Quality Aware Privacy Protection for Location-based Services

Zhen Xiao[1,2], Xiaofeng Meng[1,2], and Jianliang Xu[3]

[1] School of Information, Renmin University of China
[2] Key Laboratory of Data Engineering and Knowledge Engineering, MOE
{xiaozhen, xfmeng}@ruc.edu.cn
[3] Department of Computer Science, Hong Kong Baptist University
xujl@comp.hkbu.edu.hk

Abstract. Protection of users' privacy has been a central issue for location-based services (LBSs). In this paper, we classify two kinds of privacy protection requirements in LBS: *location anonymity* and *identifier anonymity*. While the location cloaking technique under the k-anonymity model can provide a good protection of users' privacy, it reduces the resolution of location information and, hence, may degrade the quality of service (QoS). To strike a balance between the location privacy and QoS, we present a quality-aware anonymity model for protecting location privacy while meeting user specified QoS requirements. In the model, a mobile user can specify the minimum anonymity level requirement upon location privacy as well as the maximum cloaking latency and the maximum cloaking region size requirements upon QoS. In accordance with the model, we develop an efficient directed-graph based cloaking algorithm to achieve both high-quality location anonymity and identifier anonymity. The performance objective is to maximize the cloaking success rate under the privacy and QoS constraints. Furthermore, we introduce an option of using dummy locations to achieve a 100% cloaking success rate at the cost of communication overhead. Experimental results show the effectiveness of our cloaking algorithm under various privacy and QoS requirements.

Key words: Privacy, Location-based Services, QoS

1 Introduction

With the advances in wireless communication and mobile positioning technologies, location-based services (LBSs) have become increasingly popular for mobile users. In these applications, mobile users[4] send their location information to service providers and enjoy various types of location-based services, such as mobile yellow page (e.g.,"Where is my nearest restaurant"), mobile buddy list (e.g.,"Where is my nearest friend"), traffic navigation (e.g.,"What is my shortest path to the Summer Palace"), and emergency support services (e.g.,"I need help and send me the nearest police") [1, 2].

While people get much benefit from the useful and convenient information provided by LBSs, the privacy threat of revealing a mobile user's personal information (including the identifier and location) has become a severe issue [3,

[4] In this paper, we use "mobile user", "mobile client", and "user" interchangeably.

4]. A traditional solution to protecting privacy is the use of pseudonymity [5]. That is, for any LBS request, a trusted middleware is employed to replace the real identifier of the user with a pseudonym before forwarding the request to a service provider [11, 12]. However, the location information enclosed in the request may lead to personal re-identification. An attacker can link the location to some particular individual based on external knowledge. For example, if we know the location exclusively belongs to some owner, the corresponding request can thus be linked to the location owner [8, 9].

In general, there are two kinds of privacy protection requirements in LBS:

- **Location anonymity**. This is to protect a user's location from being disclosed when the location information is sensitive (e.g., in a clinic or pub). A common technique is to cloak the user's location by an extended region. Under the k-anonymity model [6], the region is large enough such that it contains at least $k-1$ other users.
- **Identifier anonymity**. This is to hide a user's identifier when the message content is sensitive (e.g., political or financial data). Again location cloaking can be applied to provide identifier anonymity. Under the k-anonymity model [6], user locations are cloaked such that a location is covered by at least $k-1$ other requests. In this way, a request is not identifiable from the other $k-1$ requests.

While the k-anonymity model can provide a good protection of users' privacy, it reduces the resolution of the location information and, hence, may degrade the quality of service (QoS). It is often desirable to strike a balance between the location privacy and QoS requirements. In this paper, we present a quality-aware anonymity model for protecting location privacy while meeting user specified QoS requirements. In our model, a mobile user can specify the following requirements in each LBS request: 1) the minimum anonymity level k, indicating the location cloaking should satisfy both k-location-anonymity and k-identifier-anonymity; 2) the maximum cloaking latency Δt, representing the maximum cloaking delay that the user can tolerate; 3) the maximum cloaking region size δ, indicating the maximum tolerable error in location data. While k reflects the user's requirement upon location privacy, Δt and δ represent the user's QoS requirements. Since the privacy/QoS tradeoff for a user may change over time under different circumstances, we allow these requirements to vary from one request to another even for the same user.

