Personalized Semantic Location Privacy Preservation Algorithm Based on Query Processing Cost Optimization

Mengzhen Xu, Hongyun Xu(✉), and Cheng Xu

School of Computer Science and Engineering,
South China University of Technology, Guangzhou 510006, China
hongyun@scut.edu.cn

Abstract. Location-Based Services (LBSs) are playing an increasingly important role in our daily life with the development of GPS or WiFi enabled space positioning technologies and the popularization of mobile devices. While LBSs bring great conveniences to users, the exposure of users' location privacy becomes a growing concern. To address this issue, researchers propose several kinds of location preservation techniques such as location cloaking, dummies and etc. However, these methods are rather vulnerable when encounter a location semantic attack. Taking this into consideration, a few semantic location preservation methods are proposed. Based on the existing semantic location preservation frameworks, we propose a novel personalized semantic location privacy preservation method named Incremental Search (IS). In our method, an optimal anonymous location set is generated according to a certain rule, during which, two parameters are introduced to limit the number of locations in the final anonymous location set and the number of the anonymous location sets recorded temporarily so as to reduce the query processing cost. The evaluation on NGMO simulation platform validates that our method has a better performance than the two baseline algorithms.

Keywords: Semantic location · Privacy preservation · Incremental search
Query processing cost · Road network

1 Introduction

With the popularization of mobile and locator devices, Location-based services (LBSs) have been used widely. However, people are suffering from the threats of leaking their identities and location information when they are enjoying LBSs. Attackers can infer user's ongoing activities or analyze his habits according to user's information such as current location and time.

Therefore, many researchers aimed at studying the privacy preservation methods in LBSs and proposed a lot of solutions mainly based on K-anonymity [1] and L-diversity [2]. The main idea of K-anonymity is guaranteeing at least other $K-1$ users in the obfuscation region together with the user, then the user's practical position will be

G. Wang et al. (Eds.): SpaCCS 2017, LNCS 10656, pp. 153–168, 2017.
https://doi.org/10.1007/978-3-319-72389-1_14

replaced with the obfuscation region and then sent to the LBS server, so the probability for the attacker to infer the user's location correctly is $\frac{1}{K}$. K-anonymity can protect user's identity and location information, but it does not consider the distribution of the K users in the obfuscation region. L-diversity is a supplementary to K-anonymity which requires anonymity area to contain at least L different locations so as to prevent attackers from linking the user to a location.

However, L-diversity may not always be so effective for all semantic location privacy preservation in some cases. On the one hand, L locations may be the same type, thus user's location information is revealed, on the other hand, the probability of user being in each location may not be equal, thus the attacker can filter out the locations with low probabilities. To the former problem, researchers extract L-diversity to L different types of locations [3] rather than L-occurrence locations. To the latter problem, numerical values are used to quantify the probability of user being in each location, for example, popularity is used in [4, 5].

Generally speaking, a user may be sensitive to some locations while not to other locations. If users stay at sensitive locations, they need privacy preservation, otherwise, they do not need. Yigitoglu et al. [4, 5] use privacy profile to define user's sensitive locations, use popularity to define all kinds of locations and use the ratio of the aggregated popularity of the sensitive locations to the aggregated popularity of all locations to judge whether the anonymous area meets the privacy requirement. However, the algorithms [4, 5] just aim to find an anonymous area satisfying privacy requirement without considering the number of locations, which may lead to high query processing cost. Obviously, on the condition of meeting the privacy requirement, fewer locations will lead to lower query processing cost.

Our paper proposes a novel personalized semantic location privacy preservation algorithm named incremental search (IS), which aims to reduce query processing cost.

The main contributions of our paper are listed below:

(1) We propose to select anonymous locations based on global optimization and local optimization to protect users' location privacy under semantic environment.
(2) Two parameters are introduced to limit space cost and time cost.
(3) We compare our algorithm with two existing algorithms based on a real map, the result shows that our algorithm can improve anonymous success rate and reduce query processing cost.

The rest parts are organized as follows. In Sect. 2 we review the related work. Section 3 introduces the system model. Section 4 describes the algorithm in detail. The experimental results are shown in Sect. 5. And we conclude the paper in Sect. 6.

2 Related Work

Previous work in location privacy preservation mainly includes dummies [6], mix-zone [7], K-anonymity [1, 8–11], obfuscation and coordinate transformation [12], cryptography based method [13]. K-anonymity is widely used which means user cannot be

recognized with at least other $K-1$ users by using location or identity. Based on K-anonymity, many other approaches have been proposed, such as Interval-cloak [8], Casper [9], Clique-Cloak [10] and HC [11].

