Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8862652, 13 pages
https://doi.org/10.1155/2020/8862652

Research Article

Privacy-Preserving Vehicle Assignment in the Parking Space Sharing System

Tingting Fu,[1] Peng Liu,[1,2] Kun Liu,[1] and Peng Li[3]

[1]*School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China*
[2]*Guangdong Provincial Key Laboratory of Information Security Technology, Guangzhou, China*
[3]*School of Computer Science and Engineering, University of Aizu, Japan*

Correspondence should be addressed to Peng Liu; perrypliu@hotmail.com

Received 17 May 2020; Revised 25 July 2020; Accepted 28 September 2020; Published 17 October 2020

Academic Editor: Ximeng Liu

Nowadays, the availability of parking spaces is far behind the quick rising number of cars. Rather than building more lots, a better way is to share private-owned parking spaces. However, this faces the challenge that users are not willing to expose their privacy to the public. To solve this problem, we propose a new architecture for parking space sharing, integrating homomorphic cryptography into the design of a secure protocol for parking space searching and booking. The proposed privacy-preserving matching scheme (PPMS) is constructed in an untrusted third-party service system including two independent entities, namely, a server and an intermediary platform. Via the participant comparison protocol (PCP), a driver can choose from the matching result and be navigated to the parking space near his destination, without knowing any information of the provider and vice versa. In the meanwhile, in order to further improve the efficiency of matching, we also propose a block algorithm based on the longitude and latitude (BABLL), which utilizes a novel partitioning scheme. The feasibility of the architecture is validated through the detailed theoretical analysis and extensive performance evaluations, including the assessment of the resilience to attacks.

1. Introduction

The number of private cars in cities continues to increase against limited parking resources year by year, which leads to the desperate parking phenomenon and brings huge inconvenience to the transportation system. According to our previous work [1], finding a parking space in a city has become the most challenging task. It is estimated that 30% of traffic jams in urban areas every day are caused by vehicles looking for free parking lots. It is one of the major cause of traffic congestion, air pollution, and social anxiety. As shown in Figure 1, it is difficult for a user to find a parking space near the destination, either because a parking lot is full or the space is reserved. In the meantime, a lot of private parking spaces are unoccupied during daytime as owners may have left for work. The obvious spatial and temporal complementarity between the owners and visitors, e.g., rational renting of parking space, could increase the utilization of private parking resources as well as profits of providers. Therefore, to build a parking space sharing system [2], which is able to match private parking space providers with requestors, is a promising way to solve the problem of parking difficulty.

Currently, the smart vehicle parking system (SVPS) can alert drivers via mobile phones if a parking space is available in a particular area [3], through 5G and edge computing [4]. However, this type of system may not work in an extremely crowded area. In addition, if a parking space is not allocated specifically for the vehicle, it may no longer be available by the time the vehicle arrives. Assigning parking spaces in advance can reduce the time and gasoline spent by drivers looking for parking vacancies, as well as ensure that they meet specified requirements of drivers.

One major challenge of implementing the parking space sharing system is that users are very concerned about privacy disclosure [5] (e.g., location and time). For example, if a malicious user used your parking space, he would know that your parking space is empty for a certain period of time. Next time, he may take over your parking space without notifying you. In the meantime, the sharing service platform may sell your personal information. Therefore, it is necessary to

Figure 1: Finding a parking space in big cities.

design an efficient spatial-temporal matching algorithm for the shared parking space system, so that users can be assigned to the nearest place while keeping their privacy protected. Many matching algorithms have been proposed in the sharing economy, such as bike-sharing business. Although these algorithms may be very efficient, the information of users is open to the service provider. As shown in Figure 2, if following the traditional system architecture, both parking space providers and parking space requestors have to upload their information to the server, then wait for the matching results. Therefore, to protect privacy of all participators, a completely trusted platform is required. However, for a variety of reasons, such as sudden external threats, internal adversaries, and insecure platforms, a fully trusted third-party platform is difficult to achieve.

In order to solve the privacy protection problem of the parking space sharing system, we propose to introduce homomorphic cryptography, so that the matching process is calculated over cryptographic data. In the system, we design a privacy-protection matching scheme (PPMS) to meet needs of privacy protection. Specifically, the solution is based on a system architecture with two separate entities: (i) a server and (ii) an intermediate platform, which come from different service providers. Under the architecture we designed, the server is only able to process the encrypted information, while the intermediate platform can only access information disguised with a participant comparison protocol [6]. This design ensures no sensitive information will be disclosed to other parties. To implement matching under encryption, we use the additional feature of the Paillier cryptosystem and design an algorithm of parking space matching based on geographical location by longitude and latitude.

To evaluate the performance of the proposed PPMS scheme, we first theoretically analyze the validity and security of PPMS, which measures efficiency by evaluating time and space complexity, and then apply various attack behaviors to verify security resiliency. Finally, we practically implement the system and conduct extensive experiments on various scenarios. Experimental results show that PPMS can work efficiently while preventing information leakage.

Our contributions in this paper are as follows:

(i) We are the first to focus on the privacy-preserving problem in the parking space sharing system and try to allocate appropriate parking spaces under anonymous situations

(ii) We propose a scheme called privacy-protection matching scheme (PPMS) to realize the matching with encrypted information, which is based on a new block algorithm (BABLL) utilizing longitude and latitude

(iii) We evaluate the effectiveness, efficiency, and security of the proposed PPMS scheme by both theoretical

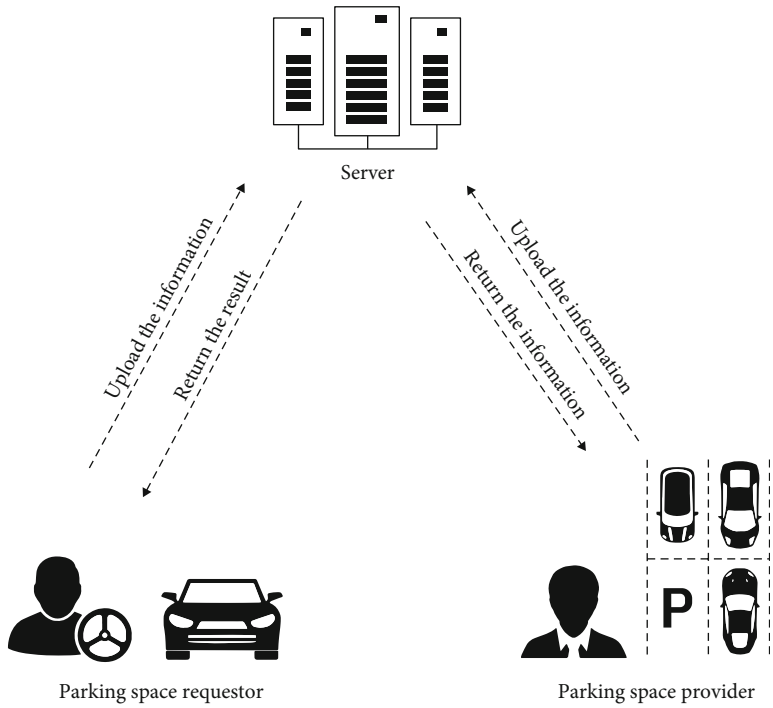

Figure 2: Traditional system architecture.

analysis and practical implementation. The results show that our scheme totally meets the needs of parking space sharing systems

The rest of this paper is organized as follows: in Section 2, literature review is conducted. We provide a brief review of the preliminary background of homomorphic cryptography in Section 3. In Section 4, we present our benchmark solution, the privacy protection matching scheme (PPMS) for shared parking systems, as well as analysis, performance evaluation, and considerations. The security analysis together with the performance evaluation are discussed in Section 5. We draw conclusion of this work in Section 6.