In accordance with the quality-aware anonymity model, we develop an efficient directed-graph based cloaking algorithm to perform anonymization over LBS requests. The performance objective is to maximize the cloaking success rate with privacy protected and QoS guaranteed. Furthermore, we introduce an option of using dummy locations to achieve a 100% cloaking success rate at the cost of communication overhead. Under this scenario, we would like to make use of as few dummies as possible to minimize the communication overhead. We conduct a series of experiments to evaluate the effectiveness of the proposed algorithms. The results show that our algorithms are superior under various privacy and QoS requirements.

The rest of this paper is organized as follows. We first review some related work on protecting location privacy in Section 2. In Section 3, we describe our quality-aware anonymity model. An efficient cloaking algorithm is proposed in

Section 4. Some discussions and improvements are presented in Section 5. Section 6 presents the implementation and experimental results of our proposed algorithms. Finally, Section 7 concludes the paper.

2 Related Work

Several different models have been used for protecting location privacy. Kido *et al.* [13] proposed a dummy-based approach, in which a user sends the actual location with several fake locations ("dummies") to a service provider. The service provider processes and returns an answer for each received location. The user finally refines the result based on the actual location.

The k-anonymity model was originally introduced for privacy protection in conventional database applications [7]. As defined in [6], a release of data provides k-anonymity protection if the information for each individual contained in the release cannot be distinguished from at least $k-1$ individuals whose information also appear in the release. In the context of LBS, Gruteser and Grunwald [11] first adopted the k-anonymity model and proposed a quad-tree based cloaking algorithm. They assume a static anonymity requirement k_{min} for all users. To achieve k-anonymity, the algorithm recursively subdivides the area around a user's location into four quadrants until the number of users in the area falls below k_{min}, and then returns the previous quadrant as the cloaking region. This technique does not differentiate the privacy requirements of different users. Moreover, no restriction is imposed on the cloaking region size. Thus, a cloaking region can be very large, which may lead to an inaccurate query result and a poor service quality.

Gedik and Liu [12] recently proposed the technique of supporting personalized privacy requirements, capturing the privacy and QoS requirements on a per-user basis. A location cloaking algorithm called *CliqueCloak* was developed. CliqueCloak constructs an undirected graph for all the requests that have not been anonymized yet. Each time the server receives a new request, it attempts to identify a clique involving the new request and some existing requests, and cloak them together with the same region. However, this method has several drawbacks. First, the effectiveness of this method is limited to users with small k values (i.e., 2-5). As shown in [12], we can hardly find the anonymity set for requests with larger k values. Second, the cost of searching a clique in a graph is costly. Third, some requests that cannot be anonymized will be dropped when their lifetimes expire. This will affect the user experience towards the service. Different from [12], our proposed cloaking algorithm considers the tradeoff between privacy and QoS requirements to achieve a higher cloaking success rate.

A different framework named *Casper* was proposed in [14]. Casper employed a grid-based pyramid structure to index all user locations. Besides the anonymity level k, a user can specify A_{min}, indicating that the user wants to hide the location information within an area of at least A_{min}. This model has the following concerns. First, k and A_{min} have a similar functionality. In fact, the higher is the value of k, the larger is the cloaking area. Second, the cloaking region may expand arbitrarily large if k is set to a large value and few users present nearby. To address this problem, Casper uses a privacy-aware query processor to return a list of candidate query results to the anonymizing proxy, who has to locally

refine the actual result from the candidate list. This approach incurs a high query processing cost, a high communication cost, and a high local computation cost. In contrast, to reduce such costs, we enforce some temporal and spatial QoS requirements when performing location cloaking.

3 System Model

We consider a LBS system consisting of mobile clients, a trusted anonymizing proxy, and LBS providers [11, 12]. Upon a user query, the mobile client first sends the LBS request to the anonymizing proxy through an authenticated and encrypted connection. The request consists of the user's identifier *id*, current location $l = (l.x, l.y)$, current time t, as well as the service related content such as the query (denoted by *data*). Additionally, the mobile client can specify in the request its privacy and QoS requirements, which include the desired anonymity level k, the tolerable maximum cloaking delay Δt, and the acceptable maximum cloaking region size (denoted by a radius of δ). Thus, a request from user i is defined as: $r_i = (id, l, k, \Delta t, \delta, data, t)$.

Based on the request's privacy and QoS requirements, the anonymizing proxy expands the location l into a cloaking region $\mathcal{L}$ (to be detailed later in this section). Moreover, the identifier *id* is replaced with a pseudonym id' (e.g., a secure hash number). The original request is transformed into a new anonymized request, $r'_i = (id', \mathcal{L}, data)$, and is forwarded to the LBS provider. Finally, the anonymized request is processed by LBS provider. The query result is sent back to the anonymizing proxy, which, after refining the result, returns the final result to the mobile client.

