Chen *et al.*
EURASIP Journal on Advances in Signal Processing (2023) 2023:123
https://doi.org/10.1186/s13634-023-01082-3

EURASIP Journal on Advances in Signal Processing

RESEARCH Open Access

Efficient and privacy-preserving multi-agent systems for smart city carpooling with *k*-regret queries and differential privacy

Fei Chen[1], Xinjian Zhang[1], Bo Ning[1*], Chao Yang[2] and Xiao Jia[3]

*Correspondence:
ningbo@dlmu.edu.cn

[1] School of Information Science and Technology, Dalian Maritime University, Dalian 116026, LiaoNing, China
[2] Information and Communication Branch, State Grid Liaoning Electric Power Co., Ltd., ShenYang 110000, LiaoNing, China
[3] Jiangxing Intelligence inc., ShenZhen 518000, GuangDong, China

Abstract

Multi-Agent Systems are characterized by the presence of multiple independent agents and find diverse applications. In the context of smart cities, MAS is employed in traffic management to enhance operational efficiency, optimize resource utilization, and improve the quality of life for residents. This research paper focuses on the design of a multi-agent intelligent scheduling system, where passengers, vehicles, and carpooling platforms serve as intelligent agents. The primary objective of passengers is to identify suitable shared vehicles based on criteria such as waiting time, budget constraints, and willingness to carpool. Vehicles, on the other hand, organize their schedules based on passenger demands and designated routes. The carpooling platform takes into account resource allocation priority and optimization problems to ensure the efficient operation of the system. To address the issue of vehicle ordering, *k*-regret queries are utilized, while passenger preferences provide insight into determining loss factors. To safeguard privacy, differential privacy techniques and a random response mechanism are employed when dealing with multiple passenger queries. Furthermore, a direction-preserving insertion verification method is implemented to mitigate computational complexity. The effectiveness and efficiency of the proposed approach are validated through experimentation.

Keywords: Multi-agent systems, Ride sharing, *k*-regret query, Differential privacy

1 Introduction

Multi-agent systems are primarily applied in smart cities to achieve intelligent, efficient operations and optimize resource utilization [1, 2]. Technologies such as information technology and the Internet of Things (IoT) are leveraged in smart cities to enhance urban management and service levels. Various subsystems within a city can be connected by multi-agent systems, forming a collaborative network. Areas such as transportation, energy, environment, security, healthcare, and education are encompassed by these subsystems. The field of transportation provides an example of the application of multi-agent systems in smart cities [3]. The use of ridesharing or taxi services is facilitated by multi-agent traffic systems, reducing the reliance on private vehicles, alleviating traffic congestion, and mitigating environmental pollution. Furthermore, passenger

safety, comfort, and convenience are enhanced by intelligent traffic systems, thereby increasing satisfaction and usage of public transportation. Consequently, the implementation of intelligent traffic systems plays a crucial role in promoting the popularity of shared mobility solutions and ensuring the sustainable development of the future transportation industry.

In the realm of intelligent transportation systems employing multiple agents, both industries and researchers are developing methods to combat difficulties, such as aggravated traffic congestion, intensified pollution, and heightened urban energy consumption. By providing shared commuting, carpooling holds the potential to satisfy travel needs, reduce energy consumption, and relieve traffic congestion [4–6]. For example, recent statistical data indicate that in Shanghai, there are around 120,000 intersections, 40,000 taxis, and over 400,000 taxi trips per day. However, the low vehicle utilization rate implies that most cars run without passengers, and the average occupancy rate for taxis or private cars is typically less than two people per car. Carpooling is a service that utilizes geographic location devices to organize shared rides dynamically, mainly employing taxis and private cars. These platforms transmit ride requests, which include pick-up location and destination, to nearby taxi drivers who determine whether or not to accept the request. Although some studies [7, 8] have proposed advanced improvement methods, the previous methods cannot adequately measure the degree to which taxis meet constraints that reflect the passenger experience. Furthermore, the problem of passenger privacy infringement has been neglected when pairing vehicles and travelers, potentially resulting in loss or privacy violations for passengers. For example, consider a scenario where a user regularly prefers carpooling and is willing to pay more for an optimal experience on a shared transportation platform. However, if the platform's data on user preferences were to be leaked, it could reveal users' carpooling preferences and payment records. Malicious parties could exploit this information for targeted phishing attacks, leading users to pay unreasonable fees. Furthermore, some applications employ "big data discriminatory pricing" (BDDP) techniques to analyze user preferences for their financial gain. Unfortunately, this has led to records such as Didi Global Inc. being fined a record 8.026 billion yuan for a data breach in 2022. Hence, to maintain user privacy while understanding their preferences and requirements, the implementation of differential privacy technology is necessary. This approach assists in providing improved travel services, elevating user satisfaction and loyalty, increasing competitiveness, and encouraging sustainable development.

In response to the above problems, the Dynamic Ridesharing Model [9] researched by academia provides a new idea, which dynamically matches the trip requests submitted in real-time with the vehicles traveling on the road network. This type of travel is more flexible and realistic and does not require passengers to have the same pick-up and drop-off locations, nor does it require passengers to submit their travel plans in advance, which places higher demands on the real-time vehicle-request matching. When ridesharing occurs, the route of the vehicle depends on the order of the boarding and alighting locations of the passengers on board, and the vehicle picks up and drops off passengers along the way, continuously completing the travel requests of different passengers, while the fare is recalculated based on certain criteria. As the route of different passengers will be

different due to different starting and ending points, to let new passengers on board, the vehicle will generally deviate from the original route of travel, which will cause negative effects on the passengers on board such as detour and delayed arrival time and in order to deliver each passenger, the route will not be the best route for each person; these factors will cause a certain degree of passenger dissatisfaction. On the other hand, when the degree of loss is quantified, i.e., when the loss appears, the ridesharing problem can be transformed into a problem of matching the most suitable vehicle to complete the trip. In this way, the loss factors are taken into account and are the main aspect to be considered. In this paper, furthermore, these factors are the privacy (preferences) of each passenger that needs to be protected, and we use Laplace noise to protect the user's preference information while computing the loss. Finally, to preserve the privacy of multi-passenger ridesharing information in subsequent queries, we utilize a random response mechanism to prevent the inquirer from deducing whether two people are sharing a car. As a result, this study aims to delve into the dynamic ridesharing problem, with the expectation to improve the quality of ridesharing trips and passenger satisfaction, which will further facilitate the sharing economy concept by enabling easier travel and ameliorating traffic congestion.

The contributions of this paper can be concluded as follows:

1. In this work, we introduce a novel ranking mechanism in the car-sharing problem. This mechanism incorporates a multi-intelligence system that efficiently handles dynamic requests. We rank candidate cabs based on their regret value, which is calculated using the min–max regret method. This approach allows us to quickly satisfy user requests while considering the potential regrets associated with each cab choice.
2. A function f is defined for computing the regret value, which is based on the user's choice during the matching process and incorporates differential privacy so that the user's preference can be protected. Furthermore, we also protect co-rider information using the Random Response Mechanism in the subsequent query.
3. To enhance the efficiency of the proposed algorithm, we propose a flexible direction-preserving-based filtering approach. This approach can quickly identify and ignore unqualified taxis, reducing redundant computation.
4. We conduct experiments to verify the effectiveness and efficiency of our approach.

The rest of the paper is organized as follows. In Sect. 2, we define the problem and discuss the classification of existing privacy-preserving methods, related definitions, and properties. In Sect. 3, we describe how to rank taxis with differential privacy, preserve information for subsequent queries and the method that accelerates the algorithm. Experimental results are presented in Sect. 4, and conclusions are drawn in Sect. 5.

2 Preliminary work

2.1 Related work

2.1.1 Ridesharing problem

A road network $G = (V, E, W)$ consists of a vertex set V and an edge set E. Each edge e_{ij} from v_i to v_j is associated with a weight w_{ij} indicating the traveling cost along this edge.

Given two nodes in the road network, a path along with them is a vertex sequence. The path cost $w(p) = \sum w_{ij}$ is the sum of all edge costs $w_{i,i+1}$ along the path, the minimum sum of the path is corresponding to the shortest path.

Use A_p for Agent passenger and A_t for Agent taxi. A trip request R represents a A_p's request for a taxi ride that is associated with a timestamp $R.t$ indicating when the request is submitted, a picked-up point $R.p$, and a destination point $R.d$. A request has details such as the length of the shortest path $R.sl$, the length of the estimated path $R.el$, and the estimated arrival time $R.et$. A A_t status T represents the real-time state of a A_t and is identified by a unique identifier $T.id$, it also contains the current geographical location $T.l$, and the number of on-service requests $T.n$, a current schedule $T.s$ and the arranged path $T.path$. A schedule is a temporally-order sequence of unfinished picked-up and destination points of n requests. The arranged path is a sequence of ordered vertex on the road network according to the schedule calculated by the shortest path programming algorithm.

In practice, the shortest paths are generally not entirely consistent between two requests. So compared with the no-sharing trip, ridesharing will incur a delay due to the inserted sharing request in general. The delay combines multiple factors such as travel distance increment, travel time increment, passenger waiting time. These are called constraints or objective functions. General ridesharing systems aim to select the right taxi which meets the constraints or achieves the minimum objective function of that request.

The basic framework of the multi-agent ridesharing service is shown in Fig. 1. A A_p submits a request R to the system, the system invokes the A_t *searching* module to search for a set of candidate taxi set T_i which is filtered based on the A_ts' index. Then, this set is pushed to the *scheduling* module to try to insert R into the schedule of a A_t in the set T_i. Next, the system chooses the right one. Finally, the system informs both the A_p and A_t and its submit updated status to the *index* module.

It should be noted that A_t's schedule will be reordered after accepting the new request, and it is a special case of the *traveling salesman problem*(a well-known NP-complete problem, which has already been proved to be NP-complete [10]) with order constraint and without the demand for returning the starting point. And the problem of minimizing

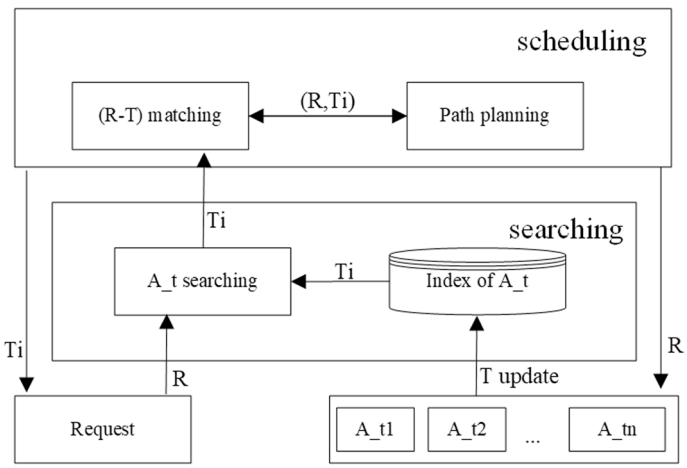


Fig. 1 Framework of multi-agent ridesharing service

the total travel distance of all A_ts for the whole query stream is also NP-complete and can be proved to be a generalization of the *traveling salesman problem*. In [11] for proof. Some literature [7, 11] choose to remain the intact order of points in the current schedule when inserting a new request. Some, like [12, 13], choose to reorder the A_t schedule completely by the kinetic tree approach combined with the pruning strategy.

