Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9358531, 13 pages
https://doi.org/10.1155/2022/9358531

Research Article

Protecting Check-In Data Privacy in Blockchain Transactions with Preserving High Trajectory Pattern Utility

Xiufeng Xia, Tingting Hou, Xiangyu Liu, Chuanyu Zong, and Shengsheng Mu

School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China

Correspondence should be addressed to Tingting Hou; 192106074080@email.sau.edu.cn

Received 16 December 2021; Revised 25 January 2022; Accepted 9 February 2022; Published 15 March 2022

Academic Editor: Qingqi Pei

Because the blockchain is secure and untamperable, it has been widely used in many industries, such as the financial industry, digital tokens, and e-commerce logistics. The remarkable security feature of the blockchain is that the blockchain verifies the transaction initiated on each block through the node, and its process is broadcast throughout the whole network to let everyone know. On the one hand, this ensures the security of every transaction, but on the other hand, it is easy to cause privacy disclosure problems for transaction users. Therefore, under the premise of ensuring the security of the blockchain, it has become a hot issue to protect the sensitive information of transaction users. A check-in privacy protection (CPP) algorithm based on check-in location generalization is proposed in this paper, which can be applied to blockchain transactions to solve the privacy leakage problem of transaction users' sensitive information. CPP algorithm not only protects the privacy of check-in data but also keeps the high utility of trajectory pattern data. Firstly, location types are recommended in the sensitive check-in location generalization based on the user's trajectory pattern by using Markov chain technology. Secondly, to make sure that the generalized locations can be scattered as much as possible to prevent the attacker from deducing back, a heuristic rule is designed to select the generalized location based on the recommended location types, and at the same time, the similarity between the anonymous trajectory and the original trajectory is maintained. In addition, a generalized location search strategy is designed to improve the efficiency of the algorithm. Based on the real spatial-temporal check-in data, the results of the experiment indicate that our algorithm can effectively protect the privacy of sensitive check-in while ensuring the high utility of trajectory pattern data.

1. Introduction

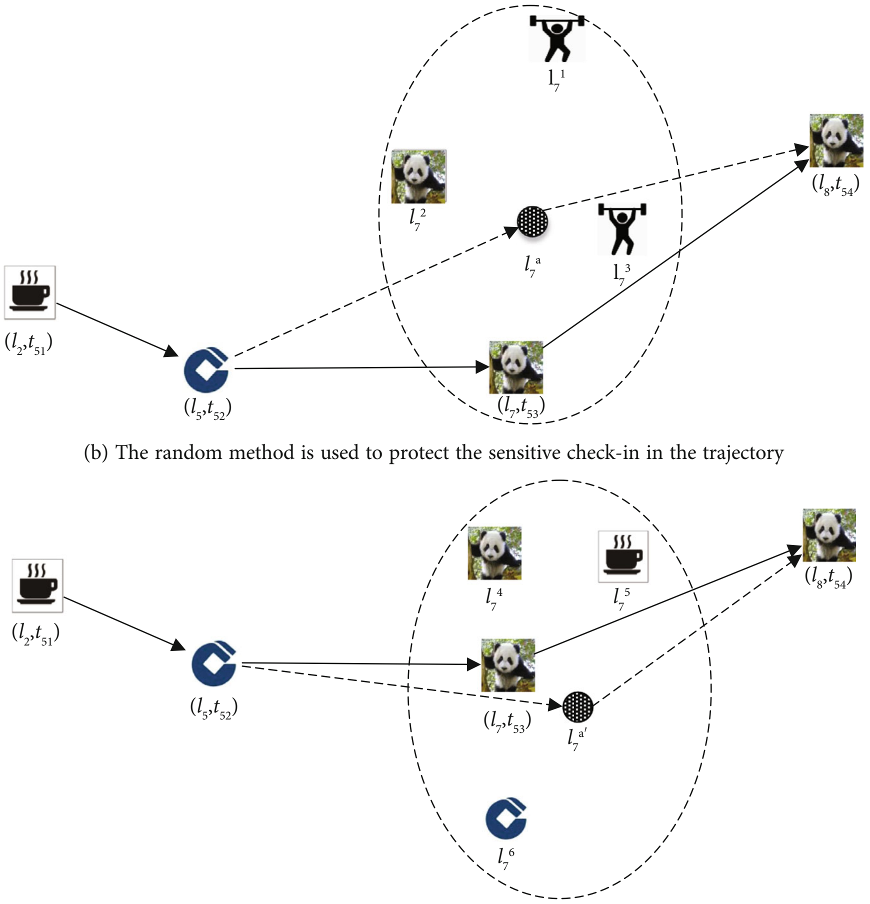
In recent years, the blockchain [1] has been broadly used in the financial industry, digital tokens, e-commerce logistics, and many other industries due to its characteristics of security and untampering. The significant security feature of the blockchain is that the blockchain authenticates each transaction initiated on each block through the node, and its process is broadcast throughout the network for everyone to know. This not only ensures the security of the transaction but also brings privacy harm to the transaction users. Hence, under the premise of ensuring the security of the blockchain [2, 3], it is already an issue worthy of attention to protect the sensitive information of transaction users. With the constant development of mobile networks [4, 5], vehicular networks [6–9], wireless communications network [10], and GPS-enabled devices, a mass of check-in data [11] of mobile users has been collected and utilized.

Check-in data contains the characteristics of human behavior, which plays a key role in major social science issues such as disease transmission, epidemic prevention and control, poverty eradication, urban planning, and other important life applications such as route recommendation and bus travel. Government and many research institutions hope to create more value through data mining. The trajectory contains many sensitive check-in data. Users' private information (home address, religious belief, interests, health, and other private information) will be obtained and used by malicious attackers, assuming these sensitive check-in data is leaked. Therefore, protecting sensitive check-in data in trajectory has become a challenging problem.

ID	Trajectory
tr_1	$<(l_6,t_{11}),(l_1,t_{12}),(l_4,t_{13}),(l_9,t_{14}),(l_3,t_{15})>$
tr_2	$<(l_4,t_{21}),(l_8,t_{22}),(l_2,t_{23})>$
tr_3	$<(l_9,t_{31}),(l_5,t_{32}),(l_4,t_{33}),(l_7,t_{34}),(l_1,t_{35})>$
tr_4	$<(l_2,t_{41}),(l_4,t_{42}),(l_6,t_{43})>$
tr_5	$<(l_2,t_{51}),(l_5,t_{52}),(l_7,t_{53}),(l_8,t_{54})>$
tr_6	$<(l_4,t_{61}),(l_2,t_{62}),(l_7,t_{63})>$

Type of location	Sign	Locations
Zoo (T_1)	🐼	l_7,l_8,l_9
Fitness room (T_2)	🏋	l_1,l_3
Coffee shop (T_3)	☕	l_2,l_6
Bank (T_4)	🏦	l_4,l_5
Anonymous central location	●	$l_7^a,l_7^{a'}$

(a) A set of user history trajectory



(b) The random method is used to protect the sensitive check-in in the trajectory

(c) This experiment is used to protect the sensitive check-in in the trajectory

Figure 1: User trajectory set and anonymous trajectory.

Check-in data means the user visits a certain place at a certain time. Sensitive check-in refers to user hopes to keep check-in data from being leaked. The user u's historical trajectory set $T_r = \{\mathrm{tr}_1, \mathrm{tr}_2, \mathrm{tr}_3, \mathrm{tr}_4, \mathrm{tr}_5, \mathrm{tr}_6\}$ is shown in Figure 1(a), and four check-in data in chronological order are included in the trajectory $\mathrm{tr}_5 = <(l_2, t_{51}), (l_5, t_{52}), (l_7, t_{53}), (l_8, t_{54})>$. The check-in data (l_7, t_{53}) indicates that user u visits location l_7 at time t_{53}. The location type of l_7 is the zoo, and t_{53} belongs to user u's office time. User u does not want to disclose the check-in data; therefore, (l_7, t_{53}) is set to sensitive check-in of user u. Currently, there is no privacy protection technology for sensitive check-in, and location privacy protection [12–15] is nearest to the problem.

