CLPP: Context-aware Location Privacy Protection for Location-based Social Network

Hongli Zhang[1], Zhikai Xu[1], Zhigang Zhou[1], Jiantao Shi[1] and Xiaojiang Du[2]
[1]School of Computer Science Engineering, Harbin Institute of Technology, Harbin, 150001, China
Email: {zhanghongli, xuzhikai, zhouzhigang, shijiantao}@pact518.hit.edu.cn
[2]Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA
Email: dux@temple.edu

***Abstract*—location-based social network (LBSN) has grown exponentially over the past several years. Given its high utility value, LBSN, however, has raised serious concerns about users' location privacy. Although users may avoid releasing geo-content in sensitive locations, this, however, does not necessarily prevent the adversary from inferring users' privacy through spatial-temporal correlations and historical information. In this paper, we introduce a new location privacy problem: context-aware location privacy protection (CLPP) problem where the privacy requirements of users are not constant and isolated. We propose a novel metric to quantify the privacy risks. Then the CLPP is formalized as how to accurately and efficiently evaluate whether the users' published geo-content meet the user's privacy requirement. To achieve online evaluating, we design two novel algorithms to calculate the correlation between the locations. Eventually, our experimental results demonstrate the validity and practicality of the proposed strategy.**

***Index Terms*—location privacy, location-based social network, inference attack, privacy preserving.**

I. INTRODUCTION

The advances in location-acquisition and wireless communication technologies enable people to release geo-location content anytime and anywhere, fostering a bunch of location-based social networking services(LBSN), e.g., Foursquare, Twitter, Google Latitude and Flickr, where users can easily check in, locate friends or share experience via mobile devices. However, the potential abuse of users' location information and relate sensitive information by unauthorized users (such as application servers, malicious users) is evolving into a serious concern.

Existing approaches (such as spatial *k-anonymity* [1, 2]) on location privacy protection mostly focus on "single shot" scenario, which, however fall to protect the privacy of LBSN users when applied to inference attack due to spatio-temporal correlations between the published geo-location content. In an example scenario of LBSN applications, a LBSN user may extremely want to share his geo-location content with his friends at cafe or store, whereas the user does not want anyone to know he had been to the hospital. Therefore, as shown in Figure 1, the user disabled location servers to avoid privacy leakage when he was at the hospital, whereas he released geo-location content at the store and cafe. This, however, does not necessarily prevent an adversary from inferring that the user visited the hospital through spatial-temporal correlations and historical information. An adversary knowing most of users follow path 2 between the cafe and the store may infer the user probably visited the hospital when the user's travel time between cafe and store is beyond the regular interval.

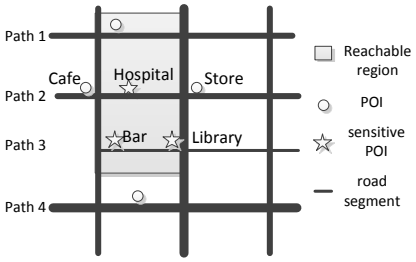

Fig.1. An example of context-aware location privacy problem

In this paper, we introduce and investigate a new location privacy problem: *Context-aware Location Privacy Protection* (CLPP) problem in LBSN, in which the privacy requirements of users are not constant and isolated along spatial and temporal dimensions. We assume that the users' privacy requirements are *diverse* and *dynamic*, depending on users' personal definition of sensitive locations. Notice that even the same user, his privacy requirement for different locations may be different and change overtime. Here, ***Location privacy*** is redefined as "the ability to prevent other parties from learning one's current or past hidden sensitive locations". In LBSN, on the one hand, the users want to release as much geo-location content as possible to get better services; on the other hand, the users are concern about their privacy, especially the privacy leakage caused by the context of published geo-location content. To tackle the problem, we need to answer the question: *when and where can the user release geo-location content to LBSN*?

To answer the question, we present a CLPP server framework. The goal of the framework is to let user publish as much geo-location content as possible while avoiding privacy risk. We propose a novel metric to quantify the privacy risks. Then, the CLPP is formalized as how to accurately and efficiently evaluate whether the users' published geo-content meet the user's privacy requirement. To achieve online evaluating, we further present two mining algorithms (basic algorithm, Collaborative Filtering based heuristic algorithm) to calculate the correlation between the locations.