2. Related Work

Quite a few studies have focused on solving parking problem in big cities, for example, in [7], Xu et al. propose an integrated auction and market design method for the parking space sharing and allocation problem. There is an IoT-based platform which implements price-compatible top trading cycles and chains (PC-TTCCs) mechanism among agents. The platform's parking spaces are reassigned via a one-sided Vickrey-Clarke-Groves (O-VCG) auction. Another work taking the price into account is proposed in [8]. Zhao et al. introduce a management framework of shared parking resource in terms of time and spatial dimension [9]. Their emphasis is on the nonpreemptive utilization of parking resources. The uncertainties of P-users' and O-users' arrival and departure are simulated in the intelligent parking management system (IPMS). A practical case of parking space sharing in Suzhou is studied, and some suggestions are given by the author [10]. Some work focuses on the design of the infrastructure or architecture, e.g., in [11], it develops time shared parking management system based on BaaS cloud infrastructure in JAVA. The shared allocation model of residential area and adjacent business district in Nantong is studied in [12]. In [13], fog computing and roadside cloud are utilized to find a vacant spot, then the matching theory is applied to solve the parking problem. In [14], the research of a shared parking system can be divided into three aspects: system structure analysis, system prediction, and privacy protection. A campus parking sharing method is developed in [15], and daytime and nighttime users can share a common campus parking lot. To ensure the QoS, the number of reserved parking spaces and a probability-based access method for premium cars are studied. However, in the above existing work, more or less, one important factor is neglected, i.e., the privacy of participants.

In the related area, many researches have been done in privacy-related problems e.g., in recommendation domain, Qi et al. consider the spatial-temporal information of the QoS data and locality-sensitive hashing (LSH), and propose a location-aware and time-aware recommendation approach considering privacy concerns [16]. Xiao protects users' privacy when they perceive data through secret sharing and multiparty security calculation to prevent leakage [17]. L-PPTD [18], a privacy-preserving truth discovery framework for mobile crowd sensing systems with two noncolluding cloud platforms, is proposed. By adopting the additively homomorphic cryptosystem, it achieves the protection of each worker's sensory data and reliability information. Tang et al. also propose a framework following the same two-server model from the best-known prior solution [19]. They leverage Yao's Garbled Circuit (GC) for data comparison. In [20], the authors present two homomorphic Paillier

encryption-based reliable and privacy-preserving truth discovery schemes, for stable and frequently moving users, respectively. Li proposes a privacy-preserving reinforcement learning framework for a patient-centric dynamic treatment regime [21]. LoPrO [22] is a location privacy-preserving online algorithm which can differentially guarantee the EV location information privacy by leveraging the Laplace mechanism. Much of above work is built on protecting privacy by blending it with the additional data. To reduce users' tuning time, the privacy-preserving spatial index (PSI) method is proposed in [23]. For Internet of Vehicle (IoV), there remains a challenge to avoid privacy conflicts for computation offloading. An edge computing-enabled computation offloading method, named ECO, with privacy preservation for IoV, is proposed in [24], to address this challenge.

Many encryption schemes have been developed to support privacy-preserving. For example, ref. [25] proves the effectiveness of homomorphic encryption. In [6], Zhang proposes a comparison method in the case of encryption. Furthermore, searchable encryption can be very useful in cloud computing [26]. Besides addition and comparison, many other operations can be conducted in the case of encryption, such as in [27], and the authors propose a practical and privacy-preserving data aggregation scheme that can compute arbitrary aggregation functions without a TA. On one hand, the scheme can ensure users' anonymity and privacy protection, while on the other, the scheme is efficient in enabling participants to join or leave the system dynamically.

A blockchain-based privacy-preserving system is briefly introduced in [28]. However, only the user ID is anonymized while other information is open to everyone. Otherwise, it is not possible for users to find their trading partners. In this paper, we are going to design and implement such a system with state-of-the-art encryption technologies.

3. Preliminary

In the proposed scenario, sensitive information must be encrypted during storing, transmission, and processing. Especially for the processing, all operation have to be done without revealing any of original contents. Sorting is one of the most important functions to effectively implement the matching process, because it determines the winning position. The cryptosystem needs to be function-effective and time-efficient. According to the above analysis, we chose the Paillier cryptosystem, which supports homomorphic encryption, to implement our design. In this section, we briefly introduce the security and attack model, concept of homomorphic cryptography, and the participant comparison protocol (PCP).

3.1. Security and Attack Model. In this design, we would like to provide strong privacy protection for all participators in the parking space sharing system. Therefore, regarding the security model, we assume there is no trustworthy entity except the government approved agency, such as CA. Malicious attackers hide in the server, the intermediate platform, requestors, and providers. They are trying to act like a sham, to steal user private information, to cheat in the transaction, and to destroy services. To resist these adversarial actions, this paper considers the following attack model:

3.1.1. DDoS Attack. In this situation, the malicious user tries to expend system resources but not to conduct any actual transactions, e.g., proposing a lot of parking requests but finally cancel them.

3.1.2. Multiple Reservation Attack. In this case, multiple malicious users try to book the same parking space so as to disrupt normal functionality.

3.1.3. Adversarial Behavior Attack. Some users may damage the parking space on purpose or possess the space violating the agreed time period.

3.1.4. Statistical Attack. A common attack from malicious users is statistical attack, i.e., using a lot of known plaintext sent to the server or the platform to guess the corresponding ciphertext.

3.1.5. Personator Attack. Malicious users may intercept and capture the normal user's encrypted ID and use it to cheat in the transaction.

3.1.6. Theft of Private Information Attack. Malicious users may use all means to reveal the private information of participators, such as ID, location, time preference, distance, and other sensitive data.

3.2. Homomorphic Encryption. Traditionally, no matter data is encrypted during transmission or storing, and to get meaningful results, it must be decrypted before processing. As more and more tasks are being offloaded to a server or a cloud, it is unavoidable to expose the plaintext of the corresponding data. Instead, the ability to perform calculation on encrypted ciphertext will ensure that the processer can never reveal the content of the original data. Homomorphic cryptography is the encryption algorithm that can achieve the goal, where the operation performed on the ciphertext, once decrypted, matches the corresponding operation performed on the original plaintext. The concept of homomorphic encryption can be presented as Eq. (1),

$$\forall m_1, m_2 \in M, E_K(m_1) \odot_c E_K(m_2) \rightarrow E_K(m_1 \odot_M m_2), \quad (1)$$

where M is the set of plaintext, m is any plaintext in M, E is the encryption function, K is the encryption key, and $\odot_M$ and $\odot_c$ stand for some operators in M and the corresponding set of ciphertext C. Regarding the ideal homomorphic encryption scheme, untrusted servers can only view, operate on, and return encrypted data, thus protecting users' privacy.