However, these approaches do not consider spatial context. Therefore, Bamba et al. propose PrivacyGrid [14], a system which can support both K-anonymity and L-diversity. In their approach, an anonymous region contains at least K users and L locations. However, the region in [14] is not L-type diverse. For example, L locations may all be the same type (e.g. hospital), which is vulnerable to location similarity attack [5]. Therefore, Xue et al. [3] define L-diversity to be L different types. These approaches do not consider that different types of locations have different probabilities of being visited; Probe [15–17] takes this situation into consideration by introducing the concept of popularity to measure the visiting probabilities of each type of locations, what's more, it classifies locations into sensitive locations and non-sensitive locations, then uses obfuscation method to complete privacy preservation. Byoungyoung [18] proposes a method that uses EMD (earth mover's distance) to mine location semantic information to avoid privacy being revealed, but this is only suitable for Euclidean space. Yigitoglu et al. [4] and Li [5] extend these approaches into road networks. In [4], they generate the city network from a map on which each location is a node, then use breadth first search algorithm to find a location set which makes the proportion of the popularity of the sensitive locations less than a certain threshold. In [5], they select all non-sensitive semantic locations from the neighbor locations of the anonymous area and add them into the anonymous area in turn, until it satisfies the privacy requirement or the terminal conditions. However, all of those approaches do not consider the number of locations in the anonymous set, which may lead to high query processing cost.

This paper proposes an algorithm to protect user's location privacy under the semantic environment, which is suitable for the road-network environment with higher success rate and lower query processing cost.

3 System Model

3.1 Semantic Location City Network

Definition 1 (Semantic location city network). A semantic location city network is modeled as a connected and undirected weighted graph G, $G = (V, E, pt, pop)$, where:

(1) V is the set of vertices, $V = Vp \cup Vi$, Vp is the set of semantic locations and Vi is the set of road intersections.
(2) E is the non-empty set of edges, $E \subseteq V \times V$, $E = \{e \mid e = (u, v)$, where $u, v \in V\}$.
(3) pop is the popularity of v ($v \in V$), the popularity of v is the probability that a user visit the location v. pt is the type of v, each location's type is represented by function pt: V –> PT (PT is the set of location types).

For simplicity, we assume that locations with the same type have the same popularity, represented by function *pop*: *PT* –> [0, 1), the popularity of road intersection is 0 (In this paper, we assume a road intersection belongs to a location which is not semantic). The popularity of semantic location v is denoted as $p(v)$, the popularity of semantic location set s is denoted as $p(s)$.

3.2 System Structure

Figure 1 depicts the Central Server Structure designed to provide location privacy preservation. We add a Location Privacy Server (LPS) between mobile users and the LBS server. LPS stores users' privacy profiles (*pp*). The process for user to complete a query is as follows: 1. User sends the query request. 2. LPS receives the query and judges whether the user's location is sensitive; if it is non-sensitive, LPS sends the real location to LBS server; if it is sensitive, LPS runs location privacy preservation algorithm and generates a location set based on the user's location, then sends the location set to LBS server. 3. LBS server calculates the candidate sets based on the real location or the location set and sends the candidate sets to LPS. 4. LPS filters the candidate sets and sends the appropriate results to the user. Our work focuses on step 2, namely providing privacy preservation for user's requests from sensitive locations.

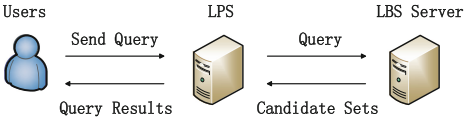


Fig. 1. Central server structure

3.3 Privacy Requirement

Let $PT = PTs \cup PTns$, where *PTs* represents the sensitive type set, *PTns* represents the non-sensitive type set. Let $Vp = Vsp \cup Vnsp$, *vsp* represents a sensitive semantic location, $vsp \in Vsp$, *vnsp* represents a non-sensitive semantic location, $vnsp \in Vnsp$.

Definition 2 (Anonymous location set, *ALS*). *ALS* is a location set used to achieve anonymity. The element in *ALS* can be a sensitive semantic location, a non-sensitive semantic location or a road intersection.

Definition 3 (θ-secure *ALS*). If an *ALS* satisfies Eq. (1)

$$\frac{\sum_{VSP \in VS \cap VSP \in ALS} P(VSP)}{\sum_{VP \in VS \cap VP \in ALS} P(VP)} \leq \theta \tag{1}$$

We denote this *ALS* as a θ-secure *ALS*. In other words, θ-secure *ALS* satisfies the following condition: the ratio of the aggregated popularity of all sensitive semantic locations (*APSSL*) to the aggregated popularity of all semantic locations (*APSL*) in an *ALS* should be less than or equal to θ.

Definition 4 (*Lmax*). It represents the maximal number of locations in an *ALS*. Although more locations in an *ALS* will provide better privacy preservation for users, it will also result in more query processing cost to the LBS server. To balance privacy preservation and query processing cost, we propose parameter *Lmax* which is decided by users.

