Soft Comput (2016) 20:1735–1747
DOI 10.1007/s00500-015-1703-8

FOCUS

Long-term location privacy protection for location-based services in mobile cloud computing

Feilong Tang[1] · Jie Li[1] · Ilsun You[2] · Minyi Guo[1]

Published online: 3 June 2015

Abstract The popularity of mobile devices, especially intelligent mobile phones, significantly prompt various location-based services (LBSs) in cloud systems. These services not only greatly facilitate people's daily lives, but also cause serious threats that users' location information may be misused or leaked by service providers. The dummy-based privacy protection techniques have significant advantages over others because they neither rely on trusted servers nor need adequate number of trustworthy peers. Existing dummy-based location privacy protection schemes, however, cannot yet provide long-term privacy protection. In this paper, we propose *four principles* for the dummy-based long-term location privacy protection (LT-LPP). Based on the principles, we propose a set of long-term consistent dummy generation algorithms for the LT-LPP. Our approach is built on soft computing techniques and can balance the preferred privacy protection and computing cost. Comprehensive experimental results demonstrate that our approach is effective to both long-term privacy protection and fake path generation for LBSs in mobile clouds.

Communicated by A. Jara, M. R. Ogiela, I. You and F.-Y. Leu.

✉ Feilong Tang
tang-fl@cs.sjtu.edu.cn

Jie Li
jieyi.lee@gmail.com

Ilsun You
ilsunu@gmail.com

Minyi Guo
guo-my@cs.sjtu.edu.cn

1 Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China

2 School of Information Science, Korean Bible University, Seoul, South Korea

Keywords Soft computing · Privacy protection · Dummy generation · Location-based services · Mobile cloud computing

1 Introduction

Mobile cloud computing enables mobile users to enjoy various information services, in which the location information of the users is a crucial context for providing useful and preferred services (Bilogrevic et al. 2014; Liang et al. 2012). With the rapid growth of mobile devices in recent years, for example, location-based services (LBSs) have been greatly developed, and people increasingly rely on LBSs in their daily lives (Lien et al. 2013). Location-based information retrieval is essential in various LBSs (Paulet et al. 2012). More specifically, users send location-based queries to cloud computing providers and acquire the corresponding information (Puttaswamy et al. 2014) such as price and evaluation of restaurants within 1 km from their current location.

Location privacy in mobile clouds is a critical concern to both researchers and practitioners. Although greatly facilitating people's lives, rapidly growing LBSs also cause potential threats against their users (Hernández et al. 2014; Zhang et al. 2014).

In LBSs, users have to send queries that contain their current locations and identifiers to cloud providers to obtain the location-specific information (Mahmoud and Shen 2012). User privacy (e.g., user name) either is explicitly given in these identifiers or can implicitly be derived from other data like IP address in queries. Furthermore, a series of queries can be linked together through identifiers to produce more information such as user mobility patterns (Dewri 2013). So, the leakage of personal private information is unacceptable

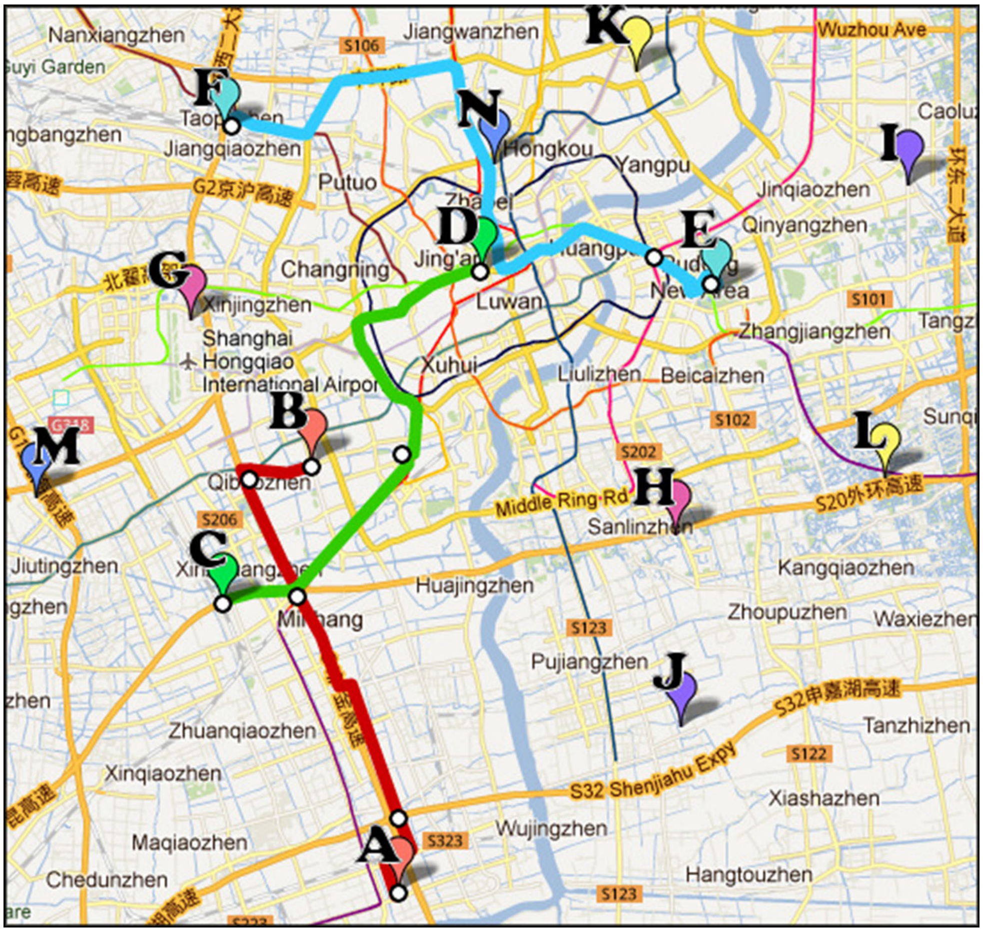


Fig. 1 Inconsistency of trip faking

in a public database (Cho et al. 2013; Tomoya et al. 2014). Although databases of cloud providers in general are not public, the sensitive location information could still be misused or leaked by incapable or malicious providers. Once adversaries get a user's location information, they can invade his privacy more easily (Pan et al. 2012).

Widespread concerns lead to extensive research aiming at protecting users' location privacy in LBSs. In Shokri et al. (2010), the authors divided the location privacy into two levels: *microscopic-level* and *macroscopic-level*. The former, which also is called as *snapshot level* location privacy, represents a user's privacy within a single query. The latter refers to the user's privacy within a whole journey with multiple queries. To capture the location privacy in a finer granularity, in this paper, we further subdivide the macroscopic-level location privacy as *journey-level* (e.g., a trip from Shanghai to Beijing) and *long-term* (e.g., activities in 1 month) location privacy.

A collection of techniques are developed for achieving different levels of privacy protection. Existing schemes on location privacy protection fall into three categories: *anonymity-based* (Gedik and Liu 2008; Samarati and Sweeney 1998), *obfuscation-based* (Ardagna and Cremonini 2011; Ardagna et al. 2007), and *dummy generation-based* (Lu et al. 2008) location privacy protection. Both the anonymity-based and the obfuscation-based schemes can protect only microscopic-level location privacy, while the dummy generation based schemes can provide macroscopic-level location privacy protection.

However, there are two serious limitations in existing dummy generation-based privacy protection schemes. The first one is that these schemes need the source and the destination of a trip before the fake path generation. In practice, however, users possibly do not know their sources and destinations, for example, they just walk around sometimes. Second, what is worse, these schemes are incompetent to protect location privacy in a long period. We illustrate this problem in the following scenario.