In summary, the paper makes the following contributions:

(1) We introduce a new location privacy problem: *Context-aware Location Privacy Protection* (CLPP) where the privacy requirements of users are not constant and isolated along spatial and temporal dimensions.

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

(2)We present a novel evaluation strategy to answer the question: when *and where can the user release geo-location content to LBSN*?

(3)We present two novel mining algorithms (basic algorithm, Collaborative Filtering based heuristic algorithm) to accurately and efficiently calculate the correlation between the locations.

The remainder of the paper is organized as follows. Section II introduces the system framework, threat model, and formally defines the secure metrics. Our proposed mining algorithm and evaluate algorithm are presented in Section III. Section IV presents the experiment results confirming the effectiveness of the proposed algorithms. Section V discusses related work on location privacy protection, followed by the concluding remarks in Section VI.

II. Problem Definitions

In this section, we will introduce the motivation of the CLPP problem and then describe the system framework and threat model. Finally, we will formally define the secure metrics.

A. Motivation

Most existing approaches [1, 2, 7-9] on location privacy protection focus on "single shot" scenario, and usually assume that the privacy requirements of users are constant and isolated in spatial and temporal dimensions. However, the assumption may not be true in location-based social network (LBSN), especially for the real-name social network. For example, in the spatial dimension, Alice may be happy to release her location to LBSN to locate her nearby friends at stores or cafes, and she does not consider these locations as privacy. But Alice does not want anyone to know she had been to the hospital by not using the server locators at all. In the temporal dimension, Alice may require higher privacy protection for workplace at weekend. Moreover, when combining the spatial dimension and the temporal dimension, the locations are not isolated. For example, as shown in Figure 1, an adversary may learn that most users follow path 2 between the cafe and the store. Then, even if Alice did not release any geo-location content at hospital, the adversary may still infer that Alice probably visited the hospital when she released her location at the cafe and the store.

Therefore, in this paper, we investigate a new location privacy problem *Context-aware Location Privacy Protection* (CLPP) and redefine **location privacy** as the ability to prevent other parties from learning one's current or past hidden sensitive locations. In this paper, we will propose a server framework to formally represent different users' personal privacy requirements, and explore novel approaches to meet the requirements of users.

B. CLPP Server Framework

The CLPP server framework is illustrated in Figure 2, which involves three critical components: *users*, *trust third party server* (TTP), and *location-based social network* (LBSN). Following are details about the three components.

(1)*Users*: Users in the system are equipped with GPS smart phones or tablets which are capable of communicating with LBSN. In order to get services from LBSN, the mobile user sends his current location p_{loc}, timestamp t, privacy requirement PR and encrypted query content qc to the TTP through secure connections. The server request is represented as ($user$, p_{loc}, t, PR, qc).

(2)*Trust third party*: TTP is fully trusted by other entities in the system. It has a collection of users' historical information and the precise locations of the POIs (map information). When the TTP gets the request message, it will evaluate whether the user's privacy requirement would be satisfied, if the current location was released to the LBSN. (The details are given in Section III). If there is no leakage of the privacy problem, the user's server request (u_k, p_{loc}, t, PR, qc) will be forwarded to the LBSN automatically; if otherwise, TTP will return the probable leaked POIs, together with the leakage probabilities, to the users. Then the user will decide whether to release or not to release the current server request. Note that we consider that TTP should avoid interfering with the users frequently. It should run automatically most of the time.

(3)*LBSN:* when receiving the user's server request, the LBSN will provide appropriate services for the users, such as locating nearby friends, or distributing the coupons around user's current location.