Location generalization [16] is a popular location privacy protection method, and it has the characteristics of retaining the user's complete location information, law computation, and simple mechanism. However, these current location generalization methods do not consider the user trajectory pattern factor, which may reduce the privacy protection degree of sensitive check-in or even directly reveal the real sensitive check-in of user.

For example, the trajectory in Figure 1(b) is an anonymous trajectory obtained by using location generalization technology under 4-anonymity privacy requirement (4-anonymity means that the probability of identifying the sensitive check-in location based on this anonymous trajectory is no more than 1/4). Generalized locations ${l_7}^1$, ${l_7}^2$, and ${l_7}^3$ are obtained by a random method in literature [17], and these and the real sensitive check-in location l_7 form an anonymous location set to participate in the trajectory release; thus, the attacker cannot guess the real check-in location of user u at time t_{53}. This anonymous trajectory has two problems: (1) it does not conform to the user trajectory pattern. Location type of l_5 is bank. Based on user history trajectory, it can be seen that the next possible location type is the zoo, the coffee shop, and the bank from the current location type with the probability of visit being 4/7, 2/7, and 1/7, respectively. However, the location type of ${l_7}^1$ and ${l_7}^3$ is the fitness room. Obviously, the attacker can easily deduce that $({l_7}^1, t_{53})$ and $({l_7}^3, t_{53})$ are false check-in based on user history trajectory. (2) The similarity between the anonymous trajectory and the original trajectory decreases. ${l_7}^a$ is the anonymous central location generated after random generalization, and ${l_7}^{a'}$ is the anonymous central location generated after another anonymization. As shown in Figure 1(b), the shape of the anonymous trajectory $<(l_2, t_{51}), (l_5, t_{52}), ({l_7}^a, t_{53}), (l_8, t_{54})>$ differs greatly from that of the original trajectory $<(l_2, t_{51}), (l_5, t_{52}), (l_7, t_{53}), (l_8, t_{54})>$. In Figure 1(c), the shape of the anonymous trajectory $<(l_2, t_{51}), (l_5, t_{52}), ({l_7}^{a'}, t_{53}), (l_8, t_{54})>$ is closer to that of the original trajectory $<(l_2, t_{51}), (l_5, t_{52}), (l_7, t_{53}), (l_8, t_{54})>$. Due to the above two problems, the probability of identifying sensitive check-in will be greater than 1/4 and lead to privacy disclosure of sensitive check-in. To solve the problem, this paper proposes a check-in privacy protection algorithm based on check-in location generalization to protect the privacy of check-in data and keep the high utility of trajectory pattern data.

The main contributions are as follows:

(1) We propose a check-in privacy protection (CPP) algorithm based on check-in location generalization

(2) We recommend generalized location types by using Markov chain technology, and design a heuristic rule to select generalized locations

(3) We optimize the generalized location search strategy to improve the efficiency of the algorithm

(4) Extensive empirical studies show that our algorithm performs efficiently to protect check-in data while preserving the high utility of trajectory pattern data

The rest of the paper is organized as follows. Section 2 analyzes related work. Section 3 presents some important concepts and problem definition. Section 4 elaborates our scheme in detail. Section 5 evaluates the performance of CPP. We conclude this paper in Section 6.

2. Related Work

Sweeney [12] first proposed the concept of the k-anonymity model, and it was first applied in the relational database. Subsequently, Gruteser and Grunwald and Gruteser and Liu [18, 19] applied the k-anonymity model to location privacy protection. The core idea of it is that the anonymous server selects $k-1$ generalized locations to form an anonymous set with user real location, and the k locations cannot be distinguished from each other. Gedik and Ling [20] proposed the Clique Cloak algorithm, which constructed the anonymous region based on the graph model combined with time and space factors, and transformed the problem of anonymous set into the problem of finding $k-1$ neighbors in the graph model. Wang et al. [21] proposed a generalized location generation scheme based on semantic information and query probability, which can generate $k-1$ generalized locations related to user location semantic information. Niu et al. [22] proposed an enhanced DLS algorithm, which can select $2k$ generalized locations with high query probability similarity to the real location by calculating the location entropy and then select $k-1$ generalized locations from them by calculating the product of location distance. Lu et al. [23] proposed two generalized location generation algorithms CirDummy and GridDummy to realize location k-anonymity considering the shape of user privacy region.

Dwork [13] first proposed the differential privacy protection method, which protects privacy by adding noise to distort data. The differential privacy protection technology with mathematical theory and strict mathematical definition has two characteristics: first, it is not affected by attackers with background knowledge, and second, it is not affected by changing the specific data. Xiong et al. [24] proposed a spatial crowdsourcing algorithm based on a reward mechanism, which protects location privacy by adding Laplace noise to location data. Xu et al. [25] proposed a hybrid location privacy protection method, which divided locations into discrete locations and nondiscrete locations. For discrete locations, differential privacy technology was directly used for noise processing; while for nondiscrete locations, a k-means clustering algorithm based on differential privacy technology was used for generalization processing. However, excessive noise will lead to poor data availability and serious errors. Thus, Ping et al. [26] proposed PriLocation, a differential privacy protection method for noise reduction, to solve effectively this problem caused by excessive noise.

The basic idea of the location privacy protection method based on encryption technology is to encrypt the user's query information. Even if the attacker obtains the query information, he cannot know the real privacy information behind the query information. Zhang and Ni [14, 15] proposed a neighbor query method PRN-KNN, which uses a spatial encryption algorithm to enable users to quickly query k-neighbor candidate sets and introduces pseudo-random number secret rules to effectively reduce algorithm processing time. Papadopoulos et al. [27] used security hardware to assist PIR protocol and protected user location privacy through KNN query. Encryption-based location privacy protection technology can better ensure data availability and service accuracy, but the disadvantage is a large amount of calculation.

3. Preliminaries and Problem Definition

The check-in data set of user u is represented as $C_u = \{c_i \mid i \in [1, m]\}$. The check-in data $c_i = (l_i, t_i)$ indicates that user u visits location l_i at time t_i, where t_i is the check-in time, and l_i is the specific location on the map, such as Northeastern University, Wanda Plaza, and Beiling Park, and (x, y) is the latitude and longitude of a specific location, respectively. T_i represents the location type of a specific location, such as universities, shopping centers, and parks.

Definition 1 Sensitive check-in. Given trajectory $\mathrm{tr} = <(l_1, t_1), (l_s, t_s), \cdots, (l_n, t_n) >$, if the user does not want to check in, (l_s, t_s) was exposed, so (l_s, t_s) is called sensitive check-in. As shown in Figure 1(b), (l_7, t_{53}) is a sensitive check-in in trajectory tr_5.

Definition 2 Trajectory pattern matrix M. Given an $m * m$ matrix, $T_1, \cdots, T_m$ represents the location type, and $M(T_i, T_j)$ represents the probability that the user travels from location type T_i to location type T_j.

As shown in Table 1(a), the location type of the zoo, the fitness room, the coffee shop, and the bank are, respectively, denoted T_1, T_2, T_3, and T_4, respectively. The user trajectory pattern matrix M is obtained according to the transfer situation of location type in user's historical trajectory set T_r, where $M(T_1, T_2)$ represents the transfer probability that the user travels from location type T_1 to the next location type T_2. In Figure 1(a), location type T_1 includes l_7, l_8, and l_9. The next location of l_7 is l_1 (T_2) and l_8 (T_1). The next location of l_8 is l_2 (T_3). The next location of l_9 is l_3 (T_2) and l_5 (T_4). Therefore, the value of $M(T_1, T_2)$ is 2/5.

Definition 3 Check-in location generalization. Given a check-in data (l_s, t_s), generalization operation refers to convert location l_s of check-in (l_s, t_s) to a location set $L' = \{l_s, l_1', l_2', \cdots, l_i'\}$, there are $1 + i$ locations in L', and the probability that any location in the set appears between moment t_{s-1} and moment t_{s+1} is equal.

