# Preventing Denial-of-request Inference Attacks in Location-sharing Services

Kazuhiro Minami
Institute of Statistical Mathematics, Tokyo, Japan
Email: kminami@ism.ac.jp

***Abstract*—Location-sharing services (LSSs), such as Google Latitude, have been popular recently. However, location information is sensitive and access to it must be controlled carefully. We previously study an inference problem against an adversary who performs inference based on a Markov model that represents a user's mobility patterns.**

**However, the Markov model does not capture the fact that a denial of a request enforced by the LSS itself implies that a target user is visiting some private location. In this paper, we develop an algorithmic model for representing this new class of inference attacks and conduct experiments with a real location dataset to show that threats posed by the denial-of-request inference attacks are significantly real.**

## I. Introduction

The rise of location-sharing services (LSSs), such as Google Latitude [1]), on a number of mobile platforms have opened up the possibilities of sharing location information with other users. However, location sharing raises significant privacy concerns [2] because location data (e.g., visiting a hospital) can be used to infer a user's personal activities. Therefore, many LSSs provide an access control mechanism that allows a user to define a privacy policy consisting of a set of private locations; that is, a LSS publishes the user's location trajectory while suppressing data points mentioned in her privacy policy.

However, such naive access-control mechanisms are not effective in protecting a user's private information since it is often possible to infer a user's suppressed location data from disclosed public locations considering strong spatial and temporal correlation among them. For example, a person traveling along a trajectory is likely to remain along that path; walking and driving paths follow a predictable pattern, following streets and sidewalks. Furthermore, each person usually exhibits some specific moving patterns reflecting their daily routine activities; that is, similar moving paths repeatedly appear in a user's trajectory data.

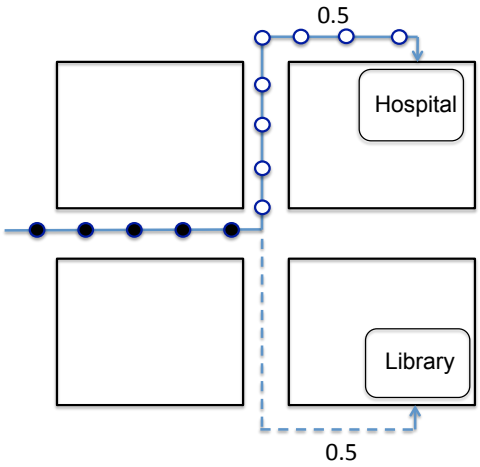
To prevent such an inference attack, Minami and Borisov [3] previously develop an access-control scheme that incorporates a user's probabilistic mobility patterns as a Markov chain. The main idea is to disclose a user's location data only if an unauthorized user possessing the Markov chain of that user as external knowledge cannot predict that the target user moves to a private location with a sufficiently high probability. Figure 1(a) shows two potential walking paths of Bob leading to a hospital and a library. Suppose that given background knowledge, it is possible for Alice to infer that Bob traveling towards the intersection is likely to visit one of these two places and that Bob wishes to hide visits to the hospital while he does not mind revealing his visits to the library. The access-control scheme in [3] stops revealing his location when he turns left at the intersection since, otherwise, Alice can predict that Bob is visiting the hospital with a high probability.

However, [3] does not consider indirect information disclosure through the *denial* of a service request. That is, if a request for a target user's location is denied, an adversary who knows the mechanism of the access-control scheme can infer that the target user is visiting some private location. Suppose that Bob visits the library in another occasion, as shown in Figure 1(b). The LSS this time publishes all the trajectory data to the library. Therefore, when Bob's location data is suppressed after he passes the intersection in Figure 1(a), the adversary can learn that Bob is visiting the hospital since there is no other possible private location nearby.

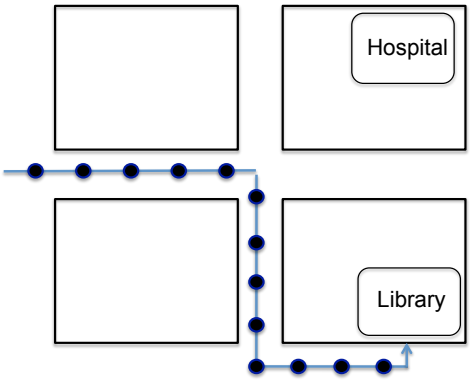
We, therefore, develop a new access-control algorithm extending [3] for preventing the denial-of-request inference attacks above. Our algorithm ensures that whenever a LSS suppresses a user's location data, there are sufficient uncertainty about the private destination of the user; that is, an adversary cannot narrow down the exact private location with a high probability though the adversary knows that the user's approaching *some* private location. Figure 1(c) shows an example trajectory of Bob's visiting the library based on our proposed method. Notice that our new scheme suppresses the location data before Bob reaches the intersection.