With regard to the development of homomorphic encryption, the first homomorphic encryption scheme implements only partial homomorphic encryption (PHE), which means that possible homomorphic operations are limited to only one operation such as addition or multiplication. In 2009, the first fully homomorphic scheme (FHE) that would allow combined computation of the encrypted data without decryption was proposed by Gentry [29]. The

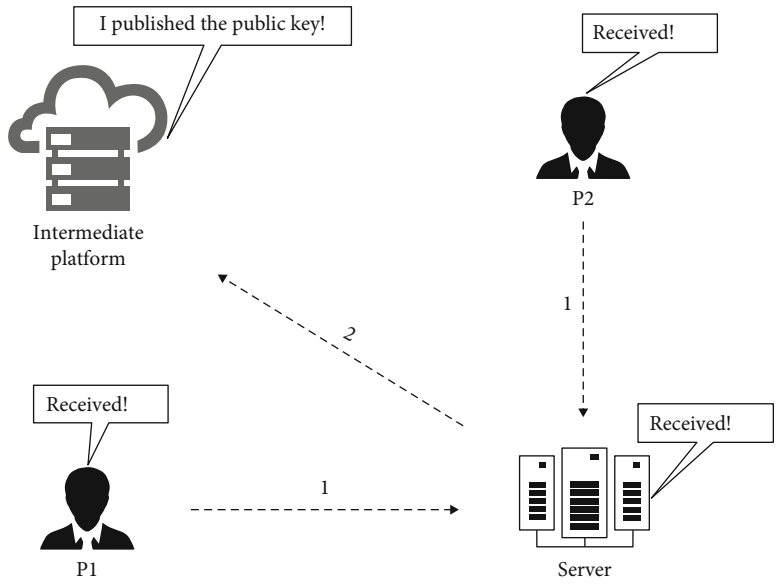

Figure 3: Participant comparison protocol.

scheme uses grid-based cryptography and guidance mechanism. Extensions to this foundation include LTV and GSW. Despite these advances, in most cases, current FHE implementations are not suited to the efficiency requirements of speed and space. As a compromise, recent trend is to use feature-reduced homomorphic encryption that has a limited number and depth of FHE operations but still yields good performance.

In the proposed parking space sharing system, we adopt and implement a Paillier cryptosystem (PC), which is an asymmetric encryption system based on the problem of calculating the n^{th} residue and is considered computationally difficult. PC is a partial homomorphic scheme that supports ciphertext addition. That is, $E((m_1) + (m_2)) = E(m_1) \cdot E(m_2)$, where E is the encryption function, and m_1 and m_2 are plaintexts. Since PC can achieve semantic security, it is impossible to determine the original plaintext in polynomial time. That is, all of data is encrypted using PC before being transmitted to an untrusted server for processing. The server will perform required operations on encrypted data and also return the encrypted outcome. Users decrypt the outcome to get the final result. Ideally, privacy is protected because the plaintext of data is not exposed to any untrusted entities.

3.3. Participant Comparison Protocol. To assign the nearest parking space to the requesting driver, we need to compare the distance of candidate places. However, distance is also part of privacy needed to be protected. Therefore, the comparison must be done on ciphertexts. Participant comparison protocol allows us to perform size comparisons without decryption. It is used to handle the returned results.

Figure 3 shows the process how participant comparison protocol works on our system. We divide it into 2 main phases as corresponding to the numbers in Figure 3.

Phase 1: the intermediate platform generates Paillier's key pair and publishes the public key. The private key is used by the platform to decrypt the ciphertext and extract the information so that it must not be leaked. The platform keeps the private key with itself at all times to ensure its safety. Participants P1 and P2 use the public key to encrypt their values and send them to the server.

Phase 2: the server uses its key pairs to interfere with the received information. The detail is shown in Eq. (2) (where *Inf* represents the values that the participants need to compare, and x and y represent the key pairs generated by the server). The server sends the interference information to the intermediate platform. The intermediate platform uses its own private key to decrypt the interference information, as shown in Eq. (3) and then directly compares the size.

$$\mathrm{PK}(\mathrm{Inf} \cdot x + y) = PK(Inf)^x \cdot \mathrm{PK}(y), \tag{2}$$

$$\mathrm{Inf} \cdot x + y = \mathrm{SK}(\mathrm{PK}(\mathrm{Inf})^x \cdot \mathrm{PK}(y)). \tag{3}$$

In Eq. (2) and Eq. (3), PK() stands for the encryption function using the public key of the intermediate platform, and SK() stands for the decryption function using the secret key of the intermediate platform. x and y are two large integers generated by the server. Inf is the information sent by users. In this design, the server can only see the encrypted information, and the intermediate platform sees only the padded information. Furthermore, neither of them would be able to uncover the original information during the calculation. In the meantime, it is possible for the intermediate platform to use the padded information to compare and sort when a proper homomorphic encryption system is selected.

4. Privacy-Preserving Matching Scheme

In this section, we will illustrate the proposed privacy-preserving matching scheme by first introducing the architecture and then the workflow, finally the detailed algorithm.

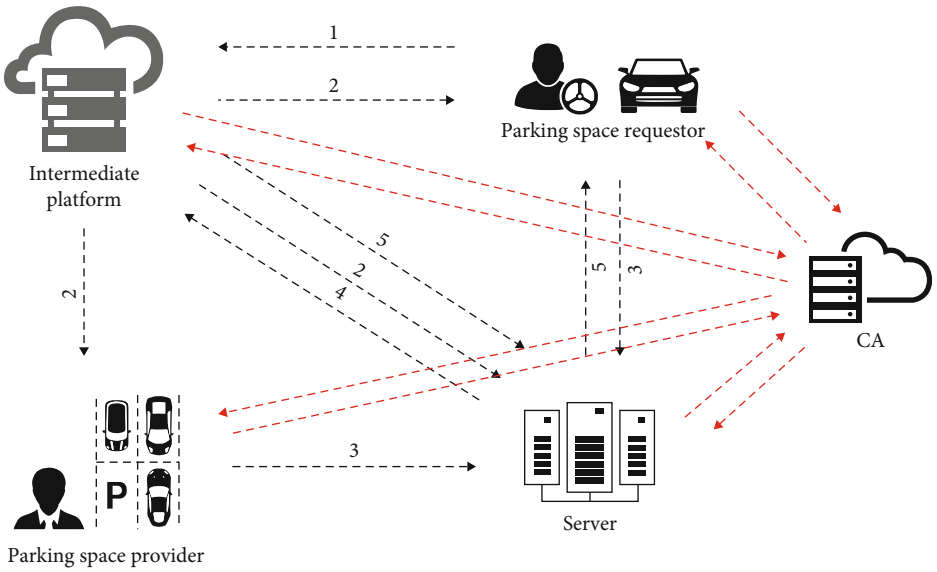

Figure 4: System architecture and workflow.