In Fig. 1, A–N are 14 locations, and a user lives at A and works at B. Let an existing fake trip generation scheme generates three paths: A → B, C → D, and E → F, where A → B is true, while C → D and E → F are synthetic and fake. As a result, an adversary cannot differentiate the true trip from other two ones, which means this scheme can protect journey-level (A → B) location privacy well. From then on, the user conducts two more trips between A and B in next days. Assume the scheme, respectively, generates two path groups: A → B, G → H, I → J and A → B, K → L, M → N. Now, the adversary can find the true trip is between A and B from the three groups of paths, which exemplifies that these existing schemes cannot provide long-term location privacy protection if the adversary traces a user in a long enough period. The authors in You et al. (2007) have pointed out this problem, but they achieve the consistency only through impractical assumptions.

Motivated by the above scenario, this paper focuses on long-term location privacy protection. We first propose four general principles for the dummy generation. Then, we propose our long-term consistent dummy generation approach based on these principles. It is well known that the more precise a solution is, the more computation and communication cost the solution needs. Unlike conventional computing, soft computing is tolerant of imprecision, uncertainty, partial truth and approximation to achieve tractability, robustness, and low solution cost (Ahmad and Ansari 2012; Metre et al. 2012; Wahab et al. 2009; Yu and Kaynak 2009). Therefore, based on soft computing techniques (Kim and Bien 2008; Mitra et al. 2002; Murakami and Honda 2008; Yager et al. 2014), we develop our dummy generation algorithms to provide preferred location privacy protection ability for mobile cloud user in the long-term level with a low cost. Experimental results demonstrate the effectiveness of our approach on trip faking as well as keeping long-term consistency. The main contributions of this paper are summarized as follows:

1. We define three categories of privacy, i.e., *microscopic-level (snapshot level)*, *journey-level*, and *long-term* location privacy to capture different application requests. In this paper, we focus on long-term location privacy protection.
2. We propose the four principles (symmetry, decongestion, practicability, and consistency) for the dummy generation to provide long-term location privacy protection. These principles are essential for protecting users' privacy in a long period.
3. Based on the above four principles, we propose and develop novel dummy generation algorithms that take both real geographical information and long-term consistency into account. The comprehensive experimental results demonstrate the effectiveness of our approach in terms of the fake path generation and the long-term consistency.

The rest of the paper is organized as follows: in Sect. 2, we briefly review related work. Section 3 proposes the four dummy generation principles for long-term location privacy protection. In Sect. 4, we propose the long-term consistent dummy generation algorithms. Section 5 evaluates the effectiveness of our approach. Section 6 concludes this paper.

2 Related work

Researches on location privacy protection for LBSs can be categorized into *microscopic-level* and *macroscopic-level* location privacy.

Microscopic-level privacy protection schemes focus on how to hide users' location in a single query (Leu et al. 2014; Leu 2009). To protect privacy, Samarati et al. first proposed a k-anonymity metric (Samarati and Sweeney 1998). Gruteser and Grunwald (2003) applied this metric in the context of location privacy and designed a scheme to achieve k-anonymity by introducing a trusted anonymizer. Its basic idea is that the trusted anonymizer first gathers at least k queries from different users in a cloaking region and then delivers the queries to a untrusted LBS provider after eliminating users' identifiers. Consequently, the LBS provider only knows there are k users in the cloaking region, but cannot find out the exact location of any user.

To improve k-anonymity metric or to eliminate the anonymizer, some enhanced schemes are designed. The authors in Mokbel et al. (2006) set a minimum acceptable resolution of the cloaked spatial region to ensure that the cloaking area is big enough. In Chow et al. (2006), decentralized approaches were developed to eliminate the central anonymizer and to achieve k-anonymity. Moreover, a distributed location-aware access control mechanism for smart buildings was developed to make authorization decisions by considering both user location data and access credentials (Hernández et al. 2014). This localization system does not require any intermediate entity, providing the benefits of a decentralized approach for smart environments.

Macroscopic-level privacy protection aims at hiding users' location in a journey with multiple queries. Dummy-based techniques were widely researched for macroscopic-level privacy protection because they need neither trusted third party nor reliable peers. Kido et al. (2005) propose to protect users' location privacy by sending user's location with fake locations called dummies, but their focus is reducing communication cost instead of dummy generation. There are many researches on improving dummy generation scheme. In Lu et al. (2008), two schemes are introduced to generate queries containing k locations which form a cloaking area always bigger than a threshold. Authors of You et al. (2007) proposed a mechanism to generate fake paths under unrealistic assumptions. In Chow and Golle (2009), a method which relies on adding noises to traces generated by a trip planner was developed.

In most recent years, Mahmoud et al. proposed a cloud-based scheme for efficiently protecting source nodes' location privacy against Hotspot-locating attack by creating a cloud with an irregular shape of fake traffic (Mahmoud and Shen 2012). Bilogrevic et al. (2014) designed a private circular query protocol (PCQP) to deal with privacy and accuracy issues of privacy-preserving LBS.

3 Soft computing-based dummy generation principles for LBSs

3.1 Preliminaries

3.1.1 Soft computing

Soft computing is usually defined as a family of techniques such as evolutionary computation, probabilistic reasoning, and chaos theory (Ogiela et al. 2014), well suited for coping with imprecision and uncertainty (Trawiński et al. 2013). Some researches have been done on privacy protection based on soft computing techniques over the last decade.

Trawiński et al. (2013) proposed a multiclassifier approach for topology-based WiFi indoor localization, which combines soft computing techniques over the data collected by smartphones. This proposal does not require additional hardware or infrastructure since the location estimation stage is embedded into smart objects.

Hernández et al. (2014) implemented a soft computing-based location-aware access control application for smart buildings. This approach set up an access control engine embedded into smart objects, which are responsible to make authorization decisions by considering both user location data and access credentials. In this system, the location-aware access control mechanism does not require any intermediate entity, providing the benefits of a decentralized approach for smart environments.

Based on soft computing approach, Wahab et al. (2009) modeled individual driving behavior to identify features that may be efficiently and effectively used to profile each driver. It has great potential applications on real-time driver identification and verification.

3.1.2 Location-based services

A query sent to a LBS provider typically contains four parts: (1) user's identification like username, (2) user's current location (e.g., latitude and longitude), (3) a question such as "what are the restaurants within one kilometer?" and (4) a timestamp. Without modification on server side, multiple requests must be made if we want to confuse adversaries with fake locations. However, excessive communication overhead are unacceptable in this way. So, we suppose users make one query including genuine and fake locations at one time. We denote a set of locations in a query as $Q^i = L^i = \{l_0^i, l_1^i, \ldots, l_k^i\}$, where i is the ith query, k ($k = 1, 2, 3, \ldots$) is the amount of fake locations, and l_0^i is user's genuine location. A fake location like l_1^i is called a dummy, and a dummy group is defined as a series of consecutive dummies, denoted as $D_j = \{l_j^1, l_j^2, \ldots, l_j^i\}$), which is similar to notation in You et al. (2007). Given a reference location l^{r} and current location l^{c}, a movement or an offset means the vector from l^{r} to l^{c}, expressed as $\mathbf{m} = l^{\mathrm{c}} - l^{\mathrm{r}} = (d, \theta)$.

3.1.3 Soft computing-based location privacy protection

We employ soft computing techniques to provide mobile users with preferred privacy protection, aiming at balancing the tolerable imprecision and the low computing cost. Without loss of generality, we divide a map into a series of squares, and each square (denoted as s) is labeled with respective geographical information g. In order to reduce the number of squares for saving storage without decreasing the precision, the adjacent squares with the same or "similar" information can be combined, and then spatial data structure such as Quad-tree can be introduced.

