Enhancing Privacy through Caching in Location-Based Services

Ben Niu*, Qinghua Li†, Xiaoyan Zhu‡, Guohong Cao§ and Hui Li‡
*State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, China
†Department of Computer Science and Computer Engineering, University of Arkansas, AR, USA
‡National Key Laboratory of Integrated Networks Services, Xidian University, China
§Department of Computer Science and Engineering, The Pennsylvania State University, PA, USA
*niuben@iie.ac.cn, †qinghual@uark.edu, ‡{xyzhu, lihui}@mail.xidian.edu.cn, §gcao@cse.psu.edu

***Abstract*—Privacy protection is critical for Location-Based Services (LBSs). In most previous solutions, users query service data from the untrusted LBS server when needed, and discard the data immediately after use. However, the data can be cached and reused to answer future queries. This prevents some queries from being sent to the LBS server and thus improves privacy. Although a few previous works recognize the usefulness of caching for better privacy, they use caching in a pretty straightforward way, and do not show the quantitative relation between caching and privacy. In this paper, we propose a caching-based solution to protect location privacy in LBSs, and rigorously explore how much caching can be used to improve privacy. Specifically, we propose an entropy-based privacy metric which for the first time incorporates the effect of caching on privacy. Then we design two novel caching-aware dummy selection algorithms which enhance location privacy through maximizing both the privacy of the current query and the dummies' contribution to cache. Evaluations show that our algorithms provide much better privacy than previous caching-oblivious and caching-aware solutions.**

I. Introduction

As one of the most popular activities, Location-Based Services (LBSs) have enriched our life with many applications over recent years. Users can download these location-based applications from Apple Store or Google Play Store easily through their smartphones or tablets. With such applications, mobile users can easily obtain information about various Point of Interests (POIs) in the vicinity, e.g., the menu of restaurants and the availabilities of nearby bars.

To enjoy these conveniences, mobile users need to submit queries to the untrusted LBS server. Since these queries include some personal information, such as users' locations and the queried interests, the LBS server can easily infer who are doing what in which place. The server may track users directly or release their personal information to third parties such as advertisers. We thus need to pay much attention to protecting user privacy.

To address the privacy issue for mobile users in LBSs, many approaches have been proposed over recent years. They include trusted anonymization server-based approaches [1], [2], [3] and mobile device-based schemes [4], [5], [6], [7], [8], [9], [10], [11]. One problem with most existing privacy preserving approaches is that they are limited to the perspective of protecting privacy for the queries sent to the LBS server. They neglect another perspective of privacy protection which is to reduce the number of queries sent to the LBS server. If less queries are submitted to the server, less location information is released, and hence there is less chance of exposing user's locations. A natural way of reducing the number of queries sent to the LBS server is to use caching; i.e., using the cached data obtained from previous queries to answer future queries. Although a few previous works [12], [13], [14] recognize the usefulness of caching for better privacy, they use caching in a pretty straightforward way, and do not provide in-depth understanding about the effect of caching on privacy.

In this paper, we propose a caching-aware, dummy-based solution to protect location privacy in LBSs. The basic idea is to cache the service data obtained for both the real location and dummy locations of the current query, and use the cached data to answer future queries so as to reduce the queries sent to the LBS server. Different from previous straightforward caching-based solutions, we rigorously explore the effect of caching on the achieved privacy. Specifically, we propose an entropy-based privacy metric which takes caching into account. Based on this metric, we design two caching-aware dummy selection algorithms. When selecting dummies for a query, these algorithms not only maximize the privacy of the current query, but also maximize the dummies' contribution to cache which in turn improves privacy.

The contributions of this paper are summarized as follows:

- We propose a privacy metric which for the first time incorporates the effect of caching on privacy. It describes the quantitative relation between cache hit ratio and the achieved privacy. Based on this metric, we rigorously explore to what extent caching can improve privacy.
- We propose a novel Caching-aware Dummy Selection Algorithm (*CaDSA*), which innovatively integrates caching with dummy selection to maximize the achieved privacy.
- We identify important factors that affect caching performances, including normalized distance and data freshness. Considering these factors, we design *enhanced-CaDSA* to further improve privacy.
- We evaluate the proposed algorithms with extensive simulations, and shed light on how caching enhances privacy.

The rest of this paper is organized as follows. In Section II,

we review related work. Section III presents some preliminaries and our privacy metrics. Section IV describes the details of our proposed schemes, security analysis and implementation issues. Section V presents performance evaluation results. Finally, we draw the conclusions in Section VI.

II. Related Work

Privacy has been one of the most popular research topics recently (e.g., privacy in LBSs [15], [16] and privacy in mobile sensing [17], [18], etc.). We review some existing research on user's privacy issues for LBSs. Although policy-based approaches [19] and cryptography-based approaches [20] have been proposed to protect location privacy for LBSs, most existing works are based on anonymization or location perturbation and obfuscation. They include trusted anonymization server-based schemes [2], [3] and mobile device-based schemes [6], [9], [10], [14], [11], [21], [22].

The former category suffers from the single point of failure due to the reliance on a trusted server, e.g, *location anonymizer* in [23]. As the result, if an adversary gains access to it, the privacy of all users will be compromised. This trusted server is also a performance bottleneck since all the submitted queries have to go through it.

Mobile device-based schemes avoid these problems. Some works focus on decreasing the computational and storage overhead by using VHC mapping [6], encountered-based solution [9], and *k-anonymous* cloaking box [10]. However, users need to communicate with each other to obtain extra data. Kido *et al.* [4] solved this problem by introducing dummy locations, which are randomly selected at user's mobile device to achieve *k-anonymity*. But they ignore the effects of *side information* [24] at the adversary. With *side information* (e.g., query probability in the local map), the adversary can easily filter out some randomly selected dummy locations from the submitted k locations, which decreases the anonymity degree of *k-anonymity*. Niu *et al.* [14], [11], [21], [22], [25] proposed a set of solutions to address this problem by carefully selecting dummy location considering that *side information* may be obtained by the adversaries; however, its associated communication and storage cost is pretty high. All of the above works overlook the use of caching to improve privacy.