4.1. System Architecture. Figure 4 illustrates the system architecture of the privacy-preserving matching scheme, which is composed of five entities: the requestor, the intermediate platform, the space provider, the server, and the CA. These entities use modern networking technologies to communicate with each other, such as wireless area network, wide area network, cellular network, and vehicular ad-hoc network [30]. Each entity is explained in detail below.

4.1.1. Requestor. The requestor is the user looking for a parking space. He needs the logic location and access code of the desired parking space near his destination. He should have no access to the privacy information such as the provider's identity, available hours of the parking space, and the physical location (in the time of matching and selecting). Regarding the physical location, it is allowable to the requestor when he is guided to and really uses the parking space. The major actions which the requestor must take are to register in the system, provide asymmetric key pairs, and upload the encrypted requirement (by the public key of the intermediate platform) to the server.

4.1.2. Provider. The provider is willing to share his parking space to gain some profit. The information he needs to provide including location, available time, price, and payee information. He must register in the system and send the said information to the server in the form of ciphertext. Later, the requestor is allowed to update above information following the specified procedure.

4.1.3. Server. The server is a party providing privacy preserving in the matching system. It initializes the new matching process based on the requestor's request and generates a large integer pair (i.e., x and y) to encapsulate the information so that neither the intermediary platform nor the system participant can decrypt the ciphertext. The major actions that the server must take are to obstruct the encrypted requirement from the requestor and remove the obstructive signal from the result returned from the intermediate platform. The final candidate set will be sent to the requestor for selection and confirmation.

4.1.4. Intermediate Platform. The intermediate platform is isolated from the server, and they are always from different service providers. They work under semitrustful condition and abide by the rules of the proposed system. The actions it takes are to distribute the Paillier cryptosystem-based public key, receive obstructed data from the server and execute matching algorithm, encrypt the result with the requestor's public key, and return it to the server.

4.1.5. CA. CA is the authority only responsible for identity verification. It is usually set up by the government. For requestors and providers, they send their real identity to CA for pseudonames. The intermediate platform and the server can verify the identity of the requestor and the provider by pseudonames through CA.

Specifically, the server and the intermediate platform can access the public keys of requestors from the database, and as long as the results are determined, the information can be guaranteed to be returned to the parking space provider.

4.2. Detailed Workflow. To realize the privacy-preserving parking space sharing in the target scenario, some interactive steps must be followed in the system. As the number shown in Figure 4, we divide the requestor-provider matching process into five main phases. All participants are assumed having registered in the system, e.g., their identities can be verified by the official authority. The parking space providers store information about their available provisions in the server and update it from time to time. The server can not disclose these information because they are encrypted by the public key of the intermediate platform. Some useful symbols in the algorithms are given in Table 1.

Table 1: Notations.

U	The requestor
P_i	The ^{ith} provider
T_U, T_{P_i}	The time information of the requestor and providers
L_U, L_{P_i}	The location information of the requestor and providers
R	The result set
$\mathrm{SK}_U, \mathrm{PK}_U$	Secret-public key pair generated by the requestor
$\mathrm{SK}_I, \mathrm{PK}_I$	Secret-public key pair generated by the intermediate platform
B_j	The j^{th} block.
$\mathrm{PK}()$	Encryption using public key
$\mathrm{SK}()$	Decryption using secret key
x, y	Key pair generated by the server
$()^{\mathrm{pk}}$	The information is encrypted

Phase 1: request posting. The requestor who wants to secure a parking space near his destination should post a request to the intermediate platform as early as possible.

Phase 2: new process launching. During the initialization of the intermediate platform, it generates Paillier key pairs (PK_I, SK_I) and makes the public key (PK_I) available to the server and all registered participants. The intermediate platform launches a new process when it receives a new request from a requestor and informs the server that it is ready for the obstructed data.

Phase 3: information collecting. After getting the notification from the intermediate platform, the requestor uses the key PK_I to encrypt his parking information including location and time as given in Eq. (4). The requestor then sends the encrypted information ($\mathrm{inf}_U^{\mathrm{pk}}$) to the server. However, he needs to let the server know the subarea of the destination so as to reduce computational cost. The information of provisions ($\mathrm{inf}_{P_i}^{\mathrm{pk}}$) of providers is already stored in the server using the key PK_I as given in Eq. (5), as well as the ID of subarea they belong to. Once the server knows which subarea the requestor is going to, it fetches the corresponding enciphered information and uses his key pairs (x and y) to twice-encrypt the information as given in Eq. (6). Finally, the server sends these information ($\mathrm{inf}_S^{\mathrm{pk}_I}$) to the intermediate platform. In Eq. (7), it represents the process of using x and y for secondary processing. The specific reason why they are equal is as shown in Eq. (3).

$$\mathrm{inf}_U^{\mathrm{pk}_I} = \mathrm{PK}_I((T_U, L_U)), \tag{4}$$

$$\mathrm{inf}_{P_i}^{\mathrm{pk}_I} = \mathrm{PK}_I\left(\left(T_{P_i}, L_{P_i}\right)\right), \tag{5}$$

$$\left(\mathrm{inf}_U^{\mathrm{pk}_I}\right)_{x,y} = \mathrm{PK}_I(x \cdot (L_U, T_U) + y), \tag{6}$$

$$\mathrm{inf}_S^{\mathrm{pk}_I} = \left(\mathrm{inf}_U^{\mathrm{pk}_I}, \mathrm{inf}_{P_i}^{\mathrm{pk}_I}\right)_{x,y} \tag{7}$$

Phase 4: assignment calculating in this step, the intermediate platform receives the encrypted information ($\mathrm{inf}_S^{\mathrm{pk}_I}$) from the server, which is under double encryption of both the platform and the server. The platform uses its own private key to unlock the cryptograph previously encrypted with its public key and get the ciphertext (still under encryption of the large integer pair by the server) as given in Eq. (8). This ensures that the platform is not able to read the original user data. Then, the intermediate platform calculates and finds the right assignment result ($R \cdot x + y$) on these ciphertext (the specific calculation process will be analyzed in detail in the next section). The requestor generates the Paillier key pairs (PK_U, SK_U) and publishes his public key (PK_U) to the intermediate platform, with which the intermediate platform uses to encrypt the assignment result information ($\mathrm{PK}_U(R \cdot x + y)$) and sends it to the server, which ensures that the server is unable to read the original user data.

$$\mathrm{inf}_S = \mathrm{SK}_I\left(\mathrm{inf}_S^{\mathrm{pk}_I}\right) = \left(\mathrm{inf}_U, \mathrm{inf}_{P_i}, \mathrm{inf}_{B_i}\right)_{x,y}. \tag{8}$$