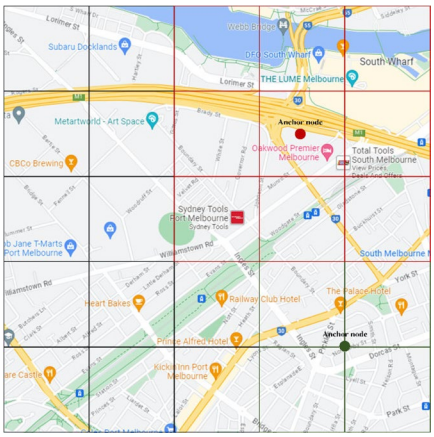
In multi-agent solving of intelligent dispatching problems, MATSim [14] is an agent-based traffic simulation framework. An iterative co-evolutionary learning approach is utilized, where each agent attempts to maximize their daily score under a given activity plan. Positive scores are given to agents for successfully completing scheduled activities (e.g., work), while negative scores are given for being late. After each iteration, the agents evaluate their previous execution plans and derive their scores. While the remaining agents choose from the existing plans based on the scores, some agents have the option to modify their plans by selecting new routes or alternative modes of transportation. Similar to MITO, MATSim is an open-source Java program that facilitates the integration of both into a single software package. Multiple extensions are included in MATSim for simulating on-demand mobility systems [15]. An extension for simulating (ride-sharing) autonomous taxis was developed by [16]. It was further extended by [17, 18], incorporating different operational strategies and algorithms for operating (ride-sharing) autonomous on-demand systems. The Demand Responsive Transportation (DRT) extension developed by [19] was utilized for this study. When trip requests with pick-up and drop-off coordinates are submitted, vehicles that can serve the request within the defined maximum waiting time without exceeding the maximum travel time of any other passenger or vehicle are searched for by the algorithm. The performance of DRT systems highly relies on service parameters, as demonstrated by [19, 20].

To meet the time constraints of request processing and solve the time-consuming problem of minimizing total travel distance, a greedy strategy, which only takes A_ts around the request into consideration based on grid indexing, is performed to speed up response time. In path re-planning, we use an insertion check method of remaining the original order likes [7, 11, 21].

2.1.2 Grid-partitioned map and adjacency matrix

For the purpose of selecting the most preferred taxi, a straightforward approach in [22] is estimating all taxis for request. Unfortunately, this idea is too time-consuming because the computation is expensive, and the number of taxis is huge. And some vehicles can be filtered directly which are outside the passenger waiting time constraints for the remote distance between them. So, we want a taxi searching process to select a small set of taxis quickly and wisely, which are likely to satisfy the requirements.

Similar to the taxi index structure used in most literature [7, 11], we adopt a road network based on grid partitioning and select road network nodes near the geographic center of the community as anchor nodes. As shown in Fig. 2(a), taxis are indexed according to the grid corresponding to their current location. Assuming that each grid unit is folded onto its anchor node, the distance between any two points is equal to the distance between the corresponding anchor nodes. Therefore, the shortest path distance from one anchor node to another can be calculated in advance and saved in the



(a) Grid-partitioned map

$$M = \begin{array}{c} \\ g_0 \\ g_1 \\ \vdots \\ g_i \\ \vdots \\ g_n \end{array} \begin{array}{c} \begin{array}{cccccc} g_0 & g_1 & \cdots & g_j & \cdots & g_n \end{array} \\ \begin{bmatrix} \varnothing & D_{01} & \cdots & D_{0j} & \cdots & D_{0n} \\ D_{10} & \varnothing & \cdots & D_{1j} & \cdots & D_{1n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ D_{i0} & D_{i1} & \cdots & D_{ij} & \cdots & D_{in} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ D_{n0} & D_{n1} & \cdots & D_{nj} & \cdots & \varnothing \end{bmatrix} \end{array}$$

$$D_{ij} = (\, t_{ij} \,,\, d_{ij} \,)$$

(b) Grid distance matrix

Fig. 2 Grid indexing

	P	D
T_1	0.88	0.79
T_2	0.89	0.71
T_3	1	1

(a) Expense of A_t and A_p

	P	D	max
T_1	0	0.08	0.08
T_2	0.01	0	0.01
T_3	0.12	0.29	0.29

(b) Final result

Fig. 3 Example pf min–max regret approach

adjacency matrix. If we obtain speed information for the road segments, travel time can also be calculated. The grid distance matrix, as shown in Fig. 2(b), provides an approximate shortest path distance between grid units.

2.1.3 k-regret query

Extracting a few tuples from the database as query answers is an important operator in the database and many applications. Top-k [23–25] and skyline [26] query are two well-studied query operator. A promising new alternative is the *regret* − *minimizing set*, introduced by [27], which is inspired by the maximum regret minimization (*min* − *max regret*) method and hybridizes the skyline operator with top-k queries.

The maximum regret minimization method is an approach to minimize the maximum difference in the objective value of the optimal solution over all scenarios, compared to the best possible objective value achievable in each scenario. This approach is usually known as min-max regret [28, 29]. It is used for uncertainty decisions in management first and aims to optimize the worst-case performance of a solution without knowledge of a probability distribution over uncertain scenarios. As shown in the following example, Fig. 3(a) indicates the travel expense of passengers and drivers when a new passenger travels on different vehicles, the lowest expense of passengers is 0.88 and the drivers is 0.71 which are outlined in black. We calculate the difference between the expense of the passenger or driver and the corresponding minimum value, called *regret*. Then, we

choose the maximum regret value in the same vehicle and fill it in the last column; the taxi which is corresponding to the minimum value marked in a black box is the final result as shown in Fig. 3(b).

However, due to its min–max structure, these problems are typically very hard to solve, both from a theoretical and practical point of view. −*regret minimizing set* was proved to be NP-hard in [30] employing a reduction from the SET-COVER decision problem. When k=1, it is similar to the optimal solution.

2.1.4 Differential privacy

Differential privacy has a rigorous mathematical proof with strong privacy protection and quantifies the degree of privacy protection, which intuitively means that query results are probabilistically almost indistinguishable, regardless of the existence of any individual in the data set or the value of the corresponding data entry. The means to achieve this is to add noise to the query results that satisfy the definition of differential privacy [31–33]. It has therefore been widely used in recent years [34–37]. The most common mechanisms are the Randomized Response mechanism, the Laplace mechanism and the Exponential mechanism, with the trend toward distributed applications. This also led to differential privacy being divided into CDP (Center Differential Privacy) [38] and LDP (Local Differential Privacy) [39, 40]. The CDP is the traditional form, which collects data from the data provider and then analyses it, satisfying the differential privacy protection in the process of analysis, but this is all based on the fact that this data center is trusted, but many cases show that it is very difficult to find a trusted third party in real life. Therefore, LDP, which is the counterpart of CDP, works without a trusted third party because it encrypts the data before it is transmitted so that the data center does not know the information of a specific individual, and then, the third party, the data collection center, eliminates the effect of the encryption on the data as much as possible by estimation.

Differential privacy has been widely applied to the problem of ridesharing services in order to protect the personal information of passengers and drivers, in particular with respect to their location and the tracking of their movements. By using differential privacy techniques, ridesharing service providers can add some noise to the collected location data, thereby avoiding revealing personal privacy information to potential attackers. For example, [41] uses smooth sensitivity, and the utility of the data set is greatly improved by reducing the amount of noise added. [42] adds some noise to these locations to protect users' privacy. [43] defines δ location set-based differential privacy to account for the temporal correlations in location data. In addition [44, 45], monitors vehicle routes by applying differential privacy techniques to the vehicle trajectory, such as fuzzification or noise addition, and the vehicle's location privacy information can be protected. In [46], the objective function of each agent is perturbed with noise in a manner that ensures differential privacy. This approach guarantees differential privacy at the functional level and is particularly suitable for systems with asymptotically stable

dynamics. Additionally, [47] focuses on preserving the privacy of the initial states of agents in average consensus.

2.2 Preliminary definitions

The definition of k-regret query, differential privacy, some common noise mechanisms satisfying differential privacy, and the problem are described as follows:

Definition 1 Utility Function f The utility function f is a mapping function $f : R_+^d \to R_+$. The utility of a piece of data p corresponding to a utility function f is $f(p)$

Definition 2 Gain Determine the utility function f and the subset S of the full set D of data, with the benefit defined as the maximum utility obtained by the points in the subset corresponding to the utility function f.

$$\text{Gain}(S,f) = \max_{p \in S} f(p) \tag{1}$$

Definition 3 Regret Ratio

$$rr_D(S,f) = \frac{r_D(S,f)}{\text{Gain}(D,f)} \tag{2}$$

where $r_D(S,f) = \text{Gain}(D,f) - \text{Gain}(S,f)$

The regret rate is in the range of [0,1], and the closer to 0, the better the subset is close to the global optimum, indicating that the subset is well chosen. The smaller the user's regret value is, the user is satisfied with the returned results.

In addition, since it is assumed that the user's utility function does not have advanced disadvantages, a utility function set F can be set up, which contains multiple utility functions. There are many utility functions, such as monotonic functions, linear functions, convex functions. In the scope of this paper, the utility function refers specifically to the linear utility function, and the sum of the weights is 1.

Definition 4 Maximum Regret Ratio:

$$\begin{aligned} rr_D(S,F) &= \sup_{f \in F} rr_D(S,f) \\ &= \sup_{f \in F} \frac{\max_{p \in D} f(p) - \max_{p \in S} f(p)}{\max_{p \in D} f(p)} \end{aligned} \tag{3}$$

Definition 5 k**-regret Query:** Given a positive integer k, the full set D of data containing n d-dimensional item is chosen to contain as many as k points in the subset S such that the maximum regret rate of the subset is minimized.

Definition 6 ϵ-**Differential Privacy**: Given a query function $f(x) : X \to R$, and the noise is represented by r, so the final output result is $M(x) = f(x) + r$. $M(x)$ satisfies ϵ-differential privacy iff. for two data sets with Hamming distance of 1, $P[M(x) \in S] \leq e^{\epsilon} P[M(x') \in S]$ is satisfied for any output set S, where ϵ is the privacy budget.

By definition, the smaller the privacy budget, the greater the protective effect.

Definition 7 **Randomized Response Mechanism** Suppose there are n users and we want to count the percentage of a certain privacy attribute to initiate a questionnaire. Each user responds to this, and the *i th* user's answer X_i is yes or no, but for privacy reasons, the user does not respond directly to the true answer. The answer is given with the help of a non-uniform coin with probability p for heads up and $1 - p$ for tails up, which is tossed to give the true answer if heads are up and the opposite answer if tails are up.

By proving that it is satisfied $ln(\frac{p}{1-p})$-DP.

Definition 8 **Laplace Mechanism** Given a query function $f(x) : X \to R$, D, D' is a pair of adjacent data sets(e.g., there is only one element difference), the sensitivity is $\triangle f = max\|f(D) - f(D')\|_1$, the output is $M(D) = f(D) + Y$, which is the noise conforming to Laplace distribution added by Y, and its parameters are $(0, \frac{\triangle f}{\epsilon})$.