In this paper, there are four assumptions to profile the ability of adversaries. First, adversaries know specific queries from a user and may record all the queries for correlation attack in a long term. This assumption allows LBS providers to require explicit identifiers such as usernames. Second, adversaries know the dummy generation algorithms so that they can find out users' true locations through analyzing these algorithms. The third assumption is that adversaries cannot distinguish user's true location by sending specific answers to user and then watching his reaction. Under this assumption, it is not necessary to care about the content of user's questions and provider's answers. Finally, we assume adversaries have no targeted information such as user's home place. This is also a common assumption in previous work.

We will present the four principles for dummy generation through the following two schemes.

3.2 Random scheme

We consider geographical information in dummy generation to make it practical. Although this scheme randomly generates fake locations, it still needs to meet some restrictions. For example, a dummy must not appear in a lake while a user is walking on a street. In our square model, when a user is at square x and corresponding geographical information is denoted as g_x, a square y that contains a fake location must have similar information g_y to keep dummies similar to genuine one. So a similarity should be defined on the set of all squares based on their geographical information. This relation must be symmetric, i.e., if x is similar to y, then y must be similar to x. Otherwise, if two locations in x and y appear

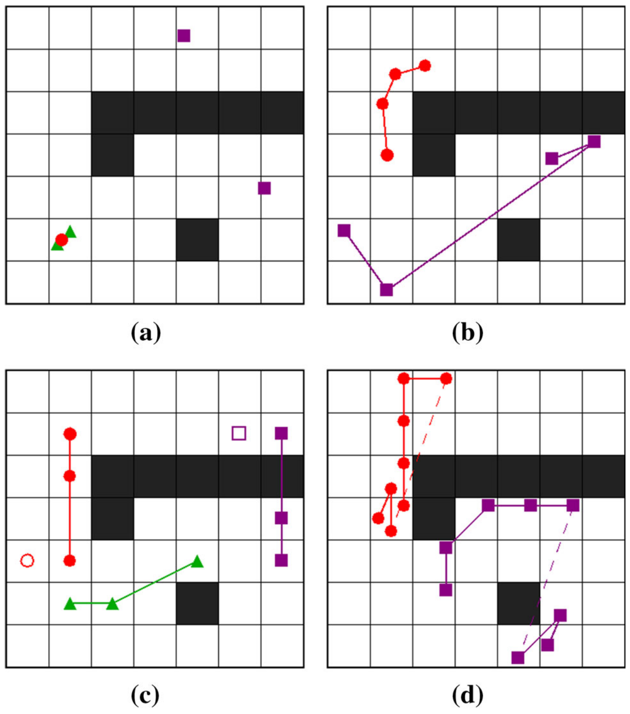


Fig. 2 Problems in defective schemes. **a** Congestion. **b** Impracticability. **c** Asymmetry. **d** Inconsistency

in one query, adversary would deduce that x must be dummy, because he knows the scheme never is generated based on a dissimilar square. This requirement can be generalized to the whole dummy generation algorithm. Specifically, if one location can be generate based on another location, the inverse side must also be established. This is the first dummy generation principle *symmetry*.

Principle 1 (Symmetry) *Dummy generation algorithm must be symmetrical.*

Based on this principle, we can generate dummies as follows: When a user generates a query at location l_0, we first determine his current square s_0, then extract all squares similar to s_0, and finally randomly choose a square for each dummy group as s_j. After that, we randomly select a position in s_j as l_j. Any adversary cannot distinguish the user's location based on a single query. If genuine and fake locations are too close, however, the adversary may compromise user's location privacy without needing to find out the genuine location. In Fig. 2a, a circle represents a genuine location while other shapes stand for dummies. If dummies are rectangles, user's privacy is properly protected. Otherwise, user's privacy is in danger. It means that too close or even overlapped locations will cause degradation of snapshot level privacy protection. So, we have the second principle *decongestion*.

Principle 2 (Decongestion) *In a query, some locations may be close to each other occasionally, but safe distance must always be met.*

3.3 Imitational scheme

Random scheme can achieve only snapshot level location privacy protection. As is shown in Fig. 2b, we assume the side length of a square to be 1 km and the time interval between adjacent two queries be 1 min. Then the adversary could easily deduce that rectangles are dummies, because if these locations are genuine, the user's moving speed is impractical. So we get the third principle *practicability*. In fact, this principle is just the reason why we should choose a similar square in a dummy generation scheme.

Principle 3 (Practicability) *Dummies and trajectories must be practical.*

Intuitively, we can generate fake trajectories by imitating the real track according to the following steps. Before dummy generation, we make a rotation degree denoted as δ_j for dummy group D_j. For the first query, we randomly choose locations which are all in squares similar to s_0^0 and have proper distance. For other queries, we assume Q_p is the nearest preceding query, now our known conditions are user's current location l_0^c and $\{l_0^\mathrm{p}, l_1^\mathrm{p}, \ldots, l_k^\mathrm{p}\}$. Based on them, we can calculate user's movement (d_0, θ_0) and then determine dummies following equation $l_j^\mathrm{c} = l_j^\mathrm{p} + (d_0, \theta_0 + \delta_j)$.

However, the above approach potentially violates the *practicability* principle that the square that contains l_j^c may be dissimilar to s_0^c. Therefore, we need to adjust results as follows. For any s_j^c dissimilar to s_0^c, we search out the square $\tilde{s_j^\mathrm{c}}$ closest to l_j^c and similar to s_0^c ; then we randomly select a location $\tilde{l_j^\mathrm{c}}$ in $\tilde{s_j^\mathrm{c}}$ as new l_j^c.

But unfortunately, sometimes trajectories generated by this scheme have deficiencies. First, as depicted in Fig. 2c, if path with triangle points is genuine, the adversary who knows the algorithm can tell that the last points of other paths will look like the hollow shapes, so this path is definitely synthetic. It is a consequence of violating *symmetry* principle which is caused by adjusting step. From Fig. 2c, we can easily find that some paths, e.g., the path with rectangle points, are bogus because they are impractical. Finally, as illustrated in Fig. 2d, this scheme shares a same defect with schemes in Krumm (2009); Shankar et al. (2009). All of them could provide only trip-level privacy protection, which means the density distribution of a dummy group should be akin to that of genuine locations. Then the fourth principle *consistency* is derived.

Principle 4 (Consistency) *Dummies must be consistent with history and form a distribution pattern like original locations.*

4 Long-term consistent dummy generation

Existing work ignores some of above four principles through impractical assumptions. We take all four principles into account, especially long-term consistency, and then propose a novel scheme to generate consistent dummies. As shown in Fig. 1, different paths have totally different orientation change patterns, but all of them are practical. Thus without information out of bounds, adversary cannot tell the real track of user. There are still two pitfalls in fulfilling *consistency* principle. The first one is how to get the same results under the same conditions. The second is how to keep distribution pattern resembling the genuine one.

Since adversaries have the full knowledge of our approach, the algorithm must be randomized. A naive method to solve the first problem is recording all random results in persistent storage. But the conditions are varying fast. As a result, it is almost impossible to record all the information. Our proposal resolves this difficulty by using a trick. The numbers generated by a typical random function in programming are pseudorandom, which means the same seed results in the same number sequence. On the other hand, if steps before a random selection is deterministic, then the same conditions will lead to same candidate results for this selection. Each user sets a fixed privacy code C. Thus, under the same conditions, the same results can be obtained for user, while adversaries can only figure out the candidates but have no idea of how to filter final results from them.

In our scheme, current fake location is generated based on the last one. But due to the offset of dummies does not match corresponding genuine one, another mechanism must be introduced to ensure consistency of distribution patterns. In our proposal, if the time interval between two consecutive queries is longer than a predefined threshold, the latter one is defined as leading query; otherwise, it is called succeeding query. When leading queries arrive, we regenerate a new start point for each dummy group to maintain long-term consistency.