Caching has been used in a few previous works. Shahriyar *et al.* [12] proposed the *Cache* system to improve user privacy. By pre-fetching the service data within a particular area before coming into that area, mobile users can search the Point of Interests (POIs) locally instead of sending queries to the untrusted LBS server. However, mobile users need to store a huge amount of service data for a large area. Shokri *et al.* [13] designed a distributed location privacy preserving algorithm for a collaborative group, termed *MobiCrowd*. Users in *MobiCrowd* query neighbors for service data before sending a query to the LBS server. Through this way, the location exposing probability is decreased. However, this scheme has several drawbacks. It pays little attention to protecting privacy for users who need to send queries to the LBS server. Additionally, it is not intentionally designed to improve cache hit ratio, which renders a low benefit brought by caching. *Mobicache* [14] simply tries to cache more data that has not been cached yet, and it does not consider the *side information* that an adversary may have. Thus, the adversary can infer the real location with the help of *side information*. There are also several common problems with these caching-based solutions. They do not have an integrated privacy metric to measure the effect of caching on privacy, and their caching design is pretty straightforward without considering important factors such as query probability and data freshness. Different from them, we propose a privacy metric to model the effect of caching, and our caching-aware dummy selection algorithms carefully combine *k-anonymity*, caching, and *side information* to achieve higher privacy degree and cache hit ratio.

III. Preliminaries

In this section, we first introduce our system model with some basic concepts adopted in this paper, and then present the motivation and the basic idea of our solution. Finally, we introduce the location privacy metrics used in our schemes.

A. System Model

The adversary may have some *side information* which helps it infer users' locations. In this paper, *side information* refers to the probability that a query can be sent from each location [11]. Users may also have such information which helps select the best dummies for privacy. This kind of information can be obtained either from some well-known social applications (i.e., Google Latitude or Yelp!) or in peer-to-peer ways like the methods in [26], [27]. For a particular user, the ideal case is that he knows all the *side information* known to the adversary and thus can perform optimal dummy locations against the adversary.

In this paper, the untrusted LBS server is considered as the adversary. It can easily obtain all the *side information* by monitoring the queries sent from users. It also knows the location privacy protection mechanism used in the system. Additionally, it knows which data has been cached. Based on these information, it tries to perform *inference attacks* to deduce and learn user's location information.

B. Motivation and Basic Idea

A mobile user of current LBS applications always submits a location-related query to the untrusted LBS server to obtain and enjoy the corresponding service data. Let us consider an example scenario. In a classroom building of a university, many mobile users may want to check for the discount information of nearby restaurants or the availability of bars in the vicinity after class. To enjoy the service data provided by particular LBS applications such as Yelp, each user needs to initiate a query to Yelp which includes his current location and the queried interest. This solution has some drawbacks that motivate our work. First, since the submitted queries include the location of users, each user's location is revealed to the untrusted LBS server. Second, if a user frequently submits queries to the LBS server, many of his locations are exposed,

and thus his moving path can be easily identified by the LBS server. Third, the LBS server needs to reply to every user, even though some users have queried for the same interest. Obviously, this solution has low efficiency.

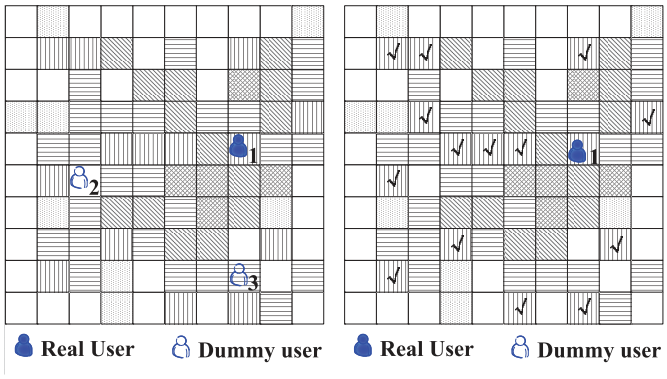
For the first problem, a straightforward method is to use *k-anonymity*, which hides user's real location into $k-1$ other locations. This aim can be achieved either by collecting real queries from nearby users or selecting dummy locations. Collecting $k-1$ real queries from neighbors is easy, but the locations in these queries may be very close to each other, and the LBS server can easily infer that the real user is in the small area surrounded by these locations. It is a kind of privacy leakage [28], [11]. Selecting dummy locations can solve this problem effectively, but the improperly selected dummy locations may fall into some unlikely places such as lakes, swamps, and rugged mountains, and can be easily filtered out by the adversary with *side information*. Therefore, how to select proper dummies is a challenge.

For the second problem, the service data obtained from previous queries can be cached to answer other future queries, since it has been shown that most LBS data has a long lifetime [12], [13], [14]. This can reduce the number of queries sent to the LBS server, and make it more difficult to track a user's moving path. Note that caching also addresses the third problem. However, it is not easy to determine which data to cache from the perspective of cache hit ratio.

To address the aforementioned problems, our basic idea is to integrate dummy and caching, through caching the service data obtained for dummy locations (as well as the real location). Dummies protect privacy for queries sent to the LBS server, while caching further enhances privacy through reducing the number of queries exposed to the LBS server. To maximize the location privacy against adversary with *side information* and improve the caching performance, we design caching-aware dummy selection schemes which carefully select dummy locations in two main steps. First, we choose dummy locations which have similar query probability as the real location. Second, when we have a set of candidates, we prefer to choose dummy locations which can bring more contributions to caching.

