RuleCache: A Mobility Pattern Based Multi-Level Cache Approach for Location Privacy Protection

Qiwei Yang  
College of Computer Science and Electronic Engineering  
Hunan University  
Changsha, China  
yangqiuwei@hnu.edu.cn

Pan Kong  
College of Computer Science and Electronic Engineering  
Hunan University  
Changsha, China  
peterkong_520@163.com

Abstract—With the widespread pervasion of Location-Based Service (LBS), location privacy attracted more and more attention. In the traditional LBSs model, users need to repeatedly send their location information to the provider’s servers or an intermediate Anonymizing Server (AS) for service, which has a lot of disadvantages, such as the low reutilization ratio of the information, the high threat of un-trusted service providers and the excessive reliance on AS. The latest cache method gives a new idea, but it does not take users’ mobile behavior into account. To address these issues, this paper presents a multi-level location privacy protection method, RuleCache, which combines the users’ mobility patterns with cache and utilize the cache content of distributed neighbors to protect location privacy. Moreover, when the user has to send query to the LBS server, we propose a Cloaking Region Generating Algorithm (CRGA) to achieve protection which considers the factor of query probability and data timeliness to increase cache contribution rate and update outdated data in time, respectively. We carry out extensive simulation and evaluation show that our RuleCache have a higher performance than many other methods on communication cost, cache hit ration and storage cost.

Keywords—Location-Based Service; location privacy; cache; cache hit ratio; k-anonymity

I. INTRODUCTION

In recent years, with the rapid development of mobile positioning technology and wireless communication technology (4G), LBS applications widely spread. With LBS, a user can easily acquire interested location-related information anytime. However, in order to obtain the accurate and convenient service, all location-based services require users to send location information as accurately as possible to the LBS provider, which makes the providers can speculate and obtain the user’s current location, future location, hobbies and other private information. Moreover, they might maliciously use these data, such as disclosing it to advertisers, which causes great inconvenience or even threat to the users’ lives [1].

In fact, the location privacy has long attracted scholars’ attentions, it is defined as the ability to block adversary for inferring the user’s real location [2]. Many methods have been proposed to protect users’ location privacy, k-anonymity is the most classic one that is introduced by Marco Gruteser et al. [3] from the data privacy protection areas. The main idea of k-anonymity is to put a certain position corresponding to k different identity information, which prevents the adversary from matching the obtained location with the right user. However, with today’s data-mining techniques, adversary can easily infer the true identity based on location information [4][5]. In the method of using intermediate Anonymizing Server [6][7], AS is responsible to hide the user’s true position in a cloaking region, then the provider’s server cannot distinguish the real user’s location from other k-1 users’ in the same region. But using AS method has many defects. First, since all the submitted queries and returned messages have to go through AS, it has a serious problem of bottleneck. Secondly, it heavily relies on AS, which may suffer from a single point of attack. If the adversary gains access to it, the privacy of all users will be leaked. Lastly, all methods assume that AS is credible, but it just adds another service provider actually, this assumption itself is accompanied by risks. Though a lot of mobile device-based schemes have been presented [8]-[14], there exists common problems of frequent request, high communication and computation costs. What’s more, in these traditional protection methods, the inherent contradiction between protection level and service quality is difficult to strike a balance. We note that the user’s mobility pattern is a key factor when designing user’s location privacy protection, but few studies take it into account. In addition, a user in these systems may simply discard the queried service information after using it which can be utilized again. So, how to increase the reutilization ratio of the information should also be addressed.

Considering these problems, we propose a new method, RuleCache, which combines the existing caching method together with users’ mobility patterns, and also utilize space k-anonymity to achieve privacy protection. The main purpose is to enhance the protection while maximizing the cache hit ratio and decreasing the communication and storage costs. The contributions of this paper are shown as follows.

- Multi-level cache protection mechanism. We designed a distributed multi-level cache location privacy protection method, RuleCache, which does not rely on trusted third party. It achieves protection on three levels: the local host, the neighbors’ collaboration and the LBS server.

- Cache standardization. According to the inherent characteristics of user mobility pattern, we design a
middleware called cacheware to control the contents of the terminal cache, which makes cache content become standardized for the first time. It optimizes the cache storage by no longer store unnecessary information. And the neighbor users can take their interested information through cacheware with collaboration.