Definition 4 Anonymous trajectory. The trajectory obtained after replacing the sensitive check-in location l_s in the original trajectory with the anonymous center location l_s^a after anonymization.

As shown in Figure 1(b), the anonymous trajectory is represented as $<(l_2, t_{51}), (l_5, t_{52}), (l_7^a, t_{53}), (l_8, t_{54}) >$.

Definition 5 Trajectory pattern similarity. Given the original trajectory pattern matrix M and the anonymous trajectory pattern matrix M' (the order of the matrix is m), the trajectory pattern similarity is shown in Formula (1):

$$\mathrm{sim}\left(M, M'\right) = \frac{\sum M(i,j) * M'(i,j)}{\sqrt{\sum M(i,j)^2}\sqrt{\sum M'(i,j)^2}}\; i, j \in (1, m). \quad (1)$$

Table 1: Trajectory pattern matrix.

(a) User trajectory pattern matrix

Matrix M	T_1	T_2	T_3	T_4
T_1	0.2	0.4	0.2	0.2
T_2	0	0	0	1
T_3	0.25	0.25	0	0.5
T_4	4/7	0	2/7	1/7

(b) Anonymous trajectory pattern matrix

Matrix M'	T_1	T_2	T_3	T_4
T_1	3/19	8/19	4/19	4/19
T_2	0	0	0	1
T_3	0.25	0.25	0	0.5
T_4	16/29	0	8/29	5/29

As shown in Figure 1(b), the anonymous trajectory pattern matrix M' is obtained by anonymizing the original trajectory pattern matrix M, and the value of trajectory pattern similarity $\mathrm{sim}(M, M')$ is 99.93%.

Definition 6 Check-in k-anonymity. Given sensitive check-in (l_s, t_s), the generalized location set $c' = \{l_s^1, \cdots, l_s^m\}$ is get through the check-in location generalization operation, where size $(c') > = k$, so that the leakage rate of check-in location is not greater than $1/k$, namely, check-in k-anonymity.

Definition 7 Location exposure rate LE. The generalized location is expressed as l', the location anonymous set is composed of the real check-in location l and $k - 1$ generalized locations, namely, $\mathrm{LAS} = \{l, l^1, l^2, \cdots, l^{k-1}\}$. Given the user location anonymous set LAS, the attacker uses background knowledge to identify LAS and infers the probability of the user real check-in location as shown in Formula (2):

$$\mathrm{LE} = \frac{1}{|\mathrm{LAS}| - |\mathrm{LAS}'|}. \quad (2)$$

$|\mathrm{LAS}|$ represents the total number of locations in an anonymous set, and $|\mathrm{LAS}'|$ indicates that the attacker can identify the number of generalized locations.

Definition 8 Distance between trajectories. Given original trajectory, sensitive check-in location l_s, anonymous trajectory, and anonymous center location l_s^a, the distance between trajectories is defined as the Euclidean distance between two locations as seen in Formula (3):

$$\mathrm{tr\_dist}(l_s, l_s^a) = \sqrt{(l_s.x - l_s^a.x)^2 + (l_s.y - l_s^a.y)^2}. \quad (3)$$

```
Input: sensitive check-in location $l_s$, privacy protection threshold $k$;
Output: an anonymous set of locations containing $k$ locations.
1. $LAS \longleftarrow \varnothing$;
2. $T$= MC-LTR ($M, r(M), sub\_T, s$);
3. $L$= GLS ($D - index, R, tr$);
4. $S$ = LATP ($k, T, M, T_s$);
5. $Cand_l$= DLS ($S, k, l_s$);
6. LAS = $Cand_l \bigcup l_s$;
7. if($LE$<1/$k$)then
8.     Return LAS.
9. else
10.    Return $\varnothing$ .
```

Algorithm 1: Check-in privacy protection algorithm based on generalization of check-in location.

Problem definition. Given check-in data set C_u, sensitive check-in set S_u of user u, real trajectory tr, and privacy protection threshold k, the location anonymous set LAS is obtained by generalizing the sensitive check-ins in sensitive check-in set based on trajectory pattern. The generalized check-ins in LAS not only meet check-in k-anonymity but also ensure the maximum similarity of trajectory pattern.

4. Check-In Privacy Protection Algorithm Based on Generalization of Check-In Location

In this section, the check-in privacy protection algorithm based on check-in location generalization (Algorithm 1) is proposed. The main idea is to select the generalized location based on the original trajectory pattern matrix in the process of check-in location generalization so that the generalization operation can change the similarity between the original trajectory pattern matrix and the anonymous trajectory pattern matrix as little as possible. Thus, high data availability of anonymous trajectory in trajectory patterns is guaranteed. The algorithm framework of this paper is shown in Figure 2. The algorithm framework can show that users' sensitive check-ins are protected by the four algorithms (Algorithms 1–4) proposed in this paper, and this method can be used to protect user identity information in blockchain transactions.

First, the Markov chain-location type recommendation (MC-LTR) algorithm is used to recommend the set of location types for sensitive check-ins (line 2). Generalizing location search (GLS) algorithm is used to search the specific location in the generalization area (line 3). The location assignment based on trajectory pattern (LATP) algorithm is adopted to allocate the number of generalized locations corresponding to the recommended location type, and the aim is to ensure that the change of the anonymized trajectory pattern matrix is minimal (line 4). The dummy location selection (DLS) algorithm is used to obtain the candidate array of generalized locations (line 5). As shown in Formula (4), score is a heuristic function, whose value measures the influence of the distance product between the generalized locations and the sensitive check-in location and the distance between trajectories before and after anonymity. The higher the value, the more scattered between the generalized locations and the sensitive check-in location, and the closer the distance between the anonymous trajectory and real trajectory is. Finally, the CPP algorithm returns an anonymous location set containing k locations (line 6).

$$\text{Score} = \frac{\prod \text{dist}\left(l_i, l_j\right)}{\text{dist}\left(l_s, {l_s}^a\right)} \left(l_i \neq l_j\right). \tag{4}$$

For example, we protect sensitive check-in (l_7, t_{53}) in trajectory tr_5.The random choice of location type is likely to expose the user's sensitive check-in, so the MC-LTR algorithm is used to ensure that the generalized location type conforms to the user's historical trajectory pattern. The location type of sensitive check-in location l_7 is T_1, and the location type of next moment predicted based on Markov chain includes T_1, T_2, T_3, and T_4, and the recommendation probability is 4/35, 0, 1/14, and 4/49, respectively. Therefore, the recommended set of location types for sensitive check-in (l_7, t_{53}) is $T = \{T_1, T_4, T_3\}$. Searching the specific location corresponding to the recommended location type mainly considers two factors: historical average speed and time accessibility. In the query area, GLS algorithm will be used to put the searched specific locations corresponding to each location type in the set into location queue L, namely, $L = [{l_1}^1, {l_1}^2, {l_4}^1, {l_4}^2, {l_4}^3, {l_3}^2]$, wherein the location type T_1 contains two specific locations ${l_1}^1$ and ${l_1}^2$, and the location type T_4 contains three specific locations ${l_4}^1, {l_4}^2$ and ${l_4}^3$, and the location type T_3 contains one specific location ${l_3}^2$. Due to need to achieve the 4-anonymity protection, three generalized locations are selected from location queue L to ensure that the anonymous set can achieve optimal protection. By using the LATP algorithm, one location type meeting the requirement of anonymity is selected at a time, and the number array S of generalized location types is obtained. Among them, $S[T_1] = 2$, $S[T_4] = 1$, and $S[T_3] = 0$. The dispersion between locations and the change situation of the original trajectory's shape and the anonymous trajectory's shape are considered. The purpose is to prevent the location exposure and ensure trajectory similarity. Finally, the DLS algorithm is used to select 3 candidates from the location

```
Input: sensitive check-in location $l_s$, privacy protection threshold $k$;
Output: an anonymous set of locations containing $k$ locations.
$LAS \leftarrow \emptyset$;
$T$= MC-LTR ($M, r(M), sub\_T$,$s$);
$L$= GLS ($D - index, R, tr$);
$S$ = LATP ($k, T, M, T_s$);
$Cand_l$= DLS ($S, k, l_s$);
LAS = $Cand_l \cup l_s$;
if（$LE < 1/k$）then
    Return LAS.
else
    Return $\emptyset$ .
```