Users' queries can be divided into two categories: *initial query* and *leading query* in terms of their request time. The initial query refers to the first query from a user. The leading query is the query following the initial query within a specified time interval. In the two categories of queries, start points for each dummy group must be elaborately selected to guarantee the long-term consistency, presented as follows.

4.1 Dummy generation algorithm for the initial query

Before dummy generation, user needs to set up several parameters. The first one is k, indicating how many groups of dummies should be created. More dummies result in stronger anonymity in the cost of more resource consumption. The second parameter is a realm R, representing the place in which dummies should be located. A typical realm is a city, because if dummies are located in another city, the LBS provider is very likely to find out they are synthetic based on network information. Last parameter is a privacy code C which was described before.

In the initial query, there is no history information, and our focus is distance between any two of all the locations and geographical information. The upper bound of distance is determined according to R, and the lower bound is set as a fraction of upper bound. The upper bound which ensures starting positions of dummy groups can be found within the bounds of R, while the lower bound is designed to ensure *decongestion* principle is fulfilled.

Algorithm 1 describes the dummy generation mechanism for the initial query. We know user's starting location l_0^{s} and set the corresponding square as s_0^{s} in step 1. Then, we select all squares similar to s_0^{s} as candidates in step 2. Note that these candidate squares are d ($d_{\min} \le d \le d_{\max}$) far from l_0^{s}. Next, for each dummy group j, Algorithm 1 randomly chooses a square having proper distance to other start locations from candidates and then randomly selects a location in the square as l_j. Finally, Algorithm 1 will generate k fake locations l_j^{s} ($1 \le j \le k$). Let a query area contain n squares. The complexity of Algorithm 1 is only $O(n)$.

Algorithm 1 Dummy generation for initial query

```
Input: $l_0^s$, $d_{min}$ and $d_{max}$;
Output: $Q_s$={$l_0^s, l_1^s, \ldots, l_k^s$}
1: $s_0^s \leftarrow$ the square contains $l_0^s$;
2: $S \leftarrow$ all the similar square $d$ far from $l_0^s$ ($d_{min} \le d \le d_{max}$)
3: for $j = 1$ to $k$ do
4:     $s_j^r \leftarrow$ a random square in $S$
5:     $l_j^r \leftarrow$ a random location in $s_j^r$
6:     remove all the squares $d'$ far from $l_j^r$ in $S$
       ($d' < d_{min}$, or $d' > d_{max}$)
7: end for
8: Return {$l_0^s, l_1^s, \ldots, l_k^s$}
```

4.2 Dummy generation algorithm for the leading query

On receiving a leading query defined above, we first find out the squares which have the maximum number of user's genuine or fake occurrences and consider them as reference squares, namely $\{s_0^{\mathrm{r}}, s_1^{\mathrm{r}}, \ldots, s_k^{\mathrm{r}}\}$. Based on the reference squares and user's current location l_0^{c}, i.e., l_0^{s}, new start locations are generated with Algorithm 2. Similar to Algorithm 1, it first sets the corresponding square as s_0^{s} in step 1. Then, Algorithm 2 calculates $d_{\min}$ and $d_{\max}$. They determine in which area fake locations will be generated, together with $\theta_{\min}$ and $\theta_{\max}$), as shown in Fig. 3. Finally, Algorithm 2 will generate k fake locations l_j^{s} ($1 \le j \le k$). Let a query area contain n squares. The complexity of Algorithm 2 is also $O(n)$.

Algorithm 2 Dummy generation for leading query

```
Input: $\{s_0^r, s_1^r, \ldots, s_k^r\}$, $l_0^s$, $\{\delta_0,\delta_1,\ldots,\delta_k\}$ and constants $C_1$,
       $C_2, C_3$;   $(0 \le C_1 \le 1 \le C_2, 0 \le C_3 \le \pi)$
Output: $Q_s=\{l_0^s, l_1^s, \ldots, l_k^s\}$
1: $s_0^s \leftarrow$ the square contains $l_0^s$;
2: $l_0^r \leftarrow$ a random location in $s_0^r$
3: $(d_0, \theta_0) \leftarrow l_0^s - l_0^r$
4: $(d, \theta) \leftarrow (d_0 \times$ RAND$(C_1, C_2)$,$\theta_0 +$ RAND$(-\pi, \pi))$
5: $(d_{min}, d_{max}) \leftarrow (C_1 \times d, C_2 \times d)$
6: for $j = 1$ to $k$ do
7:    $l_j^r \leftarrow$ a random location in $s_j^r$
8:    $\theta_j \leftarrow \theta +$ RAND$(-\pi, \pi)$
9:    $(\theta_{min}, \theta_{max}) \leftarrow (\theta_j - C_3, \theta_j + C_3)$
10:   let $l_j^r$ be the center, with $d_{min}$, $d_{max}$, $\theta_{max}$, $\theta_{min}$, a realm $R$ is
      formed, illustrated in Fig. 3;
11:   $S \leftarrow$ squares intersected with $R$ and similar to $s_0^s$
12:   Remove squares which have already been selected more than $k/2$
      times from $S$
13:   If ($S$ is not empty) {
14:      $s_j^s \leftarrow$ a random square in $S$
15:      $l_j^s \leftarrow$ a random location within $s_j^s \cap R$ }
      Else
16:      Change $l_0^s$ into a square near $s_0^s$; then jump to the start
      EndIf
17: end for
18: Return$\{l_0^s, l_1^s, \ldots, l_k^s\}$
```

In Algorithm 2, adjusting a dummy is intentionally avoided to meet *symmetry* principle and removing high frequently selected squares for adhering to *decongestion* principle. There are three constants which determine the area of candidate realm. Smaller area indicates higher consistency, but it also results in higher probability that no similar square can be found especially when "similar" is defined strictly. When there is no suitable squares for any dummy group, we modify user's current location as a compromise. The answer's accuracy of one query is sacrificed for indistinguishability of dummies. Another problem about reality may be raised. Does it matter that straight line distance, instead of route distance, is used for candidate squares selection? The answer is no, because there is a long time between start point regeneration and last query.

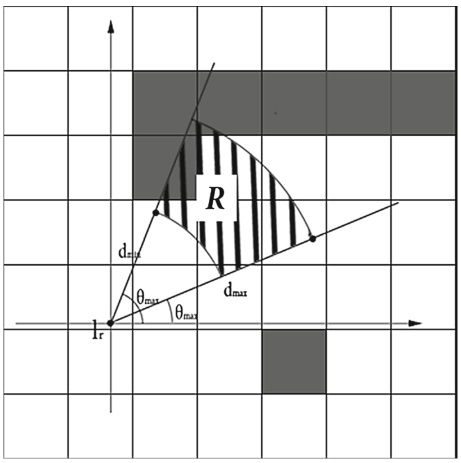


Fig. 3 Candidate region

The difference between two kinds of distance does not matter in starting location selection, but it is a serious matter when considering dummy generation for succeeding queries. Getting route distance is trivial; a simple way is using Google Map API. We denote route distance as $\tilde{d}$, different from straight line distance d. Apart from initial query which is generated by algorithm described before, all queries have a nearest previous query, namely Q_p. Based on l_0^p and user's current location l_0^c, we first get route distance $\tilde{d}_0$. Next, we calculate $\tilde{d}_\mathrm{min}$ and $\tilde{d}_\mathrm{max}$ based on $\tilde{d}_0$ and predefined constants. Then, for each dummy group j, a irregular realm R_j is formed by all points which have a route distance to l_j^p lager than $\tilde{d}_\mathrm{min}$ and smaller than $\tilde{d}_\mathrm{max}$. After R_j is determined, following steps are the same as start point regeneration.