- Cloaking Region Generating Algorithm. When the user has to send a request to the untrusted server, we propose a Cloaking Region Generating Algorithm (CRGA) to protect user’s privacy. We considered two factors, query probability and data timeliness. Query probability is utilized to increase cache hit ratio and enhance privacy security. Data timeliness is used to evaluate data freshness and help update outdated data.

- We conduct some security analysis of the proposed method, and carry out a comparative simulation with other representative methods. The results show that the communication cost of our system decreased by about 20% compared to other methods and the cache hit ratio also has a better performance than others. The storage cost is significantly low.

The rest of this paper is organized as follows. In Section II, we introduce the related work. Section III presents the motivation of our design. Following in Section IV, we describe the details of our proposed system. Then, the security analysis and performance evaluation are provided in Section V and VI, respectively. Finally, we draw the conclusions in Section VII.

II. RELATED WORK

Privacy protection has become a hot topic in recent years, and for location privacy protection, space anonymity is the most important method. We will review the space anonymity method and the cache method which is hot lately in this field and will be utilized in our algorithm.

A. Space Anonymity

Space anonymity method is to convert the exact location of the user to a spatial region which includes that exact location. And when user’s query is sent to the LBS server, the server is unable to know the real location of the user based on that cloaking region. Duckham and Kulik [15] first put forward the obfuscation as a mechanism for the location privacy protection. But it can only hide the location information of individual user and lack flexibility when a large number of users are concurrently searching. Mokbel [16] combined k-anonymity and space anonymity and introduced a trusted third party (AS) to handle a large number of queries. AS instead of the user queries information and return it to the user. However, this method poses a high performance requirements for the AS, and once AS become unreliable, the threat of privacy leakage could be great. Ghinita et al.[17][18] proposed a distributed architecture, which achieves mutual communication between users through some fixed communication infrastructure such as base station, by which the user finds other required k-1 anonymous users and forms the cloaking region. This method avoids the problems caused by AS. Chow et al. [19] presented a P2P location anonymity algorithm. In their work, users exchange information via broadcast instead of fixed communication infrastructure. The user finds anonymous neighbors through a single or multi-hop fashion and forms cloaking region, and then chooses one neighbor as the proxy, the proxy queries and returns information. This method solves many problems of location privacy protection, but the disadvantage is that the consumption of network bandwidth is larger when a large number of users query concurrently.

B. Cache Methods

The previous works always limited focus on the process of sending request to the service provider to protect privacy, in recent years, the experts began to reduce the number of requests sent to the provider to improve security and service quality. Because the number of requests sent to the provider is less, the less the user's privacy information is exposed, and the lower the chances that the adversary succeeds in obtaining the user's privacy. The cache method use the cached data obtained from prior queries to answer future queries.

Shahriyar et al.[20] first proposed the cache system to achieve user's location privacy protection. As the vast majority of Point of Interest (POI) has a fairly long time, users can download and store some of the region's POI. When they need the relevant information in the future, it can be obtained just from the local storage without sending a request to the content provider. However, this method requires the user to store huge data in a large area (probably a city), thus the cached content is blind and storage space wastes. Shokri et al. [21] designed a distributed method, MobiCrowd, based on user cooperation. Its main idea is that when the user needs information service, they inquire about the neighbor users’ storage for interested POI. But the effect of this protection method is questioned, because it does not take the cache hit rate into account and pay little attention to the scenario when user have to send query to the server. In the works of Niu et al.[22], the system selects a number of false positions to implement anonymous request, which may be attacked by data mining and background knowledge. We will propose a new method of protection based on the above related work.

III. MOTIVATION

A. User Mobility Pattern

The behavior of mobile users has some inherent characteristics, which is proved by some scholars’ studies the key factor to design LBS location privacy protection[23][24]. However, most of the existing methods have neglected this factor, which we will take full account of. We start with a simple introduction to the user's mobility pattern.

First of all, from the perspective of individual mobile users, regardless of how different life is between people, the vast majority of people always visit and stay at some fixed and limited places in a long period of time (ordinary people will not often move home or change jobs) such as residence, company, car station or shopping malls. We call these places the individual sites (ISs) in our paper. Secondly, from a global point of view, there are many places such as apartments,
companies, bus stations where there are a lot of active people. We call them the public sites (PSs).