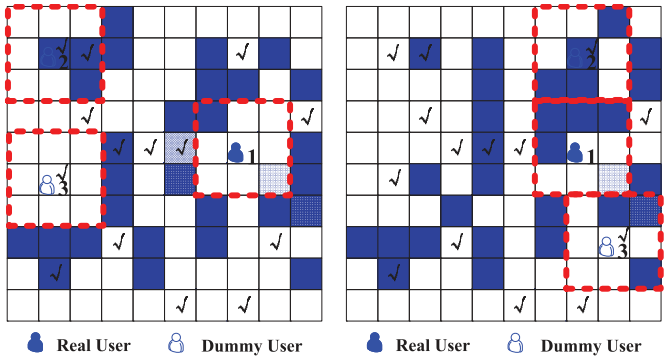
Fig. 1 further illustrates our basic idea, which focuses on two main factors, privacy (shown in Fig. 1(a) and 1(b)) and caching (shown in Fig. 1(c) and 1(d)). In Fig. 1(a) and 1(b), different shades of the cells represent different query probabilities, and cells marked with $\surd$ indicate that these cells are candidates of dummy locations. In Fig. 1(c) and 1(d), the shaded cells represent cached cells for which service data have been cached, while the blank cells have not been cached. Different gray degree means different data freshness of the cached service data in each cell. For example, darker cells represent that the service data cached on these cells is still fresh, and lighter cells mean that these service data may become out of date soon. The areas within the red dotted rectangles mean the user's query regions based on the current location and the query range.

In the first step, to effectively provide *k-anonymity* against



(a) Two dummy locations chosen by existing k-anonymity based approach

(b) 14 candidates marked with $\surd$ in our *CaDSA* considering side information



(c) A straightforward way (*CaDSA*) to obtain as many un-cached cells as possible by dummy locations

(d) Our *enhanced-CaDSA* to improve the caching performance by considering normalized distance and data freshness

Fig. 1. Our main idea to achieve 3-anonymity

adversaries with *side information*, we prefer to assign dummy locations into cells with similar query probabilities (shown in Fig. 1(b)), rather than randomly select dummy locations (shown in Fig. 1(a)). In the second step, based on the obtained 14 candidates marked with $\surd$ in Fig. 1(b), the goal of our Caching-aware Dummy Selection Algorithm (*CaDSA*) is to improve the caching performance. Therefore, we prefer to select proper dummy locations which can contribute more to caching. Fig. 1(c) shows an optimal case where all the chosen dummy locations have made maximum contributions (6 blank cells within the queried region) to the cache. As the result, both user privacy and cache hit ratio can be improved. However, *CaDSA* may not perform well in some cases due to two important factors: 1) users usually query for service data nearby, and 2) frequently queried cells in cache should be updated (i.e., cached again) before being expired. We consider these two factors to better evaluate dummies' contributions to caching. This further improves cache hit ratio as well as privacy since higher cache hit ratio means less queries sent to the LBS server. The chosen dummy locations can be found in Fig. 1(d). Here, we prefer dummy locations which are nearby and whose service data is out of date.

C. Location Privacy Metrics

There are many metrics to measure privacy, such as entropy-based metrics [29] and distortion-based metrics [30]. Entropy

can be seen as the uncertainty in determining the real location of a user from all the candidates, and it has been widely used in the literature. Based on entropy, we define two privacy metrics to measure location privacy. One metric measures the privacy degree achieved for a user when he sends a query to the LBS server. Since some queries are answered by cache, the other metric also takes the effect of caching into consideration and measures the overall privacy achieved in our system.

1) Individual privacy metric: To provide privacy protection for users who need to send queries to the LBS server, our schemes use dummy locations to achieve *k-anonymity* even when the LBS server has *side information*. In this paper, *side information* is the probability that a location can be queried (i.e., service data is requested for this location). Specifically, we divide the map area into $N \times N$ cells. Each cell has a probability of being queried (called *query probability*) which is proportional to the number of times the location was queried in the past. Let q denote this probability. Then we have $\sum_{i=1}^{N^2} q_i = 1$. For the k locations (i.e., cells) contained in a query which include one real location and $k-1$ dummies, each location has a conditional probability of being the real location. Let p_i $(i = 1, 2, ..., k)$ denote the probability that the i^{th} location is the real location. Then $p_i = \frac{q_i}{\sum_{j=1}^{k} q_i}$, and obviously $\sum_{i=1}^{k} p_i = 1$. The entropy H of identifying the real location out of the anonymity set is defined as

$$H = -\sum_{i=1}^{k} p_i \cdot \log_2 p_i. \tag{1}$$

Since larger H means higher privacy degree, our aim is to achieve the maximum entropy. It happens when all the k possible locations have the same query probability and the maximum entropy is $H_{max} = \log_2 k$.

2) Global privacy metric: The privacy metric defined in Equation 1 describes the privacy degree achieved when a user sends a query to the LBS server, but it does not consider the effect of caching on the overall privacy degree. Actually, caching improves privacy since some queries do not have to be sent to the LBS server. To capture this effect, we define another privacy metric.

Let us consider all the queries in the system. For a query answered by the LBS server, the uncertainty of the real location is calculated using Equation 1. For a query answered by cache, the LBS server obtains no information about the user's real location from this query, and every cell may be the real location. Then the uncertainty of the real location is defined as $\log_2 N^2$, which can be obtained with Equation 1 assuming that every cell is equally likely to be the real location. Let $\mathcal{Q}_{cache}$ denote the set of queries which are serviced by cache, and $\mathcal{Q}_{server}$ denote the set of queries which are sent to and serviced by the LBS server. Then our second privacy metric is defined as the *average uncertainly* (denoted by λ) of the real location in each query out of all queries:

$$\lambda = \frac{\sum_{q \in \mathcal{Q}_{server}} H_q + \log_2 N^2 \times |\mathcal{Q}_{cache}|}{|\mathcal{Q}_{server}| + |\mathcal{Q}_{cache}|} \tag{2}$$

where H_q is the uncertainty of the real location in query q calculated using Equation 1. Since cache hit ratio $\gamma = \frac{|\mathcal{Q}_{cache}|}{|\mathcal{Q}_{server}|+|\mathcal{Q}_{cache}|}$, the above formula can be rewritten as

$$\lambda = \frac{\sum_{q \in \mathcal{Q}_{server}} H_q}{|\mathcal{Q}_{server}| + |\mathcal{Q}_{cache}|} + \gamma \cdot \log_2 N^2. \tag{3}$$

It can be seen that there are two ways of increasing λ. One is to increase the entropy of each query $q \in \mathcal{Q}_{server}$. The other is to increase cache hit ratio γ, such that more users can be directly served by cache and enjoy higher privacy degree (note that $\log_2 N^2 > \log_2 k \geq H_q$). Our system considers these two factors to protect location privacy.