We also quantitatively evaluate the significance of threats posed by the denial-of-request inference attacks using the GPS trajectory dataset called "Geolife Trajectories Version 1.3" provided by Microsoft Research Asia [4]. We conduct various experiments varying several important system parameters and show that threats posed by the new inference attacks identified in this paper are significantly real.

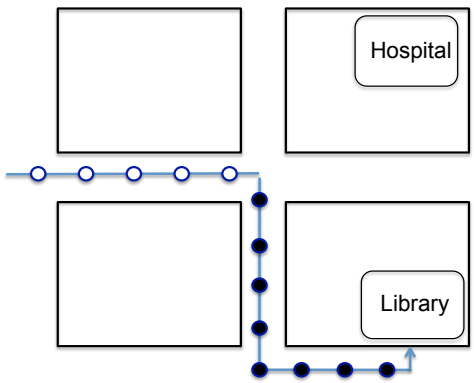
The rest of the paper is organized as follows. Section II introduces the system model and the location privacy metrics of LSSs in our previous research. Section III formally defines a denial-of-request inference attack and presents a new algorithm for preventing such inference attacks. We show our experimental results to evaluate the significance of the new inference attacks in Section IV and discuss related work in Section V. We finally conclude in Section VI.




(a) Example disclosure of location trajectory visiting the hospital based on [3]. The solid line represents an actual path of Bob visiting a hospital. We assume that Bob has 50% chance of visiting of the hospital when he is at the intersection.




(b) Example disclosure of location trajectory visiting the library based on [3].




(c) Example disclosure of location trajectory visiting the library based on our new proposed method.

Fig. 1. Examples of releasing a user's location trajectory. The black dots denote published location data while the white dots denote suppressed ones.

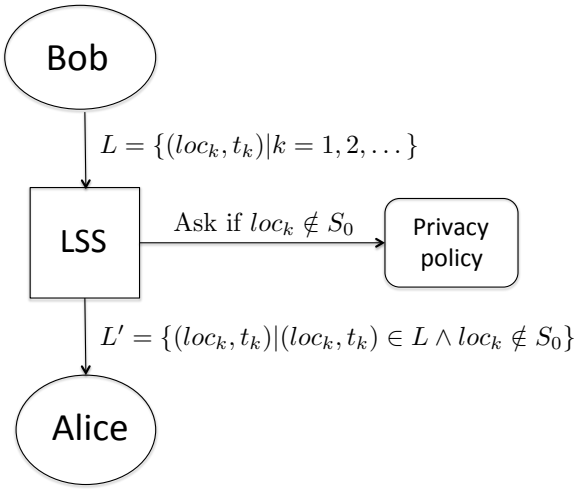



Fig. 2. System model.

## II. Background

In this section, we summarize our system model for LSSs and the metrics for location privacy in [3].

### A. System model

Figure 2 shows our system model for a LSS. We assume that a Alice is interested in receiving Bob's location movements. Bob, carrying a GPS-enabled mobile devices periodically sends location-timestamp pairs $(loc_k, t_k)$ to the LSS for $k \in \mathcal{N}$. In our model, the LSS is completely trusted and receives all of the pairs:

$$L = \{(loc_k, t_k) \mid k \in \mathcal{N}\}.$$

Bob also defines his private policy, which is a set of private locations $S_0$, which he wish to hide from Alice. The LSS releases Bob's location movement $(l_k, t_k)$ to Alice only if $l_k$ does not belong to set $S_0$, and thus Alice receives a subset of events $L' \subseteq L$.

$$L' = \{(loc_k, t_k) \mid (loc_k, t_k) \in L \wedge loc_k \notin S_0\}.$$

To simplify the presentation, we consider privacy policies that depend on location only and assume that Alice is the only user accessing the LSS. It would be easy to generalize our privacy policy to consider a timestamp $t_k$ and a request user $u$.

### B. Location privacy metrics

We next describe how we model Alice's inferences by a location predictor based on the Markov model, and give a precise definition of privacy metrics based on probabilistic inference.