5 Evaluation

In this section, we evaluate the effectiveness of our approach in terms of the fake path generation and the long-term consistence.

5.1 Simulation environment

We created a map with 49 × 26 squares according to the satellite map of our campus. The square is 50 × 50 m. They are divided into four types, which indicate normal places, roads, water areas and unreachable places. We define that two squares are similar if and only if they fall into the same type.

Then, we simulate an author's daily movements within the map and collect samples. In our scheme, different dummy generation algorithms are selected according to the time difference between two consecutive queries, so a sample is composed by a location and a time flag which indicates leading or succeeding query. Then we implement our approach and take the samples as input for our experiments.

5.2 Framework of the proposed approach

The query and response process in our approach is illustrated in Fig. 4. First, a user makes an initial query message. He, then, selects a privacy protection scheme and generates a privacy-protected query message. If the message is secret, it will be encrypted by an encryption protocol. Finally, the query message is sent to a cloud service provider, as shown in Fig. 4a.

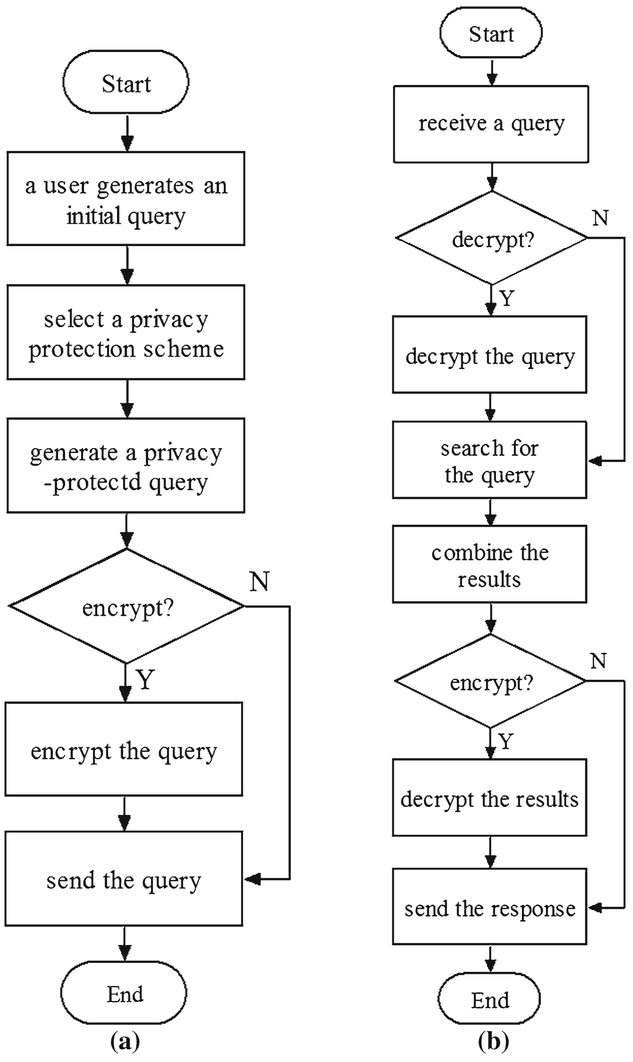


Fig. 4 The query and response process. **a** The flow of a query. **b** The flow of a response

On the other hand, on receiving a query, the service provider first decrypts the message if it is encrypted and then searches its database for the response. Similarly, the response message will be encrypted if it is secret. At that time, the corresponding response will be sent back to the user, as illustrated in Fig. 4b.

5.3 Results and evaluations

We evaluate our approach in the following aspects.

5.3.1 Fake path generation

The first group of experiments was designed to test the path faking ability of our approach to evaluate its effectiveness in microscopic-level location privacy protection. We took some paths from samples as users' genuine movements and then tested our scheme with different input paths combined with different values of parameter k, which represents the number of dummy groups. Figure 5 shows the result of a single test. In this figure, genuine locations are denoted as triangle points, while other points represent dummies grouped by different shapes.

After all tests, we carried out a user study among students in our university. The overall recognition rate is around 30 % when $k = 3$. These results show that our scheme is effective in fake path generation, but a little lower than the scheme in Shankar et al. (2009) with recognition rate of 26 %. The reason is that they rely on user's destination and mature path planer provided by Internet giant, while our fake path may take a half-turn caused by reaching a dead end. Another reason is they ignore the symmetry principle through implicitly assuming adversaries do not have the full knowledge of their algorithm.

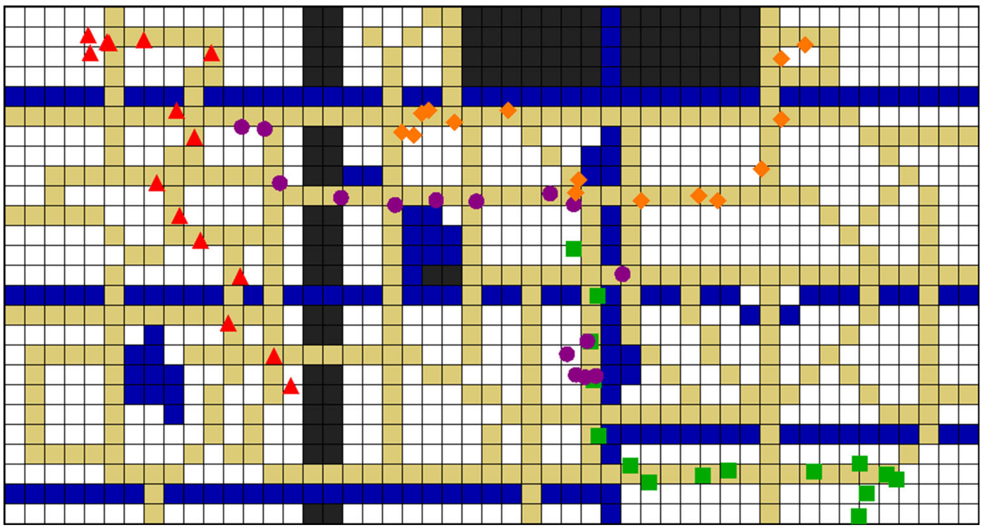


Fig. 5 Pathes for a trip in our campus: one genuine path marked with *triangles* and three fake paths marked with *diamonds*, *circles* and *squares*