Definition 5 (*Cmax*). It represents the maximal number of *ALS* to be recorded. Although recording more *ALS* will make the final *ALS* easier to contain fewer locations, it will result in larger space and time cost to LPS server. To balance the number of locations in the final *ALS* and the cost of LPS, we propose parameter *Cmax* which is also decided by users.

Definition 6 (Deficient popularity, *DP*). It represents the least popularity that an *ALS* needs to be a θ-secure *ALS*. For example, an *ALS*'s *APSSL* is 0.3 and its *APSL* is 0.5. If $\theta = 0.3$, we can calculate the *DP* of the *ALS* is $\frac{0.3}{0.3} - 0.5$, *we got* 0.5 *DP* is used to achieve local optimization.

4 Algorithm

We describe IS algorithm in this section. Before executing IS algorithm, the user should provide *pp* and the values of *Cmax* and *Lmax*. *pp* contains: (1) Sensitive location type set $PTs = \{pts1, pts2, \ldots, ptsn\}$. (2) The value of θ.

The pseudo-code of this algorithm is given in Algorithm 1. The algorithm is accomplished by looping. During each loop we search new *ALS* by adding an adjacent location to the recorded *ALS*; we keep two sets *curSet* and *nextSet*, *curSet* records the *ALS* achieved in last loop, *nextSet* records new *ALS* searched through *ALS* in *curSet*. Two hash tables *sp* and *tp* are defined to record the *APSSL* and *APSL* for every *ALS* in *curSet* and *nextSet*. Both *sp* and *tp* will be updated when a new *ALS* is found, *curSet* will be updated in the end of the loop. The algorithm terminates when a θ-secure *ALS* is found or the number of locations of an *ALS* in *curSet* reaches *Lmax*.

As mentioned before, parameter *Cmax* is defined to limit the number of *ALS* being recorded (In Algorithm 1, the *ALS* are recorded in *curSet* and *nextSet*). Therefore, if the number of *ALS* in *nextSet* is larger than *Cmax*, we will select the top *Cmax* local optimal *ALS*

Algorithm 1

Input: semantic location city network $G = (V, E, pt, pop)$, user's location loc_u, privacy profile $pp = \{PTs, \theta\}$, parameter $Lmax$, parameter $Cmax$.

Output: a θ-secure ALS.

```
1:  curSet = {{ loc_u}}, nextSet = {}, sp.put({loc_u}, p({loc_u})), tp.put ({loc_u}, p({loc_u})
2:  While (true) do
3:      For each seti in curSet do
4:          If seti.size > Lmax then
5:              return false;
6:          For each setii in seti do
7:              Temp = seti
8:              Vadj = getAdj(setii)      // Vadj is the set of all adjacent nodes of node setii
9:              For each node in Vadj do
10:                 If !Temp.contains(node) then
11:                     Temp.add(node)
12:                     nextSet.add(Temp)
13:                     tp.put (Temp, tp.get(seti) + p (node))      //update tp
14:                     If PTs.contains(node.type) then             //update sp
15:                         sp.put(Temp, sp.get(seti) + p(node))
16:                     Else
17:                         sp.put(Temp, sp.get(seti))
18:                     End
19:                     If(sp.get(Temp) / tp.get(Temp) ≤ θ) then
20:                         return Temp;
```

```
21:                     End
22:                 End
23:                 Temp = seti;
24:             End
25:         End
26:         sp.remove(seti)
27:         tp.remove(seti)
28:     End
29:     curSet.clear();
30:     curSet = Get_Cmax(nextSet, sp, tp, θ);
31:     nextSet.clear();
32: End
33:
34://Select Cmax local optimal ALS from nextSet
35: Set Get_Cmax(nextSet, sp, tp, θ) {
36:     If nextSet.size > Cmax then
37:         For each seti in nextSet do          //calculate DP for every ALS in nextSet
38:             dp = sp.get(seti) / θ - tp.get(seti)
39:         End
40:         nextSet.sort_by(dp)                  // sort nextSet as the value of DP increase
41:         newSet = nextSet.first(Cmax)         // select the first Cmax values in nextSet
42:         return newSet
43:     Else
44:         return nextSet
45:     End
46: }
```

Figure 2 depicts a semantic location city network. Each vertex has two attributes, the first one represents identity and the second one represents type. Assuming that the privacy profile pp = {{H,O}, 0.5}, the popularities of each location type are as follows, {School (S): 0.2, Hospital (H): 0.15, Office (O): 0.25, Entertainment (E): 0.15, Mall (M): 0.15, Park (P): 0.1, Intersection (I): 0}. User locates at *v1*, *Lmax* = 3, *Cmax* = 1. In the beginning, *curSet* = {{*v1*}}, *sp* = {{*v1*} -> 0.15}, *tp* = {{*v1*} -> 0.15}, *nextSe t* = {}, there is only one *ALS* ({v1}) in *curSet*.