We can represent Bob's potential locations with random variables

$$L_1, L_2, L_3, \dots$$

where each $L_i$ has a value drawn from the finite set of locations $\mathcal{L}$. For simplicity, we assume that location information is

updated at regular intervals and dispense with the timestamp $t_k$. We will use the Markov model of order 1[1] to predict the location, which assumes that a location depends *only* on the previous state.

We can represent this Markov chain as a $|\mathcal{L}| \times |\mathcal{L}|$ transition matrix, indexed by locations in $\mathcal{L}$. Each matrix entry represents the probability of moving from location $l_i$ to $l_j$:

$$M_{i,j} = Pr(L_{n+1} = l_i | L_n = l_j)$$

for every pair of $l_i$ and $l_j$ in set $\mathcal{L}$. The probability of moving from location $l_i$ to $l_j$ in $n$ time steps can be computed by multiplying the transition matrix $M$ $n$ times as follows:

$$Pr(L_{n+1} = l_i | L_1 = l_j) = M^n_{i,j}.$$

To create the transition matrix, we compute the probabilities based on the past history of a user, as stored by the LSS. Note that we include in the history locations that are not published by the LSS to be conservative. We define location privacy against an unauthorized user Alice with the knowledge of Bob's transition matrix $M$ as follows:

*Definition 1 ($(M, S_0, \delta)$-location privacy.):* Given a transition matrix $M$, a set of private locations $S_0$, and a probability threshold $\delta > 0$, we say that a LSS preserves $(M, S, \delta)$-location privacy, if, whenever a location $l_i$ is released from the LSS, for every $l_j$ such that $l_j \in S_0$,

$$M^n_{i,j} \leq \delta \text{ for all } n = 1, 2, \ldots$$

Intuitively speaking, the above definition requires that Alice cannot predict that Bob is at some private location $l_j \in S$ in some future time with a higher probability than the threshold value $\delta$.

## III. Prevention of Denial-of-Request Inference Attacks

In this section, we introduce the denial-of-request inference attack and present a new access-control algorithm for preventing them.

### A. Denial-of-request inference attacks

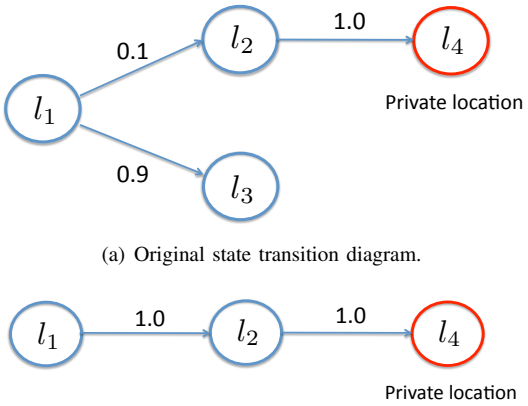
We first informally describe the denial-of-request inference attack with an example in Figure 3(a). Suppose that a target user, Bob's trajectory is $< l_1, l_2, l_4 >$. Also, suppose that a threshold probability $\delta$ is 0.8 and that location $l_4$ is a private location in set $S_0$. the LSS discloses Bob's location at $l_1$ since the probability of moving to private location $l_4$ is

$$M_{1,2} \times M_{2,4} = 0.1 < 0.8.$$

The LSS hides Bob's next location $l_2$ since the probability of moving from $l_2$ to $l_4$ is greater than the threshold value 0.8; that is,

$$M_{2,4} = 1.0 > 0.8.$$

[1]We can easily extend our privacy definition here to support a higher-order Markov model.




(a) Original state transition diagram.

(b) Compressed state transition diagram.

Fig. 3. An example of a denial of request. Each arrow from location $l_i$ to $l_j$ is labeled with a state transition probability $M_{i,j}$. We assume that $l_4$ is a private location in $S_0$.

However, if Bob moves to location $l_3$ instead, the LSS reveals it. Thus, the tracking user, Alice, knows that Bob moves to $l_2$ and then surely moves to private location $l_4$. Given such a denial-of-request event, we observe that Alice can essentially rule out transition branches that do not cause the suppression of location data and obtain a compressed state transition diagram in Figure 3(b). This alternate diagram implies that the LSS must hide Bob's location at $l_1$ as well.

We next model an adversary as a predictor with a compressed state transition matrix $M'$ that is converted from the original matrix $M$. We first introduce several notations below.