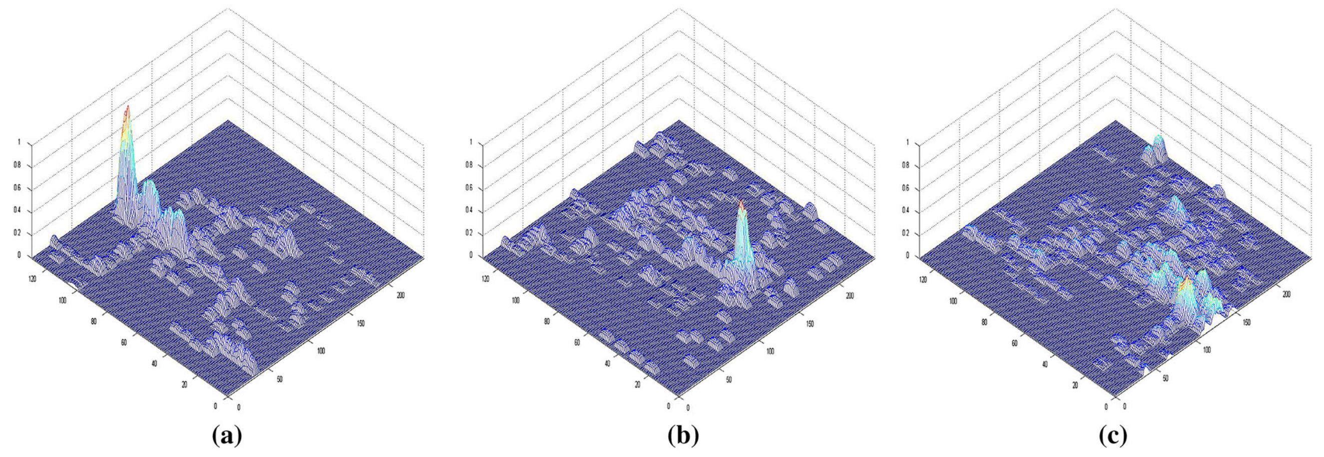
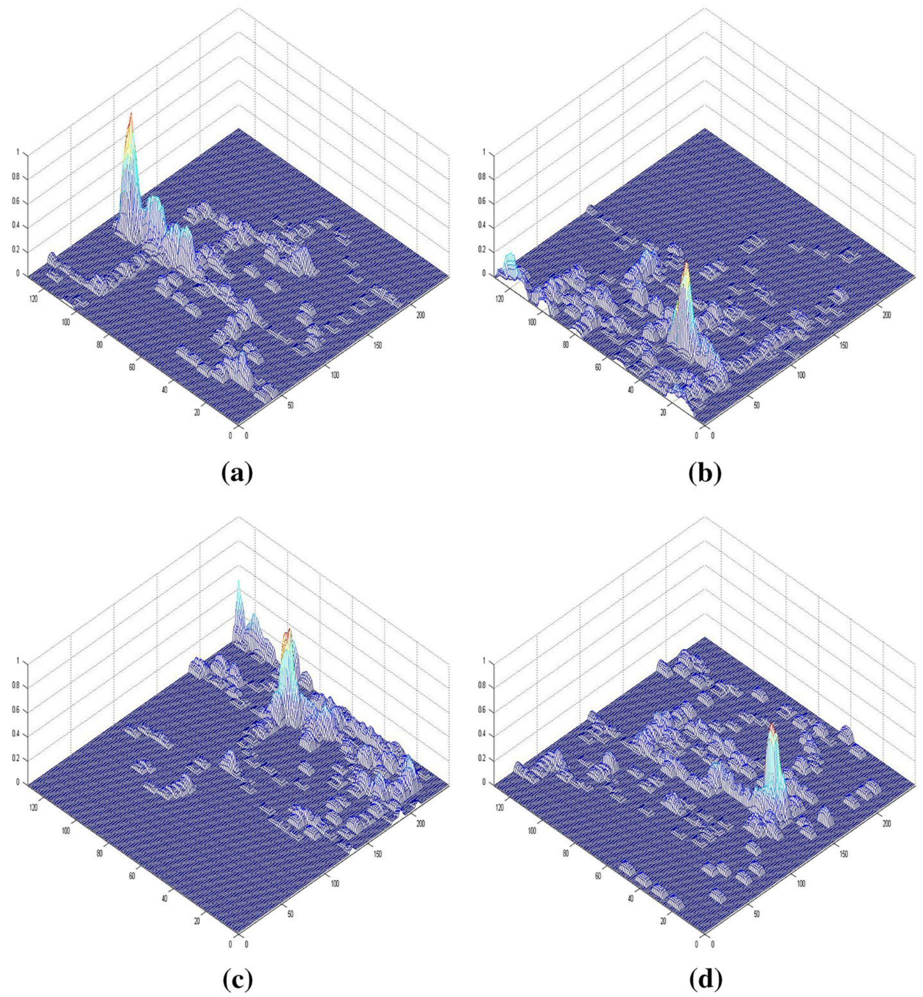


Fig. 6 Density maps of one group locations, 500 queries. **a** The genuine locations. **b** A dummy group, consistent scheme. **c** A dummy group, imitational scheme

Fig. 7 Density of each group in our long-term consistent generation scheme ($q = 500, k = 3$). **a** Real location group. **b** Fake location group 1. **c** Fake location group 2. **d** Fake location group 3



5.3.2 Density distribution consistency

Next, we conducted the second group of experiments to verify the long-term consistency of our approach in density distribution. Since any other work which takes into account both geographical information and long-term consistency could not be found, we implemented imitational scheme as the contrast. As stated before, imitational scheme

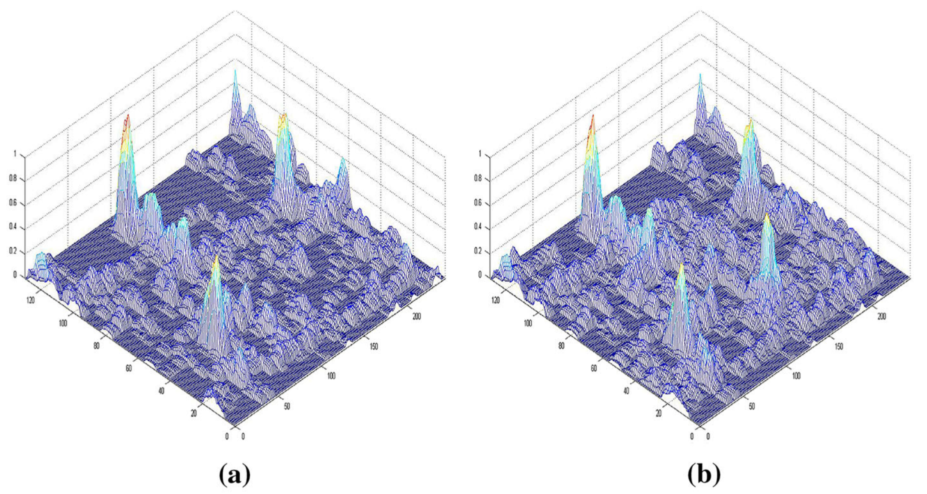
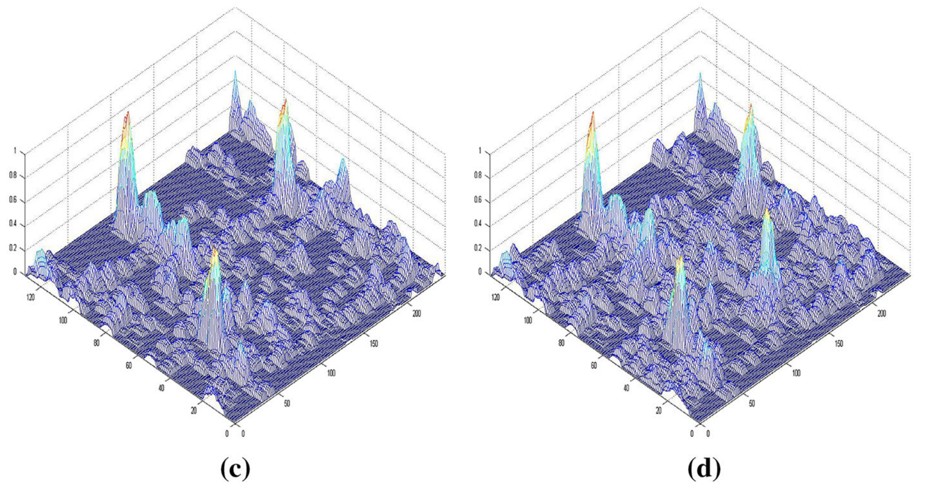



Fig. 8 Overall density maps of our consistent scheme. **a** 500 queries, $k = 2$. **b** 500 queries, $k = 3$. **c** 2500 queries, $k = 2$. **d** 2500 queries, $k = 3$

may violate symmetry principle, but it is not our concern in this experiment. Our inputs consisted of 500 continuous samples (about a week's movements) at first and then extended to 2500 samples as well. For each input and each scheme, we performed two tests with $k = 2$ and $k = 3$, respectively. In each test, queried locations are, respectively, recorded for genuine location group and each dummy group.