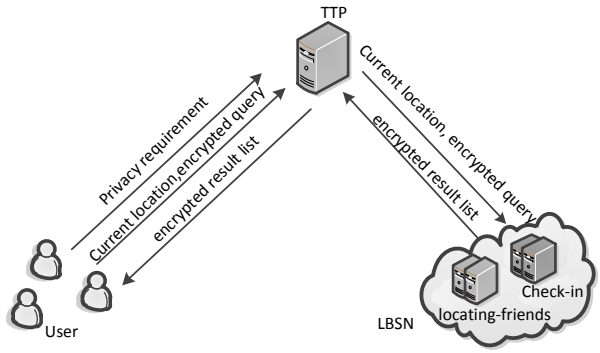

Fig.2. CLPP Server framework

C. Threat Model

We make a widely acceptable assumption that the LBSN provides "honest but curious" services. LBSN acts in an honest fashion and correctly follows the designated protocol specification. However, it is interesting in analyzing users' geo-location content so as to infer each user's sensitive information. However, this common mechanism repeatedly encounters the same issue: how to provide privacy-preserving against the newly introduced entity? To this end, our framework disassociates privacy evaluate operation from the query business logic. Here, the TTP is just responsible for evaluate the location privacy of users without knowing the query content and the relationship among users in LBSN, while the LBSN provider just processes queries from users without learning the hidden sensitive locations.

D. Secure Metrics

As mentioned, a request for LBSN service consists of five factors: user, location, time, privacy requirement and encrypted query (such as check-in, locating friends and social gaming). In this paper, we take check-in as an example for all the discussions and illustrations. However, it can be easily extended to other applications. The following are the definitions used in the article.

Definition 1: **Check-in Sequence:** Given a user u_k, his check-in sequence is a set of POIs ordered by check-in time $S(k) = < (p_1, t_1),\dots , (p_i, t_i),\dots , (p_m, t_m)>$, where t_i denotes his check-in time at POI p_i. We define a **sub-check-in sequence** as user's check-in sequence in one day.

Definition 2: ***s*-Confidence Privacy:** Let $C(k) = (c_1, c_2,\dots , c_n)$ denotes the hidden sensitive POIs set of u_k. We say the system provide s-confidence privacy for u_k, if for each c_i in C, no adversary can use $S(k)$ to identify the probability that u_k had visited c_i is bigger than s.

Definition 3: **User's Privacy Requirement:** Given a user u_k, his privacy requirement is represented as $PR(k) = < (c_1, s_1),\dots , (c_i, s_i),\dots , (c_n, s_n)>$, where c_i denotes the sensitive POI of u_k and s_i denotes s_i -confidence privacy at POI c_i. Note that different users may have personal hidden sensitive POIs set, and the privacy requirement for different POIs may be different.

III. Our Solutions: CLPP Algorithms

In this section, we will describe the details of our solutions. On the one hand, TTP calculates the correlation between the POIs through historical check-in sequences and geographical information (*offline training phase*). We first give a basic mining approach, and then we introduce a collaborative Filtering based heuristic approach. On the other hand, when the TTP gets the server request, it will evaluate whether the user's privacy requirement would be satisfied, if the current location was released to the LBSN (*online predicting phase*).

A. Basic Mining Approach

Users' behaviors usually follow certain mobility patterns in specific region [3]. Thus, a basic approach is to make use of majority users' historical check-in sequences to capture users' frequent patterns.

A frequent pattern is pattern that occurs frequently in users' check-in sequences. For example, as shown in Figure 1, if most users always follow a check-in sequence (Cafe→ Hospital → Store), then we call the sub-sequence is a frequent pattern. The confidence of the frequent pattern is computed as

$$P\left(p_j \mid p_i, p_{i+1}\right) = \frac{support(p_i, p_j, p_{i+1})}{support(p_i, p_{i+1})} \quad (1)$$

where $support(p_i, p_j, p_{i+1})$ denotes the number of sub-check-in sequences that contains POI p_i, POI p_j, and POI p_{i+1} in sequence.

TABLE I
Check-in Sequences of Users

User	Sub-check-in sequences
u_1	$< (p_1,t_i), (p_2,t_{i+1}), (p_4,t_{i+2})>$
u_2	$< (p_1,t_i), (p_2,t_{i+1}), (p_3,t_{i+2}), (p_4,t_{i+3})>$
u_3	$< ((p_4,t_i), (p_3,t_{i+1}), (p_2,t_{i+2}), (p_1,t_{i+3})>$
u_4	$< (p_1,t_i), (p_3,t_{i+1}), (p_4,t_{i+2})>$
u_5	$< (p_1,t_i), (p_2,t_{i+1}), (p_4,t_{i+2})>$
u_5	$< (p_1,t_j), (p_2,t_{j+1}), (p_5,t_{j+2})>$