*Definition 2 (Probablility $P(i, j, t)$):* Let $P(i, j, t)$ be the probability of visiting location $l_j$ $t$ steps after location $l_i$ is released.

*Definition 3:* Let $RELEASE(l_i)$ and $DENY$ be an event of a LSS's releasing location $l_i$ and that of the LSS's suppressing the current location, respectively.

*Definition 4 (Non-releasable):* We say that a location $l_i$ is non-releasable if $l_i$ is a private location in set $S_0$ or releasing $l_i$ allows Alice to predict a visit of a private location $l_k$ with a higher probability than a given threshold $\delta$.

*Definition 5 (DENY conditional probability):* We compute a conditional probability of visiting location $l_k$ at $t$ steps given a release of location $l_i$ followed by a DENY event as follows:

$$P((l_k, t) \mid RELEASE(l_i), DENY) = \frac{\sum_{l_j \in S} M_{i,j} P(j, k, t-1)}{\sum_{l_m \in S} M_{i,m}}$$

where $S$ is a set of non-releasable locations including $S_0$.

*Definition 6 (Function releasable):* The function *releasable* that takes a location $l_j$ and a set of non-releasable locations $S$ as inputs returns *true* if the following statement holds; otherwise, it returns *false*.

$$\forall l_k \in S, t \in \mathcal{N} : P((l_k, t) \mid RELEASE(l_i), DENY) \leq \delta$$

### B. Algorithm for denial-of-request inferences

We now describe Algorithm 1 that computes a compressed state transition matrix $M'$ and a set of non-releasable locations $S$ from a given state transition matrix $M$, a set of private locations $S_0$, and a threshold probability $\delta$, and the upper bound of the number of inference steps $N_{max}$. We denote the set of non-releasable locations by $S$.

Algorithm 1 first computes the set of non-releasable locations $S$, which do not satisfy the function $releasable$ in Definition 6, which computes DENY conditional probabilities in Definition 5 inside. After we obtain the final set $S$, we remove from matrix $M$ state transitions that do not cause a DENY event and normalize the probability distribution of the rest of the transitions accordingly.

We now look at the algorithm in detail. Line 4 initializes the set $S$ to be $S_0$. Lines 5–7 initializes the probability of moving location $l_i$ to $l_j$ at step 0, $P(i, j, 0)$, to 1.0 if $i = j$; otherwise to 0. The while loop in lines 8–10 add non-releasable location $l_i$ into set $S$ incrementally. The function $releasable$ in Algorithm 2 only considers transitions that cause a DENY event at the next time and checks whether it is possible to predict a visit of private location $l_k$ at step $t$ with a probability greater than threshold $\delta$. Since we cannot examine infinite number of time steps, we use $N_{max}$, which is given as an input of Algorithm 1, as the upper bound of time steps. After we obtain the final set of non-releasable locations $S$, lines 11–21 normalize the state transition probability of the original matrix $M$ by considering only transitions to non-releasable locations in set $S$. Line 22 finally outputs the converted compressed matrix $M'$ and the final set of non-releasable locations $S$.

Due to space limitation, we omit the process in Algorithm 2 of inductively computing probability $P(i, j, t)$ in Definition 2 for every pair of locations $(l_i, l_j) \in \mathcal{L} \times \mathcal{L}$ and every time step $t \in \{1, 2, \ldots, N_{max}\}$. Note that every $P(i, j, t)$ must be recomputed in Algorithm 2 every time the set of non-releasable locations $S$ is updated in line 9.

### C. Prevention of the denial-of-request inference attacks

Our new access-control scheme uses a compressed state transition matrix $M'$, which is more conservative than the original matrix $M$, and ensure the new safety definition below.

*Definition 7 ($(M', S_0, \delta)$-location privacy.):* Given a compressed transition matrix $M'$ derived from the original state transition matrix $M$, a set of private locations $S_0$, and a probability threshold $\delta > 0$, we say that a LSS preserves $(M', S_0, \delta)$-location privacy, if, whenever a location $l_i$ is released from the LSS, for every $l_j \in S_0$,

$$M'^{n}_{i,j} \leq \delta \text{ for all } n = 1, 2, \ldots$$

Our enforcement algorithm turns out be very simple; we no longer have to perform computations with a compressed matrix $M'$. The LSS releases a location $l_i$ only if $l_i$ does not belong to a set of non-releasable locations $S$ computed by Algorithm 1.