It’s not hard to notice the relations between the ISs and PSs. If lots of people share the same individual site, this individual site becomes the public site. In other words, public site must be the individual site of many people. In our system, IS provides a theoretical support for the local cache to get higher cache hit ratio, and PS provides a theoretical support for the users to complete collaboration.

B. A Scenario Example

To illustrate our motivation, we provide an example here. Staff Alice and her colleagues will have lunch during the company’s lunch break. So they should first check the information of surrounding restaurants, such as the menu, the empty seats and so on. In the traditional LBS service, they need to send requests to the service provider and then obtain the restaurant information. There are two obvious problems in this way. First, Alice exposes the privacy of her own location to an unreliable content provider (more seriously, her request may be hacked before arrive at the content provider’s server). Secondly, because so many employees in this paragraph of time query restaurant information (the same POI), the server needs to repeatedly send the same information to different users, which increases the burden of server computing overhead and network transmission overhead. In order to solve the first problem, the classical approach is to use \( k \)-anonymity method. Considering the method of selecting dummy location is easy to be attacked by data mining [9][22][25], we will use space \( k \)-anonymity. So, one of our works will focus on the design of \( k \)-anonymity algorithm. To solve the second problem, we naturally think of the current popular cache method [20][21]. It can better optimize the burden costs and the low information utilization rate. Therefore, the restaurant information used by Alice can be reused by other colleagues. Because cache method reduces the number of queries, it enhances the user’s privacy protection and solves the first problem at the same time. Our proposed RuleCache method will combine cache method and the space \( k \)-anonymity to protect the user’s location privacy.

IV. RULECACHE

A. Overview of Design

Considering the user’s mobility pattern, in our system the users will only store their interested POIs around their ISs where most of the time they are. Intuitively, the cache hit rate of local host will be high. Simultaneously, the content of cache is limited to the standard, which can avoid storing redundant information and save storage space. Notice that the IS is likely to be the PS, if any two users share the same IS, the IS becomes their PS where they can easily access information from each other. So, in our system users also share and obtain information through neighbor users’ collaboration to prevent disclosure of their privacy. Our overall vision is: when the users need service, they firstly query relevant information from their terminal storage area. Secondly, if the local information does not meet the threshold set by them (the information is insufficient), they will request to other neighbor users. And if that still can’t meet the requirements, lastly they send requests to the LBS server through our designed space anonymity algorithm, CRGA. This multi-stage protection for user location privacy is shown in Fig.1, in which the protection of level 1 is highest. Our purpose is to maximize the scenarios of using level 1 to satisfy users and send request to the LBS server as little as possible.

![Fig.1. Multistage protection of location privacy](image1)

![Fig.2. CRGA to achieve 4-anonymity](image2)

When users need to send request to the LBS server (we estimate this condition will be reduced as time goes), they will utilize the designed CRGA algorithm. Considering the impact of side information [26] on cache hit ratio and privacy security, which is ignored by many methods, our algorithm will take full account of it. Side information is the related background knowledge that can be acquired and used to infer the user’s true location by adversary. In this paper, it specially means the probability that a request is sent from each cell of the map [27], which is not difficult to obtain [22]. In Fig.2(a), the map is divided into many small regions by the square cells of the same size, different shades of the cell indicate the query probabilities are different. The greater probability means more users’ queries in this cell area. To improve the reutilization ratio of the local cached information, we select other \( k \)-1 cells that have the biggest query probabilities around the user to achieve space \( k \)-anonymity. In short, we firstly select the cells with higher user query density. As shown in the Fig.2(a), the dotted line indicates the clocking region formed by user A. Secondly, to guarantee the data fresh we should update the data in a timely manner. So our algorithm takes data timeliness into account. In Fig.2(b), the color of each cell is different, the darker color means the higher data freshness, the
white indicates that the data is invalid. We will try to select light cells with no longer fresh data in order to update the latest data. The dotted box represents the clocking region user A forms after the data timeliness is considered.

### B. Cacheware

We have described the shortcomings brought by using third anonymizing party. And strictly, security risk exists when there is a connection with the external network to send a request, which requires us to implement protection functions in the terminal. So we build a middleware called cacheware on the mobile platform. It is designed based on the concepts of IS and PS, because storage space often waste in the previous cache method which doesn’t considers users’ mobility patterns. For example, a user is sent by his company to go on a business trip, he may never use the POIs he queried and obtained there again in the future. If these data are stored in the local host, it just wastes storage space.