Raw data recorded in every test were processed to generate density maps. In each record, there are 500 or 2500 locations represented as pairs of coordinates denoted as (x, y), where $0 \leq x < 49$ and $0 \leq y < 26$. First, we scaled coordinate values up by 5, then got $0 \leq x < 245$ and $0 \leq y < 130$. Next, we created a grid with 245×130 cells, where the cell at second column first row is denoted as $c_{1,0}$. Then we counted up the number of times that locations appear in each cell. For example, a location (1.10, 1.20) in raw data will result in counter denoted as $t_{5,6}$ being increased by 1. Based on these statistics, the density value of $c_{m,n}$ is calculated by following equation:

$$d_{m,n} = \sum_{i=\max(m-S,0)}^{\min(m+S,244)} \sum_{j=\max(n-S,0)}^{\min(n+S,129)} (w_{i,j} \times t_{i,j}), \tag{1}$$

where $\sum w_{i,j} = 1$ and $w_{i,j}$ has negative linear correlation to the distance between $c_{i,j}$ and $c_{m,n}$, and S controls the overall gradient of density. We set $S = 3$. According to this equation, a density matrix is generated for every record. For each test, there are $k+1$ matrixes, respectively, corresponding to the group of all genuine locations and k dummy groups. We added them up to form another density matrix, which is corresponding to this test. Finally, all $k + 2$ matrixes were normalized by dividing the maximum density value in this test.

After data processing, density maps were created according to density matrixes. Figure 6 shows three density maps generated in two tests for different schemes using the same input and parameter k. Figure 6a is corresponding to the user's real locations, while Fig. 6b, c is corresponding to two dummy groups generated by consistent scheme and imi-

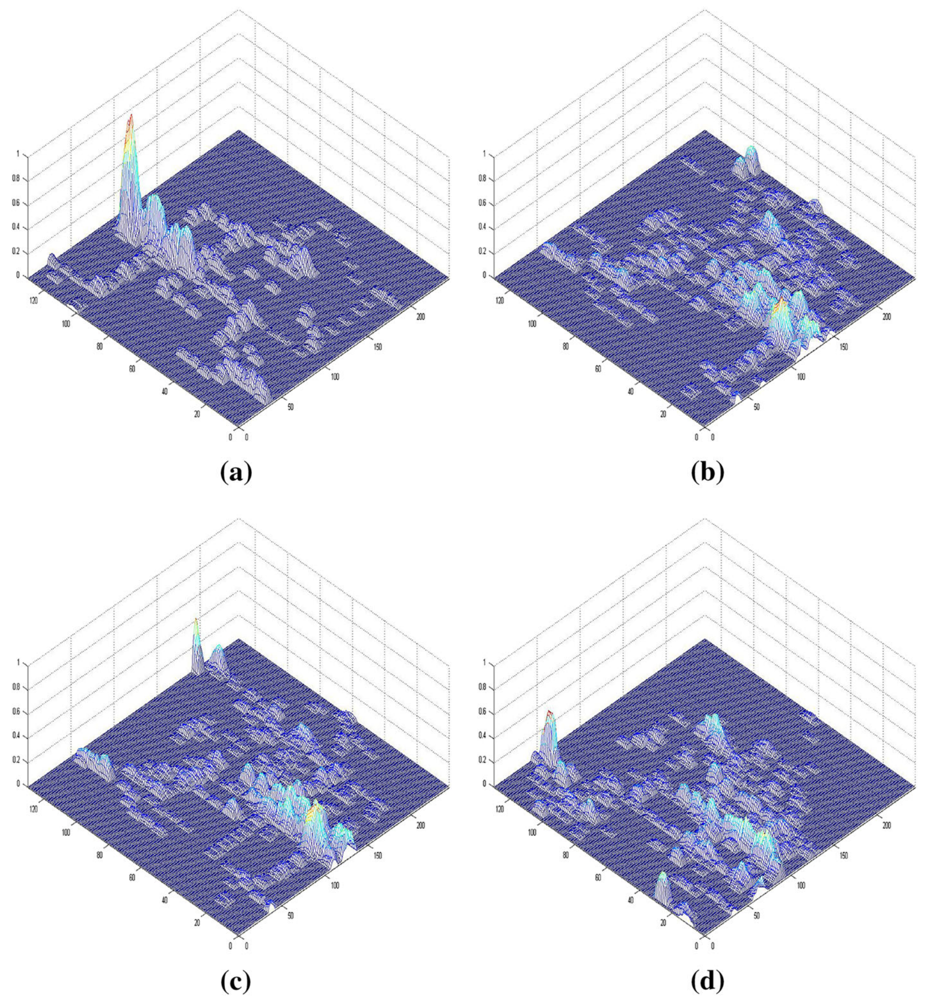


Fig. 9 Density of each group in imitational generation scheme ($q = 500, k = 3$). **a** Real location group. **b** Fake location group 1. **c** Fake location group 2. **d** Fake location group 3

tational scheme, respectively. In density maps, x and y are location coordinates and z represents density. As we can see, the distribution pattern of the dummy group generated by consistent scheme is similar to the real one, and they both have high peaks which means that the user or this dummy group always shows up there. But the distribution pattern of the dummy group generated by imitational scheme is totally different. It is not consistent with the real one. Then, we observe overall density maps. Maps of consistent scheme are presented in Figs. 7 and 8, while their counterparts are omitted due to limited space. In these maps, although distribution patterns of the genuine locations and the dummy groups are not identical, adversaries cannot distinguish the genuine one from others because they all look similar to each other.

Results in Figs. 6, 7 and 8 demonstrate that our dummy generation algorithms can protect long-term location privacy for mobile users. Dummy groups generated by consistent scheme have their own haunts like true users, and the density distribution is scarcely distinguishable among genuine and fake location groups. It is initially proved by this experiment that dummies generated by our scheme are highly consistent in density distribution. In contrast, imitational scheme cannot provide the long-term consistency, as illustrated in Fig. 9.

5.3.3 *Consistency with history*

Finally, we carried out another group of experiments to examine whether dummy groups are consistent with their own histories, i.e, when the user repeats his movements, each dummy group should also repeat its history. At first, 100 continuous samples were replicated once to form an input that contains 200 locations, i.e., when $i > 100$, location (x_i, y_i) is identical to (x_{i-100}, y_{i-100}). Then the number of original samples is enlarged to 500, which means the second input contains 1000 locations. For each input and each scheme, one test with $k = 3$ is performed. Queried locations are recorded in the same way described before.

In the input, the first half and the second half are same, but it does not always hold in every record. We define inconsistent distance as the average distance between every location in the

Table 1 Inconsistency distances

Queries	Scheme	Group 1	Group 2	Group 3	Average
200	Consistent	0.00	0.00	0.00	0.00
200	Imitational	33.79	32.13	24.37	30.1
1000	Consistent	11.51	7.50	3.10	7.37
1000	Imitational	20.38	14.05	32.78	22.4

Table 2 Adjusting times

Queries	k	Adjusting times	Probability (%)
500	2	29	5.8
500	3	25	5.0
2500	2	60	2.4
2500	3	66	2.6

first half and its counterpart in the second half and denote it as D. For example, when the input contains 200 locations, $D = \left(\sum_{i=1}^{100} \sqrt{(x_i - x_{i+100})^2 + (y_i - y_{i+100})^2}\right)/100$. We calculate the inconsistent distances for every dummy group in each test, and Table 1 presents the results (unit is 10 m). The dummy groups generated by our consistent scheme are not always consistent with their history. Instead, when the user come back to the initial location, dummy groups probably generate locations near initial ones. However, our consistent scheme always outperforms the imitational scheme.