2.3 Problem definition

The scenario involves a fixed number of A_t(Agent Taxi) (n) traveling on a road network G, along with a series of requests R. The A_ts are represented by the set $\{A_{t1}, A_{t2}, ..., A_{tn}\}$. A set of candidate vehicles $\{A'_{t1}, A'_{t2}, ..., A'_{tn}\}$ is selected by each vehicle, satisfying differential privacy constraints through the calculation of regret (loss). Furthermore, when two passengers share the same vehicle more than k times, an intimate relationship is considered to exist between them. Hence, a random response mechanism is applied to protect the privacy of shared ride queries q. In order to decrease computational complexity and enhance response time, the vehicle set $\{A_{t1}, A_{t2}, ..., A_{tn}\}$ is filtered and reduced based on angles to $\{A_{t1}, A_{t2}, ..., A_{tn'}\}$ where $n' < n$ based on angle filtering, before calculating regret. At the stage, the line connecting two points p_i and $pi + 1$ is denoted by $\overline{p_i p_{i+1}}$. The direction of $\overline{p_i p_{i+1}}$ is represented by $\theta(\overline{p_i p_{i+1}})$.

3 Our approach

In ridesharing problems, due to unexpected detours or extra time taken by car sharing, A_p(Agent Passengers) on board may experience discomfort. Additionally, if a A_t(Agent Taxi) is matched with a request that is too far out of their way, even in the opposite direction, they are likely to decline the ride. It is evident that experiencing a detour and additional service time is unavoidable, hence, unpleasantness will always be an inherent feature of carpooling services. As a result, ridesharing services inherently incur a sense of dissatisfaction, referred to as regret in this paper. We aim to make the optimal choice to minimize regret, which is referred to as a 1-regret query. There are various

methods of calculating regret, and each user may prioritize different factors such as fare, and ridesharing companionship, among others. We utilize Laplace noise to protect the user's preference information during loss computations. Additionally, in subsequent investigations that involve multi-passenger ridesharing information, the Random Response mechanism is utilized to secure the user's ridesharing data privacy, preventing the inquirer from gaining knowledge that two individuals share a ride. Finally, To accelerate the algorithm, we propose a flexible direction-preserving-based filtering approach to ignore the unqualified taxis and avoid redundant computation.

3.1 Regret calculation with differential privacy

When it comes to travel issues, cost evaluation is always essential. In non-ridesharing situations, people are mainly concerned with the distance and time of the journey, with the distance typically being translated into a fare through a pricing formula. In carpooling, the problem remains primarily one of individual travel behavior, where users are less concerned with the pick-up and drop-off locations of fellow travelers, and more concerned with the actual cost of travel, travel time and distance. Since distance is just a number, the critical aspect that users care about is how much they will have to pay on arrival and how this compares to traveling alone. These user preferences represent the sensitive data that needs to be protected, and the Laplace mechanism can be used for this purpose.

3.1.1 Price strategy

Before proposing a price strategy for carpooling, it is necessary to propose constraints on the recalculation of fares when carpooling:

1. For A_p(Agent Passengers), the unit price of the distance on the carpool section is less than the unit price when there is no carpooling;
2. For A_t(Agent Taxi), the total carpool price per unit distance on the carpool section is higher than the total price when there is no carpooling.

We discuss passengers first. As shown in Eq. 4, we define the passenger's regret value as the ratio of fare with ridesharing and fare without sharing. Denote by p the regular taxi fare per kilometer, $p * R.sl$ means the fare when a passenger travels alone without a traffic jam, it will be determined once the request is generated. $R.fare$ is the actual fare of the request. When detours increased and delivery is postponed to receive a new request, that leads to a growth of actual fare and regret value.

$$R.reg = \frac{R.fare}{p * R.sl} \tag{4}$$

Based on the above constraints, the formula for calculating the price is as follows

$$R.fare = \max\left\{ p \sum_{n-1}^{c} ((1 - \lambda(n-1)) * \mathrm{dis}_n), P \right\} \tag{5}$$

Where λ is the price floating parameter, take the value from range (0, 0.25); p is the distance unit price, similar to the cab price per kilometer; assume that the actual distance of a section is diss, passenger's fare depends on the number of people n which ride together on this section, reduce the proportion of $\lambda(n-1)$, the actual fare of this section is $p(1-\lambda(n-1)) * \text{dis}_n$; Since the complete trip of passenger travel is composed of various road sections with a different number of co-passengers, let the upper limit of the number of passengers by c. The actual trip distance is and the actual travel cost is $p\sum_{n=1}^{c}((1-\lambda(n-1)) * \text{dis}_n)$.

When the passenger's cost of the ride is determined, the driver's revenue is the sum of the passenger's costs, as shown in Eq. 6.

$$T.fare = \sum_{i=1}^{c} R_i.fare \tag{6}$$

As to the drivers, the most important factor is income, and they do not care about passengers' regret. As shown in Eq. 7, we use the ratio of non-ridesharing hourly price and hourly earnings when ridesharing to measure the value of the driver's regret. Hourly earnings should more than income when carrying a single guest. Driver's hourly earnings are calculated by Eq. 8, where $R_i.fare$ is the fare of *i-th* passenger. In general, the hourly sharing income is always higher than the non-ridesharing earnings when removing free time. Furthermore, drivers' regret arises from making the wrong pick compared to a more profitable one.

$$T.\text{reg} = \frac{p}{\text{hourly\_profit}} \tag{7}$$

$$hourly\_profit = \frac{\sum_{i=1}^{c} R_i.fare}{T.at} \tag{8}$$

In this paper, we adopt the greedy strategy based on insertion planning, that is, when adjusting the existing path to reach the boarding and alighting locations of new requests, we adopt the order of the boarding and alighting locations of accepted requests on the original path and choose the location that causes the smallest detour after inserting the starting point of new requests to insert the starting point and then, insert the request endpoint after the starting point to complete the path planning. As shown in Fig. 4, in this way, the representation of paths that retain only the starting and ending points of

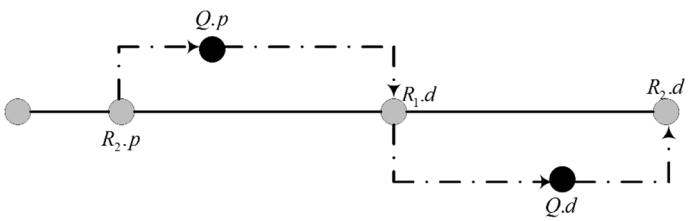


Fig. 4 Insertion path plan

the request allows the vehicle planning to be represented in segments, with the number of co-riders determined for each segment and using the optimal path internally.

It should be noted that although the planned path between key points is the optimal path, the request planning consisting of different key point line segments as a whole does not guarantee that the actual path length of the request is within a certain range or even the actual planned path quality due to the insertion strategy. Therefore, the actual path of the request needs to be measured in terms of how far it is wound compared to the optimal path.

3.1.2 Regret calculation

Due to the inevitable detours and delays in ridesharing, insertion strategies in route planning, etc., passengers' actual routes will always be weaker than the shortest routes, and the travel experience will always be weaker than the experience they would have had if they were riding alone. But how much weaker is it? Passengers do not accept these dissatisfying factors without limits. This requires a quantitative definition of these factors and reasonable constraints on the matching to ensure the quality of the matching.

No matter what task is accomplished, there is always a price to pay. No matter what good things happen when you arrive at your destination on a daily trip, in the case of a passenger's trip from the starting point to the destination, the passenger has to pay money and time to complete it. Maybe the roads are in good condition, and it took less fare and less time than usual, but the trip still costs money and time, which is called loss here. The greater the loss, the more dissatisfied the passenger is with the trip; the ideal situation is a trip with no fare and an instantaneous arrival, where the loss is 0.

Since different passengers have different factors to focus on, they need to be taken into account in the matching process. The factors that are important to passengers are private data and need to be protected. In this paper, we use time and fare, which are the factors that most passengers care about, as a measure of loss, and the fare is calculated by Eq. 5 and is determined by the distance of the trip. The reason why only the distance factor is converted into the fare and not the time as well is, firstly, because in real life it is just the distance that is chosen as the only measure of the fare; secondly, because the distance factor is more controllable than the time factor when different vehicles complete the same trip; and thirdly, because converting the time into the fare essentially predefines the weights of distance and time, no different from the previous work. Other factors are not considered.

Use $\frac{R.fare}{p*R.S_{sl}}$ to measure passenger fare loss, where $R.fare$ is the actual fare for a ride in a carpool and $p * R.S_{sl}$ is the cost when riding alone, and this value reflects the relationship between the magnitude of the carpool fare and the expected fare without the carpool. If the ratio is less than 1, the loss in the fare dimension alone will be higher than that in the non-ridership dimension, and the loss in the time dimension will increase because of the detour, and the ridesharing mode will be uncompetitive. Therefore, it is necessary to set a constraint that this ratio is less than 1 when matching is performed, to ensure the quality of matching. The range of values of passenger fare loss by constraint is [0,1].

Passenger time loss is measured using $\frac{R.S_{at}-R.S_{st}}{R.S_{st}}$, where $R.S_{at} - R.S_{st}$ represents the ridesharing delay and $R.S_{st}$ represents the minimum time spent without ridesharing. This value reflects the ratio of the delay incurred during a ridesharing trip to the minimum time spent without ridesharing. The ratio value is used instead of the delay value because it is inappropriate to use a uniform maximum delay for passengers with different travel times, e.g., a 5-minute delay must be acceptable to a passenger with a 10-minute rest trip and a passenger with a 1-hour total trip. However, the travel delay cannot be increased indefinitely, and it is a more feasible approach to set a certain percentage of the minimum usage time as the delay constraint α, such as the specified range of values [0,0.5], and the range of values of passenger time loss through the constraint is [0,1].

In summary, after defining the fare and time constraints that passengers care about when traveling together, it is necessary to define the loss of passenger travel. Passenger loss is the calculated result when the two dimensions account for different ratios, and the weights w_1 and w_2 are the coefficients in the utility function $\sum_{i=1}^{n} w_i = 1$. The specific expressions are shown in Eq 9.

$$R.loss = w_1 * \frac{R.fare}{p * R.S_{sl}} + w_2 * \frac{R.S_{at} - R.S_{st}}{\alpha * R.S_{st}} \tag{9}$$

3.1.3 Applying differential privacy in k-regret ranking

As mentioned earlier in this paper, our objective is to protect users' preferences, which are reflected in the coefficients, or weights, assigned to different factors when calculating regret values. To achieve this objective, two noise-addition methods can be employed, the first being the addition of Laplace noise with sensitivity of 1 to the weights, and the second being the addition of Laplace noise with sensitivity of $max\left(\frac{R.fare}{p*R.S_{sl}}, \frac{R.S_{at}-R.S_{st}}{\alpha*R.S_{st}}\right)$ to the outputs as a whole. In this paper, we use the former approach.