Cacheware has two function regions: compute region and cache region, as shown in Fig.3. The compute region calculates out user’s ISs using the position information obtained from terminal GPS, and the cache region is used to store user’s interesting POIs in the surrounding of the ISs. The compute region detects the position information regularly, if a user often appears in a certain position, the position is set to an IS. Accordingly, Cacheware will store the POIs around user’s ISs (default to a range of 1km radius) to prepare for their future use. By setting the non-sensitive POIs they have stored to default shared information, users can share these information to other cooperative neighbors. As for the information far away from the ISs, Cacheware deletes them after use.

The purpose of Cacheware is obvious. One is that it can take full control of the app’s LBS request and protect user’s location privacy from outer space. When an app needs to access location-based service, it will send a request to the cache region first. The other goal is that, after figuring out user’s ISs, only the POIs in vicinity of ISs are stored by Cacheware, which regulates the content of cache and save the storage space. Finally, it can help users to complete neighbor collaboration to access information. When the user can’t get enough information from his cache region, he will search it from surrounding neighbors’ cache region.

### C. System Architecture

In RuleCache system, we temporarily consider two entities, namely, mobile users and LBS providers. Mobile user is the person who holds the mobile devices, such as smart phone or tablet. LBS providers can be any network companies that provide LBS services, their LBS servers provide POIs for mobile users. The users exchange and store information through an ad hoc network. When necessary, the user sends a request to the LBS server via a cellular network (e.g., 4G) of the telecom operators. Each stored POI is labeled with data timeliness. Fig.4 shows the architecture of RuleCache system. For example, When Alice (user A) wants some POIs, the system first detects relevant information at the cache region of Cacheware (step 1) if the information fully meets the threshold set by A, the query completes. Otherwise, it then queries to neighboring users (step 2). And if Alice still can’t get enough information, the system will send a request to the LBS server (step 3). After obtaining the information, the system will determine whether the information is stored on the basis of storage conditions (IS).

![System architecture](image)

Suppose the map is divided into \( m \times m \) cells in squares with equal size. The set of them are denoted as \( \Phi = \{ a_1, a_2, \ldots, a_m \} \), the service information are exchanged and stored in the form of \( (a_n, poi, data) \) where \( n = 1, 2, \ldots, m^2 \), poi represents the name of the information user queries and data is the service information about the poi in cell \( a_n \) labeled with data timeliness.

### D. Query to Neighbors

In order to explain the concept of collaboration and expectation threshold, suppose in the previous scenario, Alice wants to query the restaurant menu information within 1km, but don’t find enough information on local cache region. So she sends a request formed \((poi, h)\) to the ad hoc networks, the key word \(poi\) is restaurant, \(h\) is the number of hops that request spread within the networks. Every time the request goes through a user, the number will reduce 1, and the request is not spread by 0. The users who receive the request will match the poi keyword with the content stored in their Cacheware. If there is a match, the relevant information will be returned to Alice. All the restaurant's information returned to Alice generate set \( P \), \( P = \{a_{p1}, a_{p2}, \ldots\} \). And the ideal result set of Alice’s request is \( Q \), that is the information of all restaurants within 1 km of the map, \( Q = \{a_{q1}, a_{q2}, \ldots\} \). Take the intersection \( |P \cap Q| \) as the effective content Alice obtains. In
order to better utilize the local cache content as well as strike a balance between service quality and privacy protection (the user’s privacy is often different with the change of time and place), we use the information threshold \( \lambda = \frac{|P \cap Q|}{|Q|} \), which indicates the user’s tolerance for local access information, the maximum value is 1 (if some related information has been stored on user’s mobile device, \( P \) is a collection of the relevant information in the first stage of calculating information threshold, and then in the second stage calculation, \( P \) is the union set of related information got from localhost and collaborative users). Alice will set a desired threshold \( \gamma \) (or set by the system default) after weighing the service quality and privacy. If \( \lambda \geq \gamma \), it means local information meet Alice’s needs, if \( \lambda < \gamma \), Alice will use our CRGA algorithm to issue a request.

E. Cloaking Region Generating Algorithm

When we design the CRGA, we identify two important factors, query probability (or side information) and data timeliness. Query probability of the anonymous area is utilized to increase cache contribution rate which related with cache hit ratio and enhance privacy security. Data timeliness is utilized to evaluate data freshness and help update outdated data.