To fully leverage the historical information, we use Frequent-Pattern Tree [4] to search and store all the frequent patterns in the sub-check-in sequences. The build process is illustrated by using the sample data shown in Table I. We construct the tree as follows: Firstly, we will scan all the sub-check-in sequences and get a list of start nodes$< (p_1: 5), (p_4: 1)>$, (the number after ":" indicate the support). Only the start nodes that satisfy the minimal support (here, the requirement for the minimal support is 1) will be preserved. Secondly, we will create the root node of the tree, and label it with "NULL". Note that the check-in sequences is directional, we will insert the node in sequence. Then, as shown in Figure 3-(a), we will scan the first sub-check-in sequence to construct the first branch of tree $< (p_1: 1), (p_2: 1), (p_4: 1)>$. For the second sequence, as shown in Figure 3-(b), since $< p_1, p_2, p_3, p_4>$shares a common prefix $< p_1, p_2>$ with the existing branch$< p_1, p_2, p_4>$, thus, we will increment the count of each node along the common prefix by one, and then create new nodes $(p_3: 1)$ $(p_4: 1)$ and add them as child nodes of $(p_2: 2)$. Step by step, after scanning all the sequences in Table I, finally, we get a frequent pattern tree as shown in Figure 3-(c).

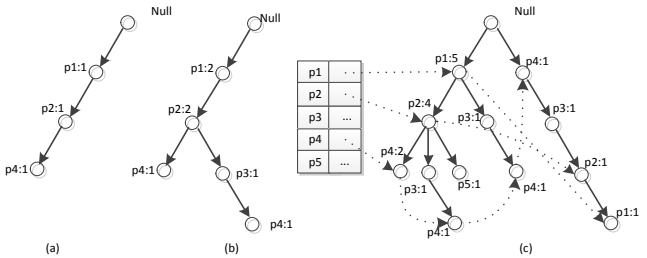

Fig.3. The search and store of the frequent pattern

To facilitate the traversal of the tree, we further build a node header table where each node points to its occurrences in the tree via a node-pointer, and we link nodes with the same names via such node-pointers in sequence.

Then, we will use the tree to calculate the confidence of the frequent pattern. Take $P(p_2| p_1, p_4)$ as example, we first find the positions that node p_4 occurs in the tree through the header table and node pointer. Then for each node p_4 in the tree, we will walk up the tree recursively to search node p_1, once p_1 is the ancestor node of p_4, *support* (p_1, p_4) will be incremented by the count of the node p_4. In the same way, we obtain the value of *support* (p_1, p_2, p_4). Then we calculate $P(p_2| p_1, p_4)$ use (1). The time complexity of the calculate process is $O(m*h)$, where m denotes the number of nodes p_4 that occurs in the tree and h denotes the average height of the tree.

B. Collaborative Filtering based Heuristic Approach

The basic approach gives no considerations to the users' individualized characteristics. In real life, people actually like to visit certain POIs that fit their own personal interest. For example, as shown in Figure 1, suppose that most people will visit the bar when driving along path 3, however, those who often went to the book stores tend to visit the library instead of the bar. It's obviously that the basic approach will not deal with the above situation. Therefore, we present a collaborative filtering (CF) based heuristic algorithm to provide users with personalized privacy evaluation.

Definition 4: **User's Rating Array:** Given a user u_k , his personal characteristic is represented as an rating array $R(k)=<r_{k1},\dots , r_{ki},\dots , r_{kn}>$, where r_{ki} denotes u_k's rating of POI p_i and it reflects how much the user u_k like POI p_i. A straightforward solution to calculate r_{ki} is to use the

number of repeat visits to POI p_i from u_k, but the method is insufficient. For example, u_k check-in 10 times at Carrefour hypermarket where plenty of people check-in whereas u_k check-in 10 times at the Science Museum where only a few people check-in. It's apparently that the Science Museum is more important when reflecting user's personal preference. Inspired by the TF-IDF scheme in information retrieval system, we treat a check-in record as "a term", and treat the user's check-in sequence as "a document". Formally, user's rating r_{ki} is computed as

$$r_{ki} = \frac{u_k.l_i}{\sum_{p_j \in S(k)} u_k.l_j} * \log \frac{|U|}{|\{u | u.l_i \geq 1, u \in U\}|} \quad (2)$$

where U denotes the collection of all the users and l_i denotes number of repeat visits to POI p_i from u_k. The first part of (2) is the TF value of POI p_i in the check-in sequence of u_k, and the second part denotes the IDF value of POI p_i.