Limma 1 **Parallel Combination** It is applied to different data sets and different query functions. Similarly, M_i represents a ϵ_i-differential privacy algorithm, D_i represents disjoint data sets $M_i(X \bigcap D_i)$ indicates that the action data sets between various differential privacy algorithms do not intersect each other, so the overall satisfaction satisfies $\max\{\epsilon_i\}$-differential privacy

Theorem 1 *Given a function* $R.loss = w_1 * \frac{R.fare}{p*R.S_{sl}} + w_2 * \frac{R.S_{at}-R.S_{st}}{\alpha*R.S_{st}}$, *where* w_1, w_2 *are within the range* [0,1], *if Laplace noise with a sensitivity of* 1 *is added on* w_1, w_2 *with a privacy budget of* ϵ, *then the overall privacy budget of the computation is given by* ϵ.

1 ***Proof***

For the purpose of convenience of proof, it is assumed that the loss is comprised of a single influencing factor, that is, $R.loss = w_1 * \frac{R.fare}{p*R.S_{sl}}$.

$$
\begin{aligned}
&Pr[R.loss(w_1') = z] \\
&= Pr\left[\left(w_1 + Laplace\left(0, \frac{b}{\epsilon}\right)\right) * \frac{R.fare}{p * R.S_{sl}} = z\right] \\
&= Pr\left[Laplace\left(0, \frac{b}{\epsilon}\right) = \frac{z - w_1 * \frac{R.fare}{p*R.S_{sl}}}{\frac{R.fare}{p*R.S_{sl}}}\right]
\end{aligned}
$$

Substituting this into the definition of differential privacy yields

$$
\begin{aligned}
&\frac{Pr[R.loss(w_1) = z]}{Pr[R.loss(w_1') = z]} \\
&= \frac{Pr\left[Laplace(0, \frac{b}{\epsilon}) = \frac{z - w_1 * \frac{R.fare}{p*R.S_{sl}}}{\frac{R.fare}{p*R.S_{sl}}}\right]}{Pr\left[Laplace(0, \frac{b}{\epsilon}) = \frac{z - w_1' * \frac{R.fare}{p*R.S_{sl}}}{\frac{R.fare}{p*R.S_{sl}}}\right]} \\
&= \frac{\frac{\epsilon}{2\Delta f} exp\left(\frac{-\epsilon\left|z - w_1 * \frac{R.fare}{p*R.S_{sl}}\right|}{\Delta f * \frac{R.fare}{p*R.S_{sl}}}\right)}{\frac{\epsilon}{2\Delta f} exp\left(\frac{-\epsilon\left|z - w_1' * \frac{R.fare}{p*R.S_{sl}}\right|}{\Delta f * \frac{R.fare}{p*R.S_{sl}}}\right)} \\
&= exp\left(\frac{-\epsilon\left(\left|z - w_1 * \frac{R.fare}{p*R.S_{sl}}\right| - \left|z - w_1' * \frac{R.fare}{p*R.S_{sl}}\right|\right)}{\Delta f * \frac{R.fare}{p*R.S_{sl}}}\right) \\
&\leq exp\left(\frac{\epsilon|w_1 - w_1'|}{\Delta f}\right) \\
&= exp(\epsilon)
\end{aligned}
$$

The parallelism principle dictates that the overall privacy budget of the computation is ϵ. □

It can be determined that the noise we added follows the Laplace distribution. Therefore, the first step is to transform it into the form of $Pr[noise = x]$, which is convenient for subsequent substitution into the probability density function of the Laplace distribution. Secondly, according to the definition of differential privacy, the ratio between two distributions is calculated and the absolute value inequality is used for bounding in the final step.

3.2 Dual-side taxi matching algorithm

We propose a request-taxi matching algorithm from two sides, including the passenger side and the driver side. The so-called passenger side means that we consider only the maximum regret value among all passengers in a car. In the second aspect, we consider the driver's regret for parting with the passengers. We then use the min-max regret approach to choose the taxi with the minimum regret as the final answer.

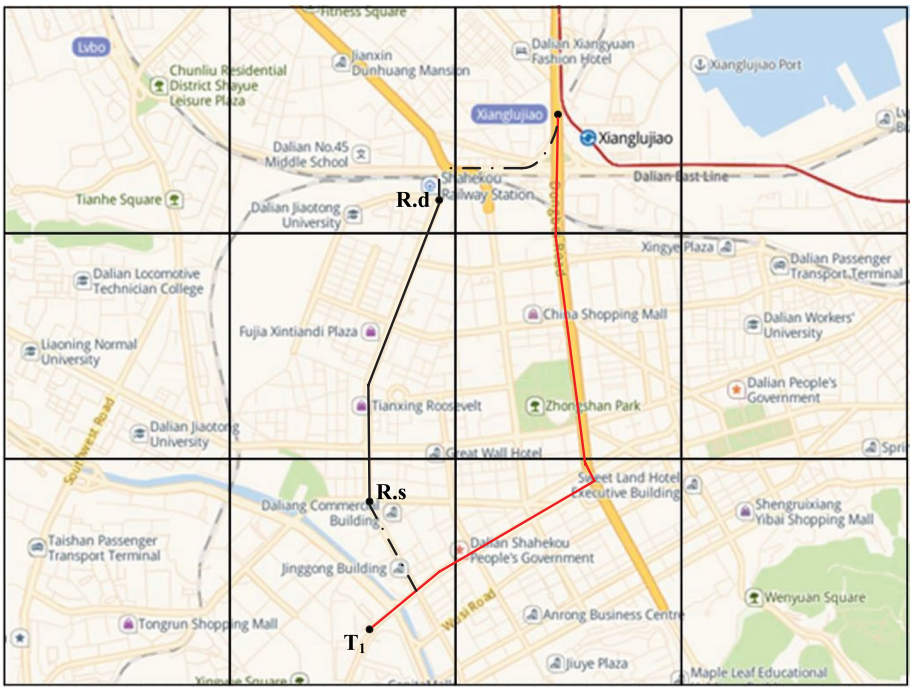


Fig. 5 A new arriving request

	P_1	P_2	P_3
T_1	0.97	0.86	0
T_2	0.95	0.8	0
T_3	1	0	0
T_4	0	0	0

(a) Initial-status

	P_1	P_2	P_{new}
T_1	0.88	0.74	0.85
T_2	0.89	0.76	0.89
T_3	1	0	1.1
T_4	0	0	1

(b) Recal-status

	P	D
T_1	0.88	0.79
T_2	0.89	0.71
T_3	1.1	0.84
T_4	1	1

(c) Regret-rep

	P	D	max
T_1	0	0.08	0.08
T_2	0.01	0	0.01
T_3	0.22	0.13	0.22
T_4	0.12	0.29	0.29

(d) Final-result

Fig. 6 Example of Dual-side matching

Require: a request R, Taxi set T
Ensure: The taxi ID with minimal regret value
for Grid cell g_i around the request **do**
  if $g_i.T_\theta < \alpha$ **then**
    $T \Leftarrow T + g_i.T$
  else
    break
  end if
end for
while T is not empty **do**
  condidateTaxi[i].receiveRequest(T_i,R)
  if $T_i.preg < minpreg$ **then**
    $minpreg \Leftarrow T_i.preg$
  end if
  if $T_i.treg < mintreg$ **then**
    $mintreg \Leftarrow T_i.treg$
  end if
end while
$ID \Leftarrow arg(min(max(T_I.preg - minpreg, T_i.treg - mingtreg)))$
return ID

Algorithm 1 Dual-side taxi matching algorithm

To dive into the details of the algorithm, consider the example illustrated in Fig. 5 and Fig. 6. It is assumed that there are at most 3 passengers in one taxi and 0 says no passenger.

At first, there is a request *R*, *R.s* is the location of the request, *R.d* is the destination, and its shortest path is marked by a red line as shown in Fig. 5. We search taxis around the request from the taxis set quickly and let us assume T_1, T_2, T_3, T_4 is searching result. T_1 is one of the candidate taxis around and its arranged path is marked by a lack line in Fig. 5. Different passengers in different taxis have different regret and their temporal status are shown in Fig. 6(a).

Secondly, since regret values of taxis are different for various newly joined passengers, we recalculate the regret value among passengers and drivers in one car after accepting the request concerning Sect. 3.1 and Fig. 6(b) as a recalculation result. Thirdly, seeing Fig. 6(c), we fill the maximum value as the representation of passengers' regret among passengers in the same taxi and the corresponding driver's regret in a new table. Finally, let us select the minimum value of each column and alters values in cells after calculating the difference between the minimum and original values as shown in Fig. 6(d). We choose the maximum value from two regret values to represent each solution and fill in a new column, deciding the taxi that corresponds to the minimum value among those as the final result marked in the black box in Fig. 6(d), then push it to the passenger and finish the service. The pseudo-code of the dual-side taxi matching is presented in Algorithm 1.

3.3 Mixed taxi matching algorithm

The Dual-said Taxi Matching algorithm can match taxi services for each request. However, there may be a situation where two requests submitted in adjacent time periods

could be matched with the same vehicle. In this case, the Dual-said taxi matching algorithm would individually match each request.

Require: a request R, Taxi set T
Ensure: The matching of request and taxi
while R is not empty **do**
  condidateTaxi[i].receiveRequest(T_i,R_i)
  if $T_i.preg < minpreg$ **then**
    $minpreg \Leftarrow T_i.preg$
  end if
  if $T_i.treg < mintreg$ **then**
    $mintreg \Leftarrow T_i.treg$
  end if
  regret(R_i,T_i)$\Leftarrow$max((T_i.preg-minpreg),(T_i.treg-mintreg))
  sort(regret)
end while
while R is not empty **do**
  result[R_i][T_i]$\Leftarrow$result(0)
  result.remove(R_i,T_i)
end while
return result

Algorithm 2 Mixed taxi matching algorithm

Then, we propose the *mixed* − *taxi matching algorithm* and it is a matching process that screens the candidate of passengers and drivers in a certain period by combining them as a whole. Concretely speaking, we do a Cartesian product of representational passengers and drivers, so we have diverse combinations. We recalculate the regret value of the passengers after joining the new requested passenger into each of the taxis as exemplified in Fig. 7(a). Then we fill in the corresponding maximum regret value of passengers and drivers in the table, seeing in Fig. 7(b). Thirdly, we select the minimum value of each column and fill values in a new column after calculating the difference between the minimum and other regret values like dual-side matching done. Finally, the

	P_1	P_2	P_{new}
(R_1,T_1)	0.88	0.74	0.85
(R_1,T_2)	0.89	0.76	0.89
(R_1,T_3)	1	0	1.1
(R_1,T_4)	0	0	1
(R_2,T_1)	0.96	0.87	0.84
(R_2,T_2)	0.82	0.95	0.93
(R_2,T_3)	1.03	0	1.32
(R_2,T_4)	0	0	1

(a) Re-calculated status

	P	D
(R_1,T_1)	0.88	0.79
(R_1,T_2)	0.89	0.71
(R_1,T_3)	1.1	0.84
(R_1,T_4)	1	1
(R_2,T_1)	0.96	0.72
(R_2,T_2)	0.95	0.7
(R_2,T_3)	1.32	0.89
(R_2,T_4)	1	1

(b) Regret representation

	P_1	P_2	max
(R_1,T_1)	0	0.09	0.09
(R_1,T_2)	0.01	0.01	0.01
(R_1,T_3)	0.22	0.14	0.22
(R_1,T_4)	0.12	0.3	0.3
(R_2,T_1)	0.08	0.02	0.08
(R_2,T_2)	0.07	0	0.07
(R_2,T_3)	0.44	0.19	0.44
(R_2,T_4)	0.12	0.3	0.3

sort

	P_1	P_2	max
(R_1,T_2)	0.01	0.01	0.01
(R_2,T_2)	0.07	0	0.07
(R_2,T_1)	0.08	0.02	0.08
(R_1,T_1)	0	0.09	0.09
(R_1,T_3)	0.22	0.14	0.22
(R_1,T_4)	0.12	0.3	0.3
(R_2,T_4)	0.12	0.2	0.3
(R_2,T_3)	0.44	0.19	0.44

(c) Final result

Fig. 7 Example of mixed matching

algorithm ranks the final regret value in ascending order and finishes the matching. See Fig. 7(c), the final service taxi for Request_1 is Taxi_2 and the taxi for Request_2 is Taxi_1 marked with a black box. The pseudo-code of the mixed taxi matching is presented in Algorithm 2.