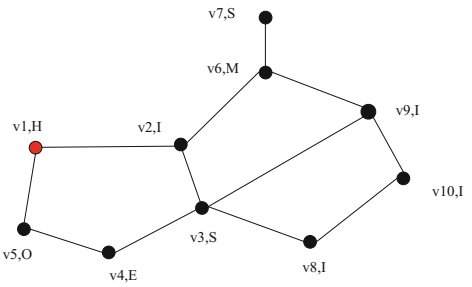


Fig. 2. An example of IS algorithm

In the first round, the number of locations for each *ALS* is 1, which is smaller than *Lmax*, so continue searching. The adjacent locations of *ALS* are *v2*, *v5*. select *v2* first, then we obtain *ALS* = {*v1*, *v2*}, *sp*({*v1*, *v2*}) = 0.15, *tp*({*v1*, *v2*}) = 0.15. Since $\frac{0.15}{0.15} > 0.5$, it does not meet the privacy requirement, add {*v1*, *v2*} to *nextSet*. Then select *v5*, and we obtain *ALS* = {*v1*, *v5*}, *sp*({*v1*, *v5*}) = 0.4, *tp*({*v1*, *v5*}) = 0.4. Since $\frac{0.4}{0.4} > 0.5$, it does not meet the privacy requirement, add {*v1*, *v5*} to *nextSet*. We obtain *nextSet* = {{*v1*, *v2*}, {*v1*, *v5*}}, *sp* = {{*v1*, *v2*} -> 0.15, {*v1*, *v5*} -> 0.4}, *tp* = {{*v1*, *v2*} -> 0.15, {*v1*, *v5*} -> 0.4}. The number of sets in *nextSet* is 2, which is larger than *Cmax*, so we need to select the top *Cmax* local optimal *ALS*. The *DP* of *ALS*{*v1*, *v2*} is $\frac{0.15}{0.5} - 0.15 = 0.15$ and the *DP* of *ALS*{*v1*, *v5*} is $\frac{0.4}{0.5} - 0.4 = 0.4$ Because 0.15 < 0.4, so we select set {*v1*, *v2*}. Thus *curSet* = {{*v1*, *v2*}}, *sp* = {{*v1*, *v2*} -> 0.15}, *tp* = {{*v1*, *v2*} -> 0.15}, *nextSet* = {}.

In the second round, the number of locations for each *ALS* in *curSet* is 2, which is smaller than *Lmax*, so continue searching. The adjacent locations of {*v1*, *v2*} are *v5*, *v3*, *v6*. Select *v5* first, then we obtain *ALS* = {*v1*, *v2*, *v5*}, *sp*({*v1*, *v2*, *v5*}) = 0.4, *tp*({*v1*, *v2*, *v5*}) = 0.4. Since $\frac{0.4}{0.4} > 0.5$, it does not satisfy the privacy requirement, add {*v1*, *v2*, *v5*} to *nextSet*. Then select *v3*, and we obtain *ALS* = {*v1*, *v2*, *v3*}, *sp*({*v1*, *v2*, *v3*}) = 0.15, *tp*({*v1*, *v2*, *v3*}) = 0.35. Since $\frac{0.15}{0.35} < 0.5$, it satisfies the privacy requirement, so {*v1*, *v2*, *v3*} is sent as the final *ALS* to LBS server. The algorithm terminates.

Assuming that the number of adjacent locations for each location is a constant value *A*, the number of recorded *ALS* is *C* and the number of locations in an *ALS* is *L*. For simplicity, the case that two *ALS* become equal after each of them adding one location respectively is ignored.

When $C < Cmax$ and $L < Lmax$: if the number of locations in *ALS* changes from m to $m + 1$, the number of *ALS* that are needed to be recorded increases $A*m$ times, so the space complexity and the time complexity are both $O(A^L*L!)$.

When $C = Cmax$ and $L < Lmax$: the number of ALS that are needed to be recorded is a constant, thus the space complexity is $O(L)$ and the time complexity is $O(L^2 logL)$.

In IS algorithm, both *Lmax* and *Cmax* may affect the running time. Larger *Lmax* or *Cmax* may result in longer running time and larger storage space, so it is important to choose the appropriate values of *Lmax* and *Cmax*. Generally speaking, *Lmax* and *Cmax* can be larger if the machine's performance is good enough. More discussions about these two parameters will be shown in next section.

5 Experimental Analysis

In this section, we evaluate the performance of the proposed algorithm. Two different *PTs* are used where PTs_1 = {Entertainment} and PTs_2 = {Hospital, Office}. Overlapping algorithm in [4] and SA algorithm in [5] are implemented for comparison.