Definition 5: **POI Similarity *sim* (p_i, p_j):** POI similarity indicates the correlation between POI p_i and POI p_j in the space of human behavior. For example, if most people who often visited the restaurant, also like going to the cinema. Then the restaurant and the cinema are similar though they are different in terms of the business categories.

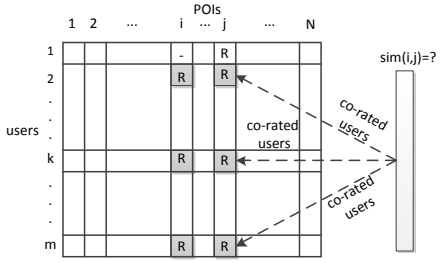

Fig.4. The process of similarity computation

The similarity between two POIs is calculated by integrating the users' check-in sequences. As shown in Figure 4, the users' ratings of the POIs are stored as entries of a matrix where the matrix rows corresponds to the users and the columns corresponds to the POIs. The basic idea to compute the POI similarity is to isolate the users who have checked in (rated) both of the POIs, and to determine the similarity by using the users' ratings (collaborative filtering algorithm). In the case, we consider the two POIs as two vectors in the *m*-dimensional user-space and then use the cosine of the angle to compute the similarity between the two vectors. Formally, *sim* (p_i, p_j) is calculated as

$$sim(p_i, p_j) = \cos(p_i, p_j) = \frac{\sum_{k \in co(i,j)} r_{ki} \cdot \sum_{k \in co(i,j)} r_{kj}}{|\sum_{k \in co(i,j)} r_{ki}|^2 | \sum_{k \in co(i,j)} r_{kj}|^2} \quad (3)$$

where *co* (*i*,*j*) denotes the set of users who have checked in both POI *i* and POI *j* and "$\cdot$" denotes the dot-product between the two vectors. We predict a user's rating of unvisited POIs by using the POI similarity and users' rating array. Intuitively, to predict the user's rating of POI p_j given the user's ratings of POI p_i and p_{i+1}, if POI p_i is more related to p_j beyond p_{j+1}, then the rating of POI p_i is likely to be a better predictor for POI p_j than the rating of POI p_{i+1} is. Formally, our approach can be represented as

$$r_{kj} = \frac{\sum_{i \in S(k)} (r_{ki} + dev_{j,i}) * sim(i,j)}{\sum_{i \in S(k)} sim(i,j)} \quad (4)$$

$$dev_{j,i} = \frac{\sum_{k \in co(i,j)} (r_{kj} - r_{ki})}{|co(i,j)|} \quad (5)$$

where *sim* (p_i, p_j) denotes the similarity between POI *i* and POI *j*, and $dev_{j,i}$ is calculated by using (5). The ratings have profound influences over one's behaviors [3]. Thus, we redefine the confidence of frequent pattern by introducing individual's rating. Formally, the confidence of the frequent pattern is computed as

$$P(p_j \mid p_i, p_{i+1}) = \frac{(1 + a * r_{kj}) support(p_i, p_j, p_{i+1})}{(1 + a * \overline{R_k}) support(p_i, p_{i+1})} \quad (6)$$

where *support* (p_i, p_j, p_{i+1}) has no difference from (1), $\overline{R}_k$ denotes u_k's average ratings of the POIs geographically located between p_i and p_{i+1}（the POIs will be given in section *C*）, and *a* is turning parameter ranging within [0, 1].

C. Online Privacy Evaluate

When the TTP gets the server request, it will evaluate whether the user's requirement would be satisfied, if the current location was released to LBSN.

Algorithm 1 Evaluate location privacy

INPUT: $PR(k) = <(c_1, s_1), \ldots, (c_i, s_i), \ldots, (c_n, s_n)>$
$S(k) = <(p_1, t_1), (p_2, t_2), \ldots, (p_i, t_i)>$
p_{i+1} // user's current location
t // current time
OUTPUT: H // probable leaked POIs, H is initialized to an empty set

```
1.  △t=t- t_i
2.  if △t≤dis(p_i, p_{i+1})/ v_max
3.      return H
4.  else
5.      R={p_m| dis(p_i , p_m)+dis(p_m , p_{i+1}) ≤Δt * v_max }
6.  if R∩C(k)= ∅      //C(k):user's sensitive POIs set
7.      return H
8.  else
9.      for each c_i in R∩C(k)
10.         if P (c_i |p_i , p_{i+1},R)> s_i
11.             H= H∪< c_i, P (c_i |p_i , p_{i+1},R)>
12. return H
```