3.4 Carpool information query

To enhance user privacy and streamline the process, the backend system of the carpool matching software may include queries that seek information about which users shared the same car at a particular time. For example, the query q_1 represents a search for the shared car of user A at a particular time. The actual return value is the taxi A_{t_1}, but if q_2 represents a search for the shared car of user B at the same time, the returned taxi would also be A_{t_1}. Thus, if both individuals repeatedly appear in the same vehicle, it can be inferred that they may have an intimate relationship. To prevent such attacks, we introduce a random response mechanism in subsequent queries. Specifically, when searching for a user's shared vehicle at a given time, we first retrieve a candidate set of vehicles A_ts with the same time period and route, and then randomly return a vehicle from A_ts. This approach ensures both the prevention of the aforementioned attacks and the availability of the data. The pseudo-code for searching the user's carpooling vehicles is presented in Algorithm 3.

Require: User A, Dataset D
Ensure: Candidate cars set C_s
Iinitiate C_s
for c in D **do**
  if c.passenger == A **then**
    Search for a set C of vehicles with the same departure time and route
    C_s.append(Random_Response(C))
  end if
end for
return C_s

Algorithm 3 Search user's carpooling vehicles algorithm

In the case of the carpool information query, shown in Fig. 8, firstly, according to the time and location of the user's carpool, We randomly draw a response from the set of candidate cars, and the final privacy budget satisfied is related to the size of the candidate set, e.g., the size of the candidate set is d, then its privacy budget is $ln(d)$.

1 ***Proof***

From Definition 6, it follows that differential privacy satisfies $P[M(x) \in S] \leq e^{\epsilon} P[M(x') \in S]$*, i.e.,* $P[M(x) = c] = 1$, $P[M(x') = c] = \frac{1}{d}$

$$\frac{P[M(x) \in S]}{P[M(x') \in S]} = d \leq e^{\epsilon}$$
$$\epsilon = ln(d)$$

□

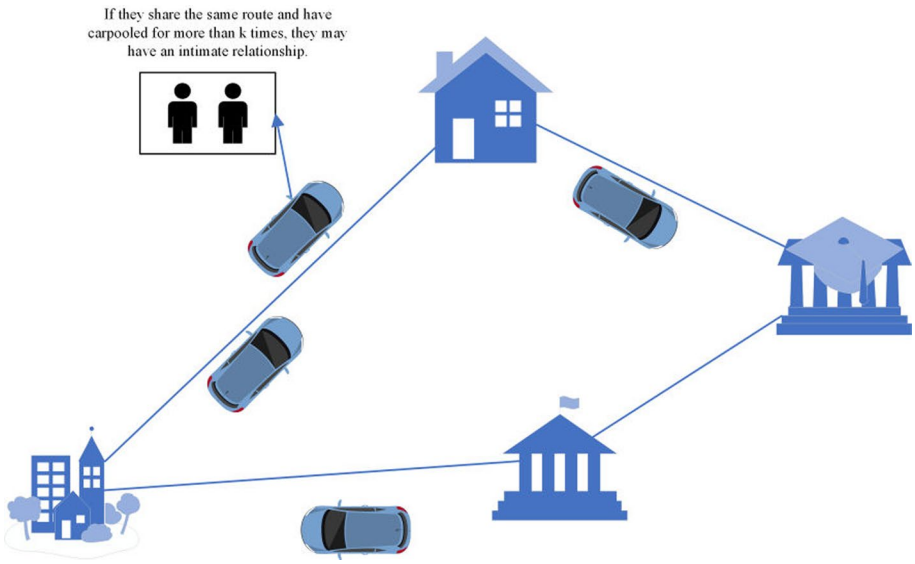


Fig. 8 Search case

Timestamp	Ride-sharing car
1676275710	c102
1678694910	c099
1679558910	c103
1681373310	c289

(a) Search case A without RR

Timestamp	Ride-sharing car
1676275710	c102
1678694910	c086
1679558910	c103
1681373310	c289

(b) Search case B without RR

Fig. 9 Example of subsequent query without RR

Timestamp	Ride-sharing car
1676275710	c202
1678694910	c010
1679558910	c143
1681373310	c276

(a) Search case A with RR

Timestamp	Ride-sharing car
1676275710	c109
1678694910	c983
1679558910	c424
1681373310	c276

(b) Search case B with RR

Fig. 10 Example of subsequent query with RR

Figure 9 shows the results without using a randomization mechanism, from which it can be observed that A and B shared three rides at the same timestamp. Therefore, it can be inferred that they have an intimate relationship. On the other hand, Fig. 10 represents the results after implementing a randomization mechanism. In Case A, c102 shares the same route segment with c202, and c099 shares the same route segment with c010. Similarly, c103 and c143, as well as c289 and c276, have identical route segments. However, only one of these cars overlaps with Case B. Consequently, the attacker is unable to deduce any intimate relationship between them.

3.5 Direction-preserving-based taxis filter

To cut down the amount of calculation further as much as possible in the final search, most of the literature is to build related index structures around the taxis, filtrate substandard candidate taxis based on time and distance constraints. However, there is little

consideration of the factor of taxi traveling direction, and it is a normal phenomenon that drivers are more willing to share the trip with passengers in the same current direction in real life. Inspired by this idea, we propose a direction filter algorithm and we will introduce it in detail next.

3.5.1 Insert verification with bounded direction error

Before getting starting the direction filter, let's introduce some concepts first.

The straight line linking two positions from p_i to p_{i+1} is denoted by $\overline{p_i p_{i+1}}$. Seeing Fig. 11(a), the *direction* of $\overline{p_i p_{i+1}}$, denoted by $\theta(\overline{p_i p_{i+1}})$, is defined to be the angle of an anticlockwise rotation from the positive x-axis to a vector from p_i to p_{i+1}. Thus, each direction falls in $[0, 2\pi)$.

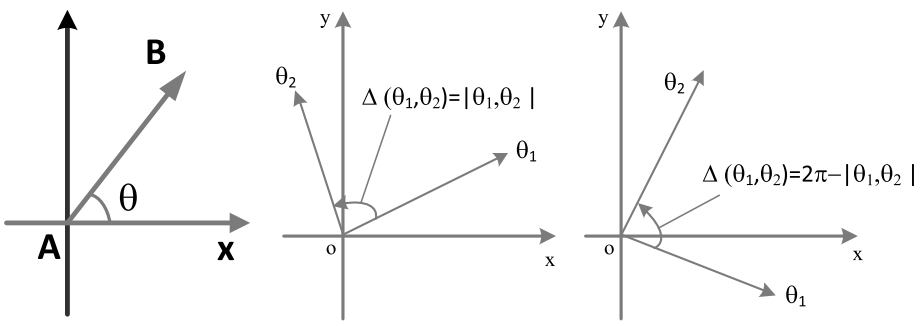
The *angular difference* between two directions θ_1 and θ_2, denoted by $\Delta(\theta_1, \theta_2)$, is defined to be the minimum of the angle of the anticlockwise rotation from θ_1 and θ_2, and that from θ_2 and θ_1, i.e.,

$$\Delta(\theta_1, \theta_2) = min\{|\theta_1 - \theta_2|, 2\pi - |\theta_1, \theta_2|\} \tag{10}$$

Figure 11(b) shows two cases where $\Delta(\theta_1, \theta_2) = |\theta_1 - \theta_2|$ and $\Delta(\theta_1, \theta_2) = 2\pi - |\theta_1, \theta_2|\}$. Note that the angular difference between any two directions falls in $[0, \pi]$.

As mentioned in Sect. 2.3, we assume that the order of points in the current schedule remains intact when inserting a new request. That is, the schedule is composed of a series of points, such as requests' pick-up and delivery point, it is a kind of trajectory data. In this case, we can regard the original schedule as a trajectory simplification of reorganized schedule after inserting a new request. The detour distance or deflected direction which caused by inserting a new request can be regarded as an error due to trajectory simplification.

The trajectory simplification algorithm generally traverse each possible simplification with its simplification error bounded by threshold α. But we only compute direction error after the insert position was known. In other words, we first utilize an algorithm similar to *insertion feasibility check* in [7] to get insert position and then use Algorithm 4 to evaluate feasibility, which according to the direction error caused by the insertion after reordering.



(a) Direction (b) Angular difference

Fig. 11 cases of direction filter

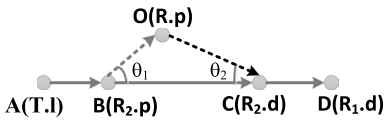


Fig. 12 Direction-based error

For instance, consider the example shown in Fig. 12. $T.l$ is the location of taxi and taxi's schedule is $T.l \rightarrow R_2.p \rightarrow R_2.d \rightarrow R_1.d$. To verify inserting $R.p$ immediately after point $R_2.p$, the algorithm first calculate the angular difference θ_1 between $\overrightarrow{BO}$ and $\overrightarrow{BC}$ and the angular difference θ_2 between $\overrightarrow{OC}$ and $\overrightarrow{BC}$. If they are both bounded by threshold α, this insert position passes the Verification. If not, the insertion fails. The pseudo-code of the mixed taxi matching is presented in Algorithm 4.

Require: threshold α,insert point O, insert position A and its next position B
Ensure: Result: direction-based error of this insert position; otherwise False
1: $\theta(AO) \Leftarrow$getDirection$(\overrightarrow{AB})$
2: $\theta(OB) \Leftarrow$getDirection$(\overrightarrow{OB})$
3: $\angle OAB \Leftarrow \min\{|\theta(AO) - \theta(AB)|, 2\pi - |\theta(AO) - \theta(AB)|\}$
4: $\angle OBA \Leftarrow \min\{|\theta(OB) - \theta(AB)|, 2\pi - |\theta(OB) - \theta(AB)|\}$
5: err $\Leftarrow \max\{\angle OAB, \angle OBA\}$
6: **if** err$> \alpha$ **then**
7: **return** False
8: **else**
9: **return** err
10: **end if**

Algorithm 4 Insertion Verification with Bounded Direction Error

3.5.2 Theoretical properties

Since *distance bounded insert verification* can get insert position and filter out some taxis which cannot meet the distance constraints, is it better than the former Inspired by [48], if insert position satisfies insertion verification with bounded direction error, it will also meet some theoretical properties.