We adopt the k-anonymity model [6] for protecting location privacy. Given a set of user requests $\{r_1, r_2, \cdots, r_n\}$ and their anonymized requests $\{r'_1, r'_2, \cdots, r'_n\}$, the location k-anonymity model is defined as follows:

Definition 1 *For any request r_i, the location k-anonymity is satisfied if and only if 1) r_i's cloaking region $r'_i.\mathcal{L}$ covers the locations of at least $k-1$ other requests (i.e., $|\{j \mid r_j.l \in r'_i.\mathcal{L}, 1 \leq j \leq n, j \neq i\}| \geq k-1$) and, 2) r_i's location $r_i.l$ is covered by the cloaking regions of at least $k-1$ other requests (i.e., $|\{j \mid r_i.l \in r'_j.\mathcal{L}, 1 \leq j \leq n, j \neq i\}| \geq k-1$).*

By this definition, a LBS provider cannot distinguish a user's location $r_i.l$ from $k-1$ other users' locations since they all present in the cloaking region $r'_i.\mathcal{L}$. Moreover, even knowing the location of a user, a LBS provider cannot tell which request is made by this user since there are k requests all covering this user's location. As such, both location anonymity and identifier anonymity are achieved. We refer to the set of users achieving location anonymity as *location anonymity set* and the set of users achieving identifier anonymity as *identifier anonymity set*. For example, Figure 1 shows four LBS requests from different users as well as their cloaking regions. Since r_1's cloaking region covers r_1, r_2, and r_3, the location anonymity set of r_1 is $\{r_1, r_2, r_3\}$. On the other hand, r_1 is covered by the cloaking regions of r_1 through r_4. Thus, the identifier anonymity set of r_1 is $\{r_1, r_2, r_3, r_4\}$.

In summary, for any request r and its anonymized request r', we specify the privacy and QoS requirements from the following three aspects:

1. **Location Privacy.** This requires to expand the user location l into a cloaking region $\mathcal{L}$ such that the k-anonymity model (Definition 1) is satisfied.
2. **Temporal QoS.** This states that the request must be anonymized before the predefined maximum cloaking delay (i.e., $t + \Delta t$).
3. **Spatial QoS.** This specifies that the cloaking region size should not exceed a threshold, i.e., the cloaking region must be inside a circle Ω centered at l and with a radius of δ (i.e., $\mathcal{L} \subseteq \Omega(l, \delta)$).

In general, a larger Δt (or δ) provides more flexibility in location anonymization but results in an extended query delay (or a less accurate query result).

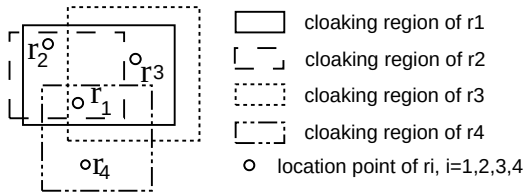


Fig. 1. Illustration of Location Anonymity Set and Identifier Anonymity Set

4 Basic Anonymization Algorithm

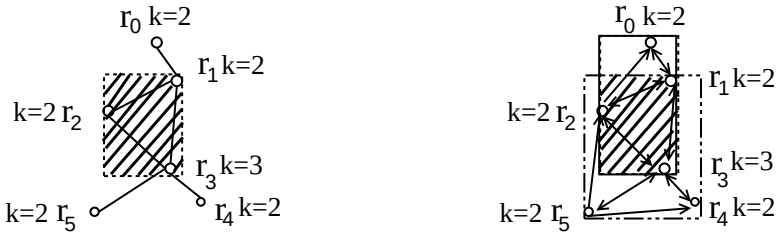
The anonymization algorithm turns the user location into a cloaking region based on the privacy and QoS constraints. In this section, we discuss the following problems faced by the anonymization algorithm: 1) when to anonymize which request? and 2) given a request under the cloaking region size constraint, how to find other requests (i.e., the location anonymity set and identifier anonymity set) to satisfy the location k-anonymity model?

Bearing in mind that our objective is to maximize the cloaking success rate, we shall delay the anonymization process of a request until right before its deadline (i.e., $t + \Delta t - \epsilon$), where ϵ is a small time offset set to be the worst cloaking time. In this way, not only can more potential requests join a request's anonymity set, but also other requests have a higher chance to include this request in their anonymity sets.

