Geoinformatica (2011) 15:571–607
DOI 10.1007/s10707-010-0117-0

Query-aware location anonymization for road networks

Chi-Yin Chow · Mohamed F. Mokbel · Jie Bao · Xuan Liu

Received: 15 December 2009 / Revised: 4 October 2010 /
Accepted: 13 October 2010 / Published online: 6 November 2010

Abstract Recently, several techniques have been proposed to protect the user location privacy for location-based services in the Euclidean space. Applying these techniques directly to the road network environment would lead to privacy leakage and inefficient query processing. In this paper, we propose a new location anonymization algorithm that is designed specifically for the road network environment. Our algorithm relies on the commonly used concept of spatial cloaking, where a user location is cloaked into a set of connected road segments of a minimum total length $\mathcal{L}$ including at least $\mathcal{K}$ users. Our algorithm is "query-aware" as it takes into account the query execution cost at a database server and the query quality, i.e., the number of objects returned to users by the database server, during the location anonymization process. In particular, we develop a new cost function that balances between the query execution cost and the query quality. Then, we introduce two versions of

The work described in this paper was supported in part by a grant from City University of Hong Kong (Project No. 7200216), by the National Science Foundation under Grants IIS-0811998, IIS-0811935, CNS-0708604, IIS-0952977 (NSF CAREER), and by a Microsoft Research Gift.

C.-Y. Chow
Department of Computer Science, City University of Hong Kong,
Kowloon, Hong Kong, China
e-mail: chiychow@cityu.edu.hk

M. F. Mokbel (✉) · J. Bao
Department of Computer Science and Engineering, University of Minnesota,
Minneapolis, MN 55455, USA
e-mail: mokbel@cs.umn.edu

J. Bao
e-mail: baojie@cs.umn.edu

X. Liu
IBM Thomas J. Watson Research Center, Hawthorne, NY 10532, USA
e-mail: xuanliu@us.ibm.com

our algorithm, namely, *pure greedy* and *randomized greedy*, that aim to minimize the developed cost function and satisfy the user specified privacy requirements. To accommodate intervals with a high workload, we introduce a *shared execution paradigm* that boosts the scalability of our location anonymization algorithm and the database server to support large numbers of queries received in a short time period. Extensive experimental results show that our algorithms are more efficient and scalable than the state-of-the-art technique, in terms of both query execution cost and query quality. The results also show that our algorithms have very strong resilience to two privacy attacks, namely, the *replay attack* and the *center-of-cloaked-area attack*.

Keywords Location privacy · Shared execution · Location-based services · Spatial network databases · GIS

1 Introduction

Combining the functionality of map software, location-detection devices, wireless communication, and database systems results in realizing *location-based services* as commercial products and research prototypes. The main promise of location-based services is to provide services for their users based on their locations. Location-based services have become a major component in transportation applications (e.g., live traffic reports), personal mobile devices (e.g., finding nearby stores or friends), advertising (e.g., sending e-coupons to nearby customers), and emergency control (e.g., dispatching ambulance). Recently, it becomes apparent that location-based services suffer from major privacy leakage where users have to disclose their private location information to untrustworthy servers. As a result, several techniques have been proposed to anonymize the user location information through *false locations* (e.g., [15, 20, 34]), *space transformation* (e.g., [10, 19]), or *spatial cloaking* (e.g., [1–3, 5–9, 11–14, 18, 24, 32, 33]). In this paper, we focus on the spatial cloaking technique as it is the most commonly used technique and is applicable to various problem settings (e.g., distributed/peer-to-peer environments [6, 7, 11, 12], sensor networks [5, 14], trajectories [33], and continuous queries [3, 32]).

Unfortunately, almost all previously proposed location anonymization techniques suffer from two main drawbacks: (1) The location anonymization process is designed completely independent from the underlying query processor. As a result, some of these techniques may end up anonymizing the user location to an extent where it is either very inefficient to execute a location-based query as a large number of queries need to be sent to the server [20] or specialized query processing techniques need to be developed [4, 10, 18, 19, 24]. (2) All techniques are designed only for the Euclidean space. Applying these techniques to the road network environment results in privacy leakage. Figure 1a depicts such a case where user q needs to be four anonymous with a cloaked area of at least four grid cells. The gray area represents the cloaked area that could be provided by a location anonymization technique designed for the Euclidean space (e.g., [1, 9, 13, 18, 24]). With such a cloaked area, an adversary knows that the user can be anywhere in the gray area. Figure 1b shows the same example with the drawing of the underlying road network. Since all the four users are in the same road segment, an adversary can pinpoint the exact road segment of user q.

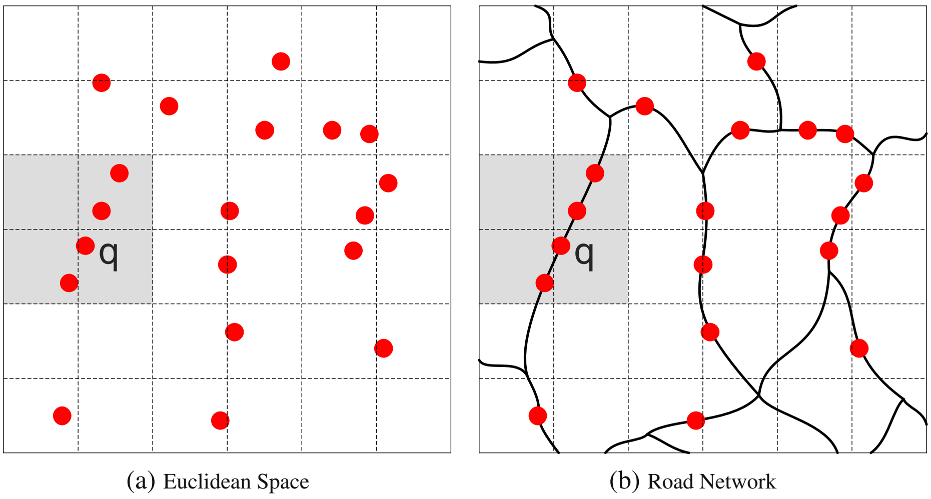


Fig. 1 Spatial cloaking for the Euclidean space

Thus, the cloaked area violates the area privacy requirement, i.e., the user cannot be anywhere in the cloaked area, as the user has to be in a single road segment only.

Although some location anonymization techniques have been proposed to preserve the user location privacy for location-based services in road network environments [22, 25, 31], they have different limitations. One of these techniques assumes a system-wide static $\mathcal{K}$-anonymity level for all mobile users [25]. This assumption not only degrades the quality of services for those mobile users whose desired privacy requirements can be satisfied by smaller $\mathcal{K}$ values, but it also not realistic because mobile users tend to have varying privacy requirements under different contexts or on different types of objects of interest [9]. One of existing techniques relies on an existing Euclidean-based anonymization technique to cloak the user location [22], so it still suffers from the drawback of the Euclidean-based techniques. The other work designs a new location anonymization algorithm that considers the query execution cost, and it supports personalized $\mathcal{K}$-anonymity privacy requirements and shared execution [31], but the underlying basic star structure degrades the query processing efficiency and the shared execution has not been fully utilized (i.e., the shared execution is limited to the queries located on the same road segment).

In this paper, we overcome the above two main drawbacks by proposing a new location anonymization algorithm that (a) is designed specifically for the road network environment, and (b) not only aims to satisfy the user privacy requirements, but it is also "query-aware" as it aims to balance between (1) the *query execution cost* at a database server and (2) the *query quality* (i.e., a smaller set of objects returned to the user indicates higher query quality). The main idea of our proposed "query-aware" location anonymization technique is to blur a user location into a cloaked set of connected road segments S such that S satisfies the user specified privacy requirements, $\mathcal{K}$-anonymity and minimum length $\mathcal{L}$. The $\mathcal{K}$-anonymity requirement indicates that S must include at least $\mathcal{K}$ users while the minimum length $\mathcal{L}$ requirement indicates that the total length of all road segments in S must be at least $\mathcal{L}$. As the database server only knows about a cloaked segment set S as a user's location information, it has to compute an answer set A that includes the exact query answer should the user be anywhere within S. The smaller the size of the returned answer set A, the better query quality the server will provide. Thus, the query quality is measured by the size of the returned answer set. To achieve our goals, we design a new objective cost function that encapsulates the query execution cost for both k-nearest-neighbor and range queries with the query quality. Then, the objective of our road network location anonymization algorithm boils down to finding a cloaked set of road segments S that minimizes our developed objective cost function while satisfying the user privacy requirements.

We present two versions of our proposed location anonymization algorithm. The first version is a *pure greedy* approach where we repeatedly select road segments to be included in a cloaked segment set S based on minimizing our developed objective cost function. Although the pure greedy approach is simple and efficient, its deterministic property would suffer from a reverse engineering attack, i.e., a replay attack, where an adversary cracks the system to know the objective cost function and the produced cloaked segment sets. To avoid such an attack, we propose another version of our algorithm, termed *randomized greedy* approach, where we inject some randomness into the greedy approach. To accommodate for cases of high workloads, e.g., traffic congestions and rush hours, we propose a *shared execution* paradigm that boosts the system scalability in terms of supporting large numbers of queries received within a short time period. The main idea of the *shared execution* paradigm is to maximize the number of shared road segments among the users' cloaked segment sets. By doing so, location-based queries at the shared road segments will be executed only once for multiple queries. In general, the contributions of this paper can be summarized as follows:

1. We design a new objective cost function that, for a given cloaked set of connected road segments S, balances between (a) the query execution cost for k-nearest-neighbor and range queries, and (b) the query quality in terms of the number of false objects returned to the user (Section 4).
2. We propose two greedy approaches, namely, *pure greedy* and *randomized greedy*, that aim to find a cloaked set of road segments S for a given user such that (a) our developed cost function is minimized, and (b) the user privacy requirements are satisfied in terms of $\mathcal{K}$-anonymity and minimum length $\mathcal{L}$ (Section 5).
3. We propose a *shared execution paradigm* that can be used in conjunction with our two proposed greedy techniques to accommodate for cases where the server is overloaded (Section 6).
4. We provide experimental evidence that our proposed algorithms effectively resist to two privacy attacks, i.e., *replay attack* and *center-of-cloaked-area attack*, and it is efficient and scalable for large numbers of users, queries, and objects, and strict privacy requirements, while preserving the user location privacy in road networks (Section 7).

The rest of this paper is organized as follows. Section 2 highlights related works. Section 3 delineates our system model. Our query execution cost model is described in Section 4. Section 5 presents our *query-aware* location anonymization algorithm. The *shared execution* paradigm is described in Section 6. Section 7 describes two privacy attacks and gives the experimental results. Finally, Section 8 concludes this paper.

2 Related works

Location anonymization Previous works in location anonymization can be classified into three categories: (1) *False locations* [15, 20, 34]. The basic idea is to send either one or more fake locations that are related to the user location. (2) *Space transformation* [10, 19]. The basic idea is to transform the location information into another space where the spatial relationships among queries and data are encoded. (3) *Spatial cloaking* [1–3, 5–9, 11–14, 18, 24, 32, 33, 35]. The main idea is to blur users' locations into spatial regions that are guaranteed to satisfy the $\mathcal{K}$-anonymity [29] and/or minimum region area privacy requirements [2, 8, 24]. The spatial cloaking technique has been applied to various problem settings that include distributed/peer-to-peer environments [6, 7, 11, 12, 35], sensor networks [5, 14], trajectory data [33], and continuous queries [3, 32]. Unfortunately, none of these works address the road network environment nor consider the query processing cost, as the focus was mainly on the Euclidean space and anonymizing user locations regardless of how difficult or inefficient is the query processing. For example, some of these techniques require special features to be developed at the query processor [4, 10, 18, 19, 24], while others result in submitting several queries [20] or an incremental nearest-neighbor query [34] to a database server, in order to get the query answer. To this end, our proposed location anonymization algorithm not only considers the user personalized privacy requirement, but it also takes in account the query execution cost during the anonymization process.

Architecture model Based on whether a privacy-preserving technique requires a third trusted party, termed *location anonymizer*, to be placed between the user and the location-based database server, the related works can be classified into two categories: (1) *Anonymized queries* [1, 3, 6, 7, 9, 13, 18, 22, 32, 33, 35]. In this category, users' locations have to go through a trusted location anonymizer that cloaks the user or query location information into cloaked areas. In this case, the original query with a point location is transformed to another query with a cloaked area. This *trusted third party* model is commercially used in other fields, e.g., the Anonymizer[1] for anonymous web surfing and the PayPal[2] for anonymous online payment. (2) *No anonymizer* [10, 19, 34]. In this category, users can directly contact the database server by utilizing space transformation [19], private information retrieval [10], or anchor points [34]. All these query processing techniques for both categories are totally independent of the anonymization process. Unfortunately, these works consider only the Euclidean space, so they cannot address the case of privacy-preserving query processing in road networks.

Location privacy in road networks Some privacy-preserving techniques have been proposed to protect the user location privacy in road network environments [22, 23,

[1] Anonymizer. http://www.anonymizer.com.

[2] Paypal. http://www.paypal.com.

25, 31]. Unfortunately, these works have different limitations. The work of [22] relies solely on the Casper's location anonymization algorithm [24], which is designed for the Euclidean space, to blur a user location into a set of road segments that intersect a cloaked area determined by the Casper's algorithm. As a result, this work inherits the drawbacks of the Euclidean location anonymization techniques. A hierarchical structure is proposed for location anonymization in road networks [23]. However, such a static hierarchical structure has a deterministic property for its cloaked areas, so it suffers from a reverse engineering attack, e.g., a replay privacy attack [31]. To satisfy the reciprocity property, the work [25] assumes a system-wide static $\mathcal{K}$-anonymity level for all mobile users. This assumption has two major drawbacks [31]: (1) It degrades the quality of services for the mobile users whose desired privacy can be satisfied by smaller $\mathcal{K}$ values. (2) This assumption is not realistic because mobile users tend to have different privacy requirements under different contexts and for different types of objects of interest. Among these related works, the work of [31], XStar, is closest to our work; however, it has some other limitations. (a) Given a road network, XStar constructs a star network by grouping neighboring road segments based on the estimated query execution and communication cost, such that each node in the star network represents a star and has a degree of at least three. Since a node in the star network may be very large, e.g., a star includes three long highways, the underlying basic star structure will result in cloaked segment sets much larger than necessary. Such larger cloaked segment sets not only lead to higher the query execution overhead at a database server, but they also increase the size of candidate lists returned to the user, and thus, the communication overhead also gets higher. (b) In XStar, the extent of shared execution is limited, as it is only applied to queries located on the same road segment. Thus, the concept of shared execution has not been fully utilized.