Phase 5: results returning when the server gets the result information ($\mathrm{PK}_U(R \cdot x + y)$), it uses its own pair keys (x and y) to decrypt the result information ($\mathrm{PK}_U(R)$) using Eq. (9). At this stage, the result is still in ciphertext so that the server cannot read it. Finally, the server sends this information ($\mathrm{PK}_U(R)$) to the requestor. The requestor uses his private key to get the result (R). From the result, the requestor should choose one desired parking space within a limited time period. If he fails to choose a parking space in time (time out) or cancels the result for three times, the requestor will be banned for a few while. This can stop DDoS attack to some extent. Upon the selected parking space is booked, the status will be updated to reserved, so that multiple reservation attack can be prevented. Moreover, two transactions are going to be conducted, i.e., a deposit will be paid to the intermediate platform, and the rent has to be paid to the space provider. To avoid exposing identifications, the payment goes through the blockchain-based strategy as proposed in [31]. When the requestor leaves the parking space in accord with the signed contract, the deposit will be returned by the intermediate platform. Otherwise, the deposit will be confiscated to punish inappropriate behaviors.

During above process, the operation of homomorphic addition satisfies our design for result and information delivery, and it removes x and y that can be realized using

$$\mathrm{PK}_u(R) = \sqrt[x]{(\mathrm{PK}_U(R \cdot x + y)/\mathrm{PK}_U(y))}, \tag{9}$$

where PK_u is the requestor's public key, and R is the result of the calculation.

4.3. Block Algorithm Based on Longitude and Latitude. In order to solve the matching problems in privacy-preserving parking space sharing, in this part, the block algorithm based on latitude and longitude (BABLL) theory is shown in Alg. 1. Next, a detailed introduction of the calculation in the intermediate platform will be given.

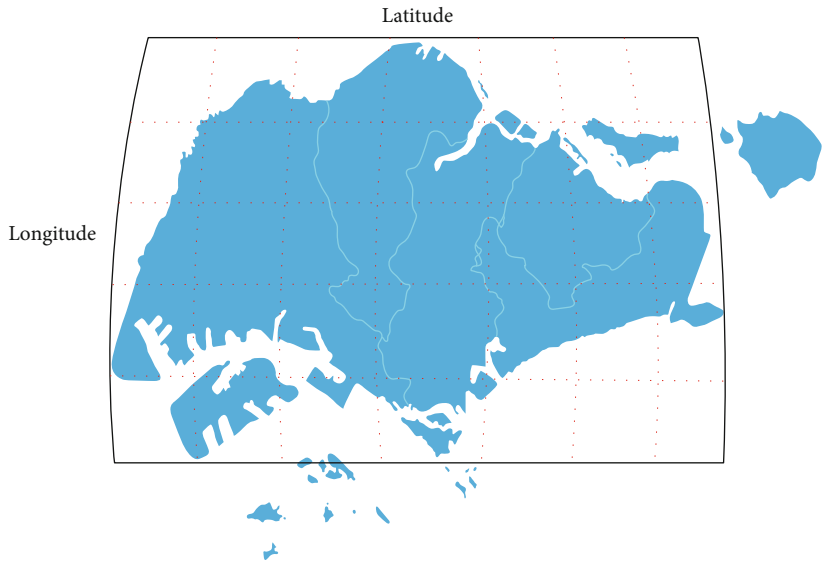

Figure 5: Divide the map of Singapore into blocks.

Neither the intermediate platform nor the server knows the actual location of each parking space; therefore, it may incur a large number of comparisons which is not efficient for the system. In this paper, we propose to divide the service area into subareas (i.e., blocks). Basically, in a city, the external trapezoid of this city may be drawn using small scales of latitude and longitude. However, it is a trade-off between the security and efficiency. Fine-grained division will bring higher efficiency but lower security, as in a small block, and there may be only a few parking spaces, so that other users may dope out the information of the provider. In this paper, we divide the trapezoid according to the length of the latitude and longitude to reduce the operation time. Figure 5 shows how we divide this trapezoid block on a rough map of Singapore. The length is about 7 km along the line of latitude and is 5.2 km for longitude.

From the previous section, we already know that we can implement the comparison and sorting mechanism in the case of encryption, and we can already specifically determine which block the requestor's parking place is. According to the requestor's preference, we need to further shrink the large block to a smaller subblock, such as the one surrounded by blue dashed lines in Figure 6. For example, the requestor may start with the small subblock such as 200 meters by 200 meters. If no available parking space is found, then the subblock can be gradually increased. Because the distance of a small segment along latitude at a certain longitude is determined, then the intermediate platform can match the available parking spaces within a certain range of the destination without knowing the exact location and the range. The calculation only needs additions, subtractions, and comparisons on ciphertexts which are supported by homomorphic encryption. The locations of these parking spaces form the result set, which will be returned to the requestor. Below, we divide the process into two situations:

The first case: the search area is within a block. Since all the data belonging to the same block as the destination has been uploaded to the intermediate platform, the search of parking spaces can limit in this block as long as the requestor's ideal range does not exceed the boundary. As shown in Figure 6, the red dots represent free parking spaces while the black dot is where the requestor wants to go. The blue box stands for the parking range that the requestor prefers.

The second case: the search area is across two or four blocks, and more data of adjacent blocks needs to be obtained from the server during matching. As shown in Figure 7, at the beginning, only the data from the block on the right is uploaded to the intermediate platform. When the requestor preferred range extends to another block, the intermediate platform will get stuck during calculation and then post a request to load parking information from the block on the left. The intermediate platform only asks for the adjacent blocks without knowing the actual block number. The server selects the block according to the request.

The matching process takes place in the intermediate platform. Here, the detailed workflow of the block algorithm based on longitude and latitude is illustrated in Algorithm 1. It mainly consists of three primary phases.

Phase 1: data fetching. The intermediate platform fetches all information ($T_U * x + y$, $T_{P_i} * x + y$, $L_U * x + y$, $L_{P_i} * x + y$ and $B_j * x + y$) from the server.

Phase 2: choose the right block. The intermediate platform makes sure whether the location (L_U) with given range is in the block based on two pairs of latitude and longitude information. If no, the intermediate platform needs to require the server to provide additional block(s).

Phase 3: find the right result set. The intermediate platform searches within the blocks to find all shared parking spaces satisfying the time preference and finally returns the sorted matching result (in descending order of distance). Note that only parking spaces located within the range given by the requestor will be checked, and the result will be encrypted by the public key of the requestor before returning it to the server.