Cache Contribution Rate. Because users’ activity degree are different in each cell on the map, the region with more people always issues more requests than the region with less people, which is also the reason for various values of side information. When we implement space k-anonymity, we select other k-1 cells with higher value of side information, which will make good contribution to improve the cache hit ratio of users’ queries in the future. On the \( m \times m \) map, service provider can calculate the probability that a query is sent out from each cell by the number of query times in the past, and all entities can easily obtain the probability distribution. We use \( P \) to represent that probability of query, obviously, there is \( \sum_{i=1}^{n} p_i = 1 \). And \( \alpha \) represents the cache contribution rate of the selected region to the cache hit ratio, we define \( \alpha_i = p_i \). In the absence of data timeliness, the k-1 regions we select should be a set that have the largest value of \( \sum_{i=1}^{k-1} \alpha_i \). We show the set as

\[
C_s = \text{set max} \sum_{i=1}^{k-1} \alpha_i. \tag{1}
\]

Data Timeliness. Every data has a certain lifetime, such as weather condition, when looking for anonymous cells, we also take opportunity to update the cached content based on the data timeliness. Considering the weather condition, although the weather forecast in a day will not be significantly changed, there are still some small differences (such as temperature, wind). Data usually becomes distorted, and in the lifetime, the longer data exists, the more obvious distorted it is. Therefore, it is necessary to update outdated information, especially for those areas with high query probabilities. We use \( T \) to represent the lifetime of a POI, and \( t \) indicates the existence time since that POI is downloaded from LBS server, \( o \) represents the data’s outdated degree, \( o \) is defined as \( o = \frac{\sqrt[t]{T}}{T} \). Because the finally cloaking region generated by \( k \) selected cells include other cells, we make the data timeliness of all cells in the piece of cloaking region into consideration, so we express average outdated degree as \( O \),

\[
O = \frac{1}{n} \sum_{i=1}^{n} \frac{t_i}{T_i}, \text{ where } n \text{ is the number of cells in the cloaking area.}
\]

Combined the two factors above, the set of anonymous cell in CRGA is represented as:

\[
C_z = \text{set max} \left( \sum_{i=1}^{k-1} \alpha_i \right) \cdot \frac{1}{n} \sum_{i=1}^{n} \frac{t_i}{T_i}. \tag{2}
\]

The detailed CRGA is shown in Algorithm 1. We first select \( z \) cells which have the highest value of side information, \( z > k \). With \( z \), we obtain all candidate set \( C \) containing k-1 cells, \( z \) is set by system (defaulted 20) or set by the user after balancing calculation cost and privacy. We will choose the optimal set \( C' \) as \( C_z \) that meets the condition best from the total candidate set \( C_z \).

**Algorithm 1:** Cloaking Region Generating Algorithm(CRGA)

**Input :** \( p \) (query probability of each cell), \( \alpha \) (real location), \( z \) (a system parameter)

**Output :** \( C_z \)

1. sort cells around 2 km based on \( \alpha \) and \( p \):
2. select \( z \) cells with the largest \( p \) as the candidate set \( C_z \);
3. if \( k > z \) then
4. return error;
5. end
6. else
7. \( C_z = \{C | C \subseteq C_z \land |C| = k - 1 \} \)
8. end
9. for each \( C' \in C_z \) do
10. compute \( \left( \sum_{i=1}^{k-1} \alpha_i \right) \cdot O \) for the \( k - 1 \) selected locations in \( C' \);
11. end
12. output \( C_z = \text{set max} \left( \sum_{i=1}^{k-1} \alpha_i \right) O \)
V. SECURITY ANALYSIS AND SYSTEM LIMITATIONS

A. Security Analysis

The adversary’s purpose is to get the location privacy of a specific user. We divide the risk that our system faces into two categories, one is external risk, the other is internal risk. The external risk refers to that the attacker invades the channel between cooperative users (or user and the server) and intercepts privacy information from outside. To prevent this eavesdropping attack, encryption technology can be introduced in our system, such as the public key infrastructure (PKI), which is currently the main method to guarantee channel security. The internal risk refers to that some entity in the system who has certain privileges intercepts and captures private information, such as a malicious service provider or a malicious neighbor user, these risks are usually more harmful. According to our system, we mainly analysis two kinds of internal attack risk.