Our work can distinguish itself from XStar, as (1) it does not rely any underlying basic structure for location anonymization, which also considers the query execution cost and the query quality, i.e., the candidate list size, and (2) it designs a more effective shared execution paradigm, where queries are dynamically grouped to share a set of cloaked segments without any pre-specified location limit. Experimental evidence shows that our proposed algorithms with the shared execution paradigm have more resilience to the replay privacy attack than the state-of-the-art technique, i.e., XStar (Section 7).

3 System model

In this section, we give the preliminaries of this paper, and then describe the underlying system architecture that consists of three main entities, mobile users, location anonymizer, and location-based database server, and present the privacy model of our system.

Preliminaries The proposed location anonymization algorithm mainly blurs a user's location into ***a cloaked set of road segments*** S that is defined as a set of connected road segments where the requesting user is residing therein, such that S satisfies the user specified privacy requirements. In a cloaked set S, a vertex v is a ***closed vertex*** if all edges connected to v are included in S; otherwise, v is an ***open vertex***. The underlying road network is modeled as a connected graph $G = (V, E)$, where V is a vertex set that represents the intersection and endpoints of the road segments, while E is an edge set that represents the road segments.

Mobile users/privacy requirements Mobile users register with the system by specifying their personalized privacy requirements, i.e., *$\mathcal{K}$-anonymity* and *minimum length $\mathcal{L}$*. $\mathcal{K}$-anonymity privacy requirement indicates that the user wants to be $\mathcal{K}$-anonymous, i.e., indistinguishable among $\mathcal{K}$ users. The minimum length privacy requirement $\mathcal{L}$ indicates that the minimum resolution of the blurred location information, i.e., the total length of the road segments in S is at least $\mathcal{L}$. Thus, a user's location must be blurred into a cloaked set S that contains at least $\mathcal{K}$ users and the total length of the road segments in S is at least $\mathcal{L}$. The minimum length privacy requirement is particularly important in a dense area, where a large $\mathcal{K}$-anonymity level results in a cloaked segment set with only a few short road segments. In a special case that a user is located in a very sensitive area, the user can specify a cloaked area that is way beyond the sensitive area, and then the user's location is blurred into a set of connected road segments intersecting the cloaked area. We assume that this special case rarely takes place, and in fact, our location anonymizer can easily deal with it, so this special case is not the focus of this paper.

Location anonymizer The *location anonymizer* is a trusted third party placed between *mobile users* and the *location-based database server*. We assume that the *location anonymizer* is placed at some cellular service provider through which the *mobile users* have access to location-based service providers. Furthermore, the *location anonymizer* maintains an *edge table* that is a hash-table on edge ID [26]. For each edge e, the tuple in the *edge table* stores (a) its endpoints, (b) its length, (c) the set of edges connected to each of its endpoints, and (d) the list of objects currently residing in e. Given n road segments in the underlying road network, m objects in the system, and the maximum degree of an intersection of road segments d, the storage complexity of the edge table is $O(n \times d + m)$. This is because each tuple stores at most $2 \times d$ adjacent edges and each of the m objects is stored by only one tuple. Basically, the *location anonymizer* blurs the location information of a user's query into a cloaked set of road segments S such that S satisfies the user's privacy requirements. While *blurring* the location information, the *location anonymizer* also removes any user identity to ensure the pseudonymity of the location information [28]. Then, the *location anonymizer* sends the anonymized query with the cloaked segment set to the database server. After the *location anonymizer* receives a *candidate list* of answers from the database server, it forwards the *candidate list* to the user. Finally, the user computes an exact answer from the *candidate list*.

Location-based database server The location-based database server is placed at an untrustworthy service provider and has the capacity to deal with private queries along with cloaked sets of road segments. Since the private queries can be easily boiled down to traditional k-nearest-neighbor and range queries (described in Section 4), the query processor embedded inside the database server only needs to employ any existing k-nearest-neighbor and range query algorithms designed for road networks (e.g., [16, 21, 27]). Furthermore, the database server also maintains an *edge table* as in the *location anonymizer*. Instead of returning an exact answer, the database server returns a *candidate list* of answers that is guaranteed to contain the exact answer to the *location anonymizer* regardless of the exact user location within the given cloaked segment set S.

Privacy model In our system, the users are required to be authenticated with the *location anonymizer* and they behave as defined in our algorithm. Since our system only preserves the user location privacy for *snapshot* location-based queries, we assume that an adversary is unable to infer that some particular snapshot queries are issued by the same user or track a particular user. This assumption is realistic, as the *location anonymizer* can guarantee the pseudonymity of the location information [28]. In other words, the output of the *location anonymizer* to the *location-based database server* is only a set of road segments and a location-based query (e.g., a range or k-nearest-neighbor query) without any user identity. Furthermore, the query can be issued by one user or a group of users formed by the shared execution paradigm. We will also describe two attack models, namely, *center-of-cloaked-area attack* and *replay attack*. For the replay privacy attack, we assume the worst scenario where an adversary knows the users' locations, but not their user identities, and the location anonymization algorithm. Given a cloaked segment set of a user along with a query, the adversary wants to employ the replay attack to find the road segment that contains the query issuer or the center-of-cloaked-area attack to find the exact location of the query. We will evaluate our system's resilience to these two privacy attacks through simulated experiments (Section 7).

4 Cost model for private queries

This section develops the cost model for both private k-nearest-neighbor (k-NN) and range queries. This cost model will be used later by the *location anonymizer* (Section 5) to find a cloaked set of road segments S that balances between minimizing the developed cost function while satisfying the user specified privacy requirements. Throughout this section, we use the following terminologies and assumptions: (a) We assume only an existing spatio-temporal query processor embedded inside the database server to deal with private k-NN and range queries. Thus, the query processor can employ any existing k-NN and range query processing algorithms designed for road networks (e.g., [16, 17, 21, 27]). (b) For any cloaked set of road segments S, we define two functions $V_o(S)$ and $E(S)$ that return the number of *open vertices* (i.e., vertices where some of its connected edges are not included in S) and the number of edges in S, respectively. (c) Without loss of generality, we assume that all k-NN and range queries ask about the same type of target objects (e.g., gas stations, taxis, or restaurants). There are T such target objects and R road segments in the system. The information about the number of target objects is given by the database server as hints. For all vertices and edges in the road network, d and l represent the average degree of connectivity of a vertex v (i.e., the number of edges connected to v) and the average edge length, respectively.

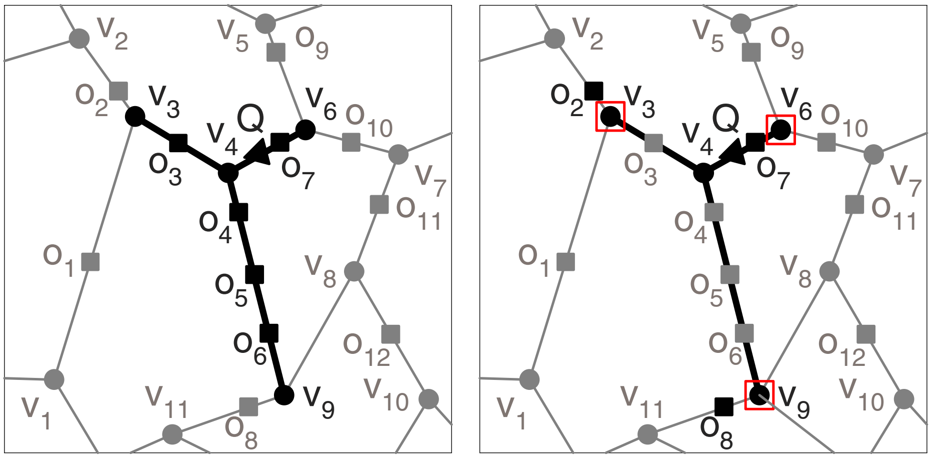
4.1 Private k-nearest-neighbor queries

A typical example of a k-nearest-neighbor (k-NN) query is "*find the k nearest objects of my location* $q = (x, y)$". However, with the anonymization process, the k-NN query is transformed to a *private* one, i.e., "*find the k nearest objects of my location, given that my location is somewhere in a cloaked set of road segments S*".

Algorithm An algorithm for a *private* k-NN query with a cloaked set of road segments S will find a *candidate list* of answers where the exact answer of the original query is guaranteed to be in the *candidate list* regardless of the actual user location within S. To facilitate the development of the query cost model, given a cloaked set of road segments S, we divide the query processing algorithm of *private* k-NN queries into two steps: (1) *Range Search Step.* In this step, we mainly execute a traditional range query of the form "*find all target objects located within the road segments in S*" where we add all target objects within S to the *candidate list*. (2) *External Search Step.* In this step, we mainly execute a traditional k-NN query at each *open vertex* in S, i.e., "*find the closest k target objects to a vertex v*", where we add the answers of these queries to the *candidate list*.

Thus, the execution of one *private* k-NN query boils down to executing one *traditional* range query and a set of *traditional* k-NN queries. Figure 2 depicts a *private* k-NN query ($k = 1$), where the actual query location Q is represented as a triangle located in edge v_4v_6. The cloaked segment set S of Q includes three edges v_3v_4, v_4v_6, and v_4v_9, one *closed vertex* v_4, and three *open vertices* v_3, v_6, and v_9. Figure 2a depicts the *range search step*, where we add all target objects of the three edges in S, o_3 to o_7, to the *candidate list*. Figure 2b depicts the *external search step*, where the nearest target object to each *open vertex* in S (enclosed in rectangles) is added to the *candidate list*. The nearest target object of the *open vertices* v_3, v_6, and v_9 are o_2, o_7, and o_8, respectively. As a result, the answer of the *private* k-NN query is a *candidate list* that contains seven objects o_2 to o_8. Notice that the exact answer, according to the exact location of Q, is o_7 which is included in the *candidate list*.

Cost model The execution cost of a *private* k-NN query is the sum of the execution cost of the *range search* and *external search* steps. We will present the cost in terms



(a) Range Search Step (b) External Search Step

Fig. 2 Private k-nearest-neighbor query

of the number of edges whose information is retrieved through the *edge table* data structure as described in Section 3. For the *range search* step, we need to retrieve the information (i.e., target objects) of all edges in S. This step results in a straightforward cost of $E(S)$, i.e., the number of edges in S. For the *external search* step, we will first consider the cost of issuing a *traditional* k-NN query at one *open vertex*. For simplicity, we assume a uniform distribution of the T target objects over all the road segments R in the road network. Thus, we will need to search within R/T segments to find one closest target object to an *open vertex*. To find k nearest target objects, we need to retrieve the information of $R/T \times k$ road segments, as we do the same for each *open vertex* in S. The total cost of the *external search* step is $V_o(S) \times R/T \times k$, where $V_o(S)$ represents the number of *open vertices* in S. Given a cloaked segment set S, the query execution cost of a *private* k-NN query is:

$$\mathrm{Cost}_{PkNN}(S, k) = E(S) + V_o(S) \times R/T \times k. \tag{1}$$

Note that our proposed location anonymization does not have any assumption for the distribution of target objects and road segments, so it can use any object distribution model, which is either provided by the service provider or estimated by our system if appropriate statistics can be collected from the service provider, and a more sophisticated distribution model for road segments. However, the development of these distribution models is out of the scope of this paper.

4.2 Private range queries

A typical example of a range query is "*find all target objects within a network range distance r of my location $q = (x, y)$*". However, with the anonymization process, the range query is transformed into a *private* one, i.e., "*find all target objects within a network range distance r of my location, given that my location is somewhere in a cloaked set of road segments S*".

Algorithm Similar to the case of *private* k-NN queries, an algorithm for a *private* range query consists of two steps: (1) *Range Search Step.* The target objects residing in the edges in S are added to the *candidate list* of answers. (2) *External Search Step.* The target objects within a network range distance r from each *open vertex* in S are added to the *candidate list*. Thus, a *private* range query boils down to $V_o(S) + 1$ *traditional* range queries.

Cost model The execution cost of the *range search step* of a *private* range query is exactly the same as in the case of *private* k-NN queries, i.e., $E(S)$. The execution cost of the *external search* step is the sum of the cost of finding the target objects within a network range distance r of each *open vertex* in S. For each *open vertex* in S, we need to expand the search to $\lceil r/l \rceil$ edges in all directions where l is the average edge length. Given that the average degree of connectivity of a vertex is d, then, approximately, we need to search a total of $\lceil r/l \rceil \times d$ road segments for each *open vertex* in S. Thus, given a cloaked set of road segments S, the query execution cost of a *private* range query is:

$$\mathrm{Cost}_{PRange}(S, r) = E(S) + V_o(S) \times \lceil r/l \rceil \times d. \tag{2}$$

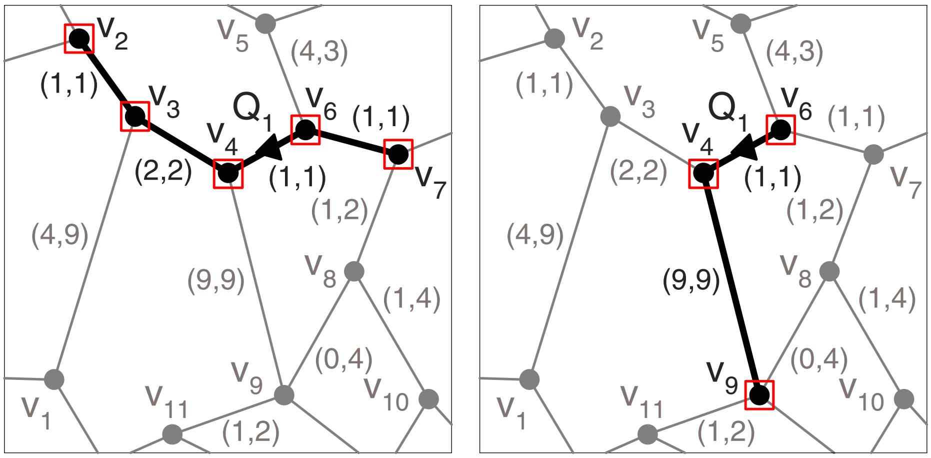
5 Query-aware anonymization

As we have indicated earlier, there are two main factors that control the quality of a cloaked set of road segments S, namely, the query execution cost and the query quality. To realize the first factor, query execution cost, we will need to select S that satisfies the user privacy requirements while minimizing the query cost models that are developed in Section 4. On the other hand, to realize the second factor, the query quality, we need to select S that satisfies the user privacy requirements and has the number of users and length as close as possible to the anonymity $\mathcal{K}$ and minimum length $\mathcal{L}$ privacy requirements, respectively. The main idea is that the shorter the length of S, the smaller the size of the *candidate list*, and hence the better query quality the query processor will provide.