IV. Our Proposed Schemes

In this section, we first present an overview of the proposed system. Then we present our Caching-aware Dummy Selection Algorithm (*CaDSA*) and the *enhanced-CaDSA*, followed by security analysis and discussion of implementation issues.

A. System Overview

When a user needs LBS, he first queries our cache system (we will show how to implement it in Section IV-F later). If there is no match or the matched service data cannot satisfy his requirements, the user sends a query to the LBS server to request service data for his real location and $k-1$ dummy locations, and the obtained data will be used to update cache. Such a user is called a *data contributor*, since he contributes data to cache. If the user's requirement can be satisfied by the cached service data, he does not need to send a query and reveal location information to the untrusted LBS server, and simply gets service data from cache. Such a user is called a *data consumer* since he consumes data from cache. For *data contributors* who submit queries to the LBS server, privacy is protected by our dummy selection algorithms which guarantee *k-anonymity*; for *data consumers*, privacy is protected by caching since they do not have to send queries to the LBS server and hence no location is revealed.

In our system, dummy selection affects the achieved privacy in two ways. First, for a particular query sent to the LBS server, dummy selection determines the privacy degree that the querying user can enjoy for this query. Second, since the service data obtained for dummy locations will be cached, dummy selection affects the cache hit ratio and in turn determines the achieved privacy degree. Considering these factors, we propose two dummy selection algorithms, *CaDSA* and *enhanced-CaDSA*.

To answer a query, cache does not need to have data for all the cells within the query range in our system. Instead, caching a high percentage of cells suffices. Specifically, each user can set a threshold τ which means the lowest fraction of cells that should be cached to answer his query. Let parameter ξ denote the percentage of cells that have actually been cached within the query range. Then a user's query can be answered by cache if $\xi \geq \tau$, but the user must send the query to the LBS server if $\xi < \tau$. By tuning τ, a user can tune the tradeoff between service quality and privacy.

B. A Cache-Oblivious Baseline Algorithm

Before describing our advanced dummy selection algorithms, we first present a baseline algorithm for comparison purposes. In this baseline algorithm, dummies are selected to maximize the entropy for the current query only, without considering dummies' effect on cache hit ratio:

$$Max(-\sum_{i=1}^{k} p_i \cdot \log_2 p_i). \tag{4}$$

This algorithm is the same as the basic scheme proposed in [11] (we omit the details here due to the space limitation), except that here the data obtained for real and dummy locations is cached.

C. Caching-Aware Dummy Selection Algorithm

Inspired by Equation 3, our main idea is to select a set of realistic dummy locations to ensure high entropy for the current query and at the same time provide more contributions to cache hit ratio. Intuitively, a dummy location's contribution to cache is mainly determined by the query probability of this location. If the query probability of a location is high, the data for this location is more likely to serve future queries, and can achieve higher cache hit ratio. Clearly, a dummy with high query probability has more contributions to cache than a dummy with low query probability. Let δ denote a dummy's contribution to cache. We define δ as

$$\delta = q \cdot g \tag{5}$$

where $g = 0$ if this location is already cached and $g = 1$ otherwise.

Since two objectives are considered, we formulate the dummy selection problem as a Multi-Objective Optimization Problem (MOP), which can be described as

$$C_{dummy} = \arg\max\{-\sum_{i=1}^{k} p_i \cdot \log_2 p_i, \sum_{i=1}^{k} \delta_i\}, \tag{6}$$

where the first objective ($\max\{-\sum_{i=1}^{k} p_i \cdot \log_2 p_i\}$) is to provide higher privacy degree for the current query and the second objective ($\max\{\sum_{i=1}^{k} \delta_i\}$) is to guarantee a better cache hit ratio. Since it is hard to satisfy all objectives at the same time, we solve the MOP in two steps, optimizing one objective function in each step. We first select a set of candidate dummies which can achieve high entropy for the current query:

$$C_c = \arg\max(-\sum_{i=1}^{k} p_i \cdot \log_2 p_i). \tag{7}$$

Then from these candidates, we further select $k-1$ dummy locations which can contribute most to cache:

$$C_{dummy} = \arg\max \sum_{i=1}^{k} \delta_i. \tag{8}$$

Algorithm 1 illustrates our idea in details. We first select $4k$ cells (this number can be adjusted based on privacy needs and allowable computation cost) which have similar query probabilities to the user's real location c_r, and randomly select $2k$ of them as candidate cells. Due to the definition of entropy in Equation 1, these candidate cells can achieve a high entropy for the current query. Out of these $2k$ cells, we further select a subset of $k-1$ dummies which have the highest contribution to cache. When k is large, since the number of subsets $\binom{2k}{k-1}$ is too large, we only consider s random subsets and select one of them which has the highest contribution to cache. Here s is a system parameter and $s = 1000$ by default. Note that, bigger s leads to higher computation cost but higher privacy degree.

Algorithm 1: Caching-Aware Dummy Selection Algorithm (CaDSA)

Input : q (query probability of each cell), c_r (real location), s (a system parameter)
Output: C_{dummy}

```
sort cells based on their query probability q;
choose 4k cells (2k cells are right before c_r and 2k cells are right after c_r in the sorted list);
randomly select 2k cells out of them as the candidate set C_c;
Ĉ_c = ∅;
if C(2k, k-1) ≤ s then
    Ĉ_c = {C' | C' ⊂ C_c & (|C'| = k-1)}
end
else
    Generate s subsets of C_c with k-1 random dummies in each subset;
    Add these subsets to Ĉ_c;
end
for each C' in Ĉ_c do
    compute Σ_{i=1}^{k} δ_i for the k-1 dummies in C';
end
output C_dummy = arg max_{C' ⊂ Ĉ_c} Σ_{i=1}^{k} δ_i
```

D. Enhanced Caching-Aware Dummy Selection Algorithm

Besides query probability, we identify two other important factors which affect dummy locations' contributions to cache. They are *normalized distance* and *data freshness*. This algorithm considers these factors into dummy selection to further improve cache hit ratio and achieved privacy degree.