To tackle the second problem, the CliqueCloak algorithm proposed in [12] constructs an undirected graph to represent the correlations of all requests. The graph is defined as: $G_u = (V, E_u)$, where V is the set of the nodes, each representing a request r received at the anonymizing proxy, and E_u is the set of edges, each representing the neighborship between the corresponding nodes (requests). For any pair of nodes $r_i, r_j \in V$, there exists an edge $e_{ij} = (r_i, r_j) \in E_u$ if and only if the distance between the two requests $|r_i r_j|$ is not more than both $r_i.\delta$ and $r_j.\delta$ ($|r_i r_j| \leq r_i.\delta$ and $|r_i r_j| \leq r_j.\delta$), which indicates the locations of r_i and r_j are within each other's predefined maximum cloaking region size. For any request r_i, its location anonymity set and identifier anonymity set are the same $r_i.U$ that includes all its neighbors except those have k larger than $r_i.k$ and those have less than $k-1$ neighbors in $r_i.U$. A clique of at least k requests is identified if they share the same anonymity set. These requests are then anonymized with the same cloaking region (the minimum bounding rectangle (MBR) of their location points) at the same time. This approach incurs a huge searching cost for

identifying cliques with the worst time complexity of $O(N2^N)$, where N is the number of nodes in the subgraph of $r_i.U$. Moreover, with a restricted definition of the anonymity set, the cloaking success rate is low.

Recall that under our location k-anonymity model, each request can have a different anonymity set and hence a different cloaking region. Thus, in contrast to [12], we build a directed graph rather than an undirected graph. In the directed graph $G_d = (V, E_d)$, for any pair of nodes $r_i, r_j \in V$, there exists an edge $e_{ij} = (r_i, r_j) \in E_d$ from r_i to r_j, if and only if the distance between the two requests is not more than $r_i.\delta$ ($|r_i r_j| \leq r_i.\delta$), which indicates r_j's location is within r_i's predefined maximum cloaking region size. Similarly, there exists an edge $e_{ji} = (r_j, r_i) \in E_d$ from r_j to r_i, if and only if r_i's location is within r_j's predefined maximum cloaking region size. The location anonymity set of a request r_i, is formed by all its outgoing neighbors $r_i.U_{out}$, i.e., $r_i.U_{out} = \{r_i\} \cup \{r_j \mid (r_i, r_j) \in G_d(V, E_d)\}$, and the identifier anonymity set, is formed by all its incoming neighbors $r_i.U_{in}$, i.e., $r_i.U_{in} = \{r_i\} \cup \{r_j \mid (r_j, r_i) \in G_d(V, E_d)\}$. For each request r_i, we maintain a $flag$ to identify its status, i.e., $flag = unanonymized$ means r_i has not been anonymized, and $flag = forwarded$ means r_i has been anonymized successfully and forwarded to the service provider but not yet deleted in the graph. A request r_i can be anonymized immediately if there are at least $k-1$ other anonymized requests in $r_i.U_{out}$ (i.e., $|\{j \mid r_j \in r_i.U_{out}, r_j.flag = forwarded, j \neq i\}| \geq k-1$) and $k-1$ other anonymized requests in $r_i.U_{in}$(i.e., $|\{j \mid r_j \in r_i.U_{in}, r_j.flag = forwarded, j \neq i\}| \geq k-1$).[5] The cloaking region of r_i is then represented by the MBR of the location points of the requests in the location anonymity set (denoted by $MBR(r_i.U_{out})$).



(a) CliqueCloak (b) Our Cloaking

Fig. 2. An Example: the differences between CliqueCloak and our proposed cloaking

We use an example to illustrate the differences between CliqueCloak(Figure 2(a)) and our proposed cloaking algorithm (Figure 2(b)). The same set of user requests with corresponding k values are shown in the figure, where they are numbered in the ascending order of their deadlines. We assume r_0 has been anonymized successfully ($r_0.flag = forwarded$ in our cloaking). With CliqueCloak, r_1, r_2, and r_3 form a clique and are cloaked with the same region (their MBR, represented by the shaded area in Figure 2(a)). Then they will be deleted from the graph.

[5] Initially, no requests are flagged as "forwarded". We employ CliqueCloak to anonymize requests in the warm-up period; our proposed cloaking algorithm follows after the warm-up period.