In this section, we start by showing that realizing any of these two important factors for S may significantly deteriorate the other factor (Section 5.1). Then, we develop an objective cost function that aims to balance these two desired factors, the query execution cost and the query quality (Section 5.2). Finally, we propose two greedy-based anonymization approaches, namely, *pure greedy* and *randomized greedy*, that aim to find a cloaked set of road segments S that minimizes our developed objective cost function (Section 5.3).

5.1 Motivation: a trade-off between query execution cost and query quality

Figure 3 gives a trade-off between query execution cost and query quality for a cloaked set of road segments S with privacy requirements $\mathcal{K} = 5$ and $\mathcal{L} = 5$. Each edge in the road network has a pair (a, b), where a indicates the number of users in that edge while b indicates the edge length. For example, the edge v_1v_3 has four users and of length nine. Figure 3a gives a cloaked segment set $S_q = \{v_2v_3, v_3v_4, v_4v_6, v_6v_7\}$ that is optimal in terms of the query quality. In this case, S_q has the minimum



(a) Optimal Query Quality (b) Optimal Query Cost

Fig. 3 A trade-off between the query quality and the query execution cost

possible length, i.e., Length$(S_q) = 5$, and the minimum possible number of users, i.e., NumUser$(S_q) = 5$. In terms of the query execution cost, S_q has five *open vertices* (enclosed in rectangles) and four edges (represented as black lines), i.e., $V_o(S_q) = 5$ and $E(S_q) = 4$. On the other hand, Fig. 3b gives a cloaked segment set $S_c = \{v_4v_6, v_4v_9\}$ that is optimal in terms of query execution cost as it is the one that has the minimum possible number of *open vertices*, i.e., $V_o(S_c) = 3$, and edges, i.e., $E(S_c) = 2$, while still satisfying the user privacy requirements. However, S_c may result in very bad query quality as the total length of all road segments in S_c is 10 which is way above the minimum length $\mathcal{L}$ privacy requirement. Table 1 gives a summary of the number of open vertices, the number of edges and the total segment length of S_q and S_c.

From these examples, we can see that trying to find the optimal S_q, i.e., maximizing the query quality, results in having five *open vertices* and four edges (Fig. 3a) which is 66.7% (i.e., $\frac{5-3}{3} \times 100\%$) and 100% (i.e., $\frac{4-2}{2} \times 100\%$) worse than what we can get from S_c (Fig. 3b). On the other hand, trying to find the optimal S_c that gives the minimal query execution cost, i.e., the minimum possible number of *open vertices* and edges, results in the total segment length of 10 (Fig. 3b) which is 100% ($\frac{10-5}{5} \times 100\%$) worse than what we can get from S_q (Fig. 3a). This raises the issue of finding an objective cost function that balances between the query execution cost and the query quality.

5.2 Objective cost function

As we have discussed in the previous section, we need an objective cost function that balances between query execution cost and query quality. The query execution cost is measured by the cost models described in Section 4. On the other hand, the query quality is measured by the number of candidate target objects returned by a database server given that the query location is within a cloaked set of road segments S. In general, the number of candidate target objects returned to the user, i.e., the query quality, is proportional to the total length of the road segments in S. Thus, the objective cost function aims to find a cloaked segment set S, such that S not only satisfies the user privacy requirements, but it also balances the query execution cost and the total segment length (i.e., the query quality).

Based on these contradicting requirements, we distinguish between two cases for the objective cost function based on whether a privacy requirement has satisfied or not. For the $\mathcal{K}$-anonymity privacy requirement, we consider two cases. **Case 1:** When a current cloaked set of segments S has not yet satisfied this privacy requirement, we should select some edges that contain more users to S. For example, if adding edge e_i or e_j to S incurs the same query execution cost QCost(S) that is computed by using the cost models described in Section 4, the objective cost function should give a smaller cost to the edge containing more users; and thus, the objective cost function adjusts QCost(S) by dividing it by a quality adjustment factor

Table 1 Two sample cloaked segments sets S_q and S_c

	No. of open vertices (V_o)	No. of edges (E)	Total segment length
S_q	5	4	5
S_c	3	2	10

NumUser(S)/$\mathcal{K}$, where an edge with more users yields a larger quality adjustment factor and NumUser(S)/$\mathcal{K} < 1$ because the total number of users in S is less than $\mathcal{K}$. Note that NumUser(S) must be at least one because the firstly selected edge contains the requesting user. **Case 2:** When a current cloaked set of segments S has satisfied the $\mathcal{K}$-privacy requirement, since further adding edges with more or less users to S does not affect the query quality, i.e., the total length of the segments in S, the objective cost function no longer considers the $\mathcal{K}$-anonymity privacy requirement; hence, the quality adjustment factor is set to one.

For the minimum length $\mathcal{L}$ privacy requirement, we also consider two cases. **Case 1:** When a current cloaked set of segments S has not yet satisfied this privacy requirement, we should select some longer edges to S. Similar to the $\mathcal{K}$-anonymity privacy requirement, the objective cost function adjusts QCost(S) by dividing it by a quality adjustment factor Length(S)/$\mathcal{L}$, where a longer edge yields a larger quality adjustment factor and Length(S)/$\mathcal{L} < 1$ because the total length of the road segments in S is less than $\mathcal{L}$. **Case 2:** In case that the length privacy requirement has been satisfied by a current cloaked set of segments S, the objective cost function gives a larger value to longer edges, i.e., dividing QCost(S) by an inverse quality adjustment factor $\mathcal{L}$/Length(S), where a longer edge yields a smaller quality adjustment factor and $\mathcal{L}$/Length(S) > 1.

Since the quality adjustment factors of these two privacy requirements are normalized to the corresponding privacy requirement, we can encapsulate them in our objective cost function. Hence, the combined quality adjustment factors are:

$$\mathsf{QualityAdj}(S) = \begin{cases} \dfrac{\text{NumUser}(S)}{\mathcal{K}} \times \dfrac{\text{Length}(S)}{\mathcal{L}}, & \text{if NumUser}(S) < \mathcal{K} \wedge \text{Length}(S) < \mathcal{L}; \\ \dfrac{\text{NumUser}(S)}{\mathcal{K}} \times \dfrac{\mathcal{L}}{\text{Length}(S)}, & \text{if NumUser}(S) < \mathcal{K} \wedge \text{Length}(S) \geq \mathcal{L}; \\ 1 \times \dfrac{\text{Length}(S)}{\mathcal{L}}, & \text{if NumUser}(S) \geq \mathcal{K} \wedge \text{Length}(S) < \mathcal{L}; \\ 1 \times \dfrac{\mathcal{L}}{\text{Length}(S)}, & \text{if NumUser}(S) \geq \mathcal{K} \wedge \text{Length}(S) \geq \mathcal{L}. \end{cases} \tag{3}$$

Therefore, for each edge e_i, we consider a new potential cloaked set of road segments $S = S \cup \{e_i\}$, and we calculate the objective cost $\mathsf{Cost}(S, Q)$ as:

$$\mathsf{Cost}(S, Q) = \frac{\text{QCost}(S, Q)}{\mathsf{QualityAdj(S)}}, \text{ where} \tag{4}$$

$$\text{QCost}(S, Q) = \begin{cases} \text{Cost}_{PkNN}(S, k), & \text{if } Q \text{ is a } k\text{-NN query;} \\ \text{Cost}_{PRange}(S, r), & \text{if } Q \text{ is a range query.} \end{cases} \tag{5}$$

5.3 Greedy approaches

Trying to find the optimal set of cloaked road segments S that minimizes a certain cost function would require an exhaustive search process. As the underlying application of road network anonymization (e.g., location-based services) is a real-time application in which it is crucial to efficiently process the anonymization and query processing in a minimal time, we avoid trying to get an optimal set S. Instead, we use a greedy approach to minimize the objective cost function developed in Section 5.2. In particular, we present two efficient greedy approaches, namely *pure greedy* and *randomized greedy* approaches, that rely on the objective cost function designed for balancing a trade-off between the query execution cost and the query quality.

5.3.1 Pure greedy approach

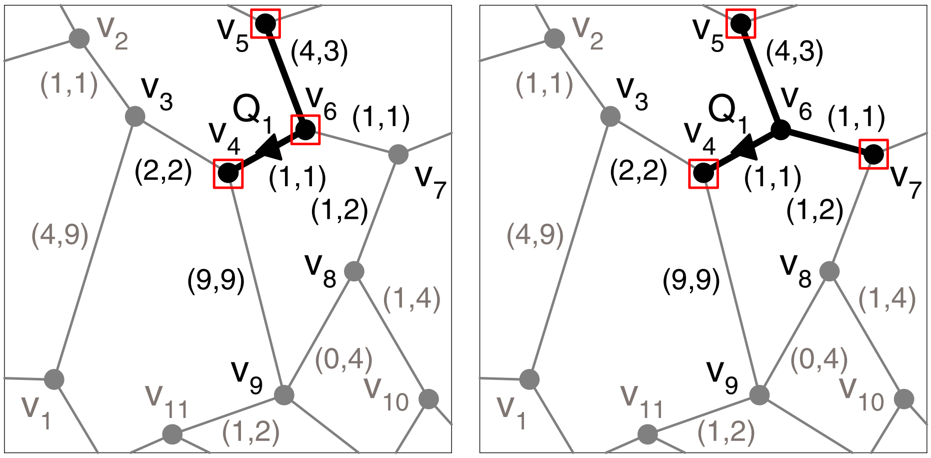
The idea of our greedy approach is to start from the road segment in which the querying user is residing. If this road segment satisfies the user's privacy requirements, we return it to a location-based database server as the cloaked road segment S. Otherwise, we check all adjacent road segments of S and greedily pick the road segment that minimizes the objective cost function. We keep adding these adjacent road segments greedily to S until S satisfies the user privacy requirements. As the objective cost function is heavily dependent on the underlying query execution cost model, such approach may result in different S for different query types.

Algorithm 1 depicts the pseudo code of the pure greedy approach of our query-aware location anonymization algorithm. The algorithm has two input parameters, the identifier of the user U who issues the query and the issued query Q. The algorithm starts by initializing a cloaked set of road segments S with the edge e that includes the user U (Line 3 in Algorithm 1). If the current S does not satisfy U's privacy requirements, i.e., $U.\mathcal{K}$ and $U.\mathcal{L}$, we do the following three steps (Lines 4 to 8 in Algorithm 1): (1) We construct the set R that consists of all road segments that are adjacent to some road segment in S. (2) For each road segment $e_i \in R$, we calculate the objective cost to decide whether e_i should be added to S for the input query Q based on the objective cost function designed in Section 5.2. Then, we choose the best edge as the one that minimizes the objective cost function should it have been added to S. (3) We add that best edge to the current cloaked set of road segments S. We keep doing these three steps until S satisfies U's privacy requirements, i.e., $\text{NumUser}(S) \geq U.\mathcal{K}$ and $\text{Length}(S) \geq U.\mathcal{L}$. In other words, the algorithm will put additional segments to S if the total number of users in S is less than $\mathcal{K}$ or the total length of the segments in S is less than $\mathcal{L}$. Finally, we return S as the cloaked segment set for the user U. It is important to note that S does satisfy the user privacy requirements while it aims to minimize the query execution cost and maximize the query quality according to our developed objective cost function.

Algorithm 1 A Greedy Approach

```
1: function GREEDY (User $U$, Query $Q$)
2: $e \leftarrow$ the road segment contains the user $U$
3: $S \leftarrow \{e\}$
4: while $\text{NumUser}(S) < U.\mathcal{K}$ or $\text{Length}(S) < U.\mathcal{L}$ do
5:     $R \leftarrow$ All adjacent road segments of $S$
6:     BestEdge $\leftarrow \mathbf{argmin}_{e_i \in R} \left(\text{Cost}(S \cup e_i, Q)\right)$
7:     $S \leftarrow S \cup$ BestEdge
8: end while
9: return $S$
```

Figure 4 gives an example to illustrate the greedy approach for the query-aware location anonymization algorithm where the user issues a k-NN query Q_1 where $k = 3$ while being five anonymous within a set of connected road segments with a total length of at least five, i.e., $\mathcal{K} = 5$ and $\mathcal{L} = 5$. For ease of comparison and



(a) Add v_5v_6 to $S = \{v_4v_6\}$ (b) Add v_6v_7 to $S = \{v_4v_6, v_5v_6\}$

Fig. 4 Example of the greedy approach

discussion, we use the same road network that was used in Fig. 3. Although we show only part of the road network, we assume that the whole road network has $R = 100{,}000$ road segments and $T = 1{,}000$ target objects. Hence, according to the query cost model developed in Section 4, we will need to search $R/T \times k = 300$ edges for each *open vertex* in a cloaked segment set S. Since edge v_4v_6 contains Q_1, we initially set the cloaked set of road segments S to $\{v_4v_6\}$, where vertices v_4 and v_6 are *open vertices* that are enclosed in rectangles. As S does not satisfy the user privacy requirements, i.e., NumUser$(S) = 1$ and Length$(S) = 1$, we compute the objective cost of the adjacent edges of v_4v_6 and add the edge with the lowest cost to S. The cost of these adjacent edges is $\mathsf{Cost}(S \cup v_5v_6) = (3 \times 300 + 2)/(\min(5/5, 1) \times \min(4/5, 5/4)) = 902/(1 \times 4/5) = 1127.5$, $\mathsf{Cost}(S \cup v_6v_7) = 902/(2/5 \times 2/5) = 5637.5$, $\mathsf{Cost}(S \cup v_3v_4) = 902/(3/5 \times 3/5) = 2505.6$, and $\mathsf{Cost}(S \cup v_4v_9) = 902/(1 \times 5/10) = 1804$. Since edge v_5v_6 has the smallest objective cost, we add v_5v_6 to S (Fig. 4a).