Normalized Distance. We consider the effect of the distance between the real location and dummy location. Since users usually query for POIs in vicinity and the data retrieved from the LBS server is cached locally around the real location, it is not very useful to cache cells far away from the real location. Thus, we prefer to select dummy locations not very far away from the real location to maintain a good cache hit ratio. We define the normalized distance between the real location l_r and the i^{th} dummy location l_i as

$$d_i = d(l_r, l_i) \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{(d - d(l_r, l_i))^2}{2}}, \tag{9}$$

where $d(l_r, l_i)$ denotes the physical distance between l_r and l_i, and $d = \frac{\sum_{i=1}^{k} d(l_r, l_i)}{k}$.

Algorithm 2: Enhanced Caching-Aware Dummy Selection Algorithm (enhanced-CaDSA)

Input : q (query probability of each cell), c_r (real location), s (a system parameter)
Output: C_{dummy}
run line 1-11 in Algorithm 1;
for *each* C' *in* $\hat{C}_c$ **do**
  compute Δ for the $k-1$ dummies in C';
end
outputs $C_{dummy} = \arg\max_{C' \subset \hat{C}_c} \Delta$

We use the total normalized distance D ($D \in [0,1]$) to describe the effect of all the $k-1$ dummy locations on the caching performance. It is defined as

$$D = \prod_{i=1}^{k-1} \sqrt{2\pi} \frac{d_i}{d(l_r, l_i)}. \quad (10)$$

Data Freshness. Since the data cached for a cell may become out of date, we prefer to update the cached data before it expires, especially for cells with high probability of being queried. For example, suppose the lifetime of cached data is 6 hours. Then the freshness of the data which has been cached for 5.5 hours is much worse than the data cached for 0.5 hours. More formally, let T denote the lifetime of cached data, and t denote the time that a cell's data has already been cached. Then the freshness value of the cell, denoted by f, is defined as

$$f = \sqrt{1 - \frac{t^2}{T^2}},\ t \leq T. \quad (11)$$

For a query submitted to the LBS server, the service data obtained for each location (being it the real location or a dummy location) covers multiples cells around this location when the query range is larger than a cell. Considering this, we compute the average data freshness (F) of those cells covered by the k submitted locations as

$$F = \frac{\sum_{i=1}^{l \cdot k} f_i}{l \cdot k}, \quad (12)$$

where f_i is the freshness value of cell i, and l indicates the number of cells within the query range.

Finally, a set of dummy locations' total contribution to cache Δ can be defined as

$$\Delta = (\sum_{i=1}^{k} \delta_i) \cdot (1 - D) \cdot (1 - F). \quad (13)$$

Algorithm 2 illustrates our idea in details. The algorithm is quite similar to Algorithm 1, expect that in the last four lines we select the set of dummies to optimize Δ.

E. Security Analysis

Since standard cryptography techniques such as encryption and decryption can be easily used upon our algorithms to deal with *eavesdropping attacks* on the wireless channel between users and other entities, in this section, we focus on analyzing the privacy of *data contributor* against *inference attacks* performed by the untrusted LBS server.

In our scheme, *data contributors* need to send queries to the untrusted LBS server to enjoy service data when their requirements cannot be satisfied by cache. Ideally, due to *k-anonymity*, the LBS server cannot distinguish the real location of a particular user from the other $k-1$ dummy locations. The probability of successful guessing should be $\frac{1}{k}$. However, the LBS server knows the query probabilities of the whole map and the historical queries of each user (each query containing the user's identifier, a mix of real and dummy locations, queried POI, query range, etc.). It can perform *inference attacks* based on these information. More formally, the information includes query probability $q_i, (0 < i \leq N^2)$ of each cell, query interest I, all the submitted k locations $l_1, l_2, \cdots, l_r, \cdots, l_k$ with their corresponding cells $c_1, c_2, \cdots, c_r, \cdots, c_k$. The LBS server's goal is to improve its probability of successfully guessing the real location from the submitted query set. In our algorithms, the *inference attack* is avoided by using randomization. Let us consider *enhanced-CaDSA* as an example. We first select $4k$ cells and then construct a candidate set C_c with $2k$ elements randomly chosen from the $4k$ cells. Further, the constructed candidate set should be refined into C' with $k-1$ elements, which improves the randomization. As a result, any combination of k locations may be selected from the $4k$ cells, which have similar query probability. Thus, the LBS server cannot use such information to improve its probability of guessing the real location. In another attack, the adversary may learn the algorithm details of *CaDSA* and *enhanced-CaDSA*. Then it can assume each of the k locations to be the real location, run our algorithms to select dummies, and compare if the obtained dummies are the same as those included in the query. If so, it concludes that the assumption is correct which means identification of the real location. However, due to the introduced randomization, different real locations may lead to the same set of k locations including one real and $k-1$ dummies. Thus, such assumption-and-verification attack will fail.

In another attack, the adversary guesses the real location as the one in the central part among the submitted k locations. However, this attack cannot improve the probability of guessing the real location due to our use of normalized distance and data freshness. In Equation 10, we use multiplication to compute the total normalized distance, which guarantee that the selected dummy locations cannot be too close to each other (this property is also proved in [11]). Although we prefer to cache nearby cells over others, data freshness helps balance the distribution of the dummy locations. As a result, cells around the real location may not always be selected as the dummy locations.

F. Implementation Issues

In our *CaDSA* and *enhanced-CaDSA*, *side information* is assumed to be known to the mobile user. Additionally, the

cached service data should be stored and disseminated properly. In this subsection, we address the issues of how to obtain *side information* and where to store cached data.