5.1 Experiment Setting

The algorithms are realized using Java and the coding environment is Eclipse. The experimental platform consists of a desktop PC equipped with an Intel(R) Pentium (R) 4 2.66 GHz CPU and 2 GB of RAM. The famous Network-based Generator of Moving Objects (i.e., NGMO) simulation platform is used to accomplish the experiments. We use the map of Oldenburg city in Germany, which contains 6105 nodes and 7035 edges. Raw data is processed according to Definition 1 to generate semantic location city network. We traverse every location and send query request. In this experiment, we just discuss six types of semantic location whose popularity are the same as we assumed in Sect. 4. Experiment parameters are shown in Table 1.

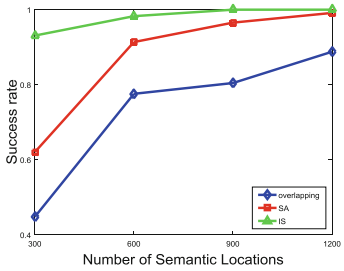
Table 1. Parameter settings

Parameter	Default value	Range
θ	0.4	[0.2, 0.8]
Number of semantic locations	600	[300, 1200]
Location types (counts)	School (S:64), Office (O:174), Hospital (H:72), Market (M:100), Entertainment (E:58), Park (P:132)	/
Lmax	30	[20, 40]
Cmax	1000	[100, 1000]

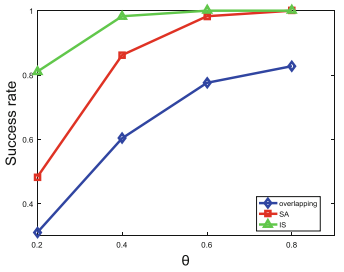
5.2 Experimental Result

Success rate is the ratio of the number of successful anonymous sensitive semantic locations to the number of total sensitive semantic locations. It is an important metric

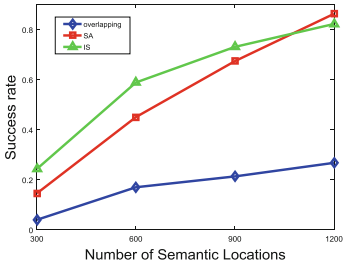
that reflects the effectiveness of a privacy preservation algorithm. If the success rate is higher, the privacy preservation algorithm is more effective. Figure 3 depicts the tendencies of success rate with varying number of semantic locations and θ when sensitive location set is PTs_1 and PTs_2 respectively.



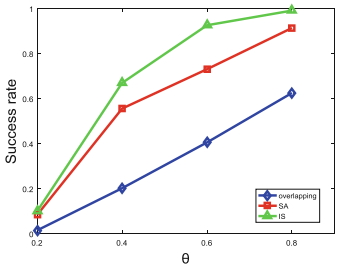
(a) θ takes default value ,using PTs_1



(b) Number of semantic locations takes default value, using PTs_1



(c) θ takes default value,using PTs_2

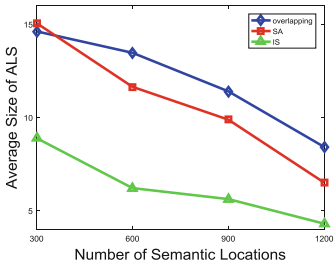


(d) Number of semantic locations takes default value, using PTs_2

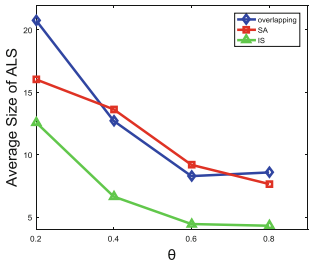
Fig. 3. Relation between success rate and the number of semantic locations or θ

From Fig. 3, we can see when the number of semantic locations or θ increases, success rates of three algorithms all increase. This is because when the number of semantic locations increases, both the number of sensitive and the number of non-sensitive semantic locations increase, which makes it easier to find a θ-secure *ALS*; when θ increases, privacy requirement is easier to be satisfied. IS has the highest success rate among three algorithms, since it records many (no more than *Cmax*) *ALSs* and is more possible to find a θ-secure *ALS*. For SA, it records only one *ALS*, thus it is harder to get a θ-secure *ALS* compared with IS. For overlapping, it fails when the new searched location is not a non-sensitive location, which makes its success rate lower.

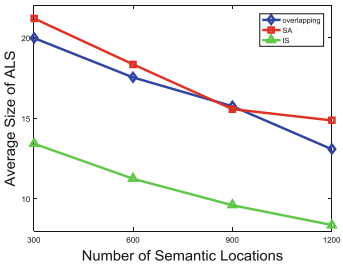
Average size of *ALS* is the average number of locations in an *ALS* during a successful query, it can be used to measure query processing cost at LBS server. Figure 4 depicts the tendencies of average size of *ALS* with varying number of semantic locations and θ when sensitive location set is PTs_1 and PTs_2 respectively.