However, $S = \{v_4v_6, v_5v_6\}$ satisfies only the $\mathcal{K}$-anonymity requirement, i.e., NumUser$(S) = 5$ and Length$(S) = 4$, so we compute the objective cost of the adjacent edges of the edges in S. The cost of these adjacent edges is $\mathsf{Cost}(S \cup v_6v_7) = (3 \times 300+3)/(\min(6/5, 1) \times \min(5/5, 5/5)) = 903/(1 \times 5/5) = 903$, $\mathsf{Cost}(S \cup v_3v_4) = (4 \times 300+3)/(\min(7/5, 1) \times \min(6/5, 5/6)) = 1203/(1 \times 5/6) = 1443.6$, and $\mathsf{Cost}(S \cup v_4v_9) = (4 \times 300+3)/(\min(14/5, 1) \times \min(13/5, 5/13)) = 1203/(1 \times 5/13) = 3127.8$. Since edge v_6v_7 has the smallest objective cost, we add v_6v_7 to S (Fig. 4b). Finally, $S = \{v_4v_6, v_5v_6, v_6v_7\}$ satisfies the user privacy requirements, i.e., NumUser$(S) = 6 \geq \mathcal{K}$ and Length$(S) = 5 \geq \mathcal{L}$, and there are three *open vertices* in S, i.e., v_4, v_5, and v_7, that are enclosed in rectangles. This example shows that the developed objective cost function effectively balances between the query execution cost and the query quality, because the cloaked segment set S has the maximal query quality, i.e., Length$(S) = \mathcal{L} = 5$, and the query execution cost of S (i.e., $V_o(S) = 3$ and $E(S) = 3$) is only slightly deteriorated compared to the optimal query cost (i.e., $V_o(S_c) = 3$ and $E(S_c) = 2$) given in Section 5.1.

5.3.2 Randomized greedy approach

Although the greedy approach is simple and effective in terms of achieving privacy requirements, minimizing the query execution cost, and maximizing the query quality, the greedy approach is vulnerable to adversary attacks. Basically, we assume the worst case that an adversary knows the number of users and length of each road segment, and the adversary is able to crack the system to know or guess the used objective cost function. Then, the adversary can know the road segment in which the user issuing the query is residing through a reverse engineering process. For example, given a cloaked set of road segments S, the adversary performs two main steps. (1) The adversary selects a certain edge $e_i \in S$, and then applies the revealed cost function in a greedy manner for e_i to compute another cloaked set of road segments S'. (2) If there is an edge e_j where $e_j \in S'$ and $e_j \notin S$, the adversary knows that the user is not in e_i. The main idea is that if the user is in e_i, S' should always be included in S, i.e., $S' \subseteq S$, before S' is equal to S. The adversary can repeat these two steps for each edge in S for all possible combinations of the user's privacy requirements, i.e., $\mathcal{K}$ and $\mathcal{L}$. When the adversary achieves a case that $S' = S$, it is likely that the user who issued the query is in e_i.

The main reason of having such a privacy attack is that the greedy approach relies on a deterministic cost function to determine a cloaked set of road segments S. To this end, we propose a randomized greedy approach for our query-aware location anonymization algorithm. The idea is to inject some randomness into the process of selecting road segments to S. Rather than solely relying on a greedy approach, we will alternate between greedy and random approaches. Although injecting randomness into the anonymization process may degrade the query processing performance, i.e., the query execution cost and the query quality, but it significantly reduces, if not prevent, the possibility of the adversary attack. To balance between the query processing performance and the vulnerability of adversary attacks, we introduce a user parameter, *random factor* (Rand_F), that controls the level of randomness injected into the greedy approach. When Rand_F is set to 0, our randomized greedy approach acts exactly as the pure greedy approach. On the other hand, when Rand_F is set to 1, our randomized greedy approach acts in a purely random way where we keep randomly selecting the adjacent edges of a current cloaked segment set S until S satisfies the user privacy requirements. In general, a larger Rand_F provides more secure privacy for the user, yet it results in lower query processing performance. It is important to note that both the pure greedy and randomized greedy approaches are guaranteed to generate a cloaked set of road segments satisfying both the $\mathcal{K}$-anonymity and minimum length $\mathcal{L}$ privacy requirements.

Algorithm 2 gives the pseudo code of our randomized greedy approach. The pseudo code is similar to that of the pure greedy approach in Algorithm 1 except for the part of choosing an edge in the set of adjacent edges of S, R, to be added to the current cloaked set of road segments S (Lines 6 to 11 in Algorithm 2). Basically, the randomized greedy approach has an option to randomly select road segments to S. Thus, we generate a random number between 0 and 1. If the input parameter Rand_F is greater than the generated random number, we opt to randomly select a road segment among the ones stored in R, i.e., the adjacent road segments of S. On the other hand, if the input parameter Rand_F is less than the generated random number, we use our greedy approach to select the road segment from R that minimizes the objective cost function described in Section 5.2. By injecting

Algorithm 2 A Randomized Greedy Approach

```
1: function RandomizedGreedy (User $U$, Query $Q$, Float $\text{Rand}_F$)
2: $e \leftarrow$ the road segment contains the user $U$
3: $S \leftarrow \{e\}$
4: while NumUser($S$) $< U.\mathcal{K}$ or Length($S$) $< U.\mathcal{L}$ do
5:     $R \leftarrow$ All adjacent road segments of $S$
6:     $\text{Rand}_N \leftarrow$ a random number between 0 and 1, i.e., [0, 1)
7:     if $\text{Rand}_F > \text{Rand}_N$ then
8:         BestEdge $\leftarrow$ a road segment is randomly selected from $R$
9:     else
10:        BestEdge $\leftarrow \mathbf{argmin}_{e_i \in R} \left(\text{Cost}(S \cup e_i, Q)\right)$
11:    end if
12:    $S \leftarrow S \cup$ BestEdge
13: end while
14: return $S$
```

randomness to the anonymization process, we can prevent the reverse engineering attack as an adversary cannot guess the generated random numbers.

6 Shared execution paradigm

As a location-based database server is likely to receive a numerous number of concurrent queries, processing these queries individually would pose a system bottleneck. To tackle this scalability issue, we propose a *shared execution* paradigm that aims to minimize the number of queries executed by the database server for a set of private queries. The main idea is to maximize the number of common vertices and edges in the cloaked set of road segments of a set of similar queries issued from nearby users. Two or more queries are similar if they belong to the same query type and interested in the same target object type. It is important to note that the proposed *shared execution* paradigm can be incorporated into both the *pure greedy* and *randomized greedy* approaches for query-aware location anonymization. In this section, we first give the motivation of the *shared execution* paradigm, and then present the algorithm with a running example.

6.1 Motivation

Figure 5 depicts a motivating example of the proposed *shared execution* paradigm, in which the *location anonymizer* receives three similar queries $\mathcal{Q} = \{Q_1, Q_2, Q_3\}$ at the same time. The privacy requirements of these queries are ($Q_1.\mathcal{K}$=5, $Q_1.\mathcal{L}$=5), ($Q_2.\mathcal{K}$=6, $Q_2.\mathcal{L}$=7), and ($Q_3.\mathcal{K}$=3, $Q_3.\mathcal{L}$=6). Without the concept of shared execution, we use the pure greedy approach to anonymize the location information of these queries individually. Figure 4b gives the cloaked set of road segments S_1 of Q_1, while the cloaked segment sets S_2 and S_3 of Q_2 and Q_3 are depicted in Fig. 5a. Anonymizing these queries individually results in three cloaked segment sets $S_1 = \{v_4v_6, v_5v_6, v_6v_7\}$, $S_2 = \{v_1v_3, v_2v_3, v_3v_4\}$, and $S_3 = \{v_6v_7, v_7v_8, v_8v_{10}\}$ that contain a total of nine edges (represented as black lines) and ten *open vertices*

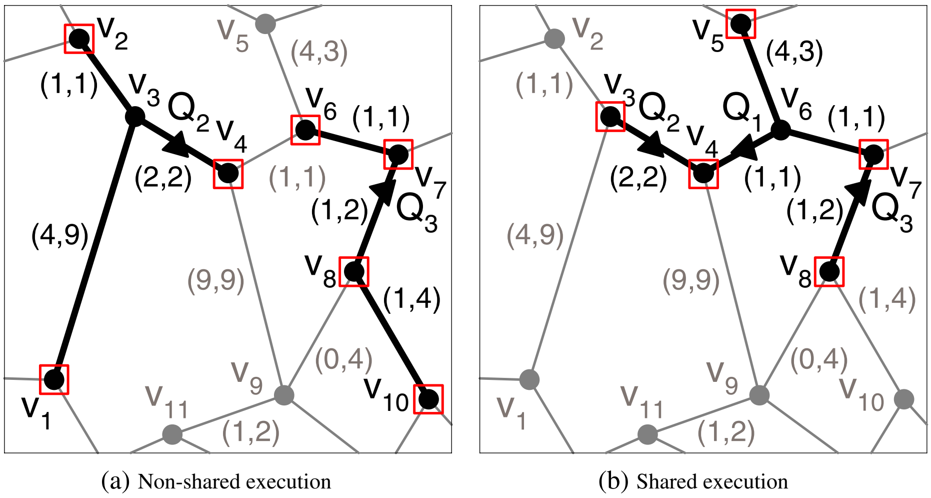


Fig. 5 Motivating example of shared execution

(enclosed in rectangles), i.e., $\{v_4, v_5, v_7\}$ in S_1, $\{v_1, v_2, v_4\}$ in S_2, and $\{v_6, v_7, v_8, v_{10}\}$ in S_3. Thus, the database server has to retrieve the target objects of these nine edges and execute the requested query at each of these ten *open vertices*.

With the proposed *shared execution* paradigm, the location anonymization algorithm aims to maximize the number of common *open vertices* and edges of the cloaked segment sets of a set of similar queries. Figure 5b depicts the set $\mathcal{S}$ of distinct edges in the cloaked segment sets S_1, S_2, and S_3 that are computed by the pure greedy approach with our *shared execution* paradigm, in which $S_1 = \{v_4v_6, v_5v_6, v_6v_7\}$, $S_2 = \{v_3v_4, v_4v_6, v_5v_6, v_6v_7\}$, and $S_3 = \{v_3v_4, v_4v_6, v_6v_7, v_7v_8\}$. Hence, $\mathcal{S}$ contains five distinct edges (represented as black lines), i.e., v_3v_4, v_4v_6, v_5v_6, v_6v_7, and v_7v_8, and five distinct *open vertices* (enclosed in rectangles), i.e., v_3, v_4, v_5, v_7, and v_8. This example shows that the *shared execution* paradigm reduces the number of edges and *open vertices* among the cloaked segment sets of the three queries by 44.4% (i.e., from nine edges to $E(\mathcal{S}) = 5$) and 50% (i.e., from ten *open vertices* to $V_o(\mathcal{S}) = 5$), respectively. As a result, the database server needs to retrieve the target objects of five edges and execute the requested query at five *open vertices*, and then share the answer of these queries among Q_1, Q_2, and Q_3.

6.2 Algorithm

Main idea Given a set of n similar queries $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_n\}$ received by the *location anonymizer* at the same time or within a short time interval, the *shared execution* paradigm aims to maximize the number of common *open vertices* and edges in the cloaked segment sets $S_1, S_2, \ldots, S_n$ for the queries in $\mathcal{Q}$. $\mathcal{S}$ is a set of distinct edges in the cloaked segment sets $S_1, S_2, \ldots, S_n$, i.e., $\mathcal{S} = \bigcup_{1 \le i \le n} S_i$. In the *shared execution* paradigm, when we are selecting an edge to a current cloaked segment set S_i for a query Q_i, we give a higher priority to select an edge that is adjacent to S_i and has been selected by other preceding queries in $\mathcal{Q}$ for their cloaked segments sets. In other words, given a set of adjacent edges R to S_i, we first select an edge from $\{R \cap \mathcal{S}\}$. The main reason is that adding any edge from $\mathcal{S}$ will have a zero query execution cost because the query execution cost of the edges in $\mathcal{S}$ is absorbed by other queries in $\mathcal{Q}$. Thus, by using the quality function described in Section 5.2, we add the edge that gives the highest quality of answers among the set $\{R \cap \mathcal{S}\}$ to S_i. However, in case that $\{R \cap \mathcal{S}\}$ is empty, we simply employ our objective cost function to select the best candidate edge to S_i. Although the *shared execution* paradigm would incur longer anonymization time for some queries, it improves the average anonymization time of the queries in the similar query set, and this sacrifice will be paid off by the significant gain in the query processing at the database server. Experimental results in Section 7 give more details on this performance gain.