Figure 2: The algorithm framework of CPP.

```
Input: trajectory pattern matrix $M$, reverse trajectory pattern matrix $R(M)$, sensitive sub-trajectory type $sub\_T = \{T_{before}, T_s, T_{behind}\}$, recommended location type quantity threshold $s$;
Output: set $T$ of location types.
Initialize $T[i]$, $i \in [1,|M|]$;
if ($sub\_T[0] == \varnothing$) then
    $T[i]=R(M)[sub\_T[2]][i], i \in [1,|M|]$;
else if ($sub\_T[2] == \varnothing$) then
    $T[i]=M[sub\_T[0]][i], i \in [1,|M|]$;
  else then
    $T[i]= M[sub\_T[0]][i] \times R(M)[i][sub\_T[2]]; i \in [1,|M|]$;
sort($T[i]$);
Select the first $s$ location types with higher probability values and put them into $T$;
Return $T$.
```

Algorithm 2: The Markov-chain location type recommendation algorithm.

```
Input: distance index $D$-$index$, query distance $R$, any location $l_i$;
Output: location queue $L$.
$L=\varnothing$;
pos =1, end =lens ($D$-$index$ [$l_i$]);
While (pos<end) do
    mid=(post+end)/2;
    if $D$-$index$[mid]<R then
        pos=mid+1;
    else if $D$-$index$[mid]>$R$ then
        end=mid-1;
    else
        pos=end=mid;
$L$=$D$-$index$ [$l_i$], $i \in (1$, pos);
Return $L$.
```

Algorithm 3: The generalized location search algorithm.

queue L and put them into the location anonymous set LAS, namely, LAS = $\{l_1^{\ 1}, l_1^{\ 2}, l_4^{\ 2}\}$.

4.1. Recommendations for Generalizing Location Types. This section mainly introduces recommendations of the location type for sensitive check-in based on the MC-LTR algorithm (Algorithm 2). For example, by using this algorithm, the location type recommendation is made during generalized sensitive check-in (l_7, t_{53}). Check-in data is an integral part of user trajectory, and user trajectory patterns can reflect user behavior characteristics, so the selection of location type should be in accordance with the user trajectory movement pattern. Because the sensitive check-in location l_7 is located in the middle of the trajectory, two predictions are needed to realize the location type recommendation. According to the previous moment location type T_4 of the sensitive

```
Input: privacy protection threshold $k$, generalized location type set $T$, trajectory pattern matrix $M$, sensitive location type $T_s$, location queue $L$;
Output: generalized location type quantity array $S$.
Initialize $S\ [T_i]$=0, $i \in [1,|T|]$;
$S\ [T_s]$=min ($k$-1, $L\ (T_s)$);
While $\sum S[T_i]<k$-1 do
    for each location type $T_i \in T\ (T_i \neq T_s)$do
      if $S[T_i]<=L\ (T_i)$ then
         $S[T_i]$++;
         update $M$ based on $S[T_i]$ to change $M$ to $M^{T_i}$;
         Calculate $sim(M, M^{T_i})$;
         $S[T_i]$–;
  $\mathrm{T}_{\max}= T_i$ which maximizes $sim(M, M^{T_i})$;
  $M$=$\mathrm{M}^{\mathrm{T}_{\max}}$;
Return $S$.
```

Algorithm 4: The location assignment based on trajectory pattern algorithm.

check-in, it is predicted that the generalized location types of the sensitive check-in are T_1, T_2, T_3, and T_4, and the probabilities are 4/7, 0, 2/7, and 1/7, respectively. The transfer probability of each generalized location type to the next moment location type for sensitive location is 1/5, 0, 1/4, and 4/7, respectively. Therefore, the recommended set of generalized location types for sensitive check-in is represented as $T = \{T_1, T_4, T_3\}$.

As shown in Table 2, the user reverse trajectory pattern matrix $R(M)$ is obtained in reverse time based on the user trajectory set T_r. In Algorithm 2, M and $R(M)$ are taken as inputs to recommend set T of location types that meets the trajectory pattern.

Algorithm 2 shows the recommendation process of location types when generalizing sensitive check-in. The algorithm takes into account three kinds of location situations of sensitive check-in in the trajectory. When the sensitive check-in location is located at the beginning of the trajectory, the reverse trajectory pattern matrix is used for the recommended location type of the sensitive check-in (lines 2 and 3). When the sensitive check-in location is located at the end of the trajectory, the trajectory pattern matrix is used for the recommended location type of the sensitive check-in (lines 4 and 5). When the sensitive check-in location is located at the nonhead-tail location of the trajectory, the combination of two trajectory pattern matrices is used to recommend location type for the sensitive check-in (lines 6 and 7). Finally, the first s location types with high recommendation probability values are selected from the recommended location types and put into the set T of location types.

4.2. Search for Specific Generalization Location. This section mainly introduces the use of a generalized location search algorithm (Algorithm 3) to generate location queue L. According to the recommended location type, the specific location of the corresponding location type should be searched in the query area. At the same time, the specific location selected should meet the time accessibility. For example, the query areas of sensitive check-in (l_7, t_{53}) are circular areas with check-in location l_5 at the previous time and check-in location l_8 at the later time as the center and $\bar{v}(t_{53} - t_{52})$ and $\bar{v}(t_{54} - t_{53})$ as the query radius, respectively, where $\bar{v}$ is the average speed calculated from the user's historical trajectory. First, the specific locations corresponding to each location type in the query area are put into location

Table 2: Reverse trajectory pattern matrix.

Matrix $R(M)$	T_1	T_2	T_3	T_4
T_1	1/6	0	1/6	2/3
T_2	2/3	0	1/3	0
T_3	1/3	0	0	2/3
T_4	0.2	0.2	0.4	0.2

Table 3: Distance index for each sensitive location.

l_1	${l_5}^6$	${l_3}^1$	${l_2}^2$			
	0.10	0.20	0.40			
l_2	${l_1}^6$	${l_5}^1$				
	0.14	0.18				
l_4	${l_1}^8$	${l_9}^4$	${l_4}^7$			
	0.20	0.22	0.65			
l_5	${l_7}^2$	${l_4}^6$	${l_1}^1$			
	0.24	0.36	0.45			
l_7	${l_1}^1$	${l_1}^2$	${l_3}^2$	${l_4}^1$	${l_4}^2$	${l_4}^3$
	0.18	0.19	0.58	0.68	0.69	0.72
l_8	${l_4}^1$	${l_4}^1$	${l_1}^1$	${l_4}^3$		
	0.11	0.32	0.50	0.75		

Table 4: Generalized location quantity allocation.