**Algorithm 1** Compute a compressed state transition matrix.

```
1:  % INPUT: M, S_0, δ, N_max
2:  % OUTPUT: M', S
3:
4:  S ← S_0
5:  for all (l_i, l_j) ∈ L × L do
6:      P(i, j, 0) ← δ_ij
7:  end for
8:  while ∃l_i ∉ S : ¬releasable(l_i, S, M, δ, N_max) do
9:      S ← S ∪ {l_i}
10: end while
11: for all (l_i, l_j) ∈ L × L do
12:     if l_j ∈ S then
13:         if ∃l_m ∈ S : M_{i,m} ≠ 0 then
14:             M'_{i,j} ← M_{i,j} / Σ_{l_m ∈ S} M_{i,m}
15:         else if ∀l_m ∈ S : M_{i,m} = 0 then
16:             M'_{i,j} ← 0
17:         end if
18:     else
19:         M'_{i,j} ← 0
20:     end if
21: end for
22: return (M', S)
```

**Algorithm 2** Function $releasable$.

```
1: % INPUT: l_i , S , M , δ , N_max
2: % OUTPUT: isReleasable
3:
4: if ∀(l_k, t) ∈ S × {1, ..., N_max} :
   P((l_k, t)|RELEASE(l_i), DENY) ≤ δ then
5:     isReleasable ← True
6: else
7:     isReleasable ← False
8: end if
9: return isReleasable
```

## IV. Experimental results

We quantitatively evaluate the significance of threats posed by the denial-of-request inference attacks in Section III using the GPS trajectory dataset called "Geolife Trajectories Version 1.3" provided by Microsoft Research Asia [4]. The Geolife dataset contains GPS trajectory data of 178 users in a period of over four years. Each timestamped data point, the majority of which is logged every 1–5 seconds, contains latitude and longitude of a mobile user. Since this dataset recorded a wide variety of users outdoor movements regarding entertainments and sports activities, we believe that the dataset is suitable to the study of location privacy.

We apply both the inference method representing the denial-of-request inference attack in Section III and the previous method in [3] to the dataset, and compare their inference power in terms of the number of locations that are necessary to hide in order to protect a given initial set of private locations.

### A. Conversion to symbolic location data

We need to convert GPS coordinates in the dataset into symbolic locations to represent users' mobility patterns as the Markov model. We consider GPS data whose data points reside within a rectangular region covering the center of Beijing, China and its surrounding areas. The dimension of the region is 39 kilometers (N. latitude 39.75–40.10) times 30 kilometers (E. longitude 116.20–116.55).

We divide each coordinate into 140 units and define 19600 unit regions in the whole rectangular region. The dimension of each region is 275 meters times 213 meters, which is about the size of a typical building in a city. We convert a GPS coordinate into the sequential identifier of a unit region containing that coordinate. We use half of a user's data to construct two state transition matrices $M$ and $M'$ and use the other half to compute the number of non-releasable locations with them.

### B. Experiments with realistic private locations

We first qualitatively evaluate the inference power of the new method with a trajectory location data of a single user whose user ID in the dataset is 68. We choose the user with the largest data points. Since the location data in the Geolife dataset was collected without running any location-based service, the dataset does not contain each user's privacy policy.

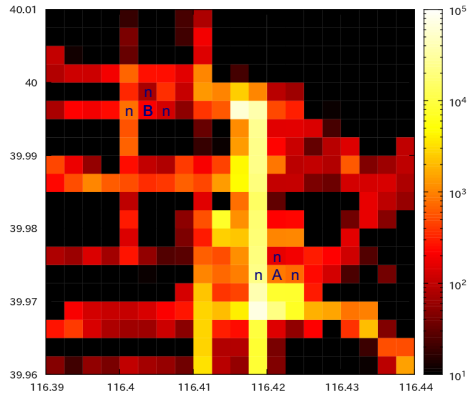
We, therefore, choose two hypothetical private locations of that user from the locations the user actually visited. One is a restaurant at (N. latitude 39.99635, E. longitude 116.40360), and the other is a hospital at (N. latitude 39.97260, E. longitude 116.42072). Since visiting either at a restaurant or a hospital is likely to imply the user's private activity, we consider our choice of two locations as an example of a reasonable privacy policy.