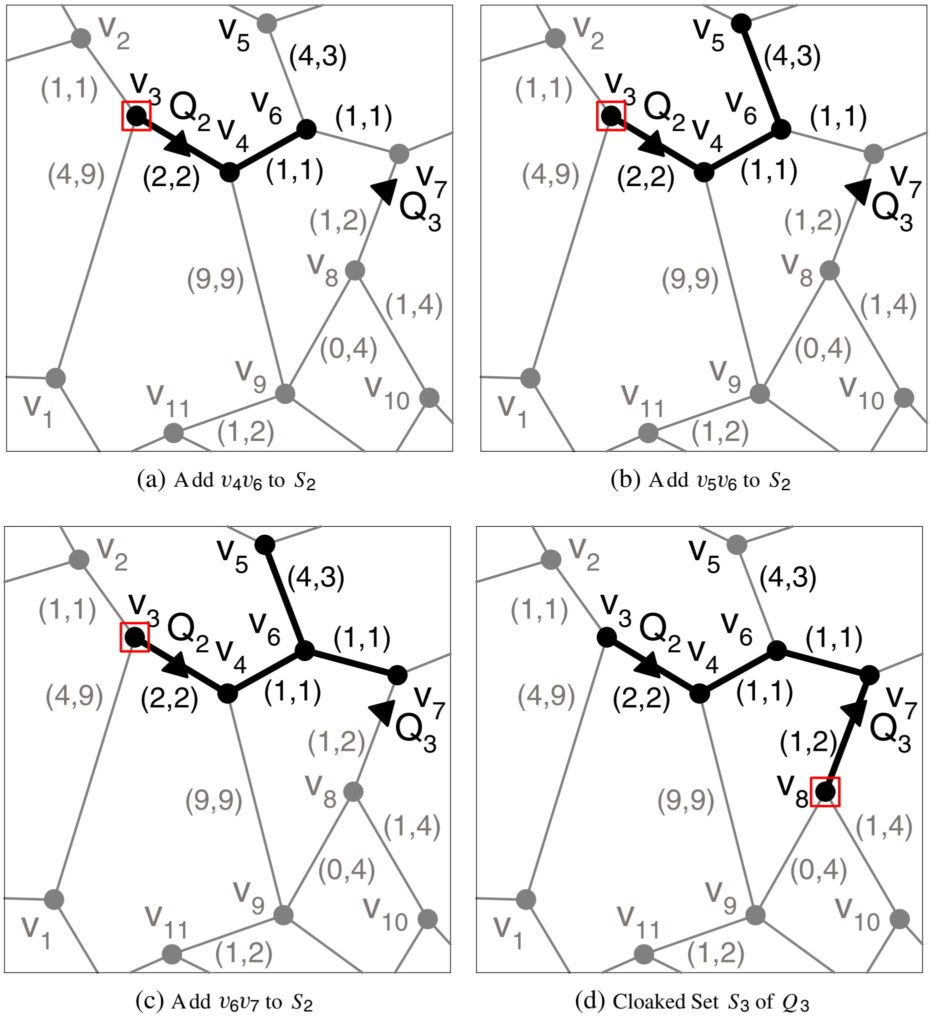
Algorithm Algorithm 3 depicts the pseudo code of the *shared execution* paradigm applied to the pure greedy approach (or the randomized greedy approach with the same modifications). The input of the algorithm is a set of similar queries $\mathcal{Q}$. We maintain a set $\mathcal{S}$ to store the distinct edges in the cloaked set of road segments of each query in $\mathcal{Q}$. Initially, we set $\mathcal{S}$ to empty and sort the queries in $\mathcal{Q}$ by their parameters in decreasing order, i.e., the value of k of k-NN queries or the network range distance r of range queries (Lines 2 to 3 in Algorithm 3). The main idea behind this sorting is to ensure that the answer of the requested query at a selected *open vertex* can be used by a database server to answer the subsequent queries in $\mathcal{Q}$. For example, a k-NN query answer can be used to answer another k'-NN query if $k \geq k'$. For each query Q_i in $\mathcal{Q}$, the proposed *shared execution* paradigm is applied to the pure greedy approach (or the randomized greedy approach) to find a cloaked set of road segments S_i. First, we set the edge that contains the querying user U_i to S_i (Line 6 in Algorithm 3). If S_i does not satisfy the user privacy requirements, i.e., $U_i.\mathcal{K}$ and $U_i.\mathcal{L}$, we repeatedly select the adjacent edges of S_i, R, to S_i until S_i satisfies the user privacy requirements. We distinguish between two cases of selecting the best edge to S_i. Case 1: $\{R \cap \mathcal{S}\}$

Algorithm 3 A Greedy Approach with Shared Execution

```
1: function SharedExecutionGreedy (QuerySet $\mathcal{Q}$)
2: $\mathcal{S} \leftarrow \{\emptyset\}$
3: Sort $\mathcal{Q}$ by the query parameter in decreasing order
4: for Each query $Q_i \in \mathcal{Q}$ do
5:     $e \leftarrow$ the road segment contains the user $U_i$ of $Q_i$
6:     $S_i \leftarrow \{e\}$
7:     while NumUser($S_i$) $< U_i.\mathcal{K}$ or Length($S_i$) $< U_i.\mathcal{L}$ do
8:         $R \leftarrow$ All adjacent road segments of $S_i$
9:         if $R \cap \mathcal{S}$ is empty then
10:            $e \leftarrow \mathbf{argmin}_{e_j \in R} \left(\mathsf{Cost}(S_i \cup e_j, Q_i)\right)$
11:        else
12:            $e \leftarrow \mathbf{argmax}_{e_j \in \{R \cap \mathcal{S}\}} \left(\mathsf{Quality}(S_i \cup e_j)\right)$
13:        end if
14:        $S_i \leftarrow S_i \cup e$
15:    end while
16:    $\mathcal{S} \leftarrow \mathcal{S} \cup S_i$
17: end for
18: return $\mathcal{S}$
```

is empty. In this case, there is no adjacent edge of S_i in $\mathcal{S}$. Thus, we simply use the objective cost function to select an edge with the best cost among the set of adjacent edges of S_i, R (Line 10 in Algorithm 3). Case 2: $\{R \cap \mathcal{S}\}$ *is not empty*. In this case, there are some adjacent edges of S_i are in $\mathcal{S}$. We select an edge with the largest quality from the set $\{R \cap \mathcal{S}\}$ (Line 12 in Algorithm 3). It is important to note that choosing any edge from $\{R \cap \mathcal{S}\}$ just adds a zero cost to the query processing. After we find a cloaked set of road segments S_i for Q_i, we update $\mathcal{S}$ accordingly (Line 16 in Algorithm 3). Finally, we send the queries in $\mathcal{Q}$ along with their cloaked segment sets S_i in a batch to the database server.

Example Figure 6 depicts an example of the *shared execution* paradigm applied to the pure greedy approach where the query set contains three similar queries



(a) Add v_4v_6 to S_2 (b) Add v_5v_6 to S_2

(c) Add v_6v_7 to S_2 (d) Cloaked Set S_3 of Q_3

Fig. 6 An example of the shared execution scheme for a query set $\mathcal{Q} = \{Q_1, Q_2, Q_3\}$

$\mathcal{Q} = \{Q_1, Q_2, Q_3\}$ that are sorted by their query parameters in decreasing order. The privacy requirements of these queries are the same as in Section 6.1. First, we compute the cloaked set of road segments S_1 of Q_1. Since Q_1 is the first query in $\mathcal{Q}$, i.e., $\mathcal{S}$ is empty, we compute S_1 based on the pure greedy approach, as depicted in Fig. 4; and hence, $\mathcal{S} = S_1 = \{v_4v_6, v_5v_6, v_6v_7\}$. Then, we process Q_2 residing in edge v_3v_4, i.e., $S_2 = \{v_3v_4\}$. Since only one of the adjacent edges of S_2 is in $\mathcal{S}$, i.e., v_4v_6, we add v_4v_6 to S_2 (Fig. 6a). Selecting v_4v_6 results in two adjacent edges of S_2 in $\mathcal{S}$, so we compute the quality of these two edges, i.e., $\mathsf{Quality}(S_2 \cup v_5v_6) = \min(7/6, 1) \times \min(6/7, 7/6) = 1 \times 6/7 = 0.857$ and $\mathsf{Quality}(S_2 \cup v_6v_7) = \min(4/6, 1) \times \min(4/7, 7/4) = 4/6 \times 4/7 = 0.381$. Thus, we add the edge with the highest quality, i.e., v_5v_6, to S_2 (Fig. 6b). Since v_6v_7 is the only adjacent edge of S_2 in $\mathcal{S}$, we add v_6v_7 to S_2 (Fig. 6c). After that, $S_2 = \{v_3v_4, v_4v_6, v_5v_6, v_6v_7\}$ satisfies the user privacy requirements, i.e., $\text{NumUser}(S_2) = 8$ and $\text{Length}(S_2) = 7$. The anonymization process of Q_2 results in adding only one *open vertex* v_3 (enclosed in a rectangle) and one edge v_3v_4 to $\mathcal{S}$. Similarly, we anonymize the location information of Q_3 residing in edge v_7v_8. The cloaked segment set of Q_3 is $S_3 = \{v_3v_4, v_4v_6, v_6v_7, v_7v_8\}$ (Fig. 6d) that results in only one *open vertex* v_8 and edge v_7v_8 to be added to $\mathcal{S}$. Finally, the cloaked segment sets of these three queries contain five distinct *open vertices* (enclosed in rectangles) and five distinct edges (represented as black lines), as depicted in Fig. 5b.

7 Experimental results

In this section, we experimentally evaluate the performance of the two versions of our proposed query-aware location anonymization algorithm, i.e., *pure greedy* (denoted as **PG**) and *randomized greedy* (denoted as **RG**). We also evaluate the performance of the proposed *shared execution* paradigm incorporated into the *pure greedy* (denoted as **SPG**) and *randomized greedy* (denoted as **SRG**) approaches.

Baseline algorithms We compare our approaches with two previous works [22, 31]. Since the work of [22] uses the Casper's location anonymization algorithm [24] (denoted as **Casper**) that is designed for the Euclidean space to blur a user location into a cloaked area. Then, a set of road segments intersecting the cloaked area forms a cloaked segment set. The other previous work [31] (denoted as **XStar**) is the state-of-the-art technique for location anonymization in road networks. To have a fair analysis of its resilience to the privacy attacks ,i.e., the replay attack and the center-of-cloaked-area attack, the same random factor of our randomized greedy approach is also applied to this previous work.

Experiment settings The simulated experiments are implemented in C++. In all experiments, we generate a set of moving objects on the road map of Hennepin County, Minnesota, USA (Fig. 7). The input road map is extracted from the Tiger/Line files that are publicly available [30]. The total area of the Hennepin County is 1,571 km^2. The road map has 57,020 edges and 42,135 vertices in which the average length of the edges (l) is about 0.1 km and the average degree of connectivity of the vertices (d) is 2.7. Mobile users are initially distributed among the vertices, and then move along the roads at speeds between 50 and 70 miles per hour. The experiments were run on an Ubuntu Linux system with an Intel Core 2 Quad processor at 2.83GHz and

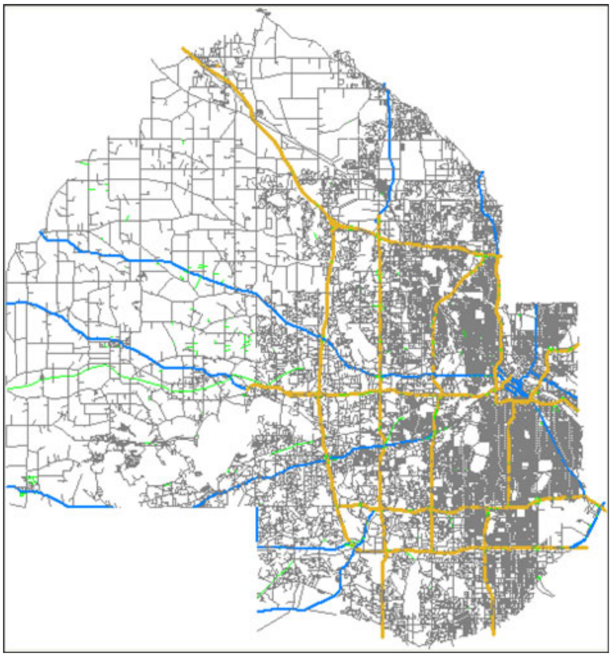


Fig. 7 The road map of Hennepin County, Minnesota, USA

4GB RAM. At the database server, we employ an *incremental network expansion* algorithm [27] and a depth-first search algorithm to process k-nearest-neighbor (k-NN) and range queries in the road network, respectively.

Parameter settings Unless mentioned otherwise, the experiments consider 100,000 mobile users and 2,000 objects in the underlying road network in which 1,000 users issue queries. The default user privacy requirements, anonymity level $\mathcal{K}$ and minimum length $\mathcal{L}$, are $\mathcal{K} = 200$ and $\mathcal{L} = 1$ km. The random factor Rand_F for our randomized greedy approach is set to 0.2. The query parameters of k-NN and range queries are k=10 and $r = 1$ km, respectively. The default size of an object is 128 bytes. The transmission bandwidth of the high-bandwidth wired communication link between the location anonymizer and the service provider is 1,000 Mbps, and the transmission bandwidth of the wireless communication link between the location anonymizer and the user is 10 Mbps. Table 2 gives a summary of the parameter settings.

Privacy attack models In this section, we evaluate the privacy resilience of a location anonymization algorithm against two privacy attack models, namely, the replay attack [31] and the center-of-cloaked-area attack [7, 35]. (1) *Replay privacy attack.* In this attack, we assume the worst scenario where an adversary knows the location anonymization algorithm, the user locations (but not their user identities) and the statistics used by the objective cost function. Given a cloaked segment set S with a query, the adversary wants to know which segment in S contains the query issuer. To do that, the adversary re-runs the location anonymization for each segment $s \in S$,

Table 2 Parameter settings

Parameter	Default value	Evaluation range
Number of users	100,000	100,000–500,000
Number of objects	2,000	2,000–10,000
Number of queries	1,000	2,000–10,000
Anonymity levels ($\mathcal{K}$)	200	100–500
Minimum total segment length ($\mathcal{L}$)	1 km	5–25 km
Requested number of objects k	10	1–20
Requested range distance r	1 km	5–30 km
Object size	128 bytes	32–512 bytes

and then calculates the linkability between the cloaked segment set of s, S', and S using the following equation:

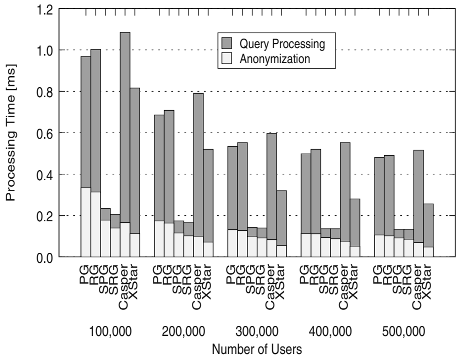
$$\text{linkability}(s, S) = |S' \cap S|/|S| \tag{6}$$

After calculating the linkability for each segment in S, the adversary infers that the segment s^* with the highest linkability contains the query issuer. The replay privacy attack succeeds if s^* contains the actual query issuer; otherwise, it fails. (2) *Center-of-cloaked-area privacy attack.* In this attack, we assume that an adversary knows the user locations (but not their user identities). Given a cloaked segment set S with a query, the adversary wants to infer which user issues the query. The adversary first computes a minimum bounding rectangle (MBR) of the segments in S. Then, the adversary determines the distance between each user located on the segment in S and the center of the MBR, and infers that the closest user u^* to the center of the MBR is the query issuer. The center-of-cloaked-area privacy attack succeeds if u^* is the actual query issuer; otherwise, it fails.