Next, r_4 and r_5 will be dropped because they cannot find enough neighbors. With our proposed cloaking algorithm, the neighborships between the requests are different and some new edges are added in the directed graph. First, r_1 is processed with $U_{out} = U_{in} = \{r_0, r_1, r_2, r_3\}$. Because $r_0.flag = forwarded$, satisfying $r_1.k = 2$, we can anonymize r_1 with cloaking region $MBR(r_0, r_1, r_2, r_3)$. Then, r_2 is processed with $U_{out} = \{r_0, r_1, r_2, r_3\}$ and $U_{in} = \{r_1, r_2, r_3, r_5\}$. Because r_0 and r_1 have been anonymized successfully, satisfying $r_2.k = 2$, we can also cloak it with $MBR(r_0, r_1, r_2, r_3)$. Similarly, all the other requests will be successfully anonymized with $r'_3.\mathcal{L} = MBR(r_1, r_2, r_3, r_4, r_5)$, $r'_4.\mathcal{L} = MBR(r_3, r_4)$, $r'_5.\mathcal{L} = MBR(r_2, r_3, r_4, r_5)$. Obviously, by allowing different cloaking regions for different requests, our proposed cloaking algorithm gets a higher success rate than CliqueCloak. Moreover, as shown in Figure 2(b), the cloaking regions of r_1, r_2, and r_3 cover some more requests in addition to r_1, r_2, and r_3, thereby providing a higher privacy level than CliqueCloak. In the following, we describe the detailed data structures and related algorithms for anonymizing user requests.

4.1 Data Structures

As mentioned, we employ a dynamic in-memory directed graph for all user requests. To facilitate the construction and maintenance of the graph, we build a spatial index (i.e., R-tree) over the location points ($r_i.l$'s) of all requests. Thus, we can use a window query to quickly find the neighbors of a request. Additionally, we maintain a min-heap to order the requests according to their cloaking deadlines (i.e., the key is $r_i.t + r_i.\Delta t$).

4.2 Algorithms

Algorithm 1: Maintenance ($r_i = \{id, x, y, t, \Delta t, k, \delta, data\}$)

```
1  insert r_i into the spatial index and the heap;
2  create a new node for r_i in the graph;
3  r_i.n ← 0; r_i.U_in ← {r_i}; r_i.U_out ← {r_i};
4  C ← a range query Q on the spatial index,
   Q = ((x − δ_max, y − δ_max), (x + δ_max, y + δ_max));
5  forall r_j ∈ C, r_j ≠ r_i do
6      if |r_i r_j| ≤ r_i.δ or |r_i r_j| ≤ r_j.δ then
7          if |r_i r_j| ≤ r_i.δ then
8              create an edge (r_i, r_j) in the graph;
9              r_i.U_out ← r_i.U_out ⋃ {r_j}; r_j.U_in ← r_j.U_in ⋃ {r_i};
10         if |r_i r_j| ≤ r_j.δ then
11             create an edge (r_j, r_i) in the graph;
12             r_i.U_in ← r_i.U_in ⋃ {r_j}; r_j.U_out ← r_j.U_out ⋃ {r_i};
13         r_i.n ← r_i.n + 1; r_j.n ← r_j.n + 1;
14     end
15 end
```

Maintenance: Algorithm 1 details the maintenance of the data structures. Given a new incoming request r_i, we first update the spatial index and the heap. Next we update the graph. We start by searching the spatial index using a range query with $r_i.l$ as the central point and δ_{max} as the radius, where δ_{max} is the

maximum cloaking region size requirement of all the requests. The requests in the search result C are the candidates for being r_i's neighbors in the graph. Each r_j in C is filtered based on whether the distance between r_j and r_i is within $r_i.\delta$ or $r_j.\delta$. In the former case, we construct an edge from r_i to r_j. In the latter case, we construct an edge from r_j to r_i. In both cases, they are added to each other's outgoing neighbor set U_{out} and incoming neighbor set U_{in}. In the algorithm, $r_i.n$ is used to represent the cardinality of $U_{in} \bigcup U_{out}$.