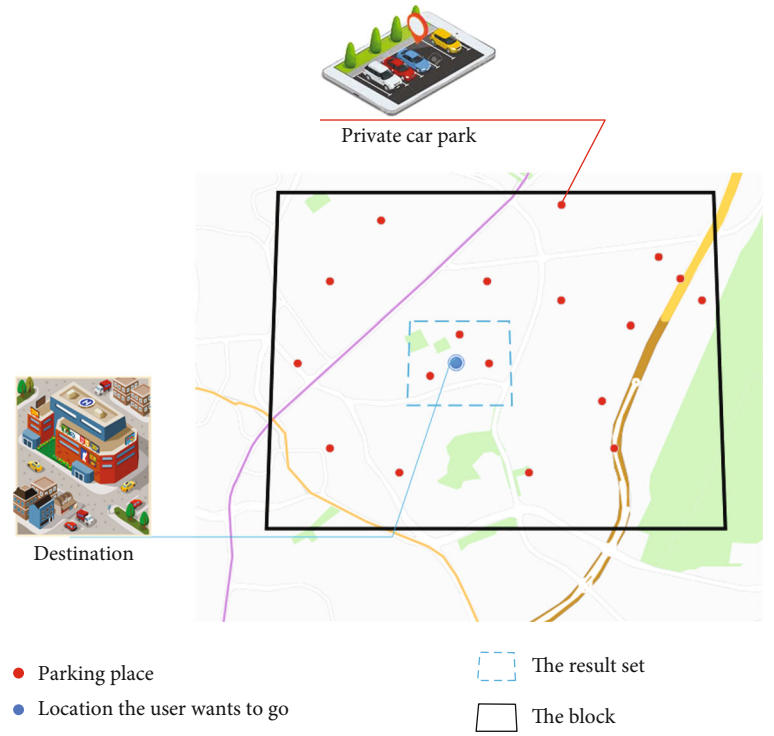

Figure 6: Search within a block.

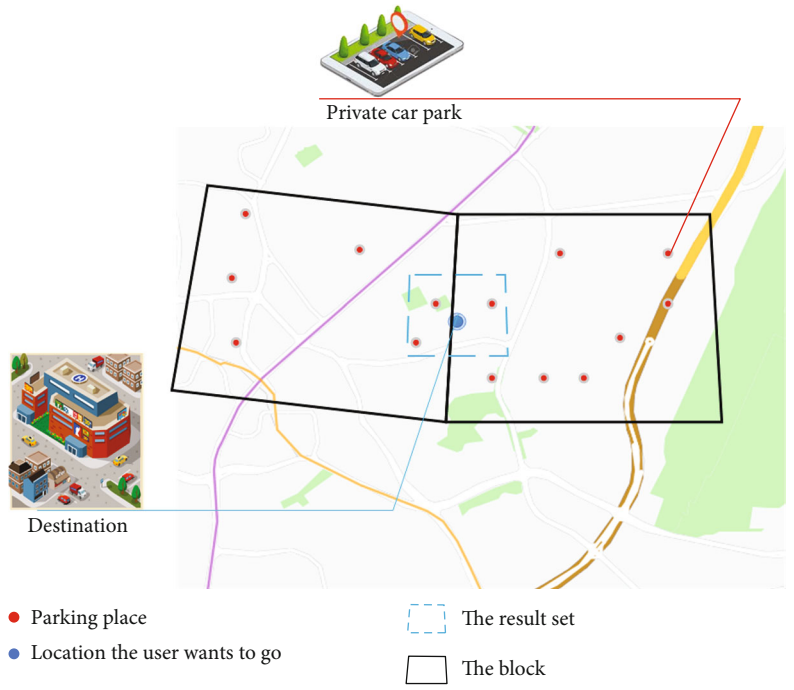

Figure 7: Search across blocks.

5. Performance Evaluation

In this section, we are going to illustrate the performance evaluation by first giving security analysis and then presenting the numerical results. Three state-of-the-art algorithms, i.e., IoT-UPM [7], campus parking [15], and blockchain-based [28], are compared in terms of security features and computation efficiency.

5.1. Security Analysis. Security is the first priority in the proposed privacy-preserving matching scheme (PPMS) and is analyzed from the following three aspects:

Fetch the information $(PK_P(T_U \cdot x + y), PK_P(T_{P_i} \cdot x + y), PK_P(L_U \cdot x + y), PK_P(L_{P_i} \cdot x + y)$ and $PK_P(B_j \cdot x + y))$ from the server; use SK_P to decrypt the information into $(T_U * x + y, T_{P_i} * x + y, L_U * x + y, L_{P_i} * x + y$ and $B_j * x + y)$

Input: $T_U * x + y, T_{P_i} * x + y, L_U * x + y, L_{P_i} * x + y$ and $B_j * x + y$;

Output: R_S;

if trapezoid of L_U is at the boundary of B_j **then**

  require adjacent blocks and merge them into B_j

end if

for T_{P_i} (i=1 to m) in B_j do

  if T_U matches with T_{P_i} then

    get T_{P_i} and merge into L_P

  end if

end for

sort L_P by distance and form the result set(R_S)

returnR_S

Algorithm 1 BABLL.

Aspects 1, manipulated message: in the PPMS, all information that involves communication between entities is encrypted to prevent eavesdropping from adversaries. In the Paillier cryptosystem, even for the same plaintext, the encryption result will be totally different each time with the same public key. Thus, statistical attacks are avoided.

Aspects 2, authentication: public certificate authority (CA) server is an essential part of the VANET; in this paper, we also assume that all participants need to register with the CA and get a few pseudonames. The intermediate platform and the server can use these pseudonames to verify the identity of participants. Unauthorized access will be immediately denied. The pseudonames will be updated at a given period of time. Therefore, the system is robust under DDoS or replay attack. CA is only responsible for ID authentication, and it does not access any data of requestors and parking space providers.

Aspects 3, privacy disclosure: true IDs of participants (requestors and providers) are protected by the public CA using pseudonames. Location, time preference, distance, and other sensitive data are calculated and matched in the form of encryption, so as to ensure no privacy leakage. Since the server will change big integers x and y constantly, there is little chance for the intermediate platform to predict the blocks, not to mentioned the actual location.

In addition to the above security analysis, we compare the proposed method (PPMS+BABLL) with the other three methods in terms of different security attributes, as shown in Table 2.

"N/A" means in the corresponding method, the data is not needed for making decision. For user identification, in the blockchain-based method, super peers and privileged users can access them. Many other attributes are evaluated as partial because privacy data (such as price and location) is open to all other peers, and only the ID is protected. In fact, many attack methods can connect the ID with the data via machine learning or big data analysis. In many cases, the attack can happen without knowing the ID, e.g., a free parking space could be maliciously occupied without necessarily knowing the owner's ID. Campus parking must be running under honesty model, such that parking permission status $X_i(k+1)$ and parking probability $p_i(k)$ for user i at time k is trustworthy. Moreover, some sensitive information, such as destination of requestors and parking duration of requestors, are only hidden from other users. Since there is only one parking lot, the location of the provider is not able to be protected. Regarding Iot-UPM, it is an auction-based method, so these sensitive information needs to be uploaded to the cloud for assignment. Compared with other methods, they either only need to accept the requestor to a fixed parking lot or users themselves are responsible for searching preferred parking space through open data, and our work has implemented high strength in privacy protection and excellent performance in resource allocation.