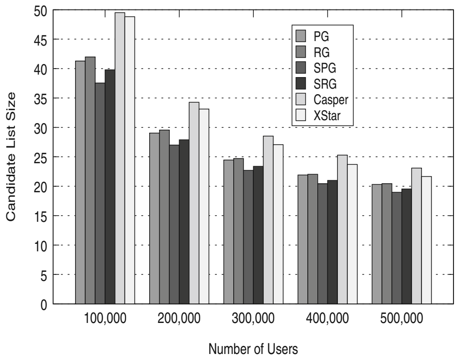
Performance metrics We evaluate our algorithms with respect to five performance measures, (1) the processing time, (2) the candidate list size, i.e., the query quality, (3) the success rate of the replay privacy attack, (4) the success rate of the center-of-cloaked-area privacy attack, and (5) the end-to-end performance. The processing time is the sum of the average time consumed in the anonymization process (i.e., anonymization time) and the average query processing time at the database server. The candidate list size measures the average number of objects returned to the users per query. The success rate of the two privacy attacks indicates the privacy resilience of our algorithms. The end-to-end performance measures the average overall query response time per query, which includes the anonymization time at the location anonymizer, the query processing time at the service provider, the transmission time of sending candidate lists from the service provider to the location anonymizer through high-bandwidth wired channels, and the transmission time of sending candidate lists from the location anonymizer to the user through wireless communication channels.

7.1 Number of users

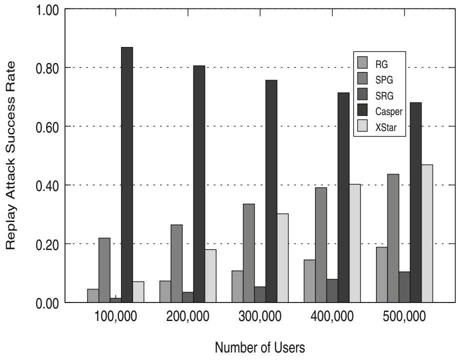
Figures 8 and 9 depict the scalability of our approaches with respect to varying the number of users from 100,000 to 500,000 for k-NN and range queries, respectively.



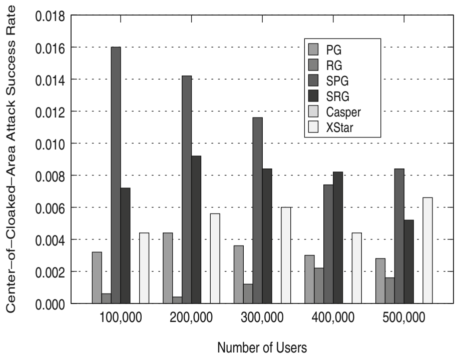
(a) Processing Time



(b) Candidate List Size

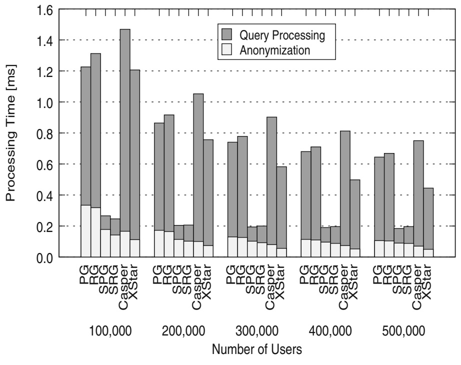


(c) Replay Attack

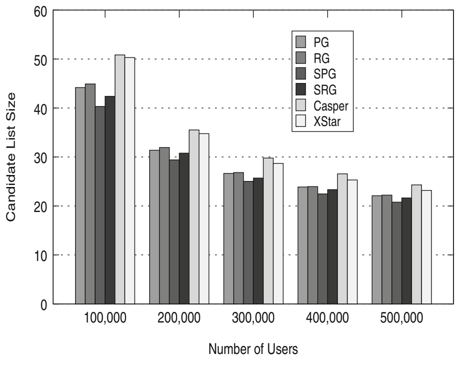


(d) Center-of-Cloaked-Area Attack

Fig. 8 Number of users (*k*-nearest-neighbor queries)



(a) Processing Time



(b) Candidate List Size

Fig. 9 Number of users (range queries)

Since the location anonymization is independent of query types, the resilience of the location anonymization algorithm to any privacy attack is the same for any query types, and thus, we only show the success rates of the replay and center-of-cloaked-area attacks for k-NN queries.

Figure 8a gives that our approaches with the shared execution paradigm, SPG and SRG, outperform the approaches without the shared execution paradigm, PG, RG, and the baseline algorithm, Casper, and even XStar having the concept of shared execution, in terms of query processing time. The main reason is that our shared execution paradigm dynamically finds a shared set of cloaked segments for a group of queries without any limitation on their locations. However, XStar only shares a cloaked segment set for the queries on the same segment. Casper gives the worst performance in terms of query processing time because it does not consider the query execution cost in road network environments. In general, the location anonymization time and the query processing time of all the approaches improve as there are more users in the system. This is due to the fact that smaller cloaked areas are generated to satisfy the same required $\mathcal{K}$-anonymity level when the number of users increases.

Figure 8b also gives that our approaches perform better than Casper and XStar in terms of candidate list size. Since Casper does not take into account the underlying road network environment, the total segment length of the cloaked segment sets generated by Casper is longer than other approaches, which are designed for road network environments. As a cloaked segment set with a longer segment length leads to larger candidate list size, Casper gives the worst query quality. Since our approaches do not rely on any specific basic unit structure for location anonymization, while XStar needs to group neighboring segments to form stars as the basic unit structure for its anonymization process, the total segment length of the cloaked segment sets of our approaches is shorter than XStar, and thus, our approaches give better query quality than XStar.

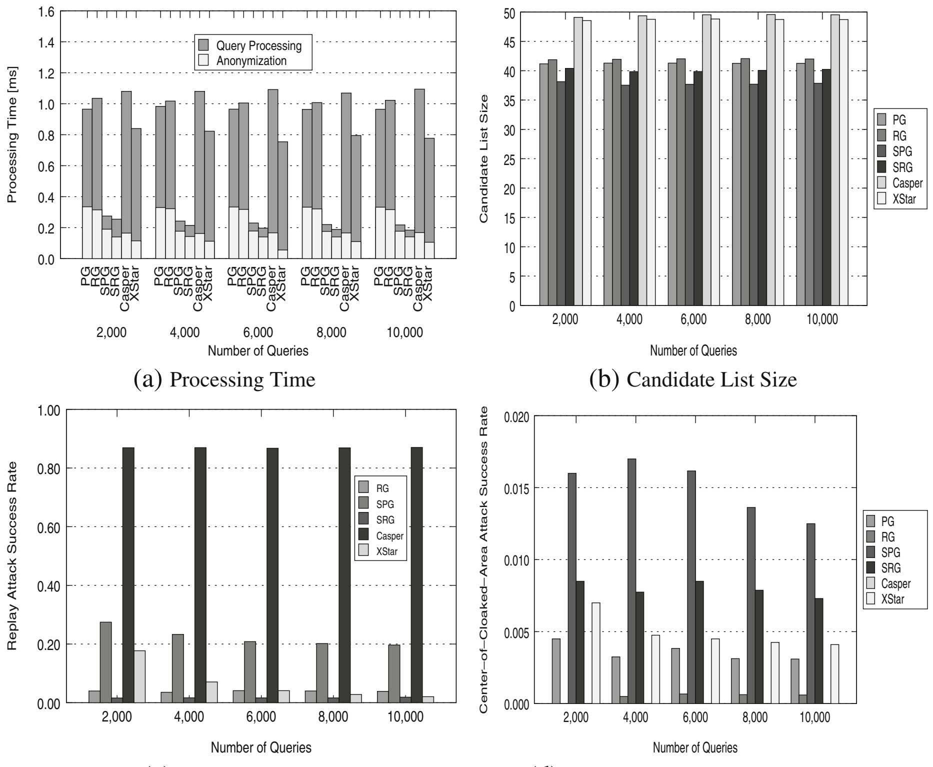
Figure 8c and d depict the resilience of all the approaches to the replay and center-of-cloaked-area privacy attacks, respectively. Since the success rate of the replay attack of the pure greedy approach (PG) is always one, we do not show its result for the replay attack. The experimental evidence shows that injecting randomness to a location anonymization process effectively increases its resilience to the replay attack. As Casper uses a data structure to do location anonymization, it is more vulnerable to the replay attack. It is interesting to see that our shared execution paradigm also improves the resilience to the replay attack. The reason is that the generation of a shared set of cloaked segments depends on a set of queries; instead of only one query in the non-shared execution paradigm, so an adversary is much more difficult to use the replay attack for a shared set of cloaked segments. The result shows that our SRG provides much better privacy guarantee than XStar for the replay attack. In terms of the center-of-cloaked-area privacy attack, all the approaches have very strong resilience to the attack, i.e., its success rate is less than 0.016 for all the approaches.

Figure 9 shows similar results for range queries, where our approaches with the shared execution paradigm, SPG and SRG, outperform our non-shared execution approaches, PG and RG, and the baseline algorithms, Casper and XStar, in terms of processing time (Fig. 9a). Since our approaches generate cloaked segment sets with shorter total segment lengths, they provide better query quality, i.e., a smaller candidate list size, than the baseline algorithms, Casper and XStar (Fig. 9b).

7.2 Number of queries

Figures 10 and 11 give the performance of all the approaches with respect to increasing the number of objects from 2,000 to 10,000 for *k*-NN and range queries, respectively. The processing time of the non-shared execution approaches, PG, RG, and Casper, is only slightly affected by the increase of the number of queries in the system for both *k*-NN and range queries, as depicted in Figs. 10a and 11a, respectively. On the other hand, the processing time of the shared execution approaches, SPG, SRG, and XStar, improves when there are more querying users. The reason is that when there are more querying users, it has a higher chance that a cloaked segment set shared by more queries, and thus, more queries share the query execution cost of the cloaked segment set. Our approaches with the shared execution paradigm also generate cloaked segment sets with shorter total segment lengths, so they provide better query quality than other approaches (Figs. 10b and 11b).

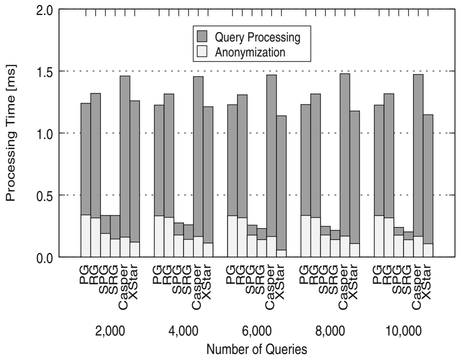
Figure 10c shows that the resilience to the replay privacy attack of the shared execution approaches, SPG, SRG, and XStar gets stronger, as the number of queries increases. The reason is that the generation of the cloaked segment set not only depends on the cost function, but it also depends on the properties of other queries,



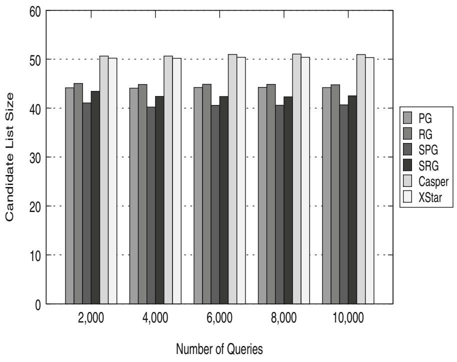
(a) Processing Time (b) Candidate List Size

(c) Replay Attack (d) Center-of-Cloaked-Area Attack

Fig. 10 Number of queries (*k*-nearest-neighbor queries)



(a) Processing Time



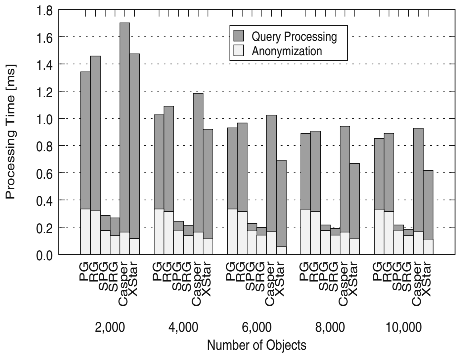
(b) Candidate List Size

Fig. 11 Number of queries (range queries)

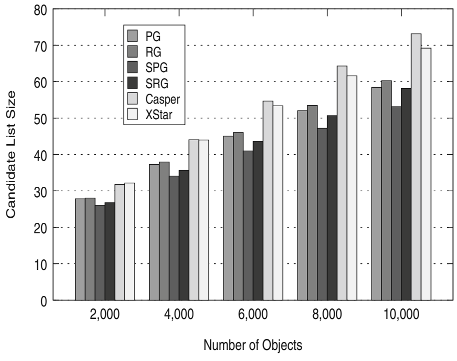
e.g., their distribution, their privacy requirements, and the order of anonymizing them; and thus, when more segments in a cloaked segment set are shared by other queries, it is more difficult for an adversary to infer which segment in the cloaked segment set contains the actual query issuer. The result provides evidence that the shared execution paradigm can reduce the query processing overhead and improve the resilience to the replay privacy attack.