Take user u_k as an example, suppose that u_k has checked in POI p_i at t_i. Given in a new check-in request (u_k, p_{i+1}, t, PR (k), qc), Firstly, TTP will calculate the interval $\triangle t$ between two check-ins: $\triangle t = (t - t_i)$. Since the map information and the maximum velocity of the road segments v_{max} are publicly observed, TTP will evaluate whether u_k has visited other POIs from the above observation. If $\Delta t \leq (dis(p_i, p_{i+1})/ v_{max})$, TTP can infer that u_k definitely does not have time to visit any other POIs between the two check-ins ($dis(p_i,p_{i+1})$ denotes the Manhattan Distance between p_i and p_{i+1}). Then the check-in request will be forwarded to the LBSN. Otherwise, as shown in Figure 1, the TTP will calculate the reachable POIs set $R = \{p_m |$ dis (p_i, p_m) +dis (p_m, p_{i+1}) + $\leq \Delta t * v_{max}\}$. Then the privacy evaluate question is formalized as a posterior probability P (p_j|

p_i, p_{i+1}, R). We use Bayesian reasoning to derive the posterior probability. Formally,

$$P(p_j \mid p_i, p_{i+1}, R) = \frac{P(p_i, p_j, p_{i+1})}{P(p_i, R, p_{i+1})} = \frac{P(p_j \mid p_i, p_{i+1})}{\sum_{p \in R} P(p \mid p_i, p_{i+1})} \quad (7)$$

where P (p_i, p_j, p_{i+1}) is calculated by using (1) or (6). Then, for each c_i in $R \cap C$ (k), TTP will use P ($c_i \mid p_i$, p_{i+1}, R) to evaluate whether the user's privacy requirement would be satisfied. If there is no leakage of the privacy problem, the user's check-in request will be forwarded to the LBSN; if otherwise, TTP will return the probable leaked POIs, together with their leakage probabilities, to the users. Then the user will decide whether to release or not to release the check-in request. The detail of this algorithm is shown in Algorithm 1.

IV. EXPERIMENTS

In this section, we evaluated our system using real-world Foursquare dataset, made available by Gao [5]. It contains the check-in history of 18107 users ranging from March 2010 to January 2011. For each user, we have his previous check-in locations and the corresponding check-in time. We first evaluate the accuracy of CLPP algorithms and then explore the tradeoff between privacy and utility. Both algorithms are implemented in C++.

A. Accuracy Metrics

In our dataset, the real sensitive locations are hidden by users, thus we generate two sensitive location sets for each user artificially in our experiment. Given a user u_k, the first hidden location set A is generated by randomly marking off a portion of POIs that he has checked in, and the second hidden location set B is generated by adding POIs which are geographically located between the POIs that he has checked in.

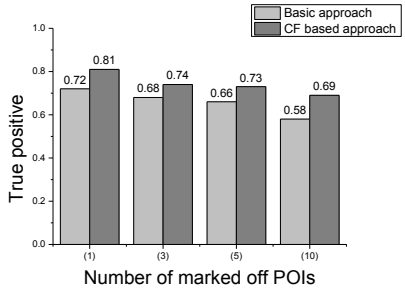

Fig.5. Performance evaluation on true positive

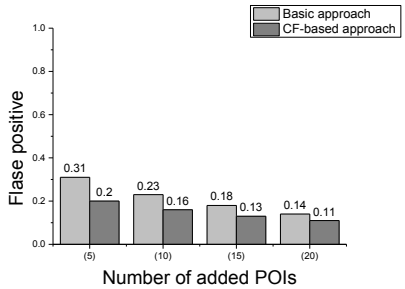

Fig.6. Performance evaluation on false positive

We then use two performance metrics to evaluate the accuracy of our system: (1) Average confidence of hidden location set A, represented as *true positive*. (2) Average confidence of hidden location set B, represented as *false positive*. For each approach, we randomly select 1,000 users and choose p_i and p_j in each user's check-in sequence (the interval Δt between p_i and p_j should exceed the average check-in interval). For each user, we random mark off 1, 3, 5, and 10 visited POIs between p_i and p_j as hidden location set A, and add 5, 10, 15, and 20 (different interval Δt between p_i and p_j) un-checked-in POIs that geographically located between p_i and p_j as hidden location set B.