Limma 2 Let α be the threshold of direction error, if direction error, which is caused by inserting a new point, is bounded by α, the detour distance and deviation degree(i.e., vertical distance) are also bounded by α, we have

$$0 \leq \text{detour} \leq \left(\frac{1}{\cos\alpha} - 1\right) * dis_{NS} \tag{11}$$

$$d_{\perp} \leq \frac{1}{2}\tan\alpha * dis_{NS} \tag{12}$$

1 ***Proof***

Seeing Fig. 13, we assume to insert node O between nodes A and B in the original path. If it meets the direction constraint, there are $\theta_1 \leq \alpha$ *and* $\theta_2 \leq \alpha$. *Thus, we have* $\cos\alpha \leq cos\theta_1 = \frac{AO'}{AO} \leq 1$ *and* $\cos\alpha \leq cos\theta_2 = \frac{O'B}{OB} \leq 1$. *After simplification, they are* $0 \leq (AO - AO') \leq \frac{1-\cos\alpha}{\cos\alpha} * AO'$ *and* $0 \leq (OB - O'B) \leq \frac{1-\cos\alpha}{\cos\alpha} * O'B$. *And obviously,* $\text{detour} = ((AO - AO') + (OB - O'B))$. *Equation 11 is right.*

Values of $\angle PAO'$ and $\angle PBO'$ are equals to α, thus points are in $\bigtriangleup PAB$ and there is $PQ = \frac{1}{2} * AB * tan\alpha$. We know $PQ = \frac{1}{2} * AB * tan\alpha$, so Eq. 12 is workable. □

In addition, if we rule passengers' regret value can't be more than 1, detour distance is constrained by regret referring to Sect. 3.1. We have

$$\text{detour} \leq \frac{(\text{num} - 1) * \lambda}{1 - (\text{num} - 1) * \lambda} * dis_{NS} \tag{13}$$

1 ***Proof***

If passengers' sharing travel cost is no more than ride alone, $(\text{dis}_{NS} \geq \sum_{\text{num}=1}^{c} (1 - (\text{num} - 1) * \lambda) * d_{\text{num}})$ *is tenable. In the optimal case, i.e., there is more than one person during the whole trip, the right side of inequality obtains the maximum* $(1 - \lambda(\text{num} - 1)) * \text{dis}_S$. *So, there is* $\text{dis}_{NS} \geq (1 - \lambda(\text{num} - 1)) * \text{dis}_S$. *Now we know* $\text{dis}_S - \text{dis}_{NS} = \text{detour}$. *We can get Eq. 13 after simplifying according to above formula.* □

There are Eqs. 11 and 13 that bound detour distance, respectively. When we assume that taxis in candidate set after direction filtering also satisfy the restraint in which passengers' regret is no more than 1, they should meet Eq. 13. Thus, they meet the distance constraint once they meet the direction constraint. So, we need the following inequality holds:

$$\text{detour} \leq \left(\frac{1}{\cos\alpha} - 1\right) * \text{dis}_{NS} \leq \frac{(\text{num} - 1) * \lambda}{1 - (\text{num} - 1) * \lambda} * \text{dis}_{NS} \tag{14}$$

If $\lambda = 0.2$ and $c = 3$, Eq. 14 can be simplified to $\cos\alpha \geq 0.6$ and $0 \leq \alpha \leq 53°$.

That is to say, even when each aspect is optimal selection, passengers' travel costs when they are served by taxis in the candidate set filtered by using the threshold

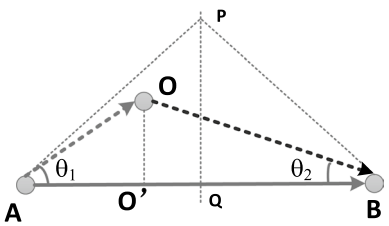


Fig. 13 Proof of theoretical properties

$\alpha = 53°$ are not higher than hailing a taxi alone. When α is set to 0, taxis cannot detour and their schedule is the request's shortest path. Only the requests whose pick-up and delivery points are both on the planning path will be accepted. The threshold can be adjusted dynamically in different situations.

3.5.3 Direction-preserving-based filter

Refer to Sect. 3.5.1, we propose *direction-preserving-based taxi filter algorithm*. After obtaining the insert position of the start and end points, we check the direction-preserving feasibility of the insert position using Algorithm 4 rather than calculating regret value directly.

For instance, consider the example shown in Fig. 14. The pseudo-code of the mixed taxi matching is presented in Algorithm 5.

Require: threshold ϵ,insert request R, taxi status T
Ensure: re-ordered schedule; otherwise False

```
initial detour
initial A, B, C, D
i ⇐ 0
while i < T.s.size do
    det1 ⇐ dis(T.s.get(i),R.p)+dis(R.p,T.s.get(i+1))-dis(T.s.get(i),T.s.get(i+1))
    if det1<detour then
        detour⇐det1
        A ⇐ i, B ⇐ i + 1
    end if
    if insertCheck(ϵ, R.p, A, B)==False then
        break
    else
        T.s.append(B,R.p)
        j⇐B
        while j < T.s.size do
            dets⇐dis(T.s.get(j),R.p)+dis(R.p,T.s.get(j+1))-
dis(T.s.get(j),T.s.get(j+1))
            if det2<detour then
                detour⇐det2
                C⇐j,D⇐j+1
            end if
            j⇐j+1
        end while
        if insertCheck(ϵ,R.d,C,D)==False then
            T.s.remove(B)
            break
        else
            T.s.append(R.d,D)
            return T.s
        end if
    end if
    i⇐i+1
end while
```

Algorithm 5 Insertion Verification with Bounded Direction Error

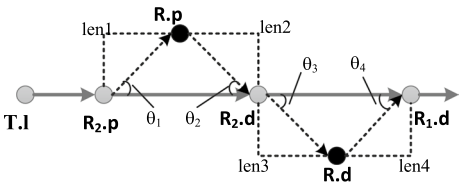


Fig. 14 Estimate of insert position

4 Experimental analyses

4.1 Metrics

The performance of differentially private ridesharing services with min-max regret is evaluated by some measures.

Satisfaction Rate(SR): is the fraction of requests that get satisfied in the ridesharing. It is a crucial criterion for measuring the effectiveness of the ridesharing system. So, $SR \in [0, 1]$ and the larger SR is, the more requests the ridesharing has satisfied.

Average Regret(AR): is the value of passengers and drivers calculated by Eq. 4 and Eq. 7. It reveals the effectiveness of ridesharing in the aspect of fare compared with no ridesharing. The smaller AR is, the more fare the passenger has saved on the basis of request being satisfied. When AR equals 1, we can assume that ridesharing does not happen.

Taxi vacancy rate(TVR): is the fraction of taxis has passengers in service during a given period. It provides an intuitive perception of how well the taxis are utilized. TVR ranges from zero to one, too. When TVR equals 1, it means a fully utilized outcome where the taxi has passengers all the time during this period.

We choose five important parameters: ϵ privacy budget, number of taxis, upper bound of regret value, taxi capacity, and fare inflating parameter. We need to first establish reasonable defaults for the parameters and then, proceed to modify the parameters one at a time to evaluate their effect. In the experiment, we set taxi capacity as 3, i.e., there are three passengers at most and the fare inflating parameter λ equals 0.2. The second and third parameters are adjusted to measure the performance of the different algorithms using the measures above.

4.2 Data set and experimental setting

We run the experiment on Windows PC with an Intel i7 processor and 16 G memory. The simulation framework is implemented in JAVA, and the simulation implementation is single-threaded.

We process 4,000 requests in the experiment, and we get them by TaxiQueryGenerator which was published by Yu Zheng (available at https://www.cs.uic.edu/sma/ridesharing/). TaxiQueryGenerator is developed based on the taxi trajectory database which contains the GPS trajectory recorded by over 33,000 taxis during a period of 87 days spanning from March to May in the year of 2011. Those requests follow *Possion* distribution.

We carry out experiments using the real road network of Beijing and two simulated road networks. The real road network is released by *Shujutang* (available at http://www.datatang.com/data/45422) which contains 171,504 road nodes and 433,392 road segments. Since we use the number of grid cells as 20*20, the cell size is 1.5km. So, we

need to process the real road network data first and select part of the data(longitude belongs in (116.2690, 116.5390) and latitude belongs in (39.7800, 40.0500)) for the next operation.

A taxi is initialized to a random vertex in the road network. At first, every taxi has an empty schedule. We set states of taxis after processing 1,000 requests as the initial states. They will follow the scheduled route when customers are on board or, otherwise, go to the nearest grid from the current grid. We assume a constant speed in the experiment, and it can be easily extended to a dynamic depending on different times and traffic conditions.

4.3 Experimental results

4.3.1 Taxi matching

We measure the performance of *dual-side taxi matching algorithm* in Sect. 3.2 and *mixed taxi matching algorithm* in Sect. 3.3. Figure 15 is the performance of the regret-based matching algorithm on a real road network.

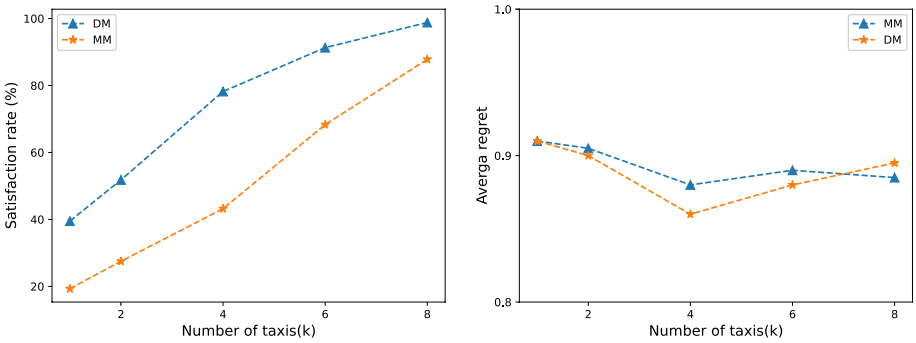
Figure 15(a) shows requests' SR(satisfaction rate) in two matching algorithms as the number of taxis changes. Two algorithms both show obvious advantages compared with non-ridesharing. This occurs naturally as there is ridesharing, and the accepted opportunities increase as well. As can be seen, since the mixed matching algorithm selects the optimal solution in a group of requests, the satisfaction rate is reduced to a little extent compared with dual-side matching. In addition, there is a steep increase in the satisfaction rate from 2K to 4K taxis and again from 6K to 8K taxis. Those make either 4K or 8K a good choice for the next experiment.

Figure 15(b) shows requests' AR(average regret) in two matching algorithms as several taxis changes. The regret values are always equal to 1 without ridesharing, it is because non-ridesharing neither allows detours nor price discounts. In contrast, two matching algorithms both first show a slight decrease and then keep rebounding in AR as the number of taxis increases. This is because as taxi quantity increases, the ridesharing opportunities increase as well. At first, the satisfaction rate of requests is low, increasing the number of taxis brings ridesharing opportunities to increase, and as a result, the satisfaction rate surges. When quantity gets larger, ridesharing opportunities do not emerge as fast as taxis arrive; therefore, AR starts increasing.