7.3 Number of objects

Figures 12 and 13 give the performance of all the approaches with respect to increasing the number of objects in the system from 2,000 to 10,000 for *k*-NN and range queries, respectively. Since varying the number of objects at the service provider does not affect the location anonymization, the execution overhead and resilience to the privacy attacks of the location anonymization algorithms are not

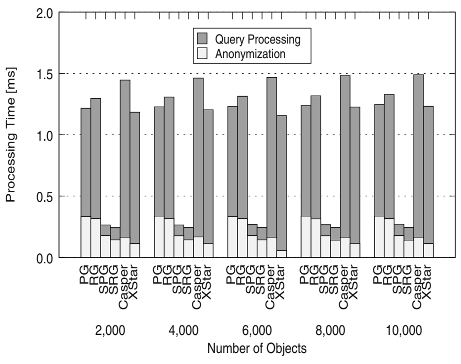


(a) Processing Time

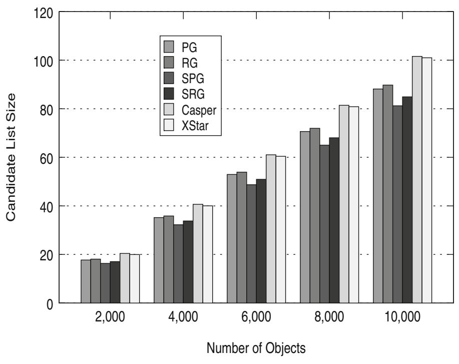


(b) Candidate List Size

Fig. 12 Number of objects (*k*-nearest-neighbor queries)



(a) Processing Time



(b) Candidate List Size

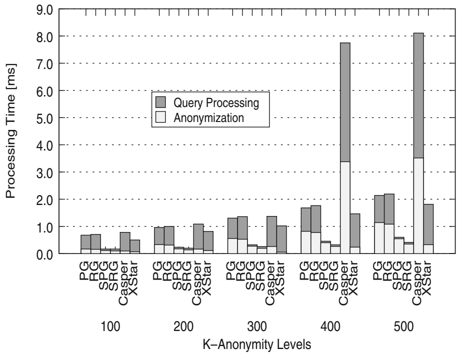
Fig. 13 Number of objects (range queries)

affected. It is interesting to see that the processing time of k-NN queries decreases as there are more objects in the system, as depicted in Fig. 12a. The reason is that the query processor can find the requested number of nearest objects for each open vertex in a cloaked segment set by searching a smaller number of road segments when the number of objects increases. On the other hand, the processing time of range queries slightly increases when there are more objects in the system (Fig. 13a). This is because the query processor has to search the requested distance from each open vertex in a cloaked segment set regardless of the object distribution. When there are more objects in the system, the query processor has to retrieve more objects for a segment, and thus, the query processing time slightly increases. Since a candidate answer list must contains the objects located within a cloaked segment set for both k-NN and range queries, when the number of objects increases, the candidate answer list contains more objects (Figs. 12b and 13b).

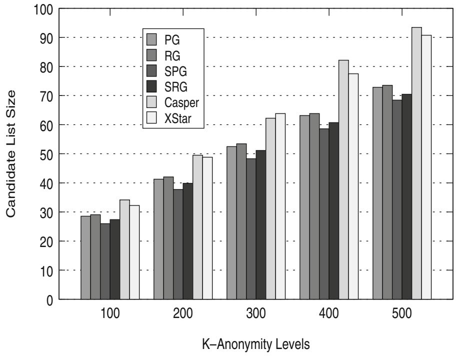
7.4 $\mathcal{K}$-anonymity privacy requirements

Figures 14 and 15 depict the performance of all the approaches for k-NN and range queries, respectively, as the user required $\mathcal{K}$-anonymity level increases from 100 to 500. It is expected that when the location anonymization algorithm has to generate larger cloaked segment sets to satisfy the stricter privacy requirements, the location anonymization overhead of all the approaches increases (Figs. 14a and 15a). Since our approaches with the shared execution paradigm, SPG and SRG, effectively share cloaked segments among queries, they perform better than the baseline algorithms, Casper and XStar, as the user requires stricter $\mathcal{K}$-anonymity levels. Such larger cloaked segment sets lead to larger candidate lists returned to the user, so the query quality gets worse when $\mathcal{K}$ increases (Figs. 14b and 15b).

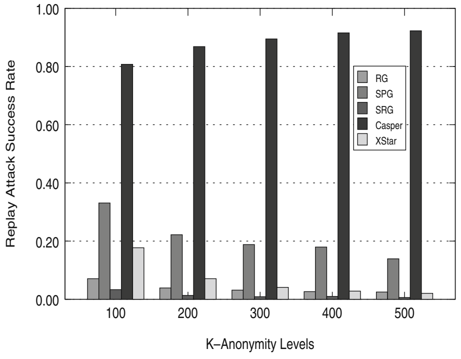
As the user requires stricter $\mathcal{K}$-anonymity levels for their queries, it is expected that the resilience to the replay privacy attack increases (Figs. 14c and 15c). The result shows that our randomized greedy approaches, RG and SRG, have better resilience to the replay attack than XStar. Similar to other experiments, Figs. 14d



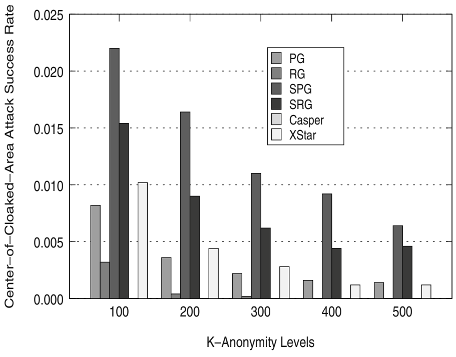
(a) Processing Time



(b) Candidate List Size

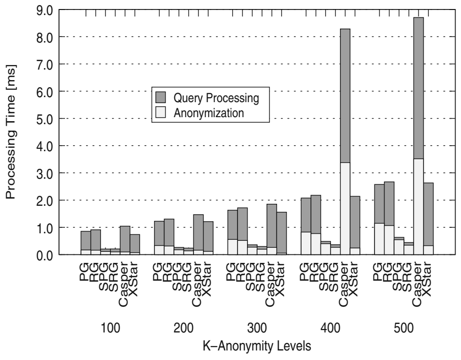


(c) Replay Attack

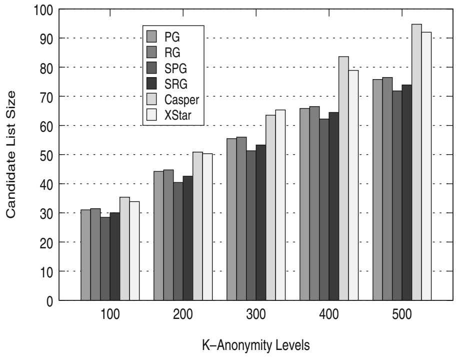


(d) Center-of-Cloaked-Area Attack

Fig. 14 $\mathcal{K}$-anonymity levels (*k*-nearest-neighbor queries)

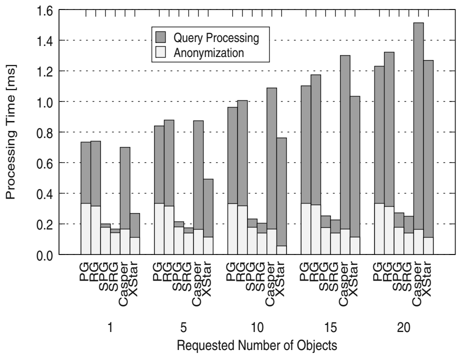


(a) Processing Time

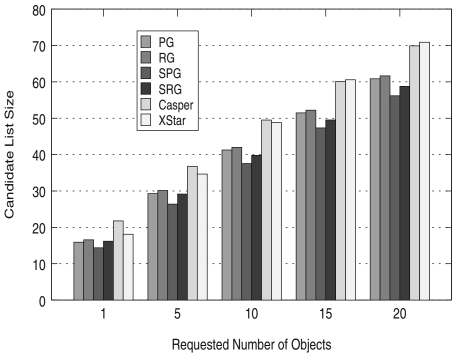


(b) Candidate List Size

Fig. 15 $\mathcal{K}$-anonymity levels (range queries)



(a) Processing Time



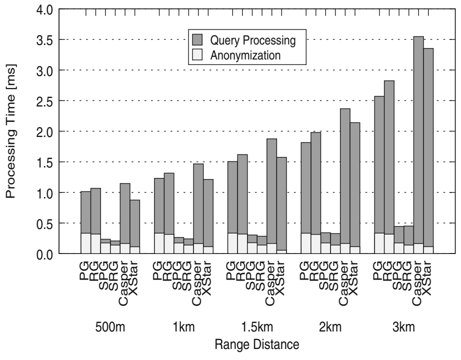
(b) Candidate List Size

Fig. 16 Requested number of objects (*k*-nearest-neighbor queries)

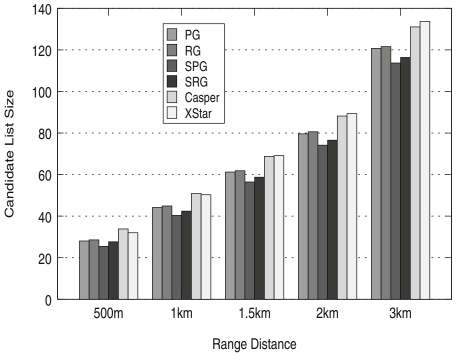
and 15d show that the success rate of the center-of-cloaked-area privacy attack is very small, i.e., 0.025, for all the approaches, even though the approach not designed for road network environments, i.e., Casper.

7.5 Query parameters

In this section, we evaluate the performance of all the approaches with respect to varying the requested number of objects for *k*-NN queries from 1 to 20 (Fig. 16) and the range distance *r* for range queries from 0.5 km to 3 km (Fig. 17). Since varying *k* and *r* for *k*-NN and range queries, respectively, does not affect the cloaked segment sets generated by the location anonymization process, the location anonymization time and the resilience to the privacy attacks of the location anonymization process are not affected. It is expected that when *k* and *r* get larger, the query execution overhead increases (Figs. 16a and 17a) and larger candidate lists are returned to

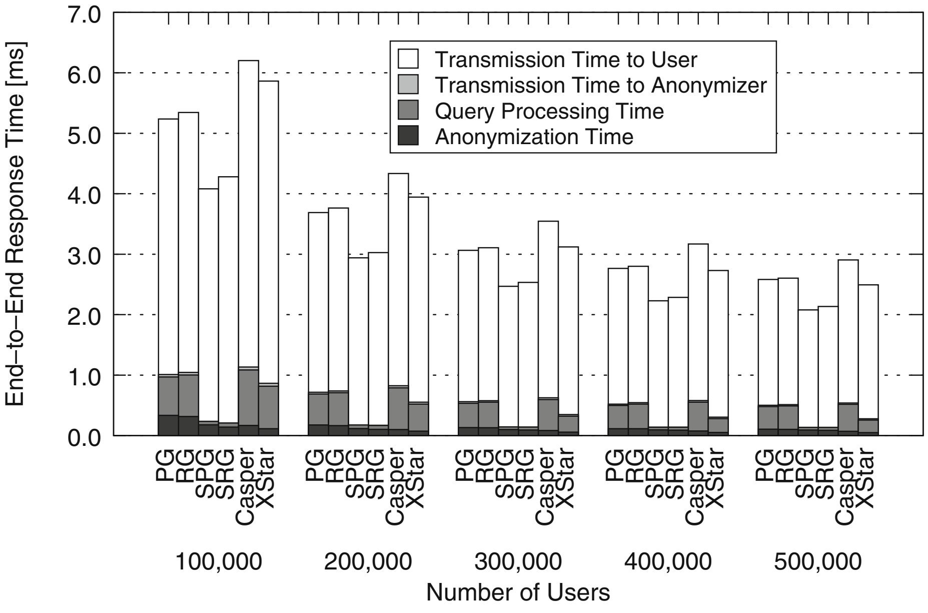


(a) Processing Time

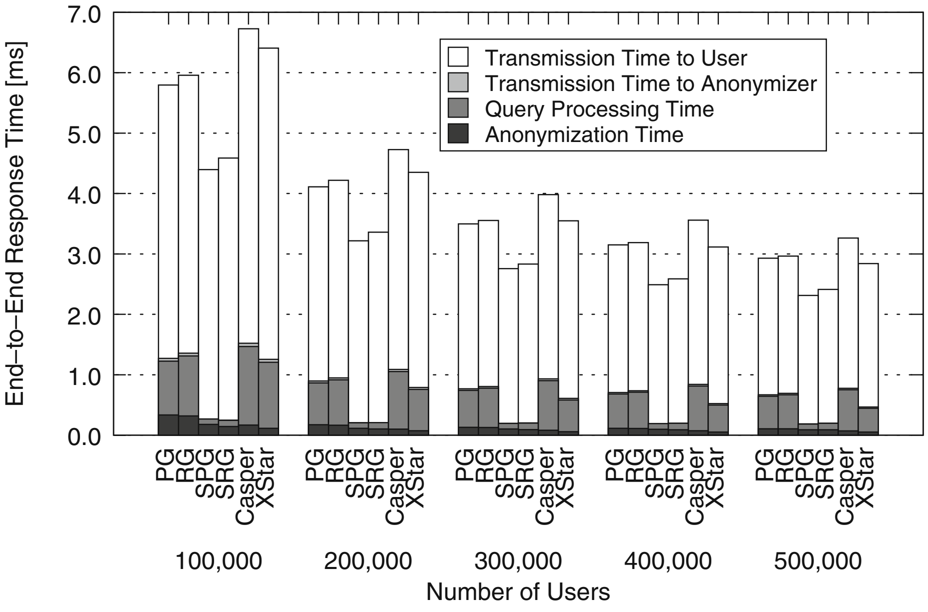


(b) Candidate List Size

Fig. 17 Range distance (range queries)