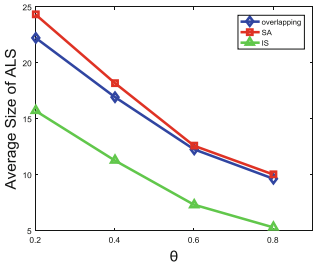
(a) θ takes default value, using PTs_1



(b) Number of semantic locations takes default value, using PTs_1



(c) θ takes default value,using PTs_2



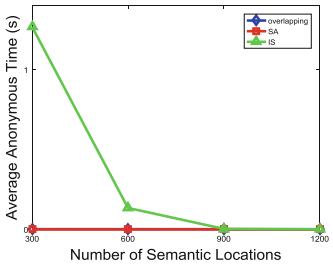
(d) Number of semantic locations takes default value,using PTs_2

Fig. 4. Relation between average size of *ALS* and the number of semantic locations or θ

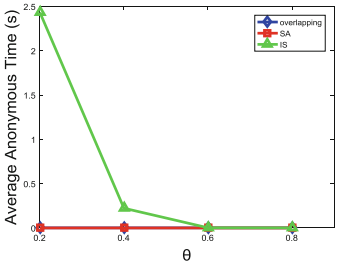
From Fig. 4, we can see when the number of semantic locations or θ increases, the average sizes of *ALS* of three algorithms all decrease. This is because when the number of semantic locations increases, it is easier to find an adjacent location which is non-sensitive, thus fewer locations are needed to get a θ-secure *ALS*; when θ increases, privacy requirement is easier to be satisfied, which leads to a smaller average size of *ALS*. IS algorithm has the smallest average size of *ALS*, since it combines global optimization and local optimization which will return an *ALS* with smaller average size compared with SA and overlapping.

Average anonymous time is a metric to evaluate the effectiveness of an anonymity algorithm. It refers to the average time of a successful query's anonymization process. An anonymity algorithm is thought to be more effective if its average anonymous time is lower. Figure 5 depicts the tendencies of average anonymous time with varying number of semantic locations and θ when sensitive location set is PTs_1 and PTs_2 respectively.

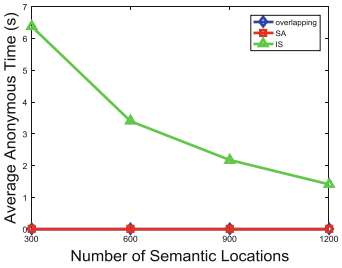
From Fig. 5, we can see, IS does not perform so well as overlapping and SA, because it records more than one *ALS* and needs more time to find a θ-secure *ALS*. However, with the number of semantic locations or θ increasing, the average anonymous time is getting closer to SA and overlapping.



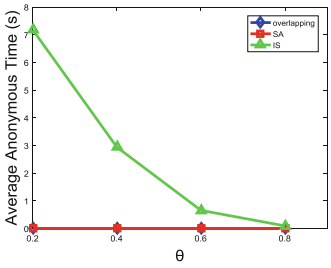
(a) θ takes default value, using PTs_1



(b) Number of semantic locations takes default value, using PTs_1



(c) θ takes default value, using PTs_2



(d) Number of semantic locations takes default value, using PTs_2

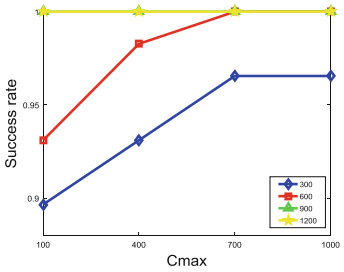
Fig. 5. Relation between average anonymous time and the number of semantic locations or θ

5.3 Parameters Discussion

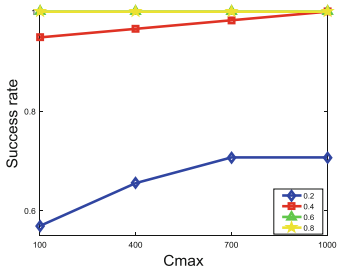
In our algorithm, both the values of *Cmax* and *Lmax* are very important. In this section we discuss them in detail. For simplicity, we only discuss the case that the sensitive location set is PTs_1.

Figure 6 depicts the tendencies of success rate, average size of *ALS*, average anonymous time with varying *Cmax* when the number of semantic locations or θ takes default value. From Fig. 6(a) and (b), we can see success rate increases and the growth rate gradually decreases when *Cmax* increases. From Fig. 6(c) and (d), we can see the average size of *ALS* decreases gradually and tends to be a stable value. This is because when *Cmax* increases, the number of recorded *ALS* enlarges which makes it easier to get a θ-secure *ALS*. From Fig. 6(e) and (f), we can see average anonymous time is high when θ is 0.2 or the number of semantic locations is between 300 and 600, but it is low in other conditions. This is because when θ is 0.2 or the number of semantic locations is between 300 and 600, the success rate is low. The failed anonymous process cost much time since IS terminates until the size of *ALS* exceeds *Lmax*. While in other conditions, success rate is high, so the value of average anonymous time is low. When

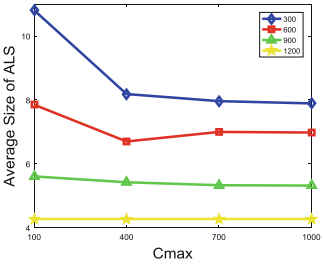
Cmax is smaller than 400, success rate is low and average size of *ALS* is large; when *Cmax* is bigger than 700, the average anonymous time becomes very large, so *Cmax* is suggested to be between 400 and 700.