Algorithm 2: Cloaking

```
1  get the top request r in the heap;
2  if t + Δt − t_now < ε then
3  |  forall r' ∈ r.U_out do if r'.flag = forwarded then k_o + +;
4  |  forall r' ∈ r.U_in do if r'.flag = forwarded then k_i + +;
5  |  if k_o ≥ k − 1 and k_i ≥ k − 1 then
6  |  |  send out R = (pid, MBR(r.U_out), r.data) for r;
7  |  |  r.flag = forwarded;
8  |  |  forall r' ∈ r.U_out ⋃ r.U_in, r' ≠ r do
9  |  |  |  r'.n ← r'.n − 1;
10 |  |  |  if r'.n = 0 and r'.flag = forwarded then delete r' from the graph;
11 |  |  end
12 |  |  if r.n = 0 then delete r from the graph;
13 |  else
14 |  |  delete r from the graph;
15 |  delete r from the spatial index and the heap;
16 end
```

Cloaking Algorithm: Algorithm 2 describes the cloaking algorithm. The input request r is the first request approaching its deadline. We first compare r's cloaking time constraint $r.t+r.\Delta t$ with current time t_{now}. If $r.t+r.\Delta t-t_{now} \leq \varepsilon$, where ε is a small time offset set to be the worst cloaking time, we cloak r immediately. Otherwise, we delay the anonymization of r to the time $r.t+r.\Delta t-\varepsilon$. In the cloaking algorithm, we compute the number of r's neighbors that have been anonymized successfully in the outgoing neighbor set and the incoming neighbor set, denoted by k_o and k_i respectively. If both k_o and k_i satisfy r's minimum anonymity level ($k_o \geq r.k-1$ and $k_i \geq r.k-1$), r can be cloaked as $r' = (pid, MBR(r.U_{out}), r.data)$ and its flag is set as "forwarded", where pid is the pseudonym that replaces $r.id$. Otherwise, the cloaking fails and the request is deleted from the graph. This algorithm has a time complexity $O(n)$, where n is the number of r's neighbors.

After r is successfully cloaked, we delay the removal of r in the graph until all its neighbors have been processed. This is because otherwise the neighborships between r and its neighbors will disappear. And when we anonymize r's neighbors later, they cannot include r in their anonymity sets and, hence, reduce the privacy level and the cloaking success rate. Thus, we then decrease the unprocessed neighbors (n) of each r's neighbor r_j in $r.U_{out} \bigcup r.U_{in}$. If r_j has been anonymized already before r and r is the last neighbor of r_j to be processed, we can remove r_j from the graph. If all neighbors of r have been anonymized before it, we can remove r from the graph. No matter whether the cloaking succeeds

or fails, finally the request r should be removed from the spatial index and the heap.

5 Improvement with Dummy Requests

This section discusses an improvement by using dummy requests in case of a cloaking failure. With dummies introduced, we can guarantee a successful cloaking for every request and a 100% success rate. Therefore, we only need to maintain for each node r the number of incoming neighbors $r.k_i$ and the number of outgoing neighbors $r.k_o$ ($r.flag$ is no longer maintained). If both $r.k_i$ and $r.k_o$ satisfy the required privacy level $r.k$, the cloaking can proceed with success. Otherwise, we use Algorithm 3 to generate enough dummies such that the dummies and the real neighbors together form r's anonymity set. The time complexity of this cloaking algorithm is $O(1)$.

We generate dummies for a request r based on the following requirements. First, dummies should be within both the location anonymity set and the identifier anonymity set such that the privacy level will be higher. Second, dummies must be indistinguishable from actual requests. Third, dummies should satisfy the spatial QoS requirement of r. Thus, to avoid expanding the existing cloaking region, the location of each dummy distributes randomly within $MBR(r.U_{out})$. The cloaking region of each dummy request, d, which also cover the location point of r, is a random spatial region between $MBR(\{r, d\})$ and $MBR(r.U_{out})$. As such, the dummies and r become both incoming neighbors and outgoing neighbors, and the service provider will have difficulty in identifying dummies.