Figure 4 shows the final sets of private locations $S$ produced by Algorithm 1. Figure 4(a) shows the case with a threshold $\delta = 0.8$ and the number of inference steps 1. The points $A$ and $B$ denote the locations of the hospital and the restaurant, respectively, that is, the initial set $S_0$ of private locations contains those two points. The locations labels as 'N' represents non-releasable locations in set $S$. The previous algorithm in [3] does not add any non-releasable location; that is, the final $S$ only contains the initial two locations $A$ and $B$.

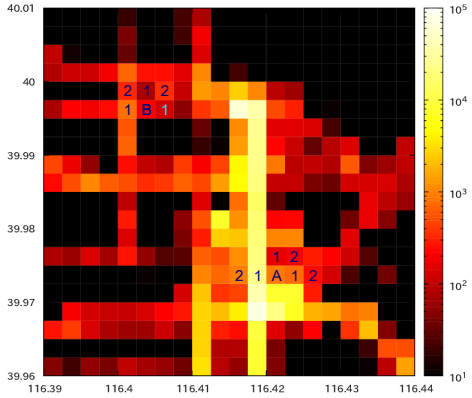
Figure 4(b) shows the case with the number of inference steps 2. The locations labeled '1' are the additional private locations inferred with the new algorithm in one step, and those labeled '2' are private locations inferred in two steps. Again, the previous algorithm does not add any new private locations in this case, either. This small experiment shows that there exist some locations that must be protected against the denial-of-request inference attack but are not properly protected by the previous algorithm.

### C. Experiments with randomly chosen private locations

We next compare the two algorithms while varying a few system parameters. We choose the top ten users in terms of the number of visited locations, and use the three hundred most frequently visited locations of each user as a list of that candidate private locations for that user. We randomly choose a given number of private locations in set $S_0$ from that list when conducting an experiment.




(a) The number of inference steps: 1.

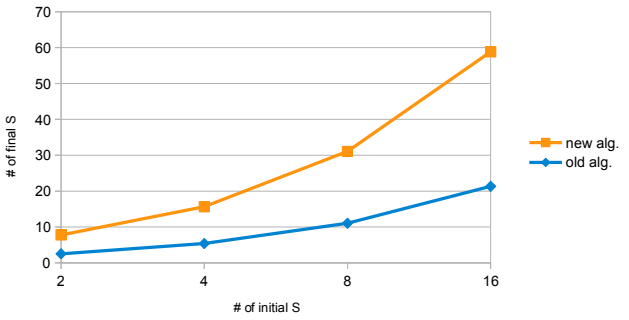



(b) The number of inference steps: 2.

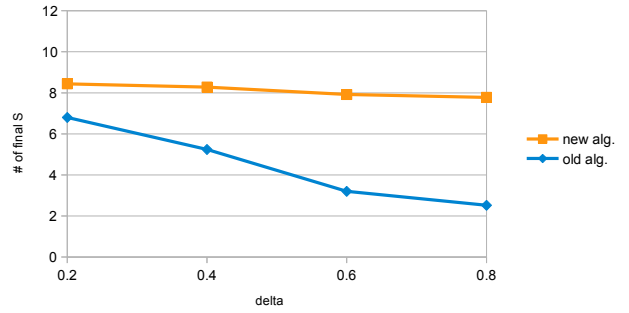
Fig. 4. Non-releasable locations computed from a given two realistic private locations (A: China-Japan Friendship Hospital, B: South Beauty Restaurant). The color of each region shows the number of the user's visits. The threshold $\delta = 0.8$.

Figure 5 shows the experimental results of the two algorithm changing three parameters. We execute five runs on each user's dataset with a given set of system parameters and take the average of the final numbers of non-releasable locations produced by those runs. Next, we take the average of those numbers of ten users.

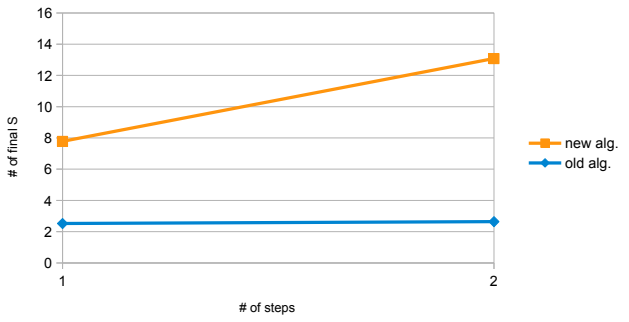
Figure 5(a) shows the dependency of the number of final non-releasable locations to the size of an initial set of private locations. As the number of initial private locations increases, the new algorithm determines significantly more locations as non-releasable than the previous algorithm. Figure 5(b) shows that the new algorithm is insensitive to the value of a threshold while the number of final non-releasable locations produced by the previous algorithm decreases as the value of a threshold increases. Figure 5(c) shows that the inference power of the new algorithm could be stronger as we increase the number of inference steps ahead. On the other hand, that is not effective




(a) Dependency on the number of initial private locations. A threshold $\delta$ is 0.8. The number of inference steps is 1.