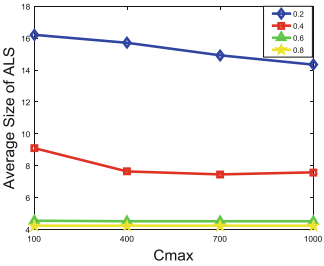
(a) θ takes default value



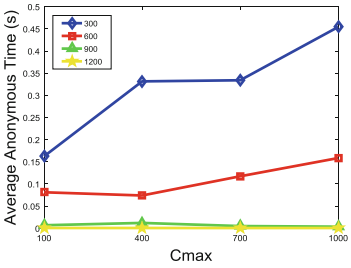
(b) Number of semantic locations takes default value



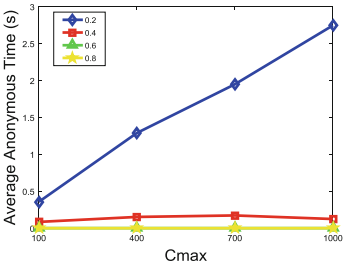
(c) θ takes default value



(d) Number of semantic locations takes default value



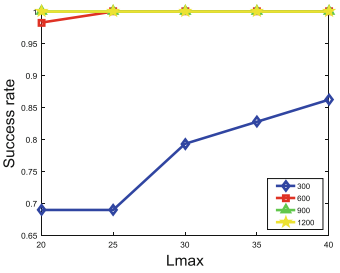
(e) θ takes default value



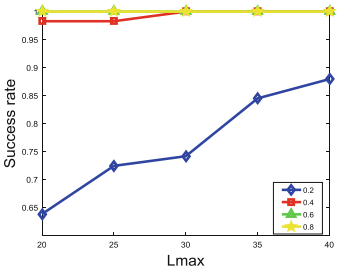
(f) Number of semantic locations takes default value

Fig. 6. *Cmax* value analysis

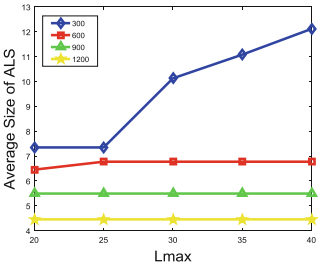
Figure 7 depicts the tendencies of success rate, average size of *ALS*, average anonymous time with varying *Lmax* when the number of semantic locations or θ takes default value. From Fig. 7, we can see when the number of semantic locations is 300 or θ is 0.2, success rate is increasing with the raise of *Lmax*. The growth rate is high when *Lmax* is smaller than 30 and low when *Lmax* is bigger than 35. Both the average size of *ALS* and the average anonymous time increase with *Lmax* increasing. Success rate



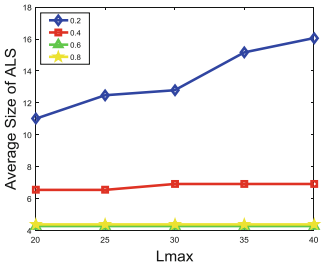
(a) θ takes default value



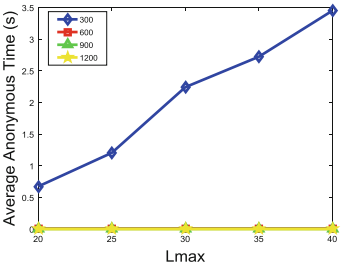
(b) Number of semantic locations takes default value



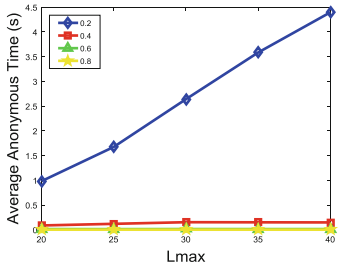
(c) θ takes default value



(d) Number of semantic locations takes default value



(e) θ takes default value



(f) Number of semantic locations takes default value

Fig. 7. *Lmax* value analysis

increases because a bigger *Lmax* makes it easier to satisfy the θ-secure *ALS* requirement. Some semantic locations which fail to find the θ-secure *ALS* with small *Lmax* eventually find a θ-secure *ALS* after *Lmax* enlarges, thus the average size of *ALS* increases. At the same time, average anonymous time becomes longer because these semantic locations cost much time to find a θ-secure *ALS*. When *Lmax* is larger than 30, the average size of *ALS* is large and average anonymous time is long, so it is better to choose 30 as the value of *Lmax*. When the number of semantic locations or θ takes other values, success rate is very close to 1, and the average size of *ALS* and average anonymous time almost keep the same, which means that *Lmax* has few impacts on IS when θ or the number of semantic locations is large.