Location type	T_1	T_2	T_3	T_4
Number of locations	2	0	1	3
Allocated quantity	2	0	0	1

```
Input: generalized location type quantity array $S$, real sensitive location $l_s$, privacy protection threshold $k$, location queue $L$;
Output: generalized location candidate array $Cand_l$.
1. Initialize $Cand_l[T_i]$=0, $i \in [1,|S|]$;
2. while lens ($Cand_l$)<$k$-1 do
3.   for each location type $T_i \in S$ && $Cand_l[T_i]$<= $S$ [$T_i$] do
4.     if lens ($Cand_l$)==0 then
5.       Select the location furthest from the real sensitive location from the generalized locations in $T_i$ and add it to $Cand_l$;
6.     else
7.       Select the location with the maximum Score in $T_i$ and add it to $Cand_l$;
8. Return $Cand_l$;
```

Algorithm 5: The dummy location selection algorithm.

queue L, namely, $L = [l_1{}^1, l_1{}^2, l_4{}^1, l_4{}^2, l_4{}^3, l_3{}^2]$. There are two locations belonging to location type T_1, there are three locations belonging to location type T_4, there is one location belonging to location type T_3, and then, $l.D(R)$ is defined to represent a group of locations within distance R of sensitive location l. As shown in Table 3, search the location of distance sensitive location l_7 within the 0.7 km, that is, $l_7.D$ $(0.7) = \{l_1{}^1, l_1{}^2, l_4{}^1, l_4{}^2, l_3{}^2\}$, and the unit of distance is kilometer (km). The generalized location search algorithm proposed in this paper implements the breadth-first search on the query area to realize the location search. $l.D$ is used to store the searched candidate locations and the corresponding distance index (D-index), and then, the binary search algorithm is used to select the qualified locations, in order to save the running time of the algorithm.

Definition 9 Distance index (D-index). Given a sensitive location l, the distance index (D-index) between this location and other locations is defined as a list $l.D$. The elements stored in the list are candidate locations and the distance data between each candidate location and the sensitive location l, and the distance data in $l.D$ are arranged in order from small to large.

4.3. Generalized Location Quantity Allocation. This section mainly introduces the generalized location quantity allocation algorithm based on the trajectory pattern (Algorithm 4). The purpose is to determine the number of specific locations allocated for the recommended location type and to ensure the maximum similarity of the trajectory pattern matrix. As shown in Table 4, five generalized locations $(l_1{}^1, l_1{}^2, l_4{}^1, l_4{}^2, l_3{}^2)$ are found for sensitive check-in (l_7, t_{53}) through the generalized location search algorithm. Because the privacy protection threshold k is 4, so three of the five generalized locations are selected to ensure the maximum similarity of the trajectory pattern matrix. In the generalized location allocation algorithm based on trajectory pattern, the same generalized location type as the sensitive check-in location is first assigned (line 2), so two generalized locations of location type T_1 are assigned. The selection of the remaining generalization location is determined by adding a generalization location of different location type at a time and calculating the similarity value of the corresponding trajectory pattern matrix (lines 4-9). When the generalized location type T_3 is added, the similarity of trajectory pattern is 99.81%. When the generalized location T_4 is added, the similarity of trajectory pattern is 99.93%. So, a generalization location of type T_4 is assigned and returns a generalization location type quantity array S.

4.4. Selection of Candidate Generalized Location. This section mainly introduces the selection of candidate generalized locations by candidate location selection algorithm (Algorithm 5). The generalized location candidate array needs to meet two conditions: (1) the locations in the array are as scattered as possible, which can effectively prevent the anti-deduction of the attacker. (2) The center location of the area formed by each location is as close as possible to the sensitive check-in location, which ensures that the anonymous trajectory is similar to the original trajectory. Among them, the traditional method to ensure the dispersion between locations is to calculate the sum of the distance between locations $\sum_{i \neq j} \mathrm{dist}(l_i, l_j)$. However, the product of the distance method $\prod_{i \neq j} \mathrm{dist}(l_i, l_j)$ can better reflect the dispersion of locations in most cases. As shown in Figure 3, A and B are selected generalization locations, and C and D are to be selected generalization locations. When selecting the third generalization location, both C and D can meet the requirements if the sum of distance method is used, because $\mathrm{tr\_dist}(D, A) + \mathrm{tr\_dist}(D, B) = \mathrm{tr\_dist}(C, A) + \mathrm{tr\_dist}(C, B)$. However, the product of distance method is used, and we should choose the generalization location C, because $\mathrm{tr\_dist}(C, A) * \mathrm{tr\_dist}(C, B) > \mathrm{tr\_dist}(D, A) * \mathrm{tr\_dist}(D, B)$, and the anonymous region formed by the generalization location A , B, and C is more scattered, so the product of distance method is adopted in this algorithm.

4.5. Algorithm Complexity Analysis. In the MC-LTR algorithm, generalized location types are recommended through the trajectory pattern matrix. Suppose $|M|$ is the order of the user trajectory pattern matrix, so, the time complexity of the MC-LTR algorithm is $O(|M| + |M|\log_2{}^{|M|})$. In the GLS algorithm, we use the binary search algorithm to find specific generalized locations that match the location type, and the algorithm complexity of the location queue at any sensitive location is $O(\log_2{}^{|D\text{-index}[l_i]|})$. In the LATP algorithm, we need to assign generalized locations for the recommended location type, because the privacy protection threshold is k,

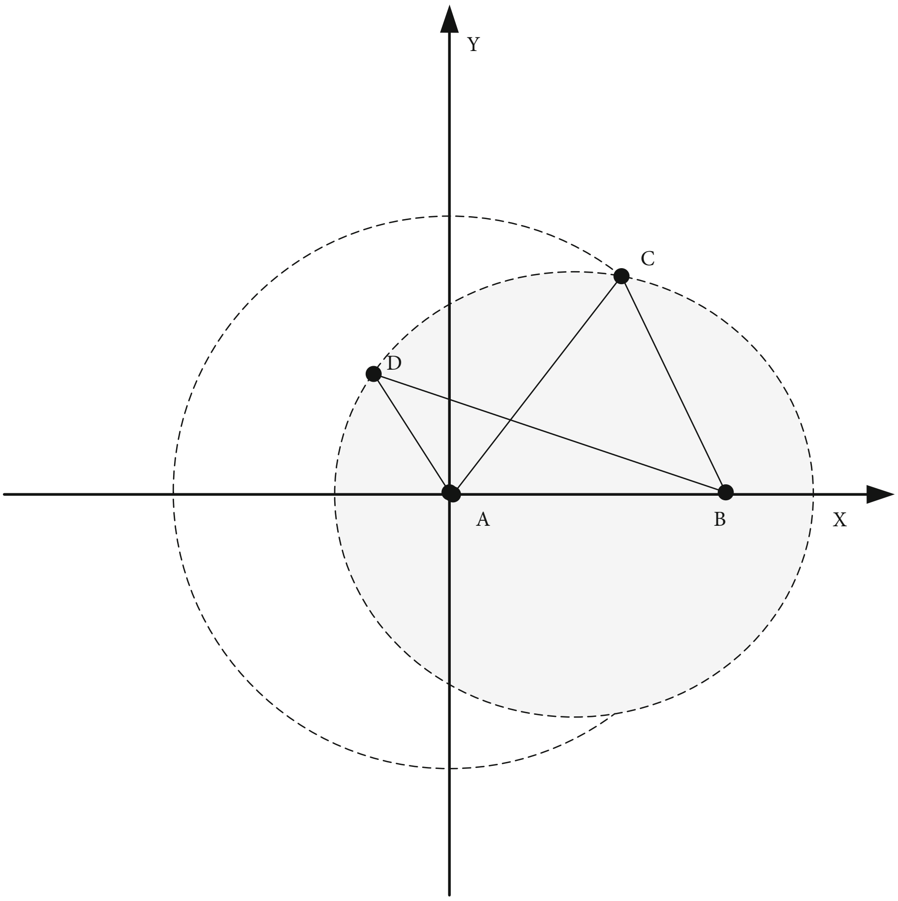


Figure 3: Distance product scene graph.

Table 5: Experimental data statistics.