We implement our ideas through utilizing WiFi Access Points (APs), which have been used for LBSs in many other works [26], [27], [11]. In our implementation, each AP acts as the local cache for mobile users connected to it. Each AP also records the number of queries issued from each cell within its range. Here such information is also called *side information* for convenience since query probability can be derived from it. Different APs exchange their cached data and *side information* with the help of mobile users, such that each AP can acquire data and *side information* of a larger area. To achieve this, mobile users downloads cached data and *side information* from each encountered AP and uploads them to other encountered APs. Specifically, when a user needs LBS, he communicates with the local AP to see if this AP caches the needed data. When receiving this query, the AP updates its *side information* by increasing the number of queries from the user's cell by one. If the AP caches the requested data, it answers the query using cached data; otherwise, the user sends a query to the LBS server, obtains data for the real and dummy locations, and adds the data to the AP's cache. To accelerate dissemination of cached data and *side information*, the user and the AP also exchange their collections of cached data and *side information*. Note that users can communicate with APs anonymously to protect privacy against APs.

Note that, our schemes can work even when there is no AP around a user, since the user stores some *side information* and cached service data locally. As the result, he can either use the cached service data locally or select $k-1$ dummy locations properly based on the *side information* in hand.

Our schemes work for each POI independently and it can serve many POIs in parallel. When serving many POIs, the storage cost may be high. To reduce the cost, we can utilize some efficient storage techniques. For example, the top-k query scheme [31] can be applied on top of our *CaDSA* and *enhanced-CaDSA* to reduce the storage cost.

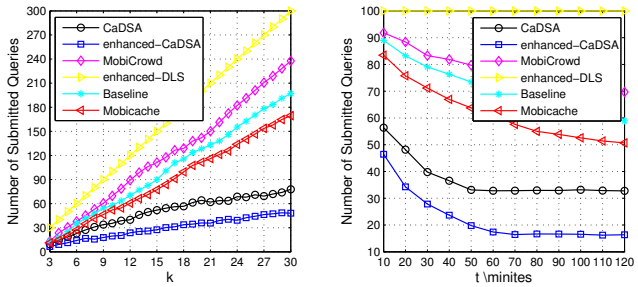
V. Performance Evaluations

A. Simulation Setup

To evaluate the performance of our proposed algorithms, we follow the simulation setting in our previous work [11] and deploy 10000 mobile users into a $8km \times 8km$ area map of New York city, which is shown in Fig. 2(a). Users follow the Levy walk mobility model [32] which has been shown to be a realistic model of describing human mobility [33], [34]. The local map is divided into 160×160 cells, and the length of each cell is 50m. The initial query probability information in this area is obtained from Google Maps API. In every 1 minute, we choose 10 users to issue a request for LBS service. Without loss of generality, only one POI is considered in the simulations. The probability that a user is chosen is proportional to the query probability of the cell where the user is currently located. Red dashed circles in Fig. 2(a) represent WiFi access points with their communication ranges. These

(a) Map of New York City (b) APs in OPNET simulator

Fig. 2. Simulation settings



(a) Communication cost vs. k, t = 120 (b) Communication cost vs. t, k = 10 minutes

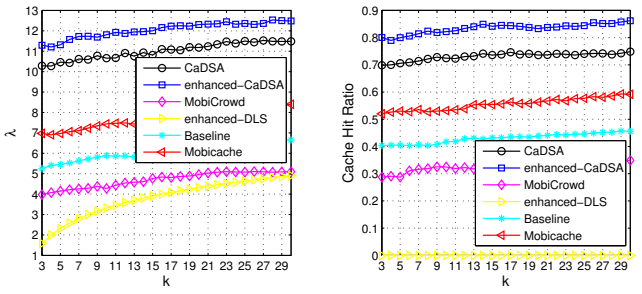
Fig. 3. The communication cost, τ = 0.8

APs are located at popular places, such as downtown areas of NYC, Brooklyn and New Jersey, shopping malls and bars, since these places usually have more users around and are covered by APs in reality. Fig. 2(b) indicates the distribution of the APs in our simulator.

In our following experiments, k is related to *k-anonymity*, which means the degree of anonymity, t is the evaluation time. τ is defined by our cache system. We compare our proposed *CaDSA* and *enhanced-CaDSA* with three existing schemes, *enhanced-DLS* which represents the dummy selection algorithm in [11], *MobiCrowd* [13], and *Mobicache* [14]. Note that *enhanced-DLS* does not use caching but *MobiCrowd* and *Mobicache* use caching. The baseline scheme described in Sec. IV-B is also compared to better understand our schemes.

B. Evaluation Results

1) Communication cost: Communication cost in our schemes should be considered from two aspects, short-range communications and 3G/4G data traffic. We pay much attention to the 3G/4G data traffic due to the extra data fee. Fig. 3 shows the effects of k and simulation time t on the number of queries submitted to the LBS server. In Fig. 3(a), we choose a snapshot when the simulation time is 120 minutes and $\tau = 0.8$. The number of submitted queries in *enhanced-DLS* [11] increases linearly with k since it does not consider caching at all. *MobiCrowd*, *Mobicache*, and the baseline scheme perform better than *enhanced-DLS* due to use of caching. However, many queries are still sent to the LBS server due to their poor cache design. Our *CaDSA* performs much better than all of them since it carefully selects the dummies which have higher contribution to cache. Our *enhanced-CaDSA* performs



(a) λ vs. k (b) Cache hit ratio vs. k

Fig. 4. Effects of k on privacy and cache, τ = 0.8, t = 120

even better due to the consideration of normalized distance and data freshness which affect caching hit ratio a lot. Similar results can be found in Fig. 3(b). These results show that our schemes make a good utilization of the service data retrieved from the LBS server and thus decrease the number of queries sent to the LBS server through cellular networks. Therefore, the communication cost is significantly reduced compared to previous solutions [13], [11], [14].