5.2. Numerical Results. To verify the performance of the proposed PPMS, we choose Singapore as the target city and randomly generate the data sets. At first, 100 requestors are simulated to demand parking spaces and then extend to 1000; finally, to 10000, so that the results can be observed under different network scale. The number of providers are 5000, 10000, and 35000. Each data set is tested for multiple times to eliminate the effect of odd locations. The experiment is running on a computer with 2.6 GHz Core i5 CPU and 8 GB memory.

As shown in Table 3, the computational complexity of each operation is listed. In Paillier cryptography, the length of the key determines the security strength. The duration of a single operation remains constant once the length is determined. As can be seen from the table, the time complexity of each password-related operation is $O(1)$. The sorting operation under encryption has a time complexity of $O(n \log n)$, where n is the number of elements in the sorting list. For each operation, we list the count used by each entity in the remaining columns of Table 3. Since the time complexity is no more than $O(n \log n)$, we consider the proposed scheme efficient.

In Table 4, we show the storage cost of different key lengths. Theoretically, a longer asymmetric key can provide greater security. However, the storage overhead is also significant. Using a 2048-bit key (which is sufficient in the near future), each ciphertext and signature require 512 bytes storage space. Then, to protect the 8-byte data, an overhead of 1536 bytes occurs including two signatures and one ciphertext.

We now analyze the effectiveness of the improved PPMS solution in terms of time and space efficiency, in comparison with one state-of-the-art-related schemes, a price-compatible top trading cycles, and chains mechanism [7] (IoT-UPM for short). We have implemented our PPMS and BABLL prototype in Java, including the Paillier cryptosystem following the workflow described in Section 4. The communication cost is omitted since the amount of data transmitted is relatively small compared with the bandwidth of 4G or WiFi.

First, we evaluate the time performance of the system. The key length that we use is 2048 bits, which provides sufficient security. The numbers of parking requests are set to

Table 2: Security and privacy comparison.

Attributes	PPMS+BABLL	Blockchain-based [28]	Campus parking [15]	IoT-UPM [7]
User identification	Yes, via CA	Partial	Partial	No
Destination of requestors	Yes	Partial	No	No
Parking duration of requestors	Partial	No	Partial	No
Cost function of requestors	N/A	N/A	Yes	N/A
Location of providers	Yes	Partial	No	No
Availability period of providers	Yes	Partial	No	No
Price of providers	Yes	Partial	N/A	No
Burst requests	Yes	No	Yes	No
Efficiency	Medium	Low	High	Medium

Table 3: Composition of operations in each entity.

	$O()$	Server	Provider	Requestor	IP
Key gen.	$O(1)$	0	0	1	1
Encryption	$O(1)$	1	2	$2n$	0
Decryption	$O(1)$	0	0	1	$n+2$
Padding	$O(1)$	$n+2$	0	0	0
Sorting	$O(n \log n)$	1	0	0	n
CA op	$O(1)$	2	2	2	n

Table 4: Per item space cost over key length.

Key length	512 bits	1024 bits	2048 bits
Key size	192 bytes	384 bytes	768 bytes
Ciphertext	128 bytes	256 bytes	512 bytes
Signature	128 bytes	256 bytes	512 bytes
Message	392 bytes	776 bytes	1542 bytes

Table 5: System Performance on time consumptions.

	100	1000	10000
Total time	2573 s	25238 s	255722 s
Avg. time	25.73 s	25.24 s	25.57 s
Success rate	100%	100%	100%

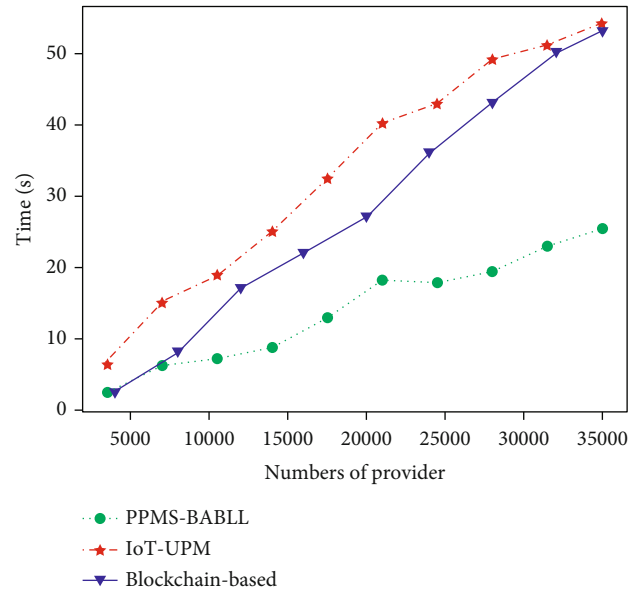

Figure 8: Comparison of time efficiency.

Table 6: System performance.

	Total number	Completed	Terminated
Process	10	10	0

100, 1000, and 10000, respectively. As shown in Table 5, we record the processing time from request posting to result set returning. The average waiting time is around 25 seconds which is acceptable as the success rate is keeping at 100%. We can also find that from 100 request to 10000 requests, the waiting time is very similar, because the major part of time is spent on encrypting not matching. The time efficiency is compared with IoT-UPM [7] and blockchain-based [28]. For BCOS adopted by the blockchain-based method, we set the ratio of privileged nodes to 0.5%. Figure 8 shows that the proposed PPMS-BABLL outperforms the time efficiency of IoT-UPM and blockchain-based under encryption. Compared to the scheme in IoT-UPM, the processing time in PPMS is significantly reduced. Regarding the blockchain-based method, with the increasing of participators, the cost of processing rises very quickly. When the number of participators is small, the blockchain-based method can achieve similar performance with our method, but it grows higher than PPMS-BABLL later to ensure security. This is due to the proposed PPMS-BABLL only fetches a small part of information for comparison and sorting in the intermediate platform.

We then evaluate the safety measurements of PPMS. In particular, we generate manipulated messages and simulate the attacks mentioned in Section 4 to verify the security of PPMS. We repeat experiments for 100 times where the total number of providers is set to 35000 and 200 for requestors. We manipulate 10% of the messages and record the rejection rate as well as the time if the process terminate due to attack. Table 6 illustrates the result of manipulating message security. As can be seen, all manipulated messages are rejected, while normal messages are accepted. Again, all 10 processes are completed correctly without termination. Therefore, it

shows that the system not only has a strong ability to resist such attacks but also has lower processing time.

6. Conclusions

In this paper, a privacy-preserving vehicle assignment problem in the parking space sharing system is studied. A homomorphic encryption-based privacy protection matching scheme (PPMS) is designed. In order to reduce the overhead of the proposed PPMS, a block algorithm based on the longitude and latitude (BABLL) is proposed. Through the security analysis, the scheme is proved to be able to protect the privacy of sensitive information such as location, time, identity of both requestors, and space providers. The scheme is also robust against attacks, e.g., DDoS or replay. We implement the prototype system and conduct comparative experiments. The results show that the proposed scheme can ensure very good success rate of matching with high time efficiency. In addition, the system resists multiple rounds of practical attacks, while maintains normal operations.