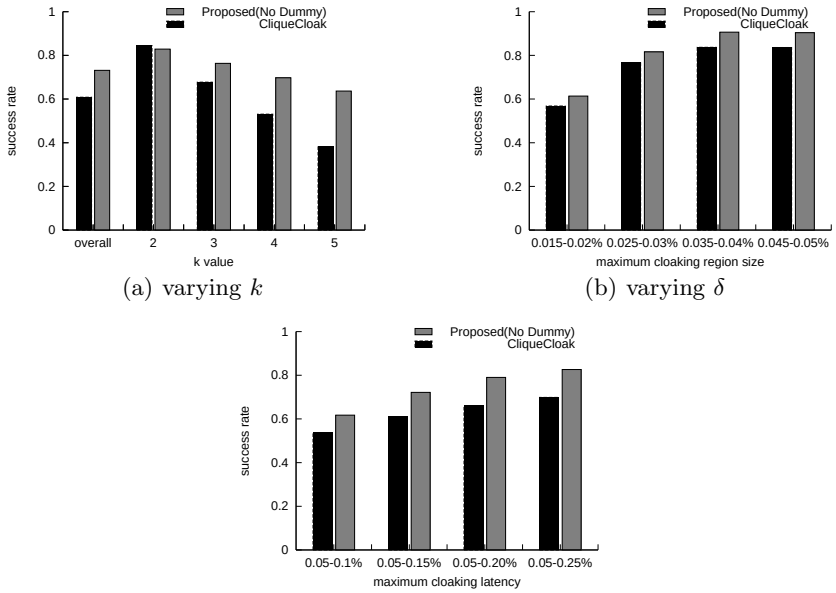
Algorithm 3: Dummy (x, y, k_d, M)

```
1 compute the MBR of M, L = ((x_min, y_min), (x_max, y_max));
2 for i = 1 to k_d do
3 |   x ← random(x_min, x); x' ← random(x, x_max);
4 |   y ← random(y_min, y); y' ← random(y, y_max);
5 |   L_i ← ((x, y), (x', y'));
6 |   send out the ith cloaked dummy request, R_i ← (pid, L_i, data);
7 end
```

Algorithm 3 shows the detailed dummy generation process. The inputs of the procedure are: location $(l.x, l.y)$ of request r to be cloaked, the number of dummies to be generated, and r's outgoing neighbor set as M. The pseudonym and service related content are also randomly generated. Finally, we send the cloaked dummy requests out to the service provider.

6 Experiments

In this section, we experimentally compare the effectiveness of our cloaking algorithm against CliqueCloak [12] under various location privacy and QoS settings. In all the experiments, we use Thomas Brinkhoff Network-based Generator of Moving Objects [15] to generate a set of moving objects. The input to the generator is the road map of Oldenburg County (with an area of about $200km^2$). We simulate 20,000 moving objects that are uniformly distributed in the spatial



(a) varying k (b) varying δ

(c) varying Δt

Fig. 3. Performance of Cloaking Success Rate under Different Settings

space at the initial time and afterwards continuously move on the road network. These moving objects issue LBS requests with their current locations at a query interval of 20,000 s. In each request, Δt, k, and δ are assigned uniformly between the range $[.05 - .15]$% of the update interval (i.e., 1,000-3,000), $[2 - 5]$ users, and [.01-0.05]% of the space (i.e., 2-10), respectively. We set the worst cloaking time offset ε as 10 s.

Recall that the goal of our cloaking algorithm is to maximize the number of requests anonymized successfully in accordance with their privacy and QoS requirements. We first evaluate the cloaking success rate with various privacy and QoS requirements. Figures 3(a), 3(b), and 3(c) show the effect of varying k, δ, and Δt on the success rate, respectively. In all cases tested, our method without using dummies always outperforms CliqueCloak, by 5-25% in terms of cloaking success rate. By using dummy requests, we can even achieve a 100% success rate. In Figures 3(a), both CliqueCloak and our method show that the requests with larger k values are more difficult to anonymize, thus getting a lower success rate. However, the performance degradation of our method is less significant than that of CliqueCloak. This indicates that our method is more robust for larger k values. Figures 3(b) and 3(c) show that a larger δ or Δt improves the flexibility in location anonymization and thus getting a higher success rate.

We then measure the efficiency of our location-anonymity model in terms of users' privacy requirements. The relative location anonymity level is measured by k'/k, where k' is the number of users actually included in the cloaking region while k is the user required number. In Figure 4, we compare the relative anonymity level of our method against CliqueCloak under different k values. In our method, by using dummies, the relative anonymity level can be up to 9 for $k = 2$, meaning that the requests are actually anonymized with $k \approx 18$ on