When the number of taxis is 4K, Fig. 16(a) shows requests' SR in two matching algorithms as upper bound of regret increases. The two algorithms both show a growing trend as the upper bound increases. It is obvious that more requests can be satisfied as the maximum regret that passengers can tolerate is added. While this does not necessarily guarantee that the average regret of requests reduces.

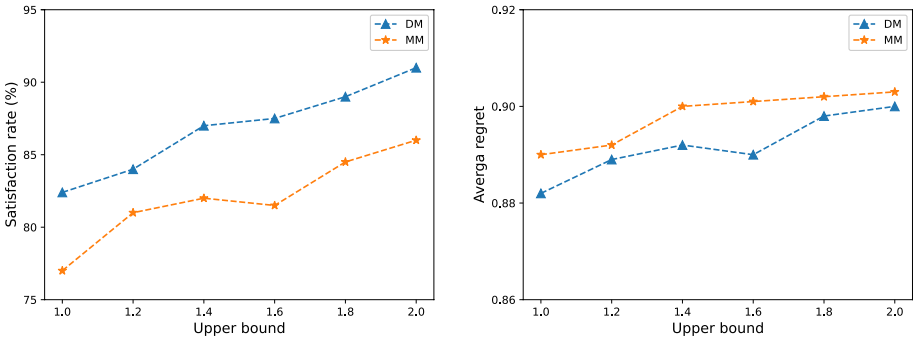
Seeing Fig. 16(b) shows requests' AR in two matching algorithms as upper bound of regret increases. It is inevitable that accepted requests after extending the upper bound have a larger regret, and it brings an inevitable worsening to the AR. Two algorithms both show these tendencies, and the mixed matching algorithm has better control of the deterioration of AR on the premise of improved satisfaction rate.

In order to consider the interests of drivers, we measure the AR and TVR of drivers and the results show in Fig. 17. Figure 17(a) shows drivers' AR(average regret) in two matching algorithms as the number of taxis changes. Drivers' average regrets are equal



(a) SR (b) AR

Fig. 15 Performance of regret-based matching algorithm on real road network



(a) SR (b) AR

Fig. 16 Performance of regret-based matching algorithm with different regret upper bound

to 1 without ridesharing when we assume a taxi will stay where it is when its schedule is empty. Drivers' AR in two algorithms are both increscents. Compared with a dual-side matching algorithm, mixed matching algorithm has a few advantages over AR. This is mainly caused by its local optimization strategy.

Figure 17(b) shows drivers' TVR(taxi vacancy rate) in two matching algorithms as the number of taxis changes. Figure 17(c) shows drivers' SE(service efficiency), i.e., the length of taxi time which has passengers in service during a given period has accounted for the entire time. With the increase in taxis number, TVR declines accordingly. This is because when the number of taxis increases, requests do not emerge as fast as new taxis arrive, more and more taxis cannot get the request.

4.3.2 Ranking with differential privacy

As depicted in Definition 6, the parameter ϵ represents the degree of privacy protection in a given privacy-preserving system. Higher values of ϵ correspond to lower levels of privacy protection, but this results in less noise addition. Empirical results from previous studies indicate a negative relationship between privacy protection and ϵ values. For instance, the work by Dwork and Roth established that the degree of differential privacy is inversely proportional to the magnitude of ϵ, and an increase in the ϵ parameter results

in a decline in privacy protection. Figure 18 shows that as ϵ increases, user satisfaction levels increase. Moreover, the DM algorithm eventually converges at 90.41 with an upper bound of 2.0, while the MM algorithm stabilizes at 83.88 under similar conditions, which is a reasonable level of convergence.

4.3.3 Direction filte

For the purpose of measuring *Direction-Preserving-based filter algorithm* in 3.5.3, we add two measurements and set the upper bound to be equal to 2.0.

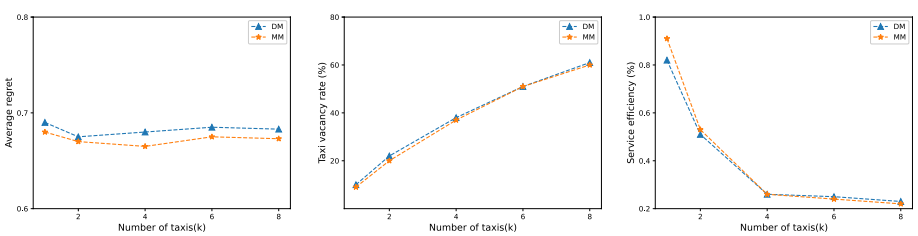
Filtration Rate(FR): is the fraction of remainder taxis after filtering. It is a crucial criterion measuring the effectiveness of the filtering and ranges from zero to one. The smaller FR is the more taxis that cannot the direction constraint the filter has been filtered.

Hit Rate(HR): is the ratio of requests that match the same taxi compared with matching result without filtering. It reveals the accuracy rate of filter algorithm. HR also ranges from zero to one. It is important to note that a higher FR does not mean a better filtering effect. While a lesser FR, the higher HR show that the filtering effect is great.

Referring to the result shown in Fig. 15, we use the number of taxis as 4k and 8k for the next experiment. Figure 19(a) shows FR of direction filter algorithms as the upper bound of threshold changes based on the set of taxis after grid-filtering. Figure 19(b) shows requests' HR of filtering as upper bound of threshold changes. The two algorithms both show small fluctuations; this is not consistent with our expectations. Combined with Fig. 17(b), the more the number of taxis is, the more taxis without passengers are. But those empty taxis will not be filtered by direction.

Next, if we make α equal to $\frac{\pi}{2}$, then, increase the number of requests in unit time and introduce an inflated parameter δ, which stands for the ratio of the total number of incremental requests to the number of initial requests. The result is greatly improved as shown in Fig. 19(c) and (d). With the increase in δ, FR is gradually dropping, and HR changes a little. This is because, with the increment in the density of requests, more and more taxis are in service. That is to say, their schedules are non-empty and they have direction errors when inserting a new request. When remainder taxis after grid-filtering grow at a speed near linearly, remainder taxis after direction filtering will have a slow growth, and filter operation can filter out more taxis out of constraints.

Further, we make the number of taxis 4K and δ equal to 5, Fig. 20 shows the result of filter operation as α changes.



(a) AR (b) TVR (c) SE

Fig. 17 Performance of regret-based matching algorithm on real road network

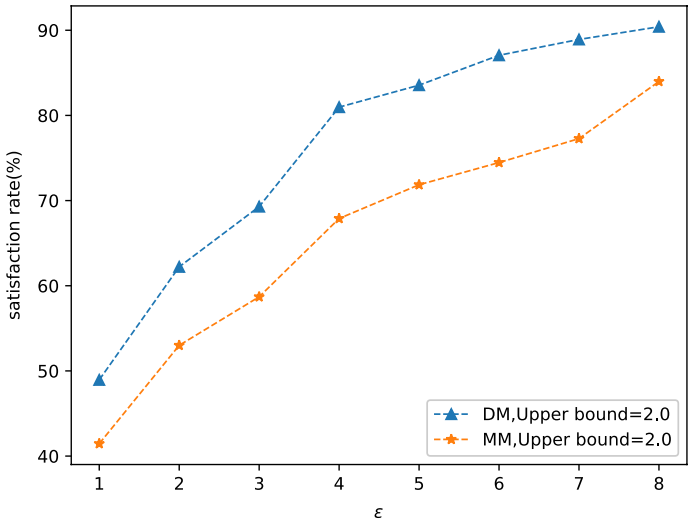


Fig. 18 Satisfaction rate with ϵ

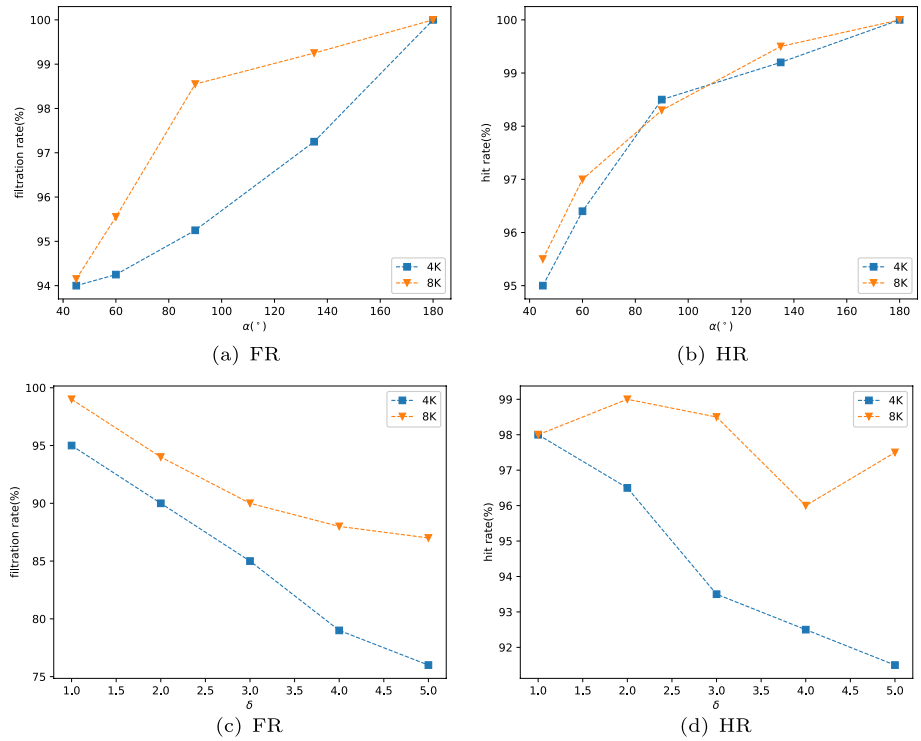
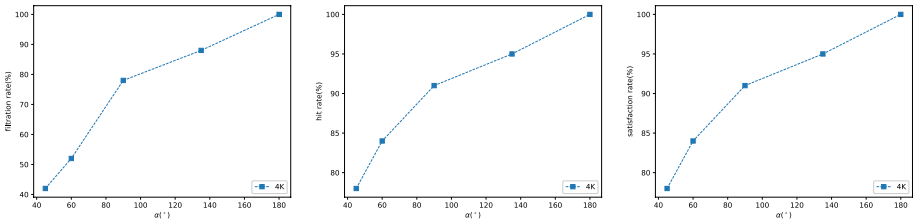


Fig. 19 Performance of direction-preserving-based filter algorithm on real road network

This indicates that the latter has a better filtering effect. And results show hit rate that reaches a higher level when the threshold is $\frac{\pi}{2}$, and this is not nearly $\frac{\pi}{3}$ proved in Sect. 3.5.2. This is mainly because we connect two points with a straight line directly, while a few paths between two points are straight lines in real life. Further, we measure the SR and AR of requests again, and the results are similar.



(a) FR (b) HR (c) SR

Fig. 20 Performance of direction-preserving-based filter algorithm on real road network

The above experimental results show that when request density is lower, using a grid-filter has achieved a good effect and this would not be better by using a direction filter. When request density and taxi density are both higher, direction filter can filter more taxis that cannot meet the constraint. Direction-Preserving-based filters can a more important role during rush hour in the city.