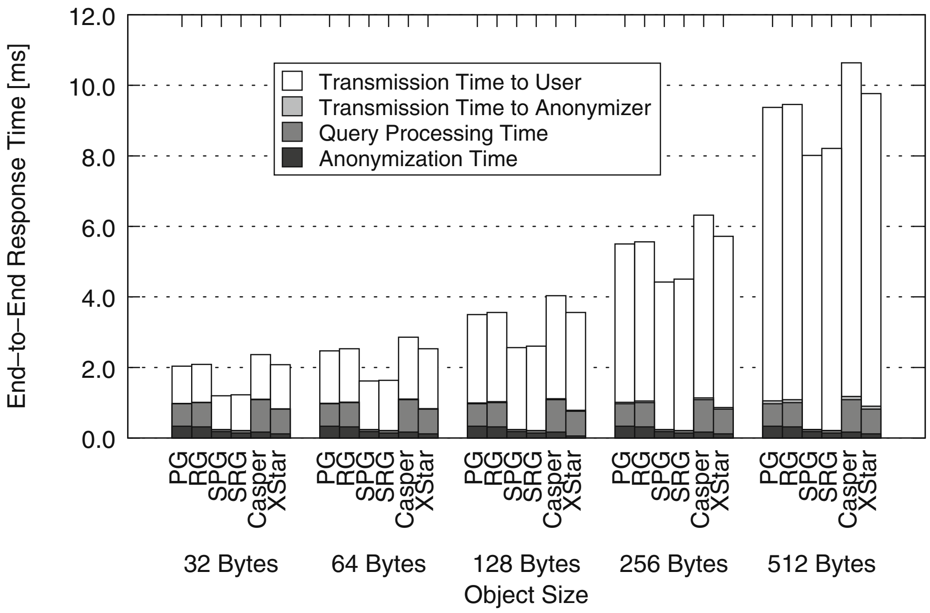
(a) *k*-nearest-neighbor queries



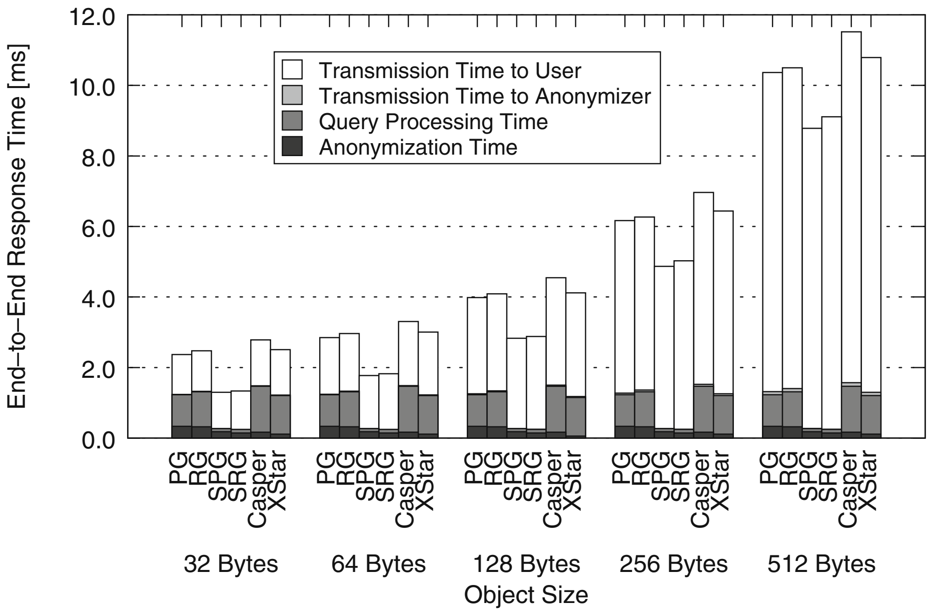
(b) Range queries

Fig. 18 End-to-end performance (number of users)

users (Figs. 16b and 17b). The results also show that our approaches with the shared execution paradigm, SPG and SRG, perform better than the baseline algorithms, Casper and XStar, when *k* and *r* get larger.



(a) *k*-nearest-neighbor queries



(b) Range queries

Fig. 19 End-to-end performance (object size)

7.6 End-to-end performance

This section evaluates the end-to-end query response time of all the approaches for k-NN and range queries with respect to increasing the number of users from 100,000 to 500,000 and the object size from 32 to 512 bytes, as the results are depicted in Figs. 18 and 19, respectively. The end-to-end query response time includes the location anonymization time at the location anonymizer (represented by black bars), the query processing time at the database server (represented by dark grey bars), the transmission of sending candidate lists from the database server to the location anonymizer (represented by light grey bars), and the transmission of sending candidate lists from the location anonymizer to the user (represented by white bars).

Figure 18 gives the same trend for both k-NN and range queries, where the transmission overhead for the user dominates the end-to-end query response time. This is because we assume the user communicates with the location anonymizer through wireless communication channels, e.g., IEEE 802.11. Since our approaches with the shared execution paradigm, SPG and SRG, effectively share cloaked segments among queries, the number of objects transmitted from the database server to the location anonymizer is much smaller than other approaches. However, after the location anonymizer gets the objects returned from a set of shared queries, it has to determine the candidate list for each individual user from the returned objects and send the candidate list to each user separately, and thus, the shared execution paradigm cannot reduce the transmission time of sending candidate lists to users. Fortunately, our approaches generate smaller cloaked segment sets than the baseline algorithms, Casper and XStar, so they incur lower communication overhead than the baseline algorithms.

Figure 19 depicts the expected results that the end-to-end query response time gets longer when the object size increases. As the object size gets larger, the transmission time between the location anonymizer and the database server and between the location anonymizer and the user increase. Since our approaches with the shared execution paradigm, SPG and SRG, generate cloaked segment sets with shorter total segment lengths than other approaches, they give the best performance in terms of the overall query response time.

8 Conclusion

This paper proposed a *query-aware* location anonymization algorithm for road network environments. Our algorithm aims to blur a user location into a set of connected road segments S such that: (a) there exist at least $\mathcal{K}$ users in S to satisfy the $\mathcal{K}$-anonymity privacy requirement, and the total segment length of the road segments in S is at least $\mathcal{L}$ to fulfill the minimum length $\mathcal{L}$ privacy requirement, (b) the query execution cost of the requested query over S is minimized, and (c) the query quality of S is maximized (i.e., the candidate list size returned to the user is minimized). Based on a developed objective cost function that takes into account the user specified privacy requirements, the query execution cost, and the query quality, we proposed two greedy-based approaches, *pure greedy* and *randomized greedy* approaches, for location anonymization in road networks. To accommodate intervals with a high workload, we also proposed a *shared execution paradigm* to improve the scalability of our location anonymization process and the query processing of a database server to support larger numbers of quires in a short time period. Extensive experimental results show that our location anonymization algorithms are efficient and scalable, while preserving the user location privacy and enabling high quality services through minimizing the developed objective cost function. Injecting randomness into our location anonymization algorithms effectively avoids the replay privacy attack. The results also depict that our algorithms with the shared execution paradigm outperform the state-of-the-art location anonymization technique designed for road network environments, in terms of both query response time and query quality.

References

1. Bamba B, Liu L, Pesti P, Wang T (2008) Supporting anonymous location queries in mobile environments with privacygrid. In: Proceedings of the international world wide web conference, WWW
2. Cheng R, Zhang Y, Bertino E, Prabhakar S (2006) Preserving user location privacy in mobile data management infrastructures. In: Proceedings of international privacy enhancing technologies symposium, PET
3. Chow CY, Mokbel MF (2007) Enabling private continuous queries for revealed user locations. In: Proceedings of the international symposium on spatial and temporal databases, SSTD
4. Chow CY, Mokbel MF, Aref WG (2009) Casper*: query processing for location services without compromising privacy. ACM Trans Database Syst 34(4)
5. Chow CY, Mokbel MF, He T (2010) A privacy-preserving location monitoring system for wireless sensor networks. IEEE Trans Mob Comput. doi:10.1109/TMC.2010.145
6. Chow CY, Mokbel MF, Liu X (2006) A peer-to-peer spatial cloaking algorithm for anonymous location-based services. In: Proceedings of the ACM symposium on advances in geographic information systems, ACM GIS
7. Chow CY, Mokbel MF, Liu X (2010) Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. GeoInformatica. doi:10.1007/s10707-009-0099-y
8. Duckham M, Kulik L (2005) A formal model of obfuscation and negotiation for location privacy. In: Proceedings of international conference on pervasive computing
9. Gedik B, Liu L (2008) Protecting location privacy with personalized k-anonymity: Architecture and algorithms. IEEE Trans Mob Comput 7(1):1–18
10. Ghinita G, Kalnis P, Khoshgozaran A, Shahabi C, Tan KL (2008) Private queries in location based services: anonymizers are not necessary. In: Proceedings of the ACM international conference on management of data, SIGMOD
11. Ghinita G, Kalnis P, Skiadopoulos S (2007) MobiHide: a mobile peer-to-peer system for anonymous location-based queries. In: Proceedings of the international symposium on spatial and temporal databases, SSTD
12. Ghinita G, Kalnis P, Skiadopoulos S (2007) PRIVÉ: anonymous location-based queries in distributed mobile systems. In: Proceedings of the international world wide web conference, WWW
13. Gruteser M, Grunwald D (2003) Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of international conference on mobile systems, applications, and services, MobiSys
14. Gruteser M, Schelle G, Jain A, Han R, Grunwald D (2003) Privacy-aware location sensor networks. In: USENIX workshop on hot topics in operating systems, HotOS
15. Hong JI, Landay JA (2004) An architecture for privacy-sensitive ubiquitous computing. In: Proceedings of international conference on mobile systems, applications, and services, MobiSys
16. Hu H, Lee DL, Xu J (2006) Fast nearest neighbor search on road networks. In: Proceedings of the international conference on extending database technology, EDBT
17. Jensen CS, Kolář J, Pedersen TB, Timko I (2003) Nearest neighbor queries in road networks. In: Proceedings of the ACM symposium on advances in geographic information systems, ACM GIS
18. Kalnis P, Ghinita G, Mouratidis K, Papadias D (2007) Preventing location-based identity inference in anonymous spatial queries. IEEE Trans Knowl Data Eng 19(12):1719–1733

19. Khoshgozaran A, Shahabi C (2007) Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In: Proceedings of the international symposium on spatial and temporal databases, SSTD
20. Kido H, Yanagisawa Y, Satoh T (2005) An anonymous communication technique using dummies for location-based services. In: Proceedings of IEEE international conference on pervasive services, ICPS
21. Kolahdouzan M, Shahabi C (2004) Voronoi-based K nearest neighbor search for spatial network databases. In: Proceedings of the international conference on very large data BAses, VLDB
22. Ku WS, Zimmermann R, Peng WC, Shroff S (2007) Privacy protected query processing on spatial networks. In: Proceedings of international workshop on privacy data management, PDM
23. Li PY, Peng WC, Wang TW, Ku WS, Xu J, Hamilton JA Jr (2008) A cloaking algorithm based on spatial networks for location privacy. In: Proceedings of IEEE international conference on sensor networks, ubiquitous, and trustworthy computing, SUTC
24. Mokbel MF, Chow CY, Aref WG (2006) The new Casper: query procesing for location services without compromising privacy. In: Proceedings of the international conference on very large data bases, VLDB
25. Mouratidis K, Yiu ML (2010) Anonymous query processing in road networks. IEEE Trans Knowl Data Eng 22(1):2–15 (2010)
26. Mouratidis K, Yiu ML, Papadias D, Mamoulis N (2006) Continuous nearest neighbor monitoring in road networks. In: Proceedings of the international conference on very large data bases, VLDB
27. Papadias D, Zhang J, Mamoulis N, Tao Y (2003) Query processing in spatial network databases. In: Proceedings of the international conference on very large data bases, VLDB
28. Pfitzmann A, Kohntopp M (2000) Anonymity, unobservability, and pseudonymity—a proposal for terminology. In: Proceedings of international privacy enhancing technologies symposium, PET
29. Sweeney L (2002) *k*-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl-based Syst 10(5):557–570
30. US Census Bureau (2009) Topologically integrated geographic encoding and referencing system (TIGER). http://www.census.gov/geo/www/tiger/
31. Wang T, Liu L (2009) Privacy-aware mobile services over road networks. In: Proceedings of the international conference on very large data bases, VLDB
32. Xu T, Cai Y (2007) Location anonymity in continuous location-based services. In: Proceedings of the ACM symposium on advances in geographic information systems, ACM GIS
33. Xu T, Cai Y (2008) Exploring historical location data for anonymity preservation in location-based services. In: Proceedings of the IEEE international conference on computer communications, INFOCOM
34. Yiu ML, Jensen C, Huang X, Lu H (2008) SpaceTwist: managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In: Proceedings of the international conference on data engineering, ICDE
35. Zhang C, Huang Y (2009) Cloaking locations for anonymous location based services: a hybrid approach. GeoInformatica 13:159–182

Chi-Yin Chow is currently an assistant professor in the Department of Computer Science, City University of Hong Kong. He received his Ph.D. degree from the University of Minnesota-Twin

Cities in 2010. His research interests are in databases, spatial and spatio-temporal databases, wireless sensor networks, mobile computing, location-based services, data privacy, and recommender systems. He was an intern at the IBM Thomas J. Watson Research Center during the summer of 2008. He is a member of ACM and IEEE.

Mohamed F. Mokbel (Ph.D., Purdue University, 2005, MS, B.Sc., Alexandria University, 1999, 1996) is an assistant professor in the Department of Computer Science and Engineering, University of Minnesota. His main research interests focus on advancing the state of the art in the design and implementation of database engines to cope with the requirements of emerging applications (e.g., location-aware applications and sensor networks). Mohamed was the co-chair of the first and second workshops on privacy-aware location-based mobile services, PALMS, 2007 (Mannheim, Germany) and 2008 (Beijing, China). He is also the PC co-chair for the ACM SIGSPATIAL GIS 2008 and 2009 conferences. Mohamed has spent the summers of 2006 and 2008 as a visiting researcher at Hong Kong Polytechnic University and Microsoft Research, respectively. He is a member of ACM and IEEE.

Jie Bao (M.Sc., 2009, Auburn University; B.A., 2007, Zhejiang University) is a Ph.D. student in the Department of Computer Science and Engineering, University of Minnesota. His main research interests are in spatial and spatio-temporal query processing, database systems, and location privacy.

Xuan Liu received her PhD in Computer Science and Engineering from University of Minnesota, Minneapolis in 2000. Since joining IBM as a Research Staff Member in 2000, Dr. Liu has served as a technical lead for various projects ranging from context aware privacy information control, knowledge based conversation management, and solutions cloud for smarter cities. Her work has contributed to IBM WebSphere product family, IBM web services toolkit, and IBM customer solutions. Dr. Liu's research interests include spatial databases and data mining, geographic information systems (GIS), cloud computing, and mobile computing. She has published many research papers in peer-reviewed journals, conferences, and workshops, and has hold four patents. She is a member of IEEE and IEEE Computer Society.