5.4 Other issues

As mentioned above, in our scheme, user's real location may be adjusted to keep the algorithm symmetric in the query, which affects query's accuracy. So we investigate the frequency of adjusting. This information was recorded in our second experiment, as shown in Table 2. The adjusting times are not increasing with k apparently, and its growth is relatively low when comparing to the quantity of queries.

6 Conclusions and future work

We examined existing dummy generation schemes and analyzed their limitations in the privacy protection. Then we proposed four principles for the dummy generation. These principles are essential for protecting user's long-term location privacy. Based on these principles, we proposed and developed a set of novel dummy generation algorithms, which take both real geographical information and long-term consistency into account. The comprehensive experimental results demonstrate that our approach achieves remarkable long-term consistency while maintaining reasonable capacity for fake path generation.

As a part of our future work, we are going to investigate how to resist various attacks in mobile cloud systems to enhance our approach.

Acknowledgments This work was supported by the National Natural Science Foundation of China (NSFC) projects (Nos. 91438121 and 61373156), the Key Basic Research Project (No. 12JC1405400) of the Shanghai Municipality, the Shanghai Pujiang Program (No. 13PJ1404600) the National Basic Research Program of China (No. 2015CB352400), and Shanghai Branch of Southwest Electron and Telecom Technology Research Institute Project (No. 2013008).

References

Ahmad MW, Ansari MA (2012) A survey: soft computing in intelligent information retrieval systems. In: 12th international conference on computational science and its applications (iccsa), pp 26–34

Ardagna C, Cremonini M (2011) An obfuscation-based approach for protecting location privacy. IEEE Trans Depend Secure Comput 8(1):13–27

Ardagna CA, Cremonini M, Damiani E (2007) Location privacy protection through obfuscation-based techniques. In: Data and applications security, vol xxi. Springer, Berlin, pp 47–60

Bilogrevic I, Jadliwala M, Joneja V (2014) Privacy-preserving optimal meeting location determination on mobile devices. IEEE Trans Inf Forens Secur 9(7):1141–1156

Cho WJ, Park Y, Sur C (2013) An improved privacy-preserving navigation protocol in VANETs. J Wire Mob Netw Ubiquitous Comput Depend Appl 4(4):80–92

Chow R, Golle P (2009) Faking contextual data for fun, profit, and privacy. In: Proceedings of the 8th acm workshop on privacy in the electronic society. Wpes '09. ACM, New York, pp 105–108

Chow CY, Mohamed MF, Liu X (2006) A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In: Proceedings of gis '06. ACM, New York, pp 171–178

Dewri R (2013) Local differential perturbations: location privacy under approximate knowledge attackers. IEEE Trans Mob Comput 12(12):2360–2372

Gedik B, Liu L (2008) Protecting location privacy with personalized k-anonymity: architecture and algorithms. IEEE Trans Mob Comput 7(1):1–18

Gruteser M, Grunwald D (2003) Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of mobisys '03. ACM, New York, pp 31–42

Hernández JL, Moreno MV, Jara AJ (2014) A soft computing based location-aware access control for smart buildings. Soft Comput 18(9):1659–1674

Kido H, Yanagisawa Y, Satoh T (2005) An anonymous communication technique using dummies for location-based services. In: Proceedings of icps '05, pp 88–97

Kim DJ, Bien Z (2008) Design of personalized classifier using soft computing techniques for personalized facial expression recognition. IEEE Trans Fuzzy Syst 16(4):874–885

Krumm J (2009) Realistic driving trips for location privacy. In: Proceedings of pervasive '09, pp 25–41

Leu FY (2009) A novel network mobility handoff scheme using sip and sctp for multimedia applications. J Netw Comput Appl 32(5):1073–1091

Leu FY, Liu JC, Hsu YT, Huang YL (2014) The simulation of an emotional robot implemented with fuzzy logic. Soft Comput 18(9):1729–1743

Liang H, Cai LX, Huang D, Shen X, Peng D (2012) An smdp-based service model for interdomain resource allocation in mobile cloud networks. IEEE Trans Veh Technol 61(5):2222–2232

Lien IT, Lin YH, Shieh JR (2013) A novel privacy preserving location-based service protocol with secret circular shift for k-nn search. IEEE Trans Inf Forens Secur 8(6):863–873

Lu H, Jensen CS, Yiu ML (2008) Pad: privacy-area aware, dummy-based location privacy in mobile services. In: Proceedings of mobide '08. ACM, New York, pp 16–23

Mahmoud M, Shen XM (2012) A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks. IEEE Trans Parallel Distrib Syst 23(10):1805–1818

Metre PB, Radhika KR, Gowrishankar G (2012) Survey of soft computing techniques for joint radio resource management. In: 2012 international conference on multimedia computing and systems (icmcs), IEEE. IEEE, New York, pp 562–565

Mitra Sushmita, Pal Sankar K, Mitra Pabitra (2002) Data mining in soft computing framework: a survey. IEEE Trans Neural Netw 13(1):3–14

Mokbel MF, Chow CY, Aref WG (2006) The new casper: query processing for location services without compromising privacy. In: Proceedings of vldb '06, pp 763–774

Murakami M, Honda N (2008) Performance of the ids method as a soft computing tool. IEEE Trans Fuzzy Syst 16(6):1582–1596

Ogiela Marek R, Castiglione Aniello, You Ilsun (2014) Soft computing for security services in smart and ubiquitous environments. Soft Comput 18(8):1655–1658

Pan X, Xu JL, Meng XF (2012) Protecting location privacy against location-dependent attacks in mobile services. IEEE Trans Knowl Data Eng 24(8):1506–1519

Paulet R, Koasar MG, Yi X (2012) Privacy-preserving and content-protecting location based queries. In: IEEE 28th international conference on data engineering (icde), IEEE. IEEE, New York, pp 44–53

Puttaswamy K, Wang S, Steinbauer T (2014) Preserving location privacy in geosocial applications. IEEE Trans Mob Comput 13(1):159–173

Samarati P, Sweeney L (1998) Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. In: IEEE symposium on research in security and privacy

Shankar P, Ganapathy V, Iftode L (2009) Privately querying location-based services with sybilquery. In: Proceedings of ubicomp '09. ACM, New York, pp 31–40

Shokri R, Freudiger J, Hubaux JP (2010) A unified framework for location privacy. In: In: Proceedings of the 9th International Symposium on Privacy Enhancing Technologies (PETS'10), pp 203–214

Tomoya T, Kazuhiro M, Ken M (2014) Evaluating data utility of privacy-preserving pseudonymized location datasets. J Wire Mob Netw Ubiquitous Comput Depend Appl 5(3):63–78

Trawiński Krzysztof, Alonso Jose M, Hernández Noelia (2013) A multiclassifier approach for topology-based wifi indoor localization. Soft Comput 17(10):1817–1831

Wahab A, Quek C, Tan C (2009) Driving profile modeling and recognition based on soft computing approach. IEEE Trans Neural Netw 20(4):563–582

Yager RR, Grichnik AJ, Yager RL (2014) A soft computing approach to controlling emissions under imperfect sensors. IEEE Trans Syst Man Cybern Syst 44(6):687–691

You T-H, Peng WC, Lee WC (2007) Protecting moving trajectories with dummies. In: Proceedings of the 2007 international conference on mobile data management, pp 278–282

Yu Xinghuo, Kaynak Okyay (2009) Sliding-mode control with soft computing: a survey. IEEE Trans Ind Electron 56(9):3275–3285

Zhang X, Zhao L, Zhao W, Xu T (2014) Novel method of flatness pattern recognition via cloud neural network. Soft Comput. doi:10.1007/s00500-014-1445-z