5 Conclusion

This paper presents a multi-agent system that utilizes the Minimax Regret method and differential privacy in order to address the carpooling problem. A new problem, referred to as regret-based carpooling, is also defined. Two matching algorithms are proposed to achieve the pairing of taxis and requests in carpooling. The first algorithm considers taxis as agents to handle each request, while the slightly more complex second algorithm can handle approximate requests made at adjacent times (closer to practical scenarios). In order to enhance user privacy protection, the adoption of privacy-preserving methods in subsequent query processes is suggested. Additionally, to improve response time, a direction-aware taxi filtering method is proposed. Several open research questions in the field of carpooling are also addressed, including traffic impact, the existence of uncertainty, passenger privacy, and the establishment of trust between drivers and passengers. The goal of this study is to tackle these issues in future research.

Acknowledgements
We thank Chao Yang and Xiao Jia for technical assistance, Fei Chen for help with the design of the model, and Professor Bo Ning for support.

Author contributions
Fei Chen performed the data analysis and wrote the manuscript; Xinjian Zhang performed the formal analysis; Bo Ning, Chao Yang, and Xiao Jia performed the validation.

Funding
This work was supported by the basic scientific research project of the Educational Department of Liaoning Province, the People's Republic of China, grant number 984230020.

Availability of data and materials
Data will be made available on reasonable request.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
All authors disclosed no relevant relationships. The authors declared no potential conflicts of interest.

Received: 12 August 2023 Accepted: 15 November 2023
Published online: 28 November 2023

References

1. J. Wang, Z. Na, X. Liu, Collaborative design of multi-uav trajectory and resource scheduling for 6G-enabled internet of things. IEEE Internet Things J. **8**(20), 15096–15106 (2020)
2. Y. Wu, S. Tang, L. Zhang, Resilient machine learning based semantic-aware MEC networks for sustainable next-G consumer electronics. IEEE Trans. Consumer Electron. **99**, 1–11 (2023)
3. Y. Guo, R. Zhao, S. Lai, L. Fan, X. Lei, G.K. Karagiannidis, Distributed machine learning for multiuser mobile edge computing systems. IEEE J. Sel. Topics Signal Process. **16**(3), 460–473 (2022)
4. L.P. Alexander, M.C. González, Assessing the impact of real-time ridesharing on urban traffic using mobile phone data. Proc. UrbComp **5**, 1–9 (2015)
5. P.M. d'Orey, R. Fernandes, M. Ferreira, Empirical evaluation of a dynamic and distributed taxi-sharing system. in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pp. 140–146 (2012). IEEE
6. F. Zwick, N. Kuehnel, R. Moeckel, K.W. Axhausen, Agent-based simulation of city-wide autonomous ride-pooling and the impact on traffic noise. Transp. Res. Part D: Transp. Environ. **90**, 102673 (2021). https://doi.org/10.1016/j.trd.2020.102673
7. S. Ma, Y. Zheng, O. Wolfson, T-share: A large-scale dynamic taxi ridesharing service. in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, pp. 410–421 (2013). IEEE
8. S. Ma, Y. Zheng, O. Wolfson, Real-time city-scale taxi ridesharing. IEEE Trans. Knowl. Data Eng. **27**(7), 1782–1795 (2014)
9. N. Boysen, D. Briskorn, S. Schwerdfeger, K. Stephan, Optimizing carpool formation along high-occupancy vehicle lanes. Eur. J. Oper. Res. **293**, 1097–1112 (2021)
10. M.W. Savelsbergh, Local search in routing problems with time windows. Ann. Oper. Res. **4**(1), 285–305 (1985)
11. L. Huang, J. Huang, Y. Xu, Z. Zhao, Z. Zhang, Driving route recommendation with profit maximization in ride sharing. Comput. J. **63**(11), 1607–1623 (2020)
12. 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 (2013)
13. Y. Xu, Y. Tong, Y. Shi, Q. Tao, K. Xu, W. Li, An efficient insertion operator in dynamic ridesharing services. IEEE Trans. Knowl. Data Eng. **34**(8), 3583–3596 (2020)
14. L. He, et al., Learning-based MIMO detection with dynamic spatial modulation. IEEE Trans. Cogn Commun. Netw. (2023). https://doi.org/10.1109/TCCN.2023.3306853
15. J. Bischoff, M. Maciejewski, Simulation of city-wide replacement of private cars with autonomous taxis in Berlin. Procedia Comput. Sci. **83**, 237–244 (2016)
16. S. Hörl, Agent-based simulation of autonomous taxi services with dynamic demand responses. Procedia Comput. Sci. **109**, 899–904 (2017)
17. C. Ruch, S. Hörl, E. Frazzoli, Amodeus, a simulation-based testbed for autonomous mobility-on-demand systems. in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3639–3644 (2018). IEEE
18. C. Ruch, C. Lu, L. Sieber, E. Frazzoli, Quantifying the efficiency of ride sharing. IEEE Trans. Intell. Transp. Syst. **22**(9), 5811–5816 (2020)
19. J. Bischoff, M. Maciejewski, K. Nagel, City-wide shared taxis: a simulation study in berlin. in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 275–280 (2017). IEEE
20. F. Zwick, K.W. Axhausen, Impact of service design on urban ridepooling systems. in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6 (2020). IEEE
21. J.-F. Cordeau, G. Laporte, A tabu search heuristic for the static multi-vehicle dial-a-ride problem. Transp. Res. Part B Methodol. **37**(6), 579–594 (2003)
22. P. Healy, R. Moll, A new extension of local search applied to the dial-a-ride problem. Eur. J. Oper. Res. **83**(1), 83–104 (1995)
23. C. Lin, J. Lu, Z. Wei, J. Wang, X. Xiao, Optimal algorithms for selecting top-k combinations of attributes: theory and applications. VLDB J. **27**(1), 27–52 (2018)
24. A. Mondal, A. Kakkar, N. Padhariya, M. Mohania, Efficient indexing of top-k entities in systems of engagement with extensions for geo-tagged entities. Data Sci. Eng. **6**, 411–433 (2021)
25. P. Liu, M. Wang, J. Cui, H. Li, Top-k competitive location selection over moving objects. Data Sci. Eng. **6**(4), 392–401 (2021)
26. H. Ying, Z. Jiande, A nonlinear service composition method based on the skyline operator. J. Syst. Eng. Electron. **31**(4), 743–750 (2020)
27. D. Nanongkai, A.D. Sarma, A. Lall, R.J. Lipton, J. Xu, Regret-minimizing representative databases. Proceed. VLDB Endow. **3**(1–2), 1114–1124 (2010)
28. R. Guillaume, A. Kasperski, P. Zielinski, A framework of distributionally robust possibilistic optimization. arXiv preprint arXiv:2210.15193 (2022)
29. S. Tang, et al., Contrastive learning based semantic communication for wireless image transmission. arXiv preprint arXiv:2304.09438 (2023)
30. S. Chester, A. Thomo, S. Venkatesh, S. Whitesides, Computing k-regret minimizing sets. Proceed. VLDB Endow. **7**(5), 389–400 (2014)
31. S. Liu, A. Liu, Z. Li, G. Liu, J. Xu, L. Zhao, K. Zheng, Privacy-preserving collaborative web services qos prediction via differential privacy. in *Web and Big Data: First International Joint Conference, APWeb-WAIM 2017*, Beijing, China, Jul 7–9, 2017, Proceedings, Part I 1, pp. 200–214 (2017). Springer

32. A. Liu, X. Shen, Z. Li, G. Liu, J. Xu, L. Zhao, K. Zheng, S. Shang, Differential private collaborative web services qos prediction. World Wide Web **22**, 2697–2720 (2019)
33. F. Wang, J. Xu, C. Liu, R. Zhou, P. Zhao, On prediction of traffic flows in smart cities: a multitask deep learning based approach. World Wide Web **24**, 805–823 (2021)
34. A. Liu, X. Shen, Z. Li, G. Liu, J. Xu, L. Zhao, K. Zheng, S. Shang, Differential private collaborative web services qos prediction. World Wide Web **22**, 2697–2720 (2019)
35. S. Ho, Y. Qu, L. Gao, J. Li, Y. Xiang, Generative adversarial nets enhanced continual data release using differential privacy. in *Algorithms and Architectures for Parallel Processing: 19th International Conference, ICA3PP 2019*, Melbourne, VIC, Australia, Dec 9–11, 2019, Proceedings, Part II 19, pp. 418–426 (2020). Springer
36. S. Ho, Y. Qu, B. Gu, L. Gao, J. Li, Y. Xiang, Dp-gan: differentially private consecutive data publishing using generative adversarial nets. J. Netw. Comput. Appl. **185**, 103066 (2021)
37. Y. Li, X. Cao, Y. Yuan, G. Wang, Privsem: protecting location privacy using semantic and differential privacy. World Wide Web **22**, 2407–2436 (2019)
38. B. Ning, X. Zhang, S. Gao, G. Li, Dp-agm: a differential privacy preserving method for binary relationship in mobile networks. Mobile Netw. Appl. **11**, 1–20 (2023)
39. Y. Wang, Y. Tong, D. Shi, Federated latent dirichlet allocation: A local differential privacy based framework. in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 6283–6290 (2020)
40. B. Ning, X. Zhang, X. Liu, C. Yang, G. Li, Q. Ma, Allocation of carbon quotas with local differential privacy. Appl. Energy **326**, 119974 (2022)
41. F. Song, T. Ma, High utility differential privacy based on smooth sensitivity and individual ranking. Int. J. Inf. Comput. Secur. **15**(2–3), 216–230 (2021)
42. U.M. Aïvodji, S. Gambs, M.-J. Huguet, M.-O. Killijian, Meeting points in ridesharing: a privacy-preserving approach. Transp. Res. Part C Emerging Technol. **72**, 239–253 (2016). https://doi.org/10.1016/j.trc.2016.09.017
43. Y. Xiao, L. Xiong, Protecting locations with differential privacy under temporal correlations. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM, (2015). https://doi.org/10.1145/2810103.2813640. https://doi.org/10.1145%2F2810103.2813640
44. Y. Liu, P. Zhou, L. Yang, Y. Wu, Z. Xu, K. Liu, X. Wang, Privacy-preserving context-based electric vehicle dispatching for energy scheduling in microgrids: an online learning approach. IEEE Trans. Emerging Topics Comput. Intell. **6**(3), 462–478 (2022). https://doi.org/10.1109/TETCI.2021.3085964
45. Y. Li, P. Zhang, Y. Wang, The location privacy protection of electric vehicles with differential privacy in v2g networks. Energies **11**(10), 2625 (2018). https://doi.org/10.3390/en11102625
46. E. Nozari, P. Tallapragada, J. Cortés, Differentially private distributed convex optimization via functional perturbation. IEEE Trans. Control Netw. Syst. **5**(1), 395–408 (2016)
47. Y. Mo, R.M. Murray, Privacy preserving average consensus. IEEE Trans. Autom. Control **62**(2), 753–765 (2016)
48. D. Zhang, M. Ding, D. Yang, Y. Liu, J. Fan, H.T. Shen, Trajectory simplification: an experimental study and quality analysis. Proceed. VLDB Endow. **11**(9), 934–946 (2018)

Publisher's Note