2) Storage cost: The storage cost in our scheme can be analyzed in three parts, the query probability, the distribution of cached service data as well as the cached service data. The former two are closely related to the number of cells divided. In our implementation, their storage cost are 22.8 KB and 53.7 KB on average when the simulation runs for 4 hours for a grid with 160×160 cells, respectively. Under this setting, each user/AP almost obtains all the query probabilities in the map. The storage cost of the cached service data is affected a lot by the number of the POIs in a particular area. In our work, each POI in the storage is recorded in the format of $\{\langle longitude, latitude\rangle, POI, timestamp\}$. Suppose the data for each POI is of less than 1KB (e.g., bank name, location, and a brief description). Then we can evaluate the storage requirement of our scheme given the number of POIs in a map. For example, there are roughly 250,000 POIs in New York City (NYC) [12]. Then the total storage cost is $250,000 \times 1\ KB = 250$ MB for a city as big as NYC. Such storage cost is not a big issue for modern computers and smart phones.

3) Effects of k on privacy and cache: Fig. 4(a) shows the effects of k on the average uncertainty λ. Obviously, the privacy degree of *enhanced-DLS* [11] is the worst since it does not use caching. *MobiCrowd* [13] performs better since it caches the obtained service data to serve other users. However, the improvement is not that much due to poor caching design. The results of the baseline scheme show a slightly higher average uncertainty since the data for dummy locations is also cached. *Mobicache* [14] performs a little better. Our *CaDSA* and *enhanced-CaDSA* have much higher privacy degree than all those schemes due to our more advanced design of caching-aware dummy selection. In particular, *enhanced-CaDSA* performs better than *CaDSA* due to consideration of normalized distance and data freshness in caching design.

Fig. 4(b) further explains how k affects cache hit ratio. *enhanced-DLS* [11] does not consider caching, and hence its cache hit ratio is 0. Since *MobiCrowd* [13] does not use any dummy locations, its cache hit ratio stays at a lower level around 32%. This value increases to 43% in the baseline scheme due to dummy locations' contribution to cache. While in our schemes, dummy selection optimizes caching performance as one objective, and such caching-aware dummy selection achieves a much higher cache hit ratio than the baseline scheme.

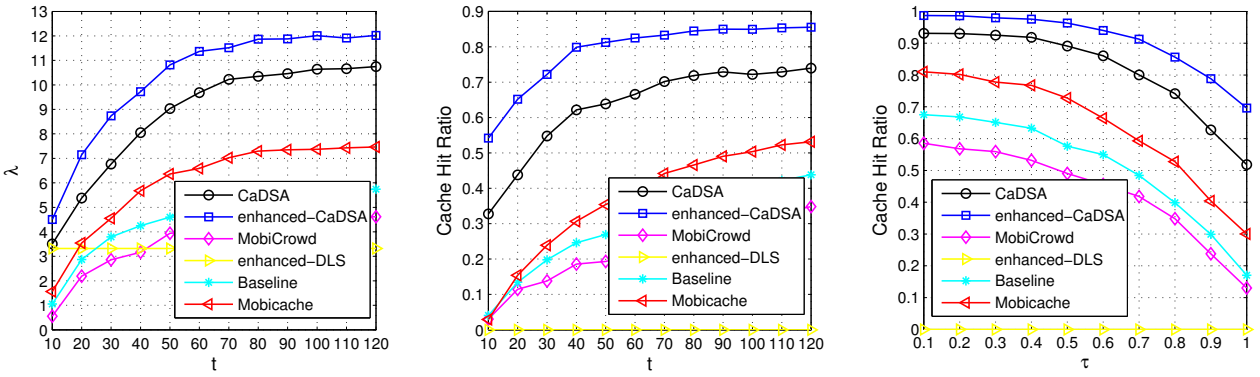
4) Effects of t on privacy and cache: We also evaluate the impact of the simulation time t on the average uncertainty λ, which is shown in Fig. 5(a). The average uncertainty increases with time in most of schemes except *enhanced-DLS* [11]. The reason is that, as time goes by, more data is cached and less queries are sent to the LBS server. Users in *MobiCrowd* [13] send queries to the LBS server without any privacy protection when their requirements cannot be satisfied by cache, and thus the average uncertainty is low. *Mobicache* [14] does not consider the *side information*, and thus the contribution to average uncertainty is limited. Our *CaDSA* and *enhanced-CaDSA* outperform all other schemes in all the tested cases since they can cache more useful data as time goes by.

These results are further explained by Fig. 5(b), which shows how cache hit ratio changes with simulation time t. We can see that the cache hit ratio in *MobiCrowd*, *baseline*, *Mobicache*, *CaDSA*, and *enhanced-CaDSA* increases with t since more data is cached as time goes by, but the cache hit ratio of *enhanced-DLS* [11] is zero due to its ignorance of caching. In all cases, our *CaDSA* and *enhanced-CaDSA* have higher cache hit ratio than the other schemes.

5) Cache hit ratio vs. τ: Fig. 5(c) shows how cache hit ratio changes with τ. Generally speaking, cache hit ratio drops with the increasing τ, since more data must be cached to answer a query. Since the contributions provided by the obtained service data are from both the real location and the carefully selected dummy locations, our schemes perform much better than *MobiCrowd*, *Mobicache* and the baseline scheme. For example, when $\tau = 0.8$, the cache hit ratios are only 34%, 40% and 53% in *MobiCrowd*, the baseline scheme and *Mobicache* respectively, but the cache hit ratios in *CaDSA* and *enhanced-CaDSA* are 74% and 86%, respectively. Compared to *CaDSA*, our *enhanced-CaDSA* improves a lot on the cache hit ratio due to consideration of more factors affecting caching performance.

VI. Conclusions

Considering the impact of caching on user privacy in LBSs, we proposed two caching-aware dummy selection algorithms to improve user's location privacy. The first algorithm *CaDSA* achieves *k-anonymity* effectively by selecting some candidate cells with similar query probabilities, and then improves caching by determining the optimal set of dummies which contributes most to cache hit ratio. The second algorithm *enhanced-CaDSA* considers a more comprehensive set of factors that affect caching performances, including normalized distance and data freshness, and further improves caching hit ratio as well as the overall privacy. Evaluation results indicate



(a) λ vs. t, τ = 0.8, k = 10 (b) Cache hit ratio vs. t, τ = 0.8, k = 10 (c) Cache hit ratio vs. τ, k = 10, t = 120

Fig. 5. Evaluation Results

that our proposed algorithms achieve much better privacy than previous solutions.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (U1401251, 61170251, 61402354), the National High Technology Research and Development Program of China (863 Program) (2012AA013102), the Fundamental Research Funds for the Central Universities (K5051201041), and China 111 Project (B08038).