Experimental data set	Number of users	Number of locations	Number of check-ins	Number of trajectories	Number of location types
Brightkite	5000	274761	3185493	168	766316
Gowalla	5000	193989	1501739	121	436665

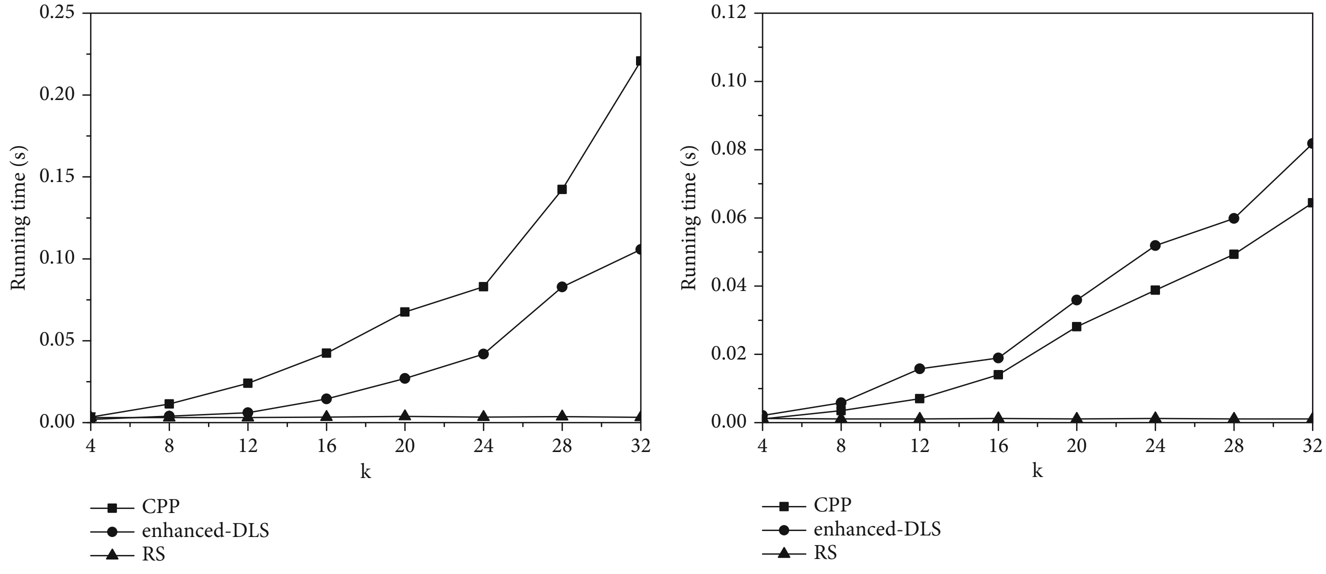
so, the generalization location needs to be allocated through $k-1$ cycles, and each cycle needs to update the matrix and calculate the matrix similarity. The algorithm's time complexity is $O((k-1)[(|s|-1)(3+|M|*|M|)]+2)=O(k*s|M|^2)$. In the DLS algorithm, we need to choose $k-1$ candidate generalization locations from the generalization region, and it is necessary to judge $|S[T_i]|$ locations every time, so the algorithm complexity is $O(k*|S|)$. Therefore, the time complexity of the CPP algorithm is $O(k*s|M|^2)$.

5. Experimental Evaluation and Analysis

This section analyzes and evaluates the performance of the proposed check-in privacy protection algorithm based on generalized check-in location. The data used in the experiment comes from two real data sets Brightkite and Gowalla disclosed by the complex network analysis platform of Stanford University. The map data of California where these two data sets are located are also obtained. Firstly, this paper deletes and filters users whose cumulative check-in days are less than 50 days in the data set and then deletes the trajectory that contains only one check-in data in a single trajectory. Finally, this article selects 5000 users and their corresponding data from the two data sets. Table 5 shows the relevant information of the experimental data.

This paper proposes a check-in privacy protection algorithm based on the generalization of check-in location (recorded as CPP), compared with the dummy location selection algorithm based on multiobjective optimization (recorded as enhanced DLS) [12] and the location privacy protection algorithm based on random selection method (recorded as RS) [7]. The performance of the algorithm is analyzed by comparison, and the influence of the parameters involved in the algorithm on the algorithm is evaluated. In the test, the value range of privacy protection anonymous parameter k is from 2 to 32.

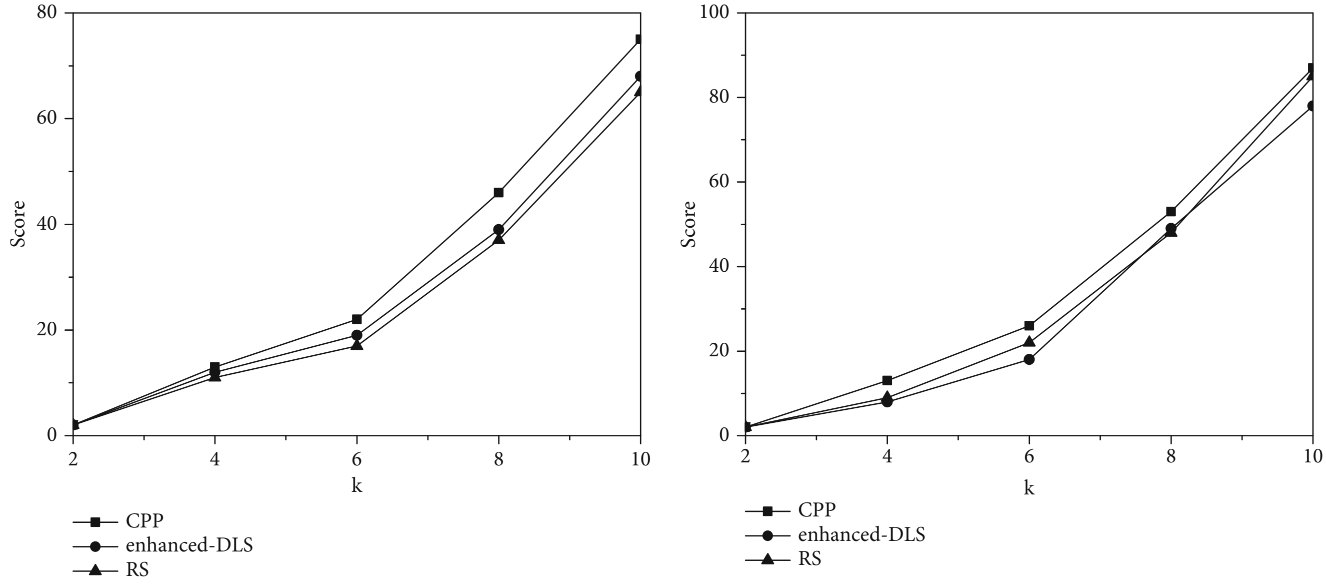
The software and hardware environment of this experiment are as follows: (1) hardware environment: Intel Xeon 3.90 GHz CPU and 256 GB; (2) operating system platform:



(a) Gowalla

(b) Brightkite

Figure 4: Change situation of running time.



(a) Gowalla

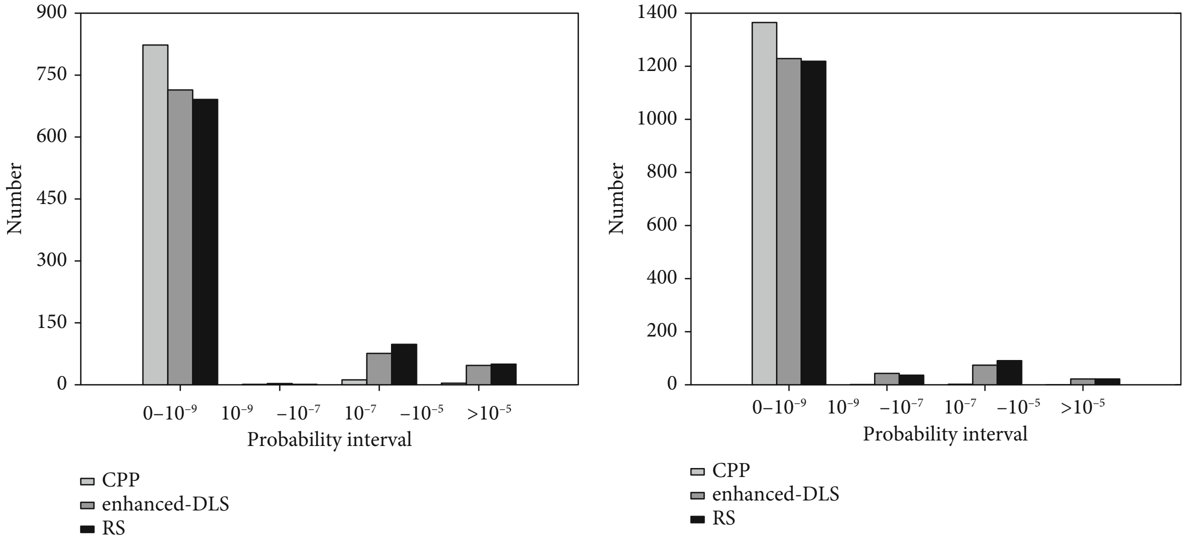
(b) Brightkite

Figure 5: Change situation of score value.