Data Availability

Data available on request, please contact the corresponding author Peng Liu (perrypliu@hotmail.com).

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the Opening Project of Guangdong Provincial Key Laboratory of Information Security Technology (Grant No. 2017B030314131) and the Natural Science Foundation of China (Grant No. 61601157).

References

[1] P. Liu, B. Xu, G. Dai, Z. Jiang, and J. Wu, "MDP: Minimum delay hot-spot parking," *Journal of Network and Computer Applications*, vol. 87, pp. 210–222, 2017.

[2] S. Marcu and A. Florea, "Smart parking system - another way of sharing economy provided by private institutions," in *2018 Thirteenth International Conference on Digital Information Management (ICDIM)*, pp. 18–23, Berlin, Germany, 2018.

[3] A. Shahzad, J.-y. Choi, N. Xiong, Y.-G. Kim, and M. Lee, "Centralized Connectivity for Multiwireless Edge Computing and Cellular Platform: A Smart Vehicle Parking System," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 7243875, 23 pages, 2018.

[4] S. Wan, X. Li, Y. Xue, W. Lin, and X. Xu, "Efficient computation offloading for Internet of Vehicles in edge computing-assisted 5G networks," *The Journal of Supercomputing*, vol. 76, no. 4, pp. 2518–2547, 2020.

[5] R. Liu, Y. Yang, D. Kwak, D. Zhang, L. Iftode, and B. Nath, "Your Search Path Tells Others Where to Park," in *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, 2017no. 3.

[6] L. Zhang, X.-Y. Li, Y. Liu, and T. Jung, "Verifiable private multiparty computation: ranging and ranking," in *2013 Proceedings IEEE INFOCOM*, pp. 605–609, Turin, Italy, 2013.

[7] X. T. R. Kong, S. X. Xu, M. Cheng, and G. Q. Huang, "IoT-enabled parking space sharing and allocation mechanisms," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 4, pp. 1654–1664, 2018.

[8] B. Tan, K. Kang, S. Xu, T. Qu, and C. Li, "First- and second-price sealed-bid auctions applied to private parking slot sharing," in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 1–5, Zhuhai, China, 2018.

[9] P. Zhao, H. Guan, and P. Wang, "Data-Driven Robust Optimal Allocation of Shared Parking Spaces Strategy Considering Uncertainty of Public Users' and Owners' Arrival and Departure: An Agent-Based Approach," *IEEE Access*, vol. 8, pp. 24182–24195, 2020.

[10] L. Yuan, "Study on the design of car-sharing parking space in the background of smart citya case study of suzhou," in *2019 International Conference on Robots Intelligent System (ICRIS)*, pp. 382–385, Haikou, China, 2019.

[11] J. Ma, L. Zhang, and Y. Xu, "Time-shared parking mechanism and application based on baas cloud infrastructure," in *2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA)*, pp. 533–536, Beijing, China, 2017.

[12] X. Liu, H. Hu, and Z. Deng, "The Shared Allocation Model of Night Parking Spaces between Residential Areas and Adjacent Business Districts," in *2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, pp. 527–532, Dublin, Ireland, 2019.

[13] O. T. T. Kim, N. Dang Tri, V. D. Nguyen, N. H. Tran, and C. S. Hong, "A shared parking model in vehicular network using fog and cloud environment," in *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 321–326, Busan, South Korea, 2015.

[14] D. Kondor, H. Zhang, R. Tachet, P. Santi, and C. Ratti, "Estimating Savings in Parking Demand Using Shared Vehicles for Home–Work Commuting," *IEEE Transactions on Intelligent Transportation Systems*, no. 8, pp. 2903–2912, 2018.

[15] W. Griggs, J. Y. Yu, F. Wirth, F. Hausler, and R. Shorten, "On the design of campus parking systems with qos guarantees," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1428–1437, 2016.

[16] L. Qi, X. Zhang, S. Li, S. Wan, Y. Wen, and W. Gong, "Spatial-temporal data-driven service recommendation with privacy-preservation," *Information Sciences*, vol. 515, pp. 91–102, 2020.

[17] M. Xiao, J. Wu, S. Zhang, and J. Yu, "Secret-sharing-based secure user recruitment protocol for mobile crowdsensing," *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, Atlanta, GA, USA, 2017.

[18] C. Miao, L. Su, W. Jiang, Y. Li, and M. Tian, "A lightweight privacy-preserving truth discovery framework for mobile crowd sensing systems," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, Atlanta, GA, USA, 2017.

[19] X. Tang, C. Wang, X. Yuan, and Q. Wang, "Non-interactive privacypreserving truth discovery in crowd sensing applications," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Honolulu, HI, USA, 2018.

[20] C. Zhang, L. Zhu, C. Xu, X. Liu, and K. Sharif, "Reliable and Privacy-Preserving Truth Discovery for Mobile Crowdsensing Systems," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2019.

[21] X. Liu, R. Deng, K. R. Choo, and Y. Yang, "Privacy-preserving reinforcement learning design for patient-centric dynamic treatment regimes," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2019.

[22] D. An, Q. Yang, W. Yu, D. Li, and W. Zhao, "LoPrO: Location Privacy-preserving Online auction scheme for electric vehicles joint bidding and charging," *Future Generation Computer Systems*, vol. 107, pp. 394–407, 2020.

[23] D. Song, M. Song, and K. Park, "A Privacy-Preserving Spatial Index for Spatial Query Processing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 2067047, 9 pages, 2018.

[24] X. Xu, Y. Xue, L. Qi et al., "An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles," *Future Generation Computer Systems*, vol. 96, pp. 89–100, 2019.

[25] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing - STOC '09*, pp. 169–178, 2009.

[26] Q. Liu, G. Wang, and J. Wu, "An efficient privacy preserving keyword search scheme in cloud computing," in *2009 International Conference on Computational Science and Engineering*, vol. 2, pp. 715–720, Vancouver, BC, Canada, 2009.

[27] C. Xu, L. Zhang, L. Zhu et al., "Aggregate in my way: Privacy-preserving data aggregation without trusted authority in ICN," *Future Generation Computer Systems*, vol. 111, pp. 107–116, 2020.

[28] J. Hu, D. He, Q. Zhao, and K.-K. R. Choo, "Parking management: a blockchain-based privacy-preserving system," *IEEE Consumer Electron. Mag*, vol. 8, no. 4, pp. 45–49, 2019.

[29] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers," in *Advances in Cryptology – CRYPTO 2010*, p. 547, Springer, Berlin, Heidelberg, 2010.

[30] P. Liu, Y. Ding, and T. Fu, "Optimal ThrowBoxes assignment for big data multicast in VDTNs," *Wireless Networks*, 2019.

[31] S. R. Pokhrel, Y. Qu, S. Nepal, and S. Singh, "Privacy-Aware Autonomous Valet Parking: Towards Experience Driven Approach," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2020.