(b) Dependency on a threshold. The number of inference steps is 1. The number of initial private locations is 2.




(c) Dependency on the number of inference steps. A threshold $\delta$ is 0.8. The number of initial private locations is 2.

Fig. 5. Comparison of two inference algorithms

with the previous algorithm.

## V. Related work

Several researchers [5], [6], [7], [8], [9] propose rule-based access-control schemes for protecting user location in pervasive environments. Hengartner [5] supports access-control policies considering the granularity of location information and time intervals. Myles [8] provides a XML-based authorization language for defining privacy policies that protect users location information. Although those schemes allow a user to specify fine-grained access-control policies, no previous work along this line considers the inference problem due to the high correlation of location data.

MaskIt [10] is the closest to our work in this paper. It controls access to a time-series of context information (e.g., location) considering an adversary who models a user's behavior as a Markov chain and makes inference about unreleased contexts based on the Hidden Markov model. However, the major difference from ours is that their privacy model considers a difference between the prior and posterior possibility of a user's being at a private location. On the other hand, our model solely considers the magnitude of posterior possibility on the private location.

## VI. Summary

In this paper, we address a new inference problem concerning a denial of service request in location-sharing services (LSSs). We formally define information gain through a denial of request and model a new adversary with a compressed state transition matrix, which eliminate state transitions to publishable locations. Our experimental results show a considerable risk of releasing location data in an unsafe way such that users' private visits can be disclosed with the inference attacks considering the denial of requests.

## Acknowledgments

This research is supported by the Strategic Joint Research Grant for NTT and Research Organization of Information and Systems (ROIS) and by the Grants-in-Aid for Scientific Research C, 11013869, of Japan Society for the Promotion of Science.

## References

[1] "Google latitude," http://www.google.com/latitude. [Online]. Available: http://www.google.com/latitude

[2] D. Anthony, T. Henderson, and D. Kotz, "Privacy in location-aware computing environments," *IEEE Pervasive Computing*, vol. 6, no. 4, pp. 64–72, 2007.

[3] K. Minami and N. Borisov, "Protecting location privacy against inference attacks," in *Proceedings of the 17th ACM conference on Computer and communications security*, ser. CCS '10. New York, NY, USA: ACM, 2010, pp. 711–713. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866406

[4] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th international conference on World wide web*, ser. WWW '09. New York, NY, USA: ACM, 2009, pp. 791–800. [Online]. Available: http://doi.acm.org/10.1145/1526709.1526816

[5] U. Hengartner and P. Steenkiste, "Access control to people location information," *ACM Transactions on Information and System Security (TISSEC)*, vol. 8, no. 4, pp. 424–456, 2005.

[6] J. I. Hong and J. A. Landay, "An architecture for privacy-sensitive ubiquitous computing," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services (MobiSys)*. New York, NY, USA: ACM, 2004, pp. 177–189.

[7] A. Kapadia, T. Henderson, J. J. Fielding, and D. Kotz, "Virtual Walls: Protecting Digital Privacy in Pervasive Environments," in *Proceedings of the Fifth International Conference on Pervasive Computing (Pervasive)*, ser. LNCS, vol. 4480. Springer-Verlag, May 2007, pp. 162–179.

[8] G. Myles, A. Friday, and N. Davies, "Preserving privacy in environments with location-based applications," *IEEE Pervasive Computing*, vol. 2, no. 1, pp. 56–64, January-March 2003. [Online]. Available: http://www.computer.org/pervasive/pc2003/b1056abs.htm

[9] V. Sacramento, M. Endler, and C. de Souza, "A privacy service for location-based collaboration among mobile users," *Journal of the Brazilian Computer Society*, vol. 14, no. 4, pp. 41–57, 2008.

[10] M. Götz, S. Nath, and J. Gehrke, "Maskit: privately releasing user context streams for personalized mobile applications," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '12. New York, NY, USA: ACM, 2012, pp. 289–300. [Online]. Available: http://doi.acm.org/10.1145/2213836.2213870