Microsoft Windows 10; and (3) programming environment: Python language, Pycharm.

This section compares the performance of the CPP algorithm, enhanced DLS algorithm, and RS algorithm by analyzing the running times of the algorithm, the change of score value, and data availability. The following can be seen from Figures 4 and 5: (1) The running time of the three algorithms increases with the increase of the privacy protection threshold k. The running time of the CPP algorithm is between the RS algorithm and the enhanced DLS algorithm. The running time of the enhanced DLS algorithm changes significantly with the increase of k, and the running time of the RS algorithm is the smallest and tends to be stable. The enhanced DLS algorithm should consider the influence of the query probability and the entropy when selecting generalized locations, and the running time will be increased with the number of generalized locations, The CPP algorithm saves the running time by proposing a reasonable and effective location search algorithm. (2) The *score* value measures the degree of dispersion between locations and the distance between the anonymous trajectory and the original trajectory.

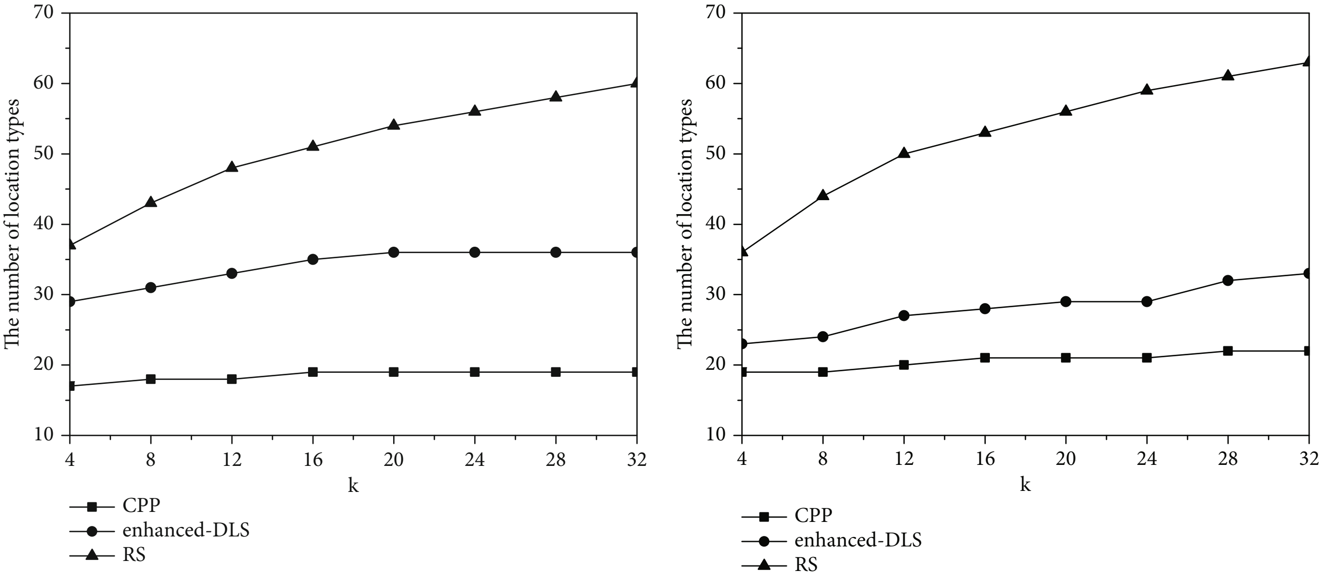
When the score value is larger, it means that the selected generalized location and sensitive check-in location are



(a) Gowalla

(b) Brightkite

Figure 6: Distribution situation of probability change.



(a) Gowalla

(b) Brightkite

Figure 7: Change situation of location types.

more dispersed, and the distance between the generalized trajectory and the real trajectory is closer. With the increase of the privacy protection threshold k, the score of the three algorithms increases significantly. The CPP algorithm has the better performance in score value because the CPP algorithm uses the heuristic rules to select each generalization location, and it ensures that each selected generalized location can keep the maximum score value. However, the random selection of each generalized location will lead to uncertainty, and the size of the *score* value to be unstable. When the order of magnitude of score value is too large, this paper uses the logarithm of *score* value to express it.

The availability of measurement data can be evaluated from three aspects: the change of user trajectory pattern, the change of access location type, and the change of access location points. The following can be seen from Figures 6–8:

(1) According to the position type transition probability difference before and after anonymity of the trajectory pattern matrix, it can be divided into four intervals: $0 \sim 10^{-9}, 10^{-9} \sim 10^{-7}, 10^{-7} \sim 10^{-5}, >10^{-5}$, counting the quantity distribution of each interval. The probability difference greater than 98% in the CPP algorithm falls in the 0~10^{-9} interval, and it shows that the change of location type transition probability is small, the similarity of the trajectory pattern matrix before and after anonymity is high, while the performance of enhanced DLS algorithm

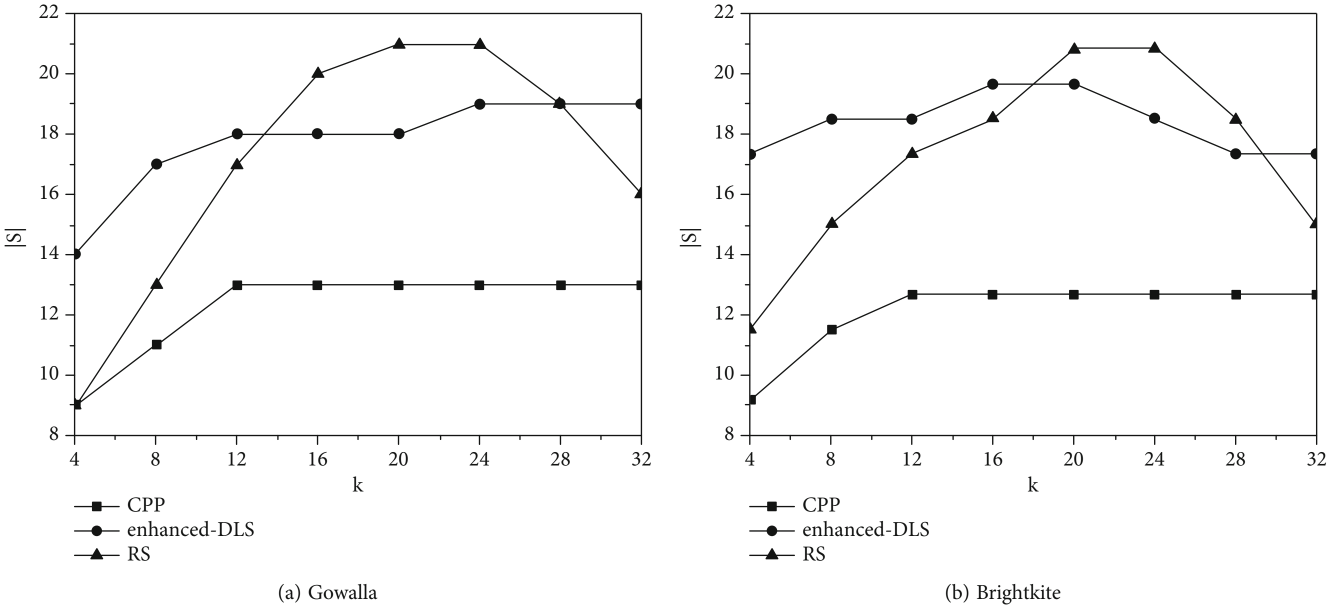


Figure 8: Change situation of specific location.

and RS algorithm is lower than that of CPP algorithm. The similarity of the trajectory pattern matrix before and after anonymity is high, and the performance of the enhanced DLS algorithm and RS algorithm is lower than the CPP algorithm

(2) *The Changes in the Type of User Access Location*. The total number of original location types visited by a user is 38. It can be seen from the figure that the number of access location types of the CPP algorithm changes little with the increasing of the *k*, and the algorithm will not generate new location types. The access location type of the enhanced DLS algorithm increases gradually as the increasing value of the *k*, while the number of access location types of the RS algorithm changes significantly with the increasing of the *k*. Because the CPP algorithm recommends the location type for the user according to the user's trajectory pattern, the generalized location selected by the random method is uncertain, and the location type of the generalized location will exceed the range of the user's original access location type.