Colluding attack resistance. Suppose several malicious users in our system want to get sensitive information. When Alice can’t get enough POI information and request surrounding collaborative users, she sends a query to them in the format of \((poi, h)\). Since the communication range of a hop in ad hoc network may transmit for hundreds of meters, and there is no relationship between the colluding users and Alice, it’s hard to determine the real location of Alice although they receive the query. Moreover, there is no real identity information in the query (if necessary, she can use a pseudonym), these colluding users even can’t know who sent the request. Therefore, our system has a strong resistance to internal colluding attack.

Inference attack resistance. Because LBS server records all request information from the users, the malicious service provider may hope to use these request information to infer the true location of the user. In our system, the request sent to server is in the form of \((ID, poi, R)\). \(ID\) is the user’s identity or pseudonym, \(R\) is the cloaking region generated by \(k\) cells that have the highest value of query probability in CRGA algorithm. So, the service providers are not only unable to judge where the request really comes from, they could even cause a greater deviation from the real location based on side information, because they generally think that the place that has higher query probability issued the request. It is effective to prevent service provider to take inference attack using background knowledge.

Due to the defects of some traditional anonymizing methods, attackers might assume that the user is actually located in the center of a cloaking region (or other geometric location). Our algorithm indeed finds anonymous cells around user’s real place, but these cells which have the higher query probability make the generated cloaking region random and uncertain. And, after we take all cells’ data timeliness into account, the uncertainty and randomness of the generated region will eventually aggravate. In a word, the user’s real position is unlikely to be the center of the generated region in our algorithm.

B. System Limitations

There exist some limitations and future work. One thing to note is that, when we design the system we can’t rule out the small number of people who engage in certain special industries and change activity place frequently, for example, the insurance salesman running around in the city. In this situation, the number of user’s IS may be very small, the information will be mainly got from neighbors and server. We do not make a detailed analysis of this situation, which may be the future point to consider. Secondly, in order to facilitate user to access the share information of other neighbor users, each user in the system should share a certain amount of non sensitive information, which may be contrary to the wishes of some users. At present, we do not make any relevant investigation and research, it is believed an interesting research direction. In addition, how to deal with real-time information acquisition is a problem of all methods using cache, such as how to obtain road traffic situation. Unfortunately, In this situation the first two levels in our approach will be largely ineffective, the user will mainly obtain information from the server directly through the third level.

VI. PERFORMANCE EVALUATION

A. System Setup

Experimental simulation and performance evaluation of RuleCache are carried out. We deploy the system on Android platform. 10000 mobile users are set up in the city’s map of \(10km \times 10km\), each mobile user follows the Levy walk mobility model[28] which has been proved to truly describe human mobility behavior. The map is divided into cells with length of 50m. The initial query probability distribution is obtained from Google Map API. We assume that each user carries on 10 POI queries per day. In the case of no loss of generality and to make simulation simple, we consider only one POI. Average values of the experimental results are taken.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(k)</td>
<td>related to (k)-anonymity and represents the degree of anonymity</td>
</tr>
<tr>
<td>(h)</td>
<td>the maximum number of hops user’s request transmits in the wireless ad hoc network</td>
</tr>
<tr>
<td>(\gamma)</td>
<td>the expectation threshold set by users</td>
</tr>
<tr>
<td>(t)</td>
<td>the time of simulation (days)</td>
</tr>
<tr>
<td>(r)</td>
<td>The cache hit ratio</td>
</tr>
</tbody>
</table>

Some parameters are used in the experiments, we list the main parameters and their interpretation in table I. We will compare our method with three existing schemes, enhanced-DLS[27], MobiCrowd[21] and CaDSA[22]. Enhanced-DLS represents the dummy selection algorithm and does not use caching, MobiCrowd preliminarily uses caching and CaDSA does better.
B. Simulation Results and Evaluation

Communication cost. We first evaluate the number of submitted queries that have acquired POI information from localhost, neighbors and LBS server respectively, as shown in Fig. 5(a). It can be seen that the number of queries that information is obtained from localhost gradually increases in early time, that is because cacheware calculating ISs and caching POIs take a certain amount of time. And then the line reached a relatively high stable state, which indicates that the users will mainly get information from their own mobile device after the interested POIs are already stored. The information obtained from neighbors is reduced to a steady value accounted for a percentage of 19% after the initial transient increases, which expresses the users’ dependence on neighbors are weakening after the strong demand for information in pre storage phase. And the role of LBS server for user to obtain information is increasingly less important, reaching about 10%. We can see that in our system the information is mainly obtained from localhost and neighbors, of which the percentage can reach almost 90%.