The experimental results in Figure 5 and Figure 6 show that increasing the number of marked off or added POIs does not much affect the true confidence and false confidence. The result further denotes that simply suppresses sensitive location does not protect user's location privacy. In Figure 5, the higher true positive rate is, the better the approach is while false positive rate in Figure 6 represents the opposite. It can be seen that the CF-based approach always exhibits the best performance in terms of true positive rate and false positive rate under all values. This is because some users' check-in sequences are personal and unpopular, making it hard for the basic approach to evaluate whether the user has visited the hidden locations through majority users' historical check-in sequences.

B. Privacy-Utility Tradeoff

What's the price in terms of utility that we have to pay for a formal privacy guarantee? Since the users want to publish as much geo-location content as possible to get better service. A performance metric called *utility ratio* is put forward to evaluate system utility. It is calculated by dividing the number of POIs that user wants to check-in by the number of POIs that user can check-in while ensuring location privacy.

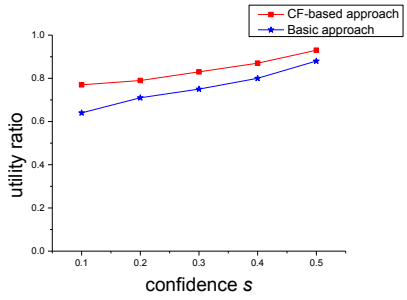

Fig.7. Utility ratio (the ratio of sensitive POIs is 5 percent)

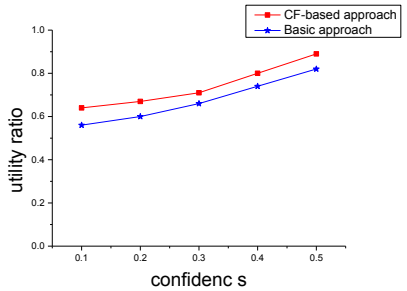

Fig.8. Utility ratio (the ratio of sensitive POIs is 10 percent)

For each user, we randomly select a part of POIs (5 percent and 10 percent separately) in the dataset and add them as sensitive locations. For simplicity, we assume that the user's requirement for different POIs is the same, but our system allows the users to define personal privacy requirement for different POIs.

We vary user's privacy requirement by varying the value of confidence *s*. For both sets of experiments, as shown in Figure 7 and Figure 8, the performance degradation of our system utility is small. This is because the false positive of our approach is very low (especially for the CF-based algorithm) and only the POIs that very relevant (which is calculated by (7)) to the sensitive POIs will be suppressed by our system. The result implies that our system can afford strong privacy guarantees (by choosing a smaller value of confidence *s*) without sacrificing too much utility.

V. Related Work

Protecting location privacy of mobile users has been widely studied in recent years. Depending on the different methods to distort the users' queries before the queries reach the location based server (LBS), we roughly classify the existing work into the following categories:

Obfuscation based approaches: Obfuscation approaches try to preserve location privacy by adding uncertainty to users' location. Spatial Location cloaking is one of the most representative approaches [2, 6-7]. It typically blurs users' real location into spatial regions to meet anonymity constraints (such as *k-anonymity*) and reports the spatial region to LBS server. However, this approach may lead to the degradation of the quality of service. Another representative approach is dummy location [8, 9], which refers to the technique that reports multiple false locations together with the real location to confuse the adversary about the real location. However, it does not apply to the social network that emphasizes content authenticity.

Anonymity based approaches: The anonymity based approaches remove users' real names or replace users' identities with pseudonyms [10-13]. The mere anonymity is not sufficient to ensure user's location privacy [10]. Barkhuus et al. [10] proposed to frequently change pseudonyms over time, and further introduced the concept of mix zone. Palanisamy et al. [11] considered the mix-zone geometry and road characteristics against timing attack and transition attack. Combing with social network, Zhao et al. [12] proposed a location privacy preserving framework based on Searchable Encryption. Li et al. [13] proposed fine grained privacy preserving location query protocol based on Homomorphic Encryption. However, the cryptographic computations are costly to the mobile users.