(3) For the user's access location change index, set the number of user visits to each access location before generalization as n_i, and after generalization, the number of user visits to each location becomes $n_i{}'$; the change in the number of visits for each location is defined as $\Delta n_i = |n_i - n_i{}' \,|$. Set a standard number threshold N and a location set S, and put the locations with the change of the number of visits greater than or equal to the standard number threshold into the set S, symbolized as $S = \{l_i | \Delta n_i \geq N\}$. The value of $|S|$ represents the number of position points in the set S. The smaller the value of $|S|$, the more stable the number of visits of the user to each access location after anonymity, and the better anonymity. It can be seen from Figure 8 that the CPP algorithm proposed in this paper has the smallest $|S|$ value, which is much lower than the other two algorithms, so the anonymous protection effect is the best. The RS algorithm is easy to generate new locations when selecting generalized locations, and the number of new locations is uncertain, so the $|S|$ value is larger

6. Conclusion

In this paper, a check-in privacy protection algorithm based on check-in location generalization for sensitive check-in protection is proposed for the first time and can be applied to blockchain transactions to solve the privacy protection problem of transaction users' sensitive information. Considering the user's trajectory pattern factor, the algorithm recommends the location type of the generalized check-in location for the user and selects generalized locations that can ensure the minimum change of trajectory pattern. Experimental research based on real check-in data sets shows that the CPP algorithm can effectively protect the sensitive check-ins in the trajectory, greatly reduce the probability of the attacker identifying the real sensitive check-ins, and maintain the high availability of the trajectory pattern data. This method is suitable for protecting the location in the area with dense geographical density. However, the *k*-anonymity method may not be implemented in areas with sparse geographical density. The solution to the above problem needs to be further studied.

Data Availability

All data included in this study are available upon request by contact with the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is partly supported by the National Natural Science Foundation of China (No. 61802268).

References

[1] L. Liu, J. Feng, Q. Pei et al., "Blockchain-enabled secure data sharing scheme in mobile-edge computing: an asynchronous advantage actor-critic learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2342–2353, 2021.

[2] L. Tan, Y. Keping, N. Shi, C. Yang, W. Wei, and L. Huimin, "Towards secure and privacy-preserving data sharing for COVID-19 medical records: a blockchain-empowered approach," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 1, pp. 271–281, 2022.

[3] Y. Sun, J. Liu, K. Yu, M. Alazab, and K. Lin, "PMRSS:privacy-preserving medical record searching scheme for intelligent diagnosis in IoT healthcare," *IEEE Transaction on Industrial Informatics*, vol. 18, no. 3, pp. 1981–1990, 2022.

[4] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: a survey," *Mobile Networks and Applications*, vol. 26, no. 3, pp. 1145–1168, 2021.

[5] J. Feng, L. Liu, Q. Pei, and K. Li, "Min-Max cost optimization for efficient hierarchical federated learning in wireless edge networks," *IEEE Transactions on Parallel and Distributed Systems*, p. 1, 2022.

[6] L. Zhao, Z. Li, A. Al-Dubai et al., "A novel prediction-based temporal graph routing algorithm for software defined vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–16, 2021.

[7] Z. Liang, T. Zheng, M. Lin, A. Hawbani, J. Shang, and C. Fan, "SPIDER: a social computing inspired predictive routing scheme for softwarized vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2021.

[8] L. Zhao, H. Li, N. Lin, M. Lin, C. Fan, and J. Shi, Eds., "Intelligent content caching strategy in autonomous driving toward 6G," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2021.

[9] L. Liu, M. Zhao, M. Yu, M. A. Jan, D. Lan, and A. Taherkordi, "Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2022.

[10] M. Wang, Y. Lin, Q. Tian, and G. Si, "Transfer learning promotes 6G wireless communications: recent advances and future challenges," *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 790–807, 2021.

[11] L. Wang and X. F. Meng, "Location privacy preservation in big data era: a survey," *Journal of Software*, vol. 25, no. 4, pp. 693–712, 2014.

[12] L. Sweeney, "K-anonymity: a model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[13] C. Dwork, "Differential privacy," in *Proc of the 33rd International Colloquium on Automata, Languages and Programming*, pp. 1–12, Springer-Verlag, Berlin, 2006.

[14] Z. Feng, *Research on Location Privacy-Preserving Nearest Neighbor Query Based PIR*, Southeast University, 2017.

[15] Z. Feng and N. Wei-Wei, "Pseudo-random number encryption based on location privacy preserving nearest neighbor querying," *Journal of East China Normal University*, vol. 2015, no. 5, pp. 128–142, 2015.

[16] F. Deldar and M. Abadi, "PDP-SAG: personalized privacy protection in moving objects databases by combining differential privacy and sensitive attribute generalization," *IEEE Access*, vol. 7, pp. 85887–85902, 2019.

[17] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proceedings of the 2005 International Conference on Pervasive Services*, pp. 88–97, Piscataway, 2015.

[18] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the First International Conference on Mobile Systems, Applications, and Services*, pp. 31–42, San Francisco, CA, USA, 2003.

[19] M. Gruteser and X. Liu, "Protecting privacy in continuous location trajectorying applications," *IEEE Security and Privacy*, vol. 2, no. 2, pp. 28–34, 2004.

[20] B. Gedik and L. Ling, "Protecting location privacy with personalized k-anonymity: architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.

[21] W. Jie, W. Chunru, and M. A. Jianfeng, "Dummy location selection algorithm based on location semantics and query probability," *Journal on Communications*, vol. 41, no. 3, pp. 53–61, 2020.

[22] B. Niu, Q. Li, and X. Zhu, "Achieving k-anonymity in privacy-aware location-based services," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, Toronto, ON, Canada, 2014.

[23] H. Lu, C. S. Jensen, and M. L. Yiu, "PAD: privacy-area aware, dummy-based location privacy in mobile services," in *Acm International Workshop on Data Engineering for Wireless & Mobile Access*, pp. 16–23, Vancouver, Canada, 2008.

[24] P. Xiong, L. Zhang, and T. Zhu, "Reward-based spatial crowdsourcing with differential privacy preservation," *Enterprise Information Systems*, vol. 11, no. 10, pp. 1500–1517, 2017.

[25] X. Qiyuan, C. Zhenping, and F. Baochuan, "Hybrid location privacy protection based on differential privacy," *Computer Applications and Software*, vol. 36, no. 6, pp. 296–301, 2019.

[26] X. Ping, T. Zhu, and P. Lei, "Privacy Preserving in Location Data Release: A Differential Privacy Approach," in *Pacific Rim International Conference on Artificial Intelligence*, pp. 183–195, Springer, Cham, 2014.

[27] S. Papadopoulos, S. Bakiras, and D. Papadias, "Nearest neighbor search with strong location privacy," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 619–629, 2010.