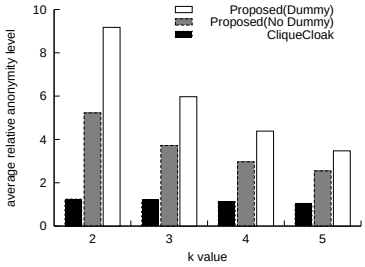


Fig. 4. Relative Anonymity Level

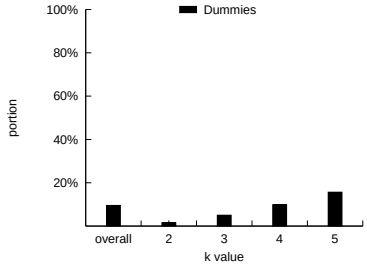


Fig. 5. Portion of Dummy Requests

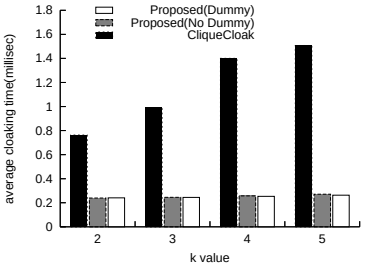


Fig. 6. Cloaking Efficiency

average. Without using dummies, the relative anonymity level is from 5.2 for $k = 2$ to 2.5 for $k = 5$, meaning that the requests are actually anonymized with $k \approx 10$ on average. CliqueCloak provides a lower level from 1.2 for $k = 2$ to 1.0 for $k = 5$. This result also demonstrates that even without dummies our method can support larger k values up to 10 while CliqueCloak is limited to smaller k values. In Figure 5, we measure the portion of dummy requests generated in the total requests under varying k values. Requests with larger k require more neighbors and hence a higher percent of dummies. On average, we can achieve a 100% success rate with about 10% dummies (and thus 10% of communication overhead), which we think is acceptable.

Finally, Figure 6 shows the effect of k on the cloaking efficiency. In all cases tested, our method shows a much shorter cloaking time than CliqueCloak. When k increases, (more neighbors are formed), the average cloaking time lengths as a result of increasing cost of searching anonymity sets. However, the performance degradation of our method is much less smaller than that of CliqueCloak.

7 Conclusion

In this paper, we have discussed the problem of quality-aware privacy protection in location-based services. We classified the privacy requirements into location anonymity and identifier anonymity. To protect both of these two anonymities, we have presented a quality-aware k-anonymity model that allows a mobile user to specify in each LBS request the location privacy requirement as well as the temporal and spatial QoS requirements. We have developed an efficient directed-graph based cloaking algorithm to achieve a high cloaking success rate while

satisfying the privacy and QoS requirements. Moreover, we have introduced the use of dummy requests to achieve a 100% cloaking success rate at the cost of communication overhead. Experimental evaluation have verified the effectiveness of our model and the proposed cloaking algorithms under various privacy and QoS requirements.

Acknowledgments

This research was partially supported by the grants from the Natural Science Foundation of China under grant number 60573091, 60273018; Program for New Century Excellent Talents in University(NCET). Jianliang Xu's work was supported by grants from the Research Grants Council, Hong Kong SAR, China (Project Nos. HKBU 2115/05E and HKBU 2112/06E).

References

1. R. José, N. Davies. Scalable and Flexible Location-Based Services for Ubiquitous Information Access. In Proceedings of First International Symposium on Handheld and Ubiquitous Computing, 1999.
2. J. Schiller and A. Voisard, editors. Location-Based Services. Morgan Kaufmann Publishers, 2004.
3. L. Barkhuus and A. K. Dey. Location-Based Services for Mobile Telephony: a Study of Users' Privacy Concerns. In INTERACT, 2003.
4. A. R. Beresford and F. Stajano. Location Privacy in Pervasive Computing. IEEE Pervasive Computing, 2(1):46-55, 2003.
5. A. Pfitzmann and M. Hansen. Anonymity, Unlinkability, Unobservability, Pseudonymity, and Identity management - A Consolidated Proposal for Terminology, 2005.
6. L. Sweeney. K-anonymity: A model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10(5):557C570, 2002.
7. P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical Report SRI-CSL- 98-04, SRI International, 1998.
8. M. Gruteser and B. Hoh. On the Anonymity of Periodic Location Samples. In SPC, 2005.
9. A. Machanavajjhala, J. Gehrke, and D. Kifer. l-Diversity: Privacy Beyond k-Anonymity. In ICDE, 2006.
10. M. J. Atallah and Keith B. Frikken.Privacy-Preserving Location-Dependent Query Processing. In ICPS, 2004.
11. M. Gruteser and D. Grunwald. Anonymous usage of location based services through spatial and temporal cloaking. In ACM/USENIX MobiSys, 2003.
12. B. Gedik and L. Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. In ICDCS, 2005.
13. H. Kido, Y. Yanagisawa, and T. Satoh. Protection of Location Privacy using Dummies for Location-based Services. In ICPS, 2005.
14. M. F. Mokbel, C. Chow and W. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In VLDB, 2006.
15. T. Brinkhoff. A Framework for Generating Network-Based Moving Objects. GeoInformatica, 6(2):153C180, 2002.