6 Conclusion

This paper proposed IS algorithm to achieve personalized semantic location privacy preservation. According to combining global optimization with local optimization, IS can improve anonymous success rate and reduce query processing cost. Two parameters *Cmax* and *Lmax* are introduced to limit space and time cost. By comparing IS with overlapping and SA in three aspects, the experimental results show that IS has good performance.

Acknowledgements. This work was partially supported by the Natural Science Foundation of China (No. 61272403), by the Fundamental Research Funds for the Central Universities (No. 10561201474).

References

1. Sweeney, L.: K-anonymity: A model for protecting privacy. Int. J. Uncert. Fuzz. Knowl. Syst. **10**(5), 557–570 (2002)
2. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: L-diversity: privacy beyond k-anonymity. Proc. ACM Trans. Knowledge Discov. Data (TKDD), 152 (2007)
3. Xue, M., Kalnis, P., Pung, H.K.: Location diversity: enhanced privacy protection in location based services. In: Choudhury, T., Quigley, A., Strang, T., Suginuma, K. (eds.) LoCA 2009. LNCS, vol. 5561, pp. 70–87. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01721-6_5
4. Yigitoglu, E., Damiani, M.L.: Privacy-preserving sharing of sensitive semantic locations under road-network constraints. In: Proceedings of International Conference on Mobile Data Management, pp. 186–195 (2012)
5. Li, M., Qin, Z., Wang, C.: Sensitive semantics-aware personality cloaking on road-network environment. Int. J. Secur. Appl. **8**(1), 133–146 (2014)
6. Kido, H., Yanagisawa, Y., Satoh, T.: An anonymous communication technique using dummies for location-based services. In: Proceedings of IEEE International Conference on Pervasive Services, ICPS (2005)
7. Beresford, A.R., Stajano, F.: Mix zones: user privacy in location-aware services. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PerCom 2004 Workshops), pp. 127–131 (2004)

8. Gruteser, M., Grunwald, D.: Anonymous usage of location based services through spatial and temporal cloaking. In: Proceedings of the 1st International Conference on Mobile Systems, Applications and Services (MobiSys 2003), San Francisco, California, pp. 31–42 (2003)
9. Mohamed, F.M., Chow, C.Y., Walid, G.A.: The new casper: query processing for location services without compromising privacy. In: Proceedings of 32nd International Conference on Very Large Data Bases, pp. 763–774. ACM Press (2006)
10. Gedik, B., Liu, L.: Protecting location privacy with personalized k-anonymity: architecture and algorithms. IEEE Trans. Mob. Comput. **7**(1), 1–18 (2008)
11. Kalnis, P., Ghinita, G., Mouratidis, K.: Preventing location-based identity inference in anonymous spatial queries. Proc. IEEE Trans. Knowl. Data Eng. **19**(12), 1719–1733 (2007)
12. Ardagna, C.A., Cremonini, M., Damiani, E., De Capitani di Vimercati, S., Samarati, P.: Location privacy protection through obfuscation-based techniques. In: Barker, S., Ahn, G.-J. (eds.) DBSec 2007. LNCS, vol. 4602, pp. 47–60. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73538-0_4
13. Ghinita, G., Kalnis, P., Khoshgozaran, A., Shahabi, C., Tan, K.L.: Private queries in location based services: anonymizers are not necessary. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD 2008), Vancouver, Canada, pp 121–132 (2008)
14. Bamba, B., Liu, L., Pesti, P.: Supporting anonymous location queries in mobile environments with Privacy Grid. In: Proceedings of the 17th International Conference on World Wide Web, New York, pp. 237–246 (2008)
15. Damiani, M.L., Bertino, E., Silvestri, C.: The PROBE framework for the personalized cloaking of private locations. ACM Trans. Data Priv. **3**(2), 123–148 (2010)
16. Damiani, M.L., Bertino, E., Silvestri, C.: Protecting location privacy against spatial inferences: the PROBE approach. In: ACM SPRINGL 2009 (2009)
17. Damiani, M.L., Silvestri, C., Bertino, E.: Fine-grained cloaking of sensitive positions in location-sharing applications. IEEE Pervasive Comput. **10**(4), 64–72 (2011)
18. Byoungyoung, L., Jinoh, O., Hwanjo, Y.: Protecting location privacy using location semantics. In: Proceedings of the 17th ACM SIGKDD, pp. 1289–1297 (2011)