Moreover, Existing techniques mostly focus on "single shot" scenario, which, however fall to protect privacy of LBSN users when applied to inference attack due to spatio-temporal correlation between published contents. There has been some work to protect against adversaries aware of some spatial-temporal correlations. Shokri et al. [14] quantified location privacy in the scenario of sporadic location exposure. Cheng et al. [15] discussed the linkage attacks based on the knowledge about maximum velocity of uses. The work is improved by Ghinita et al. [16] and Wang et al. [7] using spatial cloaking and introducing delays. However, the work by [7, 14-16] does not provably protect privacy against adversary aware of context correlation beyond the max velocity.

VI. Conclusion

In this paper, observing that the privacy requirements of users are not constant and isolate in LBSN, we formalize this as *Context-aware Location Privacy Protection* (CLPP) problem. To this end, we use *s*-confidence to evaluate users' location privacy. We further use Bayesian inference to derive the probability of the visited POIs, and on this basis we present a CLPP server framework to securely release user's geo-location content. To achieve online evaluating, we propose two novel algorithms to calculate the correlation between the locations offline. The Basic algorithm makes use of majority users' behavioral patterns. The CF-based algorithm further considers users' own personal characteristic. Extensive experiments demonstrate the validity and practicality of the proposed strategy.

VII. Acknowledgments

This research was partially supported by the National Basic Research Program of China (973 Program) under grant No. 2011CB302605, the National High Technology Research and Development Program of China (863 Program) under grant No. 2011AA010705, the National Science Foundation of China (NSF) under grants No.61100188, No.61173144, No.61402137, and No. 61402149.

References

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

[2] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In Proc. *of IEEE ICDCS*, 2005.

[3] A. Noulas, S. Scellato, C. Mascolo, et al. An Empirical Study of Geographic User Activity Patterns in Foursquare. In Proc. *of ICWSM*, 2011.

[4] J. Han, J. Pei, Y. Yin, et al. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data mining and knowledge discovery*, 1: 53-87, August 2004.

[5] H. Gao, J. Tang, and H. Liu. Exploring Social-Historical Ties on Location-Based Social Networks. In Proc, *of the Sixth International AAAI Conference on Weblogs and Social Media*, 2012.

[6] H.P. Li, H. Hu, J. Xu. Nearby friend alert: location anonymity in mobile geosocial networks. *Pervasive Computing*, 4: 62-70, December, 2013.

[7] Y. Wang, D. Xu, X. He, et al. L2P2: Location-aware location privacy protection for location-based services. In Proc. *of IEEE INFOCOM*, 2012.

[8] B. Niu, Q. Li, X. Zhu, et al. Achieving k-anonymity in privacy-aware location-based services. In Proc. *of IEEE INFOCOM*, 2014.

[9] B. Niu, Q. Li, X. Zhu,et al. Enhancing privacy through caching in location-based services, in Proc. *of IEEE INFOCOM*, 2015.

[10] L. Barkhuus, A. Dey, Location-based services for mobile telephony:a study of users' privacy concerns. In Proc. *of the 9th International Conference on Human-Computer interaction*, 2003.

[11] B. Palanisamy, L. Liu. Attack-resilient mix-zones over road networks. *Architecture and algorithms*, 2013.

[12] X. Zhao, L. Li, G. Xue. Checking in without worries: Location privacy in location based social networks. In Proc. *of IEEE INFOCOM*, 2013.

[13] X. Li, T. Jung. Search Me If You Can: Privacy-preserving Location Query Service, In Proc. *of IEEE INFOCOM*, 2013.

[14] R. Shokri, G. Theodorakopoulos, G. Danezis, et al. Quantifying location privacy: the case of sporadic location exposure. *Privacy Enhancing Technologies. Springer Berlin Heidelberg*, 57-76, 2011.

[15] R. Cheng, Y. Zhang, E. Bertino, et al. Preserving User Location Privacy in Mobile Data Management Infrastructures. In Proc. *of Privacy Enhancing Technologies* (PET), 2006.

[16] G. Ghinita, M.L. Gabriel, et al. preventing velocity-based linkage attacks in location-aware applications. In Proc. *of ACM SIGSPATIAL*, 2009.