Fig. 5(b) represents the condition of comparing the number of queries submitted to LBS server by different methods. From the figure, we can see the query number sent by enhanced-DLS keeps a horizontal line, because it does not use cache. Though MobiCrowd and CaDSA issue less number of queries than enhanced-DLS, there are still a lot of requests at LBS server (about 50% and 30%, respectively). Since our RuleCache have considered the factors of user's mobility pattern, the neighbor collaboration and the data update mechanism, few queries (about 10%) are sent to the server. This result shows that our multi-level cache system is better at using cache content than other methods, reducing the requests between user and LBS server by more than 20%, which not only saves the communication cost, but also obviously reduces the risk of privacy leakage.

Cache Hit Ratio. We also make comparative analysis of local cache hit ratio that indicates the percentage of information got from localhost and neighbors. Fig.6 (a) reflects the relationship among cache hit ratio of several methods and parameter k when t=7, h=4, γ =0.8. As MobiCrowd does not specifically enhance cache hit ratio, the value of which is relatively low, hovering between 50% and 60%. Without using cache, the ratio of Enhanced-DLS keeps 0. CaDSA makes use of dummy locations to improve cache hit rate, it can reach a higher level in about 70%. Since Our RuleCache have considered the impact of mobility pattern, the regional query probability and user cooperation on cache hit ratio in the process of design, it enhances the possibility effectively for user to access information from local, and the value can reach 90%. Fig.6 (b) shows the relationship between cache hit rate and t. We note that enhanced-DLS still keeps 0 as before. The ratio of MobiCrowd, CaDSA and RuleCache increases with time going and finally reaches a stable value, and our method always remains at a relatively high level. The results also show RuleCache has a higher cache hit ratio than CaDSA by 20%.

Storage cost. Obviously, the cost of terminal storage depends on the size of the storage map and the number of POIs in the region. Assume that the size of each stored POI

\[ \left( a_p, \text{ poi, data} \right) \text{ is less than 1KB. Take New York City (with the largest number of POI in the world) as an example, there are 250000 POIs. The traditional cache method, such as CaDSA, may consume increasingly large device storage with the extension of use time, and the maximum stored size is } 250,000 \times 1KB = 250MB . \] While our method only stores the related POIs around user’s ISs, the average space cost to meet user’s needs is only about 27MB, significantly reducing the storage overhead.

TABLE II. Security degree and communication cost of RuleCache

<table>
<thead>
<tr>
<th>Stage</th>
<th>Security degree</th>
<th>Communication cost</th>
<th>Information source</th>
<th>Information percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>level 1</td>
<td>high</td>
<td>low</td>
<td>localhost</td>
<td>71%</td>
</tr>
<tr>
<td>level 2</td>
<td>middle</td>
<td>middle</td>
<td>neighbor</td>
<td>19%</td>
</tr>
<tr>
<td>level 3</td>
<td>low</td>
<td>high</td>
<td>LBS server</td>
<td>10%</td>
</tr>
</tbody>
</table>

VII. CONCLUSION

In this paper, we propose a multi-level cache privacy protection system called RuleCache after considering the defects of low reutilization ratio of the POI contents, the untrusted service providers and the excessive reliance on anonymizing server in traditional LBS service model. We combine users’ mobile behavior pattern with the cache method to enhance protection, and through the design of cacheaware we make the terminal stored content standardized, which
avoids the terminal device stores useless and redundant content. When user needs to send a query to the server, we propose CRGA algorithm which takes side information and data timeliness into account. The system is mainly designed to improve cache hit ratio of the stored content in terminal device and prevent user’s privacy leakage. The simulation results show that the performance of our method is better than that of others by more than 20% on the communication cost and cache hit ratio, and the storage cost of our system is significantly low.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions. This work was supported in part by the National Science Foundation of China under Grant 61300036, the Projects in the National Science & Technology Pillar Program under Grant 2013BAH38F01 and the Foundation for University Key Teacher by the Ministry of Education.

REFERENCES

[22] Ben Niu,Qinghua Li,Xiaoayan Zhu,Guohong Cao,Hui Li. Enhancing Privacy through Caching in Location-Based Services.in Proc. Of IEEE INFOCOM 2015.