REFERENCES

[1] L. Sweeney, "k-anonymity: a model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[2] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. of ACM MobiSys 2003*.

[3] T. Xu and Y. Cai, "Feeling-based location privacy protection for location-based services," in *Proc. of ACM CCS 2009*.

[4] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proc. of IEEE ICPS 2005*.

[5] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *Proc. of ACM VLDB 2006*.

[6] A. Pingley, W. Yu, N. Zhang, X. Fu, and W. Zhao, "Cap: A context-aware privacy protection system for location-based services," in *Proc. of IEEE ICDCS 2009*.

[7] H. Lu, C. S. Jensen, and M. L. Yiu, "Pad: privacy-area aware, dummy-based location privacy in mobile services," in *Proc. of ACM MobiDE 2008*.

[8] A. Pingley, N. Zhang, X. Fu, H.-A. Choi, S. Subramaniam, and W. Zhao, "Protection of query privacy for continuous location based services," in *Proc. of IEEE INFOCOM 2011*.

[9] J. Manweiler, R. Scudellari, and L. P. Cox, "Smile: encounter-based trust for mobile social services," in *Proc. of ACM CCS 2009*.

[10] H. Hu and J. Xu, "Non-exposure location anonymity," in *Proc. of IEEE ICDE 2009*.

[11] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving k-anonymity in privacy-aware location-based services," in *Proc. of IEEE INFOCOM 2014*.

[12] S. Amini, J. Lindqvist, J. Hong, J. Lin, E. Toch, and N. Sadeh, "Cache: Caching location-enhanced content to improve user privacy," in *Proc. of ACM MobiSys 2011*.

[13] R. Shokri, G. Theodorakopoulos, P. Papadimitratos, E. Kazemi, and J.-P. Hubaux, "Hiding in the mobile crowd: Locationprivacy through collaboration," *Dependable and Secure Computing, IEEE Transactions on*, vol. 11, no. 3, pp. 266–279, May 2014.

[14] X. Zhu, H. Chi, B. Niu, W. Zhang, Z. Li, and H. Li, "Mobicache: When k-anonymity meets cache," in *Proc. of IEEE GLOBECOM 2013*.

[15] Z. Zhu and G. Cao, "Applaus: A privacy-preserving location proof updating system for location-based services," in *Proc. of IEEE INFOCOM 2011*.

[16] K. Shin, X. Ju, Z. Chen, and X. Hu, "Privacy protection for users of location-based services," *Wireless Communications, IEEE*, vol. 19, no. 1, pp. 30–39, 2012.

[17] Q. Li and G. Cao, "Efficient privacy-preserving stream aggregation in mobile sensing with low aggregation error," in *ACM PETS 2013*.

[18] ——, "Providing efficient privacy-aware incentives for mobile sensing," in *IEEE ICDCS 2014*.

[19] W3C. (2011, Apr.) Platform for privacy preferences (p3p) project. [Online]. Available: http://www.w3.org/P3P/

[20] I. Bilogrevic, M. Jadliwala, K. Kalkan, J.-P. Hubaux, and I. Aad, "Privacy in mobile computing for location-sharing-based services," in *Proc. of the 11th Privacy Enhancing Technologies Symposium (PETS) 2011*.

[21] B. Niu, Q. Li, X. Zhu, and H. Li, "A fine-grained spatial cloaking scheme for privacy-aware users in location-based services," in *Proc. of IEEE ICCCN 2014*.

[22] B. Niu, X. Zhu, W. Li, and H. Li, "Epcloak: An efficient and privacy-preserving spatial cloaking scheme for lbss," in *Proc. of IEEE MASS 2014*.

[23] C.-Y. Chow, M. F. Mokbel, and W. G. Aref, "Casper*: Query processing for location services without compromising privacy," *ACM Trans. Database Syst.*, vol. 34, no. 4, 2009.

[24] C. Y. Ma, D. K. Yau, N. K. Yip, and N. S. Rao, "Privacy vulnerability of published anonymous mobility traces," in *Proc. of ACM MobiCom 2010*.

[25] B. Niu, X. Zhu, W. Li, H. Li, Y. Wang, and Z. Lu, "A personalized two-tier cloaking scheme for privacy-aware location-based services," in *Proc. of IEEE ICNC 2015*.

[26] S. Saroiu and A. Wolman, "Enabling new mobile applications with location proofs," in *Proc. of ACM HotMobile 2009*.

[27] W. Luo and U. Hengartner, "Veriplace: a privacy-aware location proof architecture," in *Proc. of ACM GIS 2010*.

[28] R. Kato, M. Iwata, T. Hara, A. Suzuki, X. Xie, Y. Arase, and S. Nishio, "A dummy-based anonymization method based on user trajectory with pauses," in *Proc. of ACM SIGSPATIAL 2012*.

[29] A. Serjantov and G. Danezis, "Towards an information theoretic metric for anonymity," in *Proc. of the 3rd Privacy Enhancing Technologies Symposium (PETS) 2003*.

[30] B. Hoh and M. Gruteser, "Protecting location privacy through path confusion," in *Proc. of IEEE SECURECOMM 2005*.

[31] Y. Zhang, Y. Zhang, and C. Zhang, "Secure top-k query processing via untrusted location-based service providers," in *Proc. of IEEE INFOCOM 2012*.

[32] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong, "On the levy-walk nature of human mobility," in *Proc. of IEEE INFOCOM 2008*.

[33] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "Slaw: A new mobility model for human walks," in *Proc. of IEEE INFOCOM 2009*.

[34] S. Yang, X. Yang, C. Zhang, and E. Spyrou, "Using social network theory for modeling human mobility," *Network, IEEE*, vol. 24, no. 5, pp. 6–13, 2010.