# Utility-Aware and Privacy-Preserving Mobile Query Services

Emre Yigitoglu, M. Emre Gursoy, *Member, IEEE,* and Ling Liu, *Fellow, IEEE*

**Abstract**—Location-based queries enable fundamental services for mobile users. While the benefits of location-based services (LBS) are numerous, exposure of mobile users' locations to untrusted LBS providers may lead to privacy concerns. This paper proposes StarCloak, a utility-aware and attack-resilient location anonymization service for privacy-preserving LBS usage. StarCloak combines several desirable properties. First, unlike conventional approaches which are indifferent to underlying road network structure, StarCloak uses the concept of *star*s and proposes *cloaking graphs* for effective location cloaking on road networks. Second, StarCloak supports user-specified $k$-user anonymity and $l$-segment indistinguishability, for enabling personalized privacy protection and for serving users with varying privacy preferences. Third, StarCloak achieves strong attack-resilience against replay and query injection attacks through randomized star selection and pruning. Finally, to enable efficient query processing with high throughput and low bandwidth overhead, StarCloak makes cost-aware star selection decisions by considering query evaluation and network communication costs. We evaluate StarCloak on two datasets using real-world road networks, under various privacy and utility constraints. Results show that StarCloak achieves improved query success rate and throughput, reduced anonymization time and network usage, and higher attack-resilience in comparison to XStar, its most relevant competitor.

**Index Terms**—Privacy, location privacy, location-based services, mobile query services, Internet of Things

## 1 Introduction

The growth of location-based services (LBSs) is fueled by ubiquitous wireless connectivity, universal presence of smart mobile devices, and increased investments from industry and government in the Internet of Things and connected vehicles. As more and more users and vehicles are connected continuously and automatically, they are embraced by life-enriching location-based services, including but not limited to emergency assistance, real-time traffic alerts, and location recommendations.

Despite growing research in providing services for mobile users traveling on road networks [1], [2], [3], [4], users' location privacy poses an important concern. Unauthorized location exposure may cause vulnerability for abuse such as unwanted advertisement, spam, stalking, and location spoofing. In addition, when private location data of a mobile user is linked to sensitive public locations such as health clinics, cancer treatment centers, nightclubs or religious organizations, such unauthorized linkage may cause ethical, professional, and social risks both to individuals and the society at large. As a result, it becomes imperative to protect road network travelers' location privacy as they interact with LBS providers via service queries.

In this paper, we design and develop StarCloak, a location anonymization service for protecting the location privacy of mobile users. StarCloak forms a middle layer between mobile users and untrusted LBS providers, such that for any user who wants to issue a query to the LBS provider, the user's query is first intercepted by StarCloak, their true location is cloaked according to user-desired privacy and utility specifications, and only the cloaked location is made available to the LBS provider. The design of StarCloak incorporates several desirable aspects. First, StarCloak takes into account the road network structure for effective privacy protection and efficient query processing with anonymized locations. Second, StarCloak supports user-defined, personalized privacy and utility goals such as $k$-user anonymity and $l$-segment indistinguishability to meet different users' different privacy preferences. Third, StarCloak achieves high resilience against attacks that are relevant in the mobile location privacy domain, such as replay attacks and query injection attacks. Finally, StarCloak aims to minimize the communication and IO costs of privacy protection, i.e., it incurs a small time overhead for anonymizing users' locations, and anonymized cloaked locations are compact enough to be sent through a wireless network with little bandwidth overhead.

Efficient and effective implementation of the anonymization services provided by StarCloak requires the development of several optimized data structures and algorithms. To that end, StarCloak uses optimized data structures such as *star*, *star graph*, and *cloaking graph*. StarCloak maintains its internal data structures as new queries are processed, and generates candidate star-sets as cloaked regions when it finds that certain users' queries can be successfully served. StarCloak's candidate star-set pruner, which is implemented with high parallelism, enables pruning of candidate star-sets to generate low-cost cloaked regions with improved attack-resilience, thanks to randomized pruning. In addition, we also propose two variants of StarCloak, namely spatially bounded StarCloak and hybrid StarCloak, for generating more compact cloaked regions with negligible sacrifice in query success rate and throughput.

We evaluate StarCloak and its variants through extensive experiments on real-world Georgia and California road networks of different scales, under varying privacy and utility constraints. We also compare StarCloak with two baseline anonymization

- *Emre Yigitoglu and Ling Liu are with the School of Computer Science, Georgia Institute of Technology, Atlanta, GA. e-mail: eyigitoglu@gatech.edu, ling.liu@cc.gatech.edu.*
- *M. Emre Gursoy is with the Department of Computer Engineering, Koc University, Istanbul, Turkey. e-mail: emregursoy@ku.edu.tr.*

*Manuscript received X; revised Y.*

approaches (random sampling and network expansion) as well as XSTAR [5], which is the most relevant work to ours from the literature. Results show that STARCLOAK offers significantly improved query success rate and throughput. Furthermore, compared to XSTAR, STARCLOAK achieves substantially reduced anonymization time, network bandwidth usage, and improved resilience against inference attacks.

**Contributions:** In summary, the novelty and contributions of this paper can be stated as follows:

(1) We design and develop STARCLOAK, a location anonymization service for protecting the location privacy of mobile travelers on road networks. The design of STARCLOAK achieves: (i) user-defined, personalized privacy and utility goals through $k$-user anonymity and $l$-segment indistinguishability, (ii) resilience against correlation-based replay and query injection attacks, and (iii) reduction of communication and time costs of anonymization.

(2) We propose several optimized data structures and algorithms to implement STARCLOAK efficiently and effectively.

(3) We further propose two variants of STARCLOAK: Spatially Bounded STARCLOAK and Hybrid STARCLOAK. These variants enable generating more compact cloaking regions so that lower query processing and communication costs can be attained; however, they incur increased anonymization time.

(4) We experimentally evaluate STARCLOAK and its variants against two baseline anonymization approaches and XSTAR, which is the most relevant work to ours from the literature [5]. Experiments show that STARCLOAK can achieve substantial improvements compared to XSTAR, especially in terms of efficiency and attack-resilience.

## 2 RELATED WORK

Location privacy has been an active research area in recent years and several mechanisms were developed for privacy-preserving LBS usage. We analyze related work in two categories: (i) privacy in unconstrained geographic environments, (ii) privacy for mobile travelers on road networks. In the former, users' locations may be in any point of the geographical space. However, in case of mobile travelers on road networks, since their mobility is constrained by the underlying road network, works under the first category may be vulnerable to map-matching or similar attacks. In contrast, works under the second category take into account the structure of the road network while enforcing privacy. STARCLOAK belongs to the second category.

### 2.1 Privacy in Unconstrained Environments

For preserving privacy in unconstrained environments, many approaches utilize the notion of location $k$-anonymity [6] with a trusted third party (TTP) [7]. This TTP sits between the users and the LBS provider, and acts as the anonymizer. The TTP requirement was relaxed in [8], [9] using distributed servers and in [10] to support peer-to-peer environments. A collaborative peer-to-peer communication model called CAST was proposed in [11], which leverages trusted peers and cached data to compute results locally with lower latency. Personalized $k$-anonymity was introduced in [12] so that users' personalized privacy preferences can be supported, e.g., different $k$ for each user. In addition to $k$-anonymity, application of $l$-diversity to location privacy can ensure that the location of a user is indistinguishable between $l$ different, diverse locations [13]. To circumvent continuous tracking of users' trajectories, a time-obfuscated approach was proposed by Hwang et al. [14] which extends $k$-anonymity.

After the emergence of differential privacy (DP), an extension of DP called *geo-indistinguishability (GI)* was developed and applied in the LBS domain [15], [16]. An alternative to GI is Bayesian location privacy, which obfuscates users' locations such that an optimal Bayesian adversary trying to reconstruct the user's true location from their obfuscated location will incur maximal error [17], [18]. Yu et al. [19] proposed PIVE which considers both GI and Bayesian location privacy. The Eclipse approach proposed in [20] addresses a shortcoming of PIVE, which is resilience against long-term observation attacks.

Sending fake queries to LBS providers with *dummy* locations is another prominent privacy approach [21], [22]. Lu et al. [23] segmented circular or grid regions into subregions, and distributed dummies to radiuses or subgrids. Do et al. [24] proposed a dummy generation method using conditional probabilities so that dummies will remain resistant to adversaries with external spatiotemporal knowledge. Liu et al. [25]'s method takes into account spatiotemporal correlations when generating dummies. Hara et al. [26] developed a dummy generation method to satisfy physical constraints of the environment. Sun et al. [27] proposed a method called PPCS to generate dummy locations while considering semantic information of those locations, and showed that this approach can withstand certain collusion and inference attacks.

Spatial transformations can be used to transform location coordinates and/or evaluate LBS queries in the transformed space. Gutscher [28] proposed an approach based on coordinate transformations using rotation and translation. Khoshgozaran et al. [29] proposed a one-way transformation to map spatial objects and queries to another space, and use a Hilbert curve-based approach to evaluate range and nearest neighbor queries in the transformed space. Gupta and Rao [30] proposed VIC-PRO for protecting users' location vicinity using geometrical transformations such as reflection and transformation while satisfying $k$-anonymity.

Works discussed in this section assume unconstrained geographic environments, i.e., user's location can be in any point of the geographic space. However, since the mobility of road network travelers is constrained by the underlying roads, these works are often not suitable for road network travelers. For example: (i) a $k$-anonymous circular cloaking region could remain vulnerable if there is only one plausible road within it; (ii) if some dummy locations reported by the traveler do not correspond to roads or it is physically impossible to be driving there, then the adversary can easily filter out the dummies. Furthermore, from a utility perspective, knowledge of the road network can improve utility and reduce query execution costs via various optimizations. Since STARCLOAK is a privacy mechanism for road network travelers, it is different from the works studied in this section.

### 2.2 Privacy for Mobile Travelers on Road Networks

Privacy mechanisms for mobile travelers on road networks can be divided into three subcategories: mobile permission systems, mix-zones, and location obfuscation. Mobile permission systems are used to prevent untrusted third-party apps or LBS providers to access users' locations in sensitive zones. Felt et al. [31] and Kelley et al. [32] studied users' preferences and interactions with mobile permission systems. Liu et al. [33] showed that a binary classifier can predict users' permission decisions with high accuracy. Olejnik et al. [34] developed SmarPer to enable personalized permission decision making through machine learning. Yigitoglu et al. [35] developed PrivacyZone, which protects users' locations from being disclosed to third party mobile apps in privacy-sensitive regions called *quarantine areas*. The optimizations in PrivacyZone take advantage of road network structure. However, works that fall under the permission system category are generally not comparable to StarCloak because they either completely block location access or randomly perturb the user's location when the user is in a sensitive zone. They do not guarantee privacy notions such as $k$-user anonymity or $l$-segment indistinguishability.

Mix-zones have been employed in several works to circumvent the risks of continuous location tracking on road networks. After a set of users enter a mix-zone, they change pseudonyms and exit the mix-zone such that the mapping between users' old and new pseudonyms is hidden. MobiMix considers road network, time spent in mix-zone, and travel speed constraints to build attack-resilient mix-zones [36]. Palanisamy and Liu [37] further improve effectiveness and attack-resilience by studying *continuous query correlation attacks* and non-rectangular mix-zones. The approach in [38] enables distribution of group secret keys in cryptographic mix-zones in the presence of malicious eavesdroppers, without relying on trusted dealers. Vaas et al. [39] propose using fictive chaff vehicles to establish attack-resilient mix-zones in areas with low traffic density. Mix-zones differ from location obfuscation and StarCloak in several ways. Most importantly, mix-zones do not anonymize users on demand (i.e., when user issues query to a LBS) but rather when sufficiently many users enter a mix-zone.

StarCloak falls under the location obfuscation subcategory. Under this subcategory, Mouratidis and Yiu [40] provide $k$-anonymity for road network travelers under reciprocity requirement. Chow et al. [41] support personalized privacy specifications such that a cloaked region satisfies $k$-anonymity and includes a total minimum segment length of $L$. Li and Palanisamy [42] propose *reversible* cloaking such that anonymity levels can be reduced to accommodate multi-level privacy and selective de-anonymization. Yang et al. [43] study the orthogonal problem of *path privacy*, and define the M-cut requirement to achieve path privacy. A similar path privacy problem is studied in [44]. Another orthogonal problem is semantic-aware sharing of sensitive locations under road network constraints [45], [46]. In contrast, StarCloak does not require semantic annotation. Qiu et al. [47] design a location obfuscation strategy to achieve geo-indistinguishability in spatial crowdsourcing for vehicles on road networks.

Most closely related to our work under this category is XStar [5], which also achieves personalized $k$-user anonymity and $l$-segment indistinguishability for road network travelers. However, XStar does not consider resilience against the attack models in StarCloak, such as correlation-based replay attack and query injection attack (Section 3.3). Furthermore, StarCloak has several internal algorithms and data structures to improve efficiency and attack-resilience, such as cloaking graphs and star-set pruning. As a result, our experiments comparing StarCloak and XStar show that StarCloak can achieve much higher throughput and attack-resilience while being substantially faster than XStar.

## 3 StarCloak Overview and Concepts

This section presents an overview of StarCloak's privacy, utility, attack, and cost models, followed by the problem statement which combines these models. StarCloak can be viewed as a location anonymization service that sits between mobile users and untrusted LBS providers. Assume that user Alice wants to issue a LBS query. Without StarCloak, Alice's query is directly sent to the untrusted LBS provider with her true current location; the LBS provider executes the query based on Alice's location and sends the results back to Alice's device. However, if Alice is using StarCloak, then StarCloak first computes an anonymized location for Alice and replaces her true location with the anonymized location (transparently from Alice), before the query is sent to the untrusted LBS provider. Hence, the privacy of Alice's true location is protected from the untrusted LBS provider.

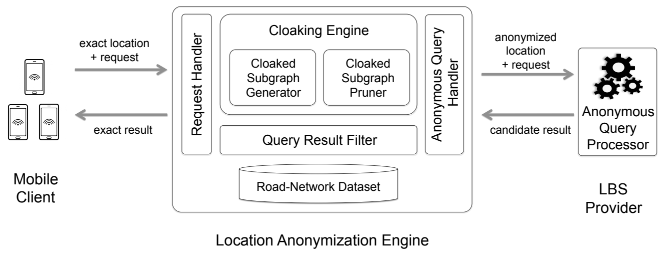

Fig. 1: Overall architecture of StarCloak

Figure 1 illustrates the reference architecture. Let $q$ denote the original query of mobile user $u$. When $u$ issues query $q$ with his/her true location, the location and query are intercepted by the location anonymization engine. The engine transforms $u$'s true location to a cloaked location $S$ while meeting the personalized privacy and utility profile of $u$. Next, the engine relays the anonymized location and query to the LBS provider. The LBS provider computes a candidate result, and the candidate result is received by the location anonymization engine. Since the cloaked location often has lower spatial resolution than the actual location to meet privacy goals, the candidate result may contain false positives. The anonymization engine performs post-processing of the candidate result to filter false positives and obtain the exact result. Finally, the exact result is delivered to $u$.

In StarCloak, we use a trusted third-party anonymization server to provide anonymization service. As shown in Figure 1, this server handles queries from many mobile users. There are mainly two reasons why having a trusted server is preferred. First, in order to provide $k$-user anonymity, StarCloak needs to know the locations of multiple users simultaneously. This would be difficult if StarCloak was deployed directly and independently on each user's device. Second, StarCloak's anonymization is based on road network information (see the *Road Network Dataset* in Figure 1), which is usually too large for mobile users to store on their device. Furthermore, there can be frequent updates to the road network structure (e.g., road closures) which would necessitate frequent application updates if StarCloak was deployed on the user's device. Therefore, we prefer to deploy StarCloak on a trusted third-party anonymization server.

### 3.1 Road Network Model

A road network is represented as an undirected graph $G = \langle V_G, E_G \rangle$ with node set $V_G$ denoting road junctions and edge set $E_G$ denoting roads, respectively. Each road connects a pair of junctions. We use $d_G(v)$ to denote the degree of a node $v$ with respect to $G$, i.e., $d_G(v) = |\{w|(v,w) \in E_G\}|$. We call $v$ an *intersection node* if $d_G(v) \geq 3$. For example, Figure 2 illustrates a sample road network. The road junctions are shown as $V_G = \{v_1, v_2, ..., v_{16}\}$. It holds that: $d_G(v_1) = 2$ and $d_G(v_5) = 3$, therefore $v_5$ is an intersection node. $q_1$ and $q_2$ are queries of two hypothetical users; first user is traveling on the road between $v_2$ and $v_6$, and second user is traveling on the road between $v_6$ and $v_7$.

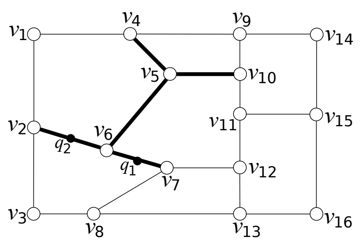

Fig. 2: Road network and query injection example

An anonymized location in the road network can be represented as a subgraph. *Border nodes* are nodes that connect a subgraph $S$ to the remainder of the main graph $G$.

**Definition 1** (Subgraph)**.** *$S$ is a subgraph of road network $G$, denoted by $S = \langle V_S, E_S\rangle$, if and only if $V_S \subset V_G$ and $E_S \subset E_G$.*

**Definition 2** (Border Node)**.** *Let $S$ denote a subgraph of $G$. The set of border nodes of $S$, denoted $BV(S)$, are nodes in both $S$ and $G$ but have edges that are in $E_G$ but not in $E_S$. Formally:*

$$BV(S) = \{v \mid \exists w \in (V_G \setminus V_S) \text{ s.t. } (v, w) \in E_G\}$$

Equivalently, border nodes are those nodes $v$ in $S$ that satisfy the condition: $d_G(v) > d_S(v)$. As an example, we can construct a subgraph $S$ in Figure 2 as: $V_S = \{v_2, v_4, v_5, v_6, v_7, v_{10}\}$ and $E_S = \{(v_4, v_5), (v_5, v_{10}), (v_5, v_6), (v_6, v_7), (v_2, v_6)\}$. In Figure 2, this subgraph is marked in bold. It can serve as the anonymized location for users with queries $q_1$ and $q_2$. Then, it holds that: $BV(S) = \{v_2, v_4, v_7, v_{10}\}$. Finally, we define a *segment* to refer to a consecutive sequence of edges.

**Definition 3** (Segment)**.** *A segment, denoted by $\overline{v_0 v_L}$, is a sequence of edges $(v_0, v_1), \ldots, (v_i, v_{i+1}), \ldots, (v_{L-1}, v_L)$, where all $v_i$ are unique and satisfy the conditions: $d_G(v_0) \geq 3$, $d_G(v_L) \geq 3$, and $d_G(v_i) = 2$ for $0 < i < L$.*

### 3.2 Personalized Privacy and Utility Model

StarCloak enforces location privacy for mobile users while considering privacy and utility simultaneously. It supports personalized location $k$-user anonymity and $l$-segment indistinguishability, such that instead of using a system-supplied fixed $k$ or $l$ for all users and queries [6], it achieves high versatility via user-specified privacy needs and specifications [48]. In addition, we introduce two utility metrics to capture location utility constraints: maximum spatial and temporal cloaking resolutions. These utility metrics constrain and regulate StarCloak so that it performs anonymization while meeting the spatial and temporal tolerances.

StarCloak performs location anonymization via **cloaking**, i.e., user's exact location is transformed into a cloaked region with lower spatial resolution (such as a subgraph of the road network) to achieve privacy. There are two formal privacy notions used in StarCloak: $k$-user anonymity and $l$-segment indistinguishability. $k$-user anonymity protects user $u$'s location by "hiding $u$ in a crowd", i.e., enforcing at least $k-1$ other users in the vicinity of $u$ report the same cloaked location. We observe that $k$-user anonymity is not sufficient to prevent the linkage of user $u$ with a sensitive public location or road segment, since the cloaked $k$-anonymized region may lack sufficient segment diversity, e.g., it may contain only a single road segment. This motivates the proposal of $l$-segment indistinguishability.

**Definition 4** ($k$-user anonymity)**.** *An anonymized location $S$ satisfies $k$-user anonymity if at least $k$ active users report $S$.*

**Definition 5** ($l$-segment indistinguishability)**.** *An anonymized location $S$ satisfies $l$-segment indistinguishability if it contains at least $l$ different road segments.*

In StarCloak, a query $q$ is allowed to specify a custom privacy requirement as $(\delta_k^q, \delta_l^q)$, such that $\delta_k^q \geq 1$ is the desired $k$-user anonymity level and $\delta_l^q \geq 1$ is the desired $l$-segment indistinguishability level.

A trivial approach to achieve maximum protection could be to assign the whole road network $G$ as the anonymized location. However, this approach clearly provides weak utility and low quality of service. Hence, we incorporate *spatial* and *temporal cloaking resolutions* as utility constraints. The spatial constraint $\sigma_s$ bounds the spatial size of the anonymized location. This is necessary so that anonymized locations are not arbitrarily large. The temporal constraint $\sigma_t$ bounds the maximum time delay resulting from anonymization. This is necessary so that the query-issuing user receives a response in a timely manner.

**Definition 6** (Query profile)**.** *For user $u$ with query $q$, we denote by $(\delta_k^q, \delta_l^q, \sigma_s^q, \sigma_t^q)$ the complete profile of $q$, where $\delta_k^q, \delta_l^q$ are the privacy parameters and $\sigma_s^q, \sigma_t^q$ are the utility parameters.*

For query $q$, combining the privacy parameters $(\delta_k^q, \delta_l^q)$ and utility parameters $(\sigma_s^q, \sigma_t^q)$, we arrive at the complete profile of $q$. By allowing each of the parameters to be query-specific, StarCloak provides maximum flexibility to end users so that each user can select different, personalized privacy and utility levels that are suitable for them.

### 3.3 Attack Model

While $k$-user anonymity and $l$-segment indistinguishability provide desirable privacy properties, they may still be vulnerable to attacks, as was shown by background knowledge attacks and minimality attacks on tabular $k$-anonymity and $l$-diversity definitions [49]. These attacks enable adversaries to leverage their knowledge of the anonymization algorithm and aggregate public statistics to violate the privacy of their victims. As a result, the adversary may be able to predict the user's true location with confidence higher than the desired limit of $1/l$. In order to capture an adversary's power of associating $u$ with a segment in the location privacy domain, we use the notion of *linkability* [5].

**Definition 7** (Linkability)**.** *For user $u$ whose anonymized location is $S$, the linkability of $u$ with a specific segment $s^* \in S$ is the probability that adversary associates $u$ with $s^*$ based on adversarial background knowledge $K_{ad}$, denoted as: link$[u \leftarrow s^* | S, K_{ad}]$.*

Next, we explain how $K_{ad}$ can be formulated and attacks can be executed in a mobile LBS ecosystem. In particular, we consider 2 attacks: replay attacks and query injection attacks. We design StarCloak to have high resilience against these attacks.

**Correlation-Based Replay Attack:** In a standard replay attack, the adversary observes the anonymized location $S$ and attempts to reverse-engineer the user's true location with understanding of the anonymization algorithm and underlying road network structure [5], [41]. Specifically, the adversary re-runs the anonymization algorithm, denoted $\mathcal{A}(\cdot)$, for each segment $s \in S$ that could potentially be the mobile user's actual location. The similarity between $S$ and the algorithm's output $S'$ generated by $\mathcal{A}$, is used to estimate the likelihood of $s$ having generated $S$:

$$\textit{like}[S|u \leftarrow s, K_{ad}] = \frac{|S' \cap S|}{|S|} \tag{1}$$

In this paper, we consider an enhanced version of the replay attack, called the *correlation-based replay attack*, in which the adversary also knows the statistical density distribution of users on the road network. For example, the adversary may know the traffic density of the city during rush hour, which enables him/her to predict that there is higher probability that the user is actually located on a dense segment rather than a sparse segment. This is a realistic assumption, given that density statistics are nowadays publicly shared by the likes of navigation services and/or Uber Movement.

In the correlation-based replay attack, the likelihood computation of Equation 1 is enhanced in two ways. First, considering that a user's location is cloaked together with other active users in the vicinity to achieve $k$-user anonymity, the placement of the remaining $k-1$ users in $S$ is integrated into the calculation. Note that there are a total of $\left(\!\!\binom{S}{k-1}\!\!\right)$ different possible placements of $k-1$ queries in $S$. We denote by $m_i$ each placement, such that $1 \leq i \leq \left(\!\!\binom{S}{k-1}\!\!\right)$. Second, we denote by Pr$[s]$ the probability of user $u$ being located on segment $s$, and by Pr$[m_i]$ the probability of remaining $k-1$ users being located as in the placement of $m_i$, both according to the knowledge of statistical density. Then, we compute the enhanced *like*$^c[S|u \leftarrow s, K_{ad}]$ as:

$$\textit{like}^c[S|u \leftarrow s, K_{ad}] = \sum_{i=1}^{\left(\!\binom{S}{k-1}\!\right)} \Pr[s] \cdot \Pr[m_i] \cdot \textit{like}[S|u \leftarrow s, m_i, K_{ad}] \tag{2}$$

Then, linkability is calculated as:

$$\textit{link}[u \leftarrow s^*|S, K_{ad}] = \frac{\textit{like}^c[S|u \leftarrow s^*, K_{ad}]}{\sum_{s \in S} \textit{like}^c[S|u \leftarrow s, K_{ad}]} \tag{3}$$

**Query Injection Attack:** In the replay attack, the adversary is passive (i.e., observer only). In the *query injection attack*, the adversary is active, i.e., he/she is also capable of injecting queries into the system. We expect anonymization algorithms with tight $\sigma_s$ constraints to be more vulnerable to the query injection attack.

A typical anonymization algorithm cloaks multiple segments into an anonymized location if and only if they include at least one active query, and through the shortest paths between the active queries. This minimizes the size of $S$ by not adding any redundant segments to $S$, therefore such an algorithm has higher chance of satisfying the $\sigma_s$ constraints. However, the query injection attack exploits this property as follows. Consider the anonymized location $S$ that consists of the bold lines in Figure 2. Suppose that $q_1$ and $q_2$ are the queries injected by the adversary with privacy profiles $(\delta_k, \delta_l) = (3, 3)$. Then, the adversary can infer that the third (actual) query was issued from either segment $\overline{v_4v_5}$ or segment $\overline{v_5v_{10}}$, since the segment $\overline{v_5v_6}$ would not have been added to $S$ otherwise.

To integrate the effects of the query injection attack, the *like*$^c$ calculation in Equation 2 is modified as follows. Let $\bar{q}$ be queries injected by an adversary, i.e., the adversary knows the true locations of these queries. Then, if the placement $m_i$ is plausible under $\bar{q}$, we use the original value of Pr$[m_i]$ in the *like*$^c$ calculation. If $m_i$ is implausible, e.g., one of the queries in $\bar{q}$ is from $\overline{v_4v_5}$ but $m_i$ does not assign any queries to $\overline{v_4v_5}$, we set Pr$[m_i] = 0$. The calculation of linkability is same as Equation 3.

## 3.4 Query Cost Model

An important challenge in finding an optimal anonymized location $S$ for a query $q$ is to minimize the *cost* of the query when executed with the anonymized location. We study two types of cost: *cost of query evaluation* and *cost of communication*.

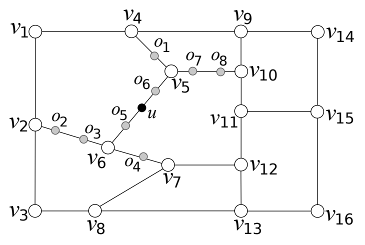

Fig. 3: Illustration of query processing on a road network

**Cost of Anonymized Query Evaluation:** Most query processing approaches for road networks are based on two types of fundamental operations: edge-based and node-based. An edge-based operation takes a query $q$ and an edge $e$ as input and returns a set of objects on $e$ denoted $\mathcal{O}_e(q, e)$ which satisfy the query condition. For segment $s$ potentially composed of a sequence of edges, we have: $\mathcal{O}_s(q, s) = \cup_{e \in s} \mathcal{O}_e(q, e)$. We denote by $\mathcal{C}_s$ the average computation cost of evaluating the query on a segment. In contrast, a node-based operation takes a query $q$ and a node $v$ as input and returns a set of objects in the vicinity of $v$ denoted $\mathcal{O}_v(q, v)$ which satisfy the query condition. The computation cost of evaluating a node-based query is denoted by $\mathcal{C}_v$.

Let $q$ denote a query issued at some position while traveling on segment $s$, and let $v_b^s$ and $v_e^s$ denote the two ends of $s$. The query result $\mathcal{R}(q, s)$ satisfies the following:

$$\mathcal{R}(q, s) \subseteq \mathcal{O}_s(q, s) \cup \mathcal{O}_v(q, v_b^s) \cup \mathcal{O}_v(q, v_e^s) \tag{4}$$

We give an example in Figure 3. A 3-nearest neighbor query is issued by a user $u$ located on segment $\overline{v_5v_6}$. The exact answer to this query is $\mathcal{R}(q, s) = \{o_5, o_6, o_7\}$, which is indeed a subset of the union of $\mathcal{O}_s(q, \overline{v_5v_6})$ = $\{o_5, o_6\}$, $\mathcal{O}_v(q, v_5)$ = $\{o_1, o_6, o_7\}$ and $\mathcal{O}_v(q, v_6)$ = $\{o_3, o_4, o_5\}$. We extend this model from a single segment $s$ to anonymized locations which potentially consist of a set of segments $S$ by employing the concept of border nodes (see Definition 2). Concretely, the result of query $q$ with $S$ as its anonymized location satisfies:

$$\mathcal{R}(q, S) \subseteq \left(\cup_{s \in S} \mathcal{O}_s(q, s)\right) \cup \left(\cup_{v \in BV(S)} \mathcal{O}_v(q, v)\right) \tag{5}$$

Consequently, the evaluation cost of $q$ with anonymized location $S$, denoted by $cost_{eval}(q, S)$ can be estimated as:

$$cost_{eval}(q, S) = \mathcal{C}_s \cdot |S| + \mathcal{C}_v \cdot |BV(S)| \tag{6}$$

where $|BV|$ denotes the number of border nodes in $S$ and $|S|$ denotes the number of segments in $S$.

**Cost of Communication:** We presented the architecture and communication phases of StarCloak in Figure 1. Here, when calculating the cost of communication, we focus specifically on the cost that is added due to the usage of a location anonymization service such as StarCloak.

For query $q$, the communication cost in mobile client's exact request sent and the exact result it receives do not change depending on whether an anonymization service is used or not, since a service request takes a fixed encoded format and the size of the exact answer is fixed. With respect to the messages exchanged between the location anonymization service and the LBS provider, we measure communication cost as the length of the sent and received messages, and use $\|x\|$ to denote the encoded length of object $x$. For the message sent from the anonymization service to the LBS provider, the query remains intact while the location is anonymized by cloaking it to a set of segments $S$. Therefore, the communication cost here is $\|q\| + \|S\|$. The message sent from the

LBS provider to the anonymization service contains the candidate result $\mathcal{R}(q, S)$; hence, the communication cost here is $\|\mathcal{R}(q, S)\|$. As discussed above, a query $q$ usually has fixed length. Also, for given privacy requirements, the number of segments in $S$ tends to be fairly stable. As such, we conclude that $\|\mathcal{R}(q, S)\|$ is the dominant and most "optimizable" communication cost.

For query $q$, let $res\_size$ denote the average exact result size of $q$, e.g., if $q$ is the popular $k$-NN query, then $res\_size = k$. Following Equation 5, given a query $q$ and anonymized location $S$, the size of the candidate result $\mathcal{R}(q, S)$ can be estimated as:

$$|\mathcal{R}(q, S)| \leq res\_size \cdot |BV(S)| + \sum_{s \in S} \sum_{e \in s} |\mathcal{O}_e(q, e)| \quad (7)$$

Then, denoting by $\rho_o$ the average number of objects on an edge and $\mathcal{C}_o$ the cost of sending/receiving an object $o$ over the wireless channel (e.g., sending unique identifier of $o$), the total communication cost for $q$ with anonymized location $S$ is:

$$cost_{comm}(q, S) = \mathcal{C}_o \cdot \left[ res\_size \cdot |BV(S)| + \rho_o \cdot \sum_{s \in S} \sum_{e \in s} |e| \right]$$

**Overall Cost:** It is desirable to combine $cost_{eval}$ and $cost_{comm}$ to find an estimation of the overall cost. In StarCloak, we consider a linear combination scheme:

$$cost(q, S) = \beta \cdot cost_{comm}(q, S) + (1 - \beta) \cdot cost_{eval}(q, S)$$

where $\beta$ is the parameter tuning the trade-off between evaluation cost (mainly CPU computation on server side) and the communication cost (mainly bandwidth of wireless channel).

### 3.5 Problem Statement

Given a road network represented as a graph $G$ with mobile users traveling on it while issuing queries, where each user $u$'s query $q$ is associated with its profile $(\delta_k^q, \delta_l^q, \sigma_s^q, \sigma_t^q)$, the principles and objectives of StarCloak are:

- User $u$'s true location is transformed to an anonymized (cloaked) location $S$, where $S$ is a subgraph of $G$.
- $S$ satisfies the privacy requirements of $q$ in terms of $\delta_k^q$-user anonymity and $\delta_l^q$-segment indistinguishability.
- $S$ satisfies the utility constraints of $q$, i.e., spatial size of $S$ is no larger than $\sigma_s^q$ and the temporal delay caused by location anonymization is no more than $\sigma_t^q$.
- $S$ achieves high resilience against the replay and query injection attacks.
- Anonymized location $S$ yields low $cost(q, S)$.

StarCloak's objectives take multiple aspects into consideration: privacy, utility, attack-resilience, and cost. However, the key challenge lies in satisfying these objectives simultaneously, as high privacy and attack-resilience are often in conflict with max utility and min cost. For example, an anonymization approach that randomly samples road segments and adds them to $S$ can provide high privacy and attack-resilience, but causes increased query processing cost and decreases the likelihood of satisfying spatial utility constraints. On the other hand, an anonymization approach that relies on network expansion (e.g., using Dijkstra's deterministic network expansion algorithm) causes vulnerability to replay and query injection attacks despite increased likelihood of satisfying spatial utility constraints.

StarCloak achieves the above objectives via three novel aspects. First, StarCloak introduces *star graph*, a road junction-based abstraction and a cost-aware randomness in road junction-based expansion to reduce the high query cost of random sampling and improve the attack-resilience of baseline network expansion. Second, StarCloak introduces a *cloaking graph* to generate candidate star graphs by grouping queries that can be cloaked together based on their privacy profiles. Third, StarCloak uses utility-aware and cost-effective pruning strategies to generate final cloaking star graph that has low query processing cost and high attack-resilience among the set of candidate star graphs.

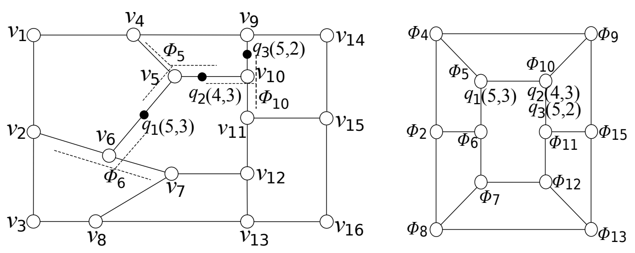

Fig. 4: Illustration of the Star concept

## 4 StarCloak Algorithms

This section explains the StarCloak constructs and algorithms in detail. We first describe *star* concept and other data structures used in StarCloak in Section 4.1. We explain how an incoming query $q$ is pre-processed by StarCloak and added to appropriate data structures in Section 4.2. Overview of the main StarCloak algorithm is presented in Section 4.3. The main algorithm relies on several methods, such as selecting a star, updating the cloaking graph (adding and removing queries from the cloaking graph), candidate star-set selection, and star-set pruning. These methods are described in Sections 4.4, 4.5, 4.6, and 4.7, respectively.

### 4.1 Star Concept and StarCloak Data Structures

StarCloak introduces *star* as the basic unit of location cloaking on road networks. Each star is defined by a vertex with its neighbor segment list in $G$.

**Definition 8** (Star). *Let $G = \langle V_G, E_G \rangle$ denote the road network of interest. We define a star $\Phi_i$ anchored at vertex $v_i \in V_G$ as a subgraph of $G$, denoted by $\Phi_i = \langle V_\Phi^i, E_\Phi^i \rangle$, and $V_\Phi^i = \{v_i\}$ and $E_\Phi^i = \{w | w \neq v_i, w \in V_G, (v_i, w) \in E_G\}$.*

Accordingly, every node $v_i$ with $d_G(v_i) \geq 3$ is associated with a unique star $\Phi_i$, which consists of vertex $v_i$ and all of its adjacent road segments, that is, those segments with $v_i$ as one of two end nodes. For example, in the left plot of Figure 4, star $\Phi_5$ is composed of node $v_5$ and segments $\{\overline{v_5 v_4}, \overline{v_5 v_6}, \overline{v_5 v_{10}}\}$.

The road network can then be transformed into a *star graph*, as shown on the right of Figure 4. Each vertex in the star graph is a star in $G$, and two vertices are adjacent in the star graph if and only if their corresponding stars in $G$ share a segment. All edges in the star graph are of unit length. The *hop distance* between two stars $\Phi_i$ and $\Phi_j$ in a road network $G$ is measured by the number of hops in the shortest path between $\Phi_i$ and $\Phi_j$. For example, in Figure 4, the hop distance between $\Phi_6$ and $\Phi_{10}$ is 2, since their shortest path in the star graph is $\Phi_6 \rightarrow \Phi_5 \rightarrow \Phi_{10}$.

In addition to the star concept, StarCloak uses some important data structures for improved effectiveness and efficiency.

***Query Queue,*** $\mathcal{Q}$: A first-in-first-out (FIFO) queue that records the incoming queries which must be anonymized before they are relayed to the respective LBS provider. Incoming queries are inserted into the queue from the tail. The anonymization engine pops each query from $\mathcal{Q}$ to find a suitable cloaked subgraph $S$.

***Expiration Heap,*** $H$: A max-min heap that maintains the queries in the order of their expiration time computed as: query arrival time + temporal delay constraint $\sigma_t^q$. StarCloak checks $H$ to identify queries that are close to their expiration time in order to prioritize them or to identify queries that have expired.

***Cloaking Graph,*** $G_C$: An undirected graph dynamically constructed in-memory, for recording the set of queries associated to a star based on their similarities with respect to their privacy requirements, their spatial proximity, and their expiration times. The cloaking graph will be explained further in Section 4.5.

***Star-Map,*** $M_S$ ***and Query-Map,*** $M_Q$: We create one hash map to index stars called a Star-Map, and similarly, one hash map to index queries associated to a node in the cloaking graph called Query-Map, for fast star and query look-up.

***Candidate Star-Set Queue,*** $\mathcal{Q}_C$: A FIFO-based queue structure that records generated candidate cloaking star-sets. The pruning method in StarCloak pops star-sets from $\mathcal{Q}_C$ to apply randomized pruning when generating the final cloaked star regions.

## 4.2 Incoming Query Pre-processing

Let $q$ be an incoming query. StarCloak pre-processes $q$ to generate the internal representation of $q$ by performing the following sequence of tasks. First, a unique identifier is assigned to $q$ using a secure hash function with user ID and query issue time, i.e., $hash(q.u||q.t)$. Second, using the latitude and longitude values of $q$'s location together with the road network graph, the true road segment of $q$ is determined. Third, $q$ is inserted to queue $\mathcal{Q}$. Fourth, $q$ is inserted to the expiration heap $H$ with query expiration time as key and query identifier as value. Query expiration time is equal to: $q.t + \sigma_t^q$.

## 4.3 Main StarCloak Procedure

The main StarCloak procedure is presented in Algorithm 1. StarCloak runs actively as long as there is at least one query in queue $\mathcal{Q}$ waiting to be served, i.e., $\mathcal{Q}$ is not empty (line 1). It continuously pops queries from $\mathcal{Q}$ and processes them to find anonymized locations that fit users' privacy and utility requirements. First, StarCloak removes expired queries from the system (lines 2-11). To find expired queries, StarCloak utilizes the expiration heap $H$. Since this heap maintains queries in order of their expiration time, queries that are more likely to have expired will be closer to the beginning of the heap. Therefore, StarCloak checks the first entry of the heap $q_e$ (line 4). If $q_e$ has not expired, then the remaining entries of $H$ need not be checked, since the heap is sorted (hence, we break out of the loop on lines 10-11). However, if $q_e$ has expired, then it is removed from the cloaking graph on line 6 (see Section 4.5 for details of removal). When an expired query $q_e$ is removed from the cloaking graph, the corresponding node $v_u$ of the cloaking graph is affected. If this node $v_u$ is not empty, it is added to the list $\mathcal{L}$ on lines 7-8 so that it can be processed later. Then, StarCloak pops $q_e$ from $H$ (line 9) and continues with the next iteration of the loop to check if there are any other expired queries in $H$.

During the procedure between lines 2-11 of Algorithm 1, list $\mathcal{L}$ accumulates, consisting of cloaking graph nodes. Between lines 12-15, StarCloak processes each cloaking graph node $v_u$ in $\mathcal{L}$ by attempting to find anonymized locations for these nodes. To do so, the candidate star-set of $v_u$ is searched (line 13). The details of this search are presented in Section 4.6 and Algorithm 3. If a plausible anonymized location is found by this search, then a non-empty return value is obtained from the *SearchCandidateStarSet* function. In this case, this return value can be added to the candidate star-set queue $\mathcal{Q}_C$ for pruning and generation of the final cloaked star region (lines 14-15).

**Algorithm 1:** Main StarCloak Procedure

```
Input: $\mathcal{Q}$: query queue, $H$: expiration heap,
       $G_C$: cloaking graph
Output: $\mathcal{Q}_C$: candidate star-set queue
1  while $\mathcal{Q} \neq \emptyset$ do
2      $\mathcal{L} \leftarrow \emptyset$
3      while $true$ do
4          $q_e \leftarrow$ first entry of $H$
5          if $q_e$ is expired then
6              $v_u \leftarrow$ RemoveQueryFromCloakingGraph($q_e$)
7              if $v_u \neq \varnothing$ then
8                  Add $v_u$ to $\mathcal{L}$
9              Pop $q_e$ from $H$
10         else
11             break
12     foreach $v_u \in \mathcal{L}$ do
13         $C \leftarrow$ SearchCandidateStarSet($v_u$)
14         if $C \neq \emptyset$ then
15             Add $C$ to $\mathcal{Q}_C$ for pruning
16     $q_i \leftarrow$ first entry of $\mathcal{Q}$
17     $\Phi_i \leftarrow$ SelectStar($q_i$)
18     $v_u \leftarrow$ AddQuerytoCloakingGraph($q_i$, $\Phi_i$)
19     $C \leftarrow$ SearchCandidateStarSet($v_u$)
20     if $C \neq \emptyset$ then
21         Add $C$ to $\mathcal{Q}_C$ for pruning
```

When Algorithm 1 reaches line 16, it is ready to process a query from the query queue $\mathcal{Q}$. The first entry of $\mathcal{Q}$ is retrieved and assigned to $q_i$ on line 16. Then, a star is selected to assign to it on line 17 using the *SelectStar* function. The details of *SelectStar* are presented in Section 4.4. Thereafter, StarCloak updates the cloaking graph by adding $q_i$ and the selected star $\Phi_i$ to the cloaking graph on line 18 (details of this function are presented in Section 4.5 and Algorithm 2). Finally, the steps taken between lines 19-21 are identical to the steps on lines 13-15.

## 4.4 Select Star

As shown in Algorithm 1, StarCloak performs anonymization by scanning through the query queue $\mathcal{Q}$. All segments that are associated with active queries are marked as *active*. StarCloak first selects a star to assign queries on the active segment as the initial cloaking star. By definition, each segment has two end nodes. If both nodes are intersection nodes, i.e., $d_G(v_s) \geq 3$ and $d_G(v_t) \geq 3$, StarCloak needs to determine to which star this active segment should be assigned: $\Phi_s$ or $\Phi_t$. For example, in Figure 4 when $q_1$ arrives and segment $\overline{v_5 v_6}$ becomes active, one of the two possible stars $\Phi_5$ or $\Phi_6$ will be determined as the initial cloaking star. When a star $\Phi$ is "selected" and segment $s$ is assigned to $\Phi$, we denote this by $s \leftarrow \Phi$.

In StarCloak, we use a cost-aware star selection strategy, taking into account the cost model described in Section 3.4. Let $q$ be the query, $AS$ denote the set of currently active segments on the road network $G$, and $\phi$ be the set of selected stars. Then, the minimization of the overall cost can be formally stated as:

$$\min_{\phi} \sum_{\Phi \in \phi} cost(q, \Phi) \tag{8}$$
$$\text{s.t. } \forall s \in AS, \exists \Phi \in \phi, s \leftarrow \Phi$$

This optimization problem aims at finding an assignment between stars and segments such that the stars cover all segments with active queries, while having the minimum total cost. It can be shown that the optimization problem in Expression 8 is NP-Hard. The proof follows from a reduction from the Vertex-Cover problem, which is a well-known NP-Complete problem. Specifically, if for all stars $\Phi$ in the star graph we set $cost(q, \Phi) = 1$, then the problem is equivalent to the classical Vertex-Cover problem. Motivated by the hardness of finding a globally optimal solution to our problem, we propose a randomized algorithm called Select Star, which finds approximate solutions with high assignment quality. The intuition is, for each query which has two endpoints as viable stars, the algorithm probabilistically selects one of the two stars with probability inversely proportional to their $cost$.

Our Select Star algorithm works as follows. Let $q$ be an incoming query with travel segment $s$, and let $\Phi_a$ and $\Phi_b$ be the two stars on the two endpoints of segment $s$. For simplicity, we assume both endpoints are stars; if not, then $s$ is trivially assigned to the endpoint which is a star. If only one of $\Phi_a$ or $\Phi_b$ is currently active, Select Star assigns $s$ to the active star. If both $\Phi_a$ and $\Phi_b$ are active, then $s$ is assigned to $\Phi_a$ with probability $cost(q, \Phi_b)/[cost(q, \Phi_a) + cost(q, \Phi_b)]$, or $\Phi_b$ otherwise. If neither star is active, then the same probabilistic assignment to either $\Phi_a$ or $\Phi_b$ is carried out, additionally, the assigned star is marked as active for next iterations. This assignment has the desirable property that the outcome of our randomized Select Star algorithm is not far from an optimal assignment. More formally, denoting by $cost^{opt}$ the cost achieved by the optimal assignment, and denoting by $cost^{rnd}$ the cost achieved by our Select Star algorithm, it holds in expectation that: $\mathbb{E}\left[cost^{rnd}\right] \leq 2 \cdot cost^{opt}$.

## 4.5 Cloaking Graph Update

We use the cloaking graph $G_C$ to group nearby queries and efficiently index query groups that can be cloaked together. The cloaking graph $G_C(V_C, E_C)$ is an undirected graph, where $V_C$ is the set of vertices each representing a set of requests grouped by the star they are assigned to. $E_C$ is the set of edges; there is an edge $e = (v_i, v_j) \in E_C$ between $v_i$ and $v_j$ iff queries associated with both vertices can be cloaked together based on $k$-user anonymity, $l$-segment indistinguishability, and spatial tolerance.

Let $\Phi$ be an active star and $v \in V_C$ be its corresponding vertex in $G_C$. For each $v$, the **query set** $v.Q$ consists of the queries assigned to $\Phi$. We compute the **combined privacy of utility requirements** $\langle \delta_k^v, \delta_l^v, \sigma_s^v \rangle$ for $v$ using queries in $v.Q$:

$$\langle \delta_k^v, \delta_l^v, \sigma_s^v \rangle := \langle \max_{q \in v.Q} \delta_k^q, \max_{q \in v.Q} \delta_l^q, \min_{q \in v.Q} \sigma_s^q \rangle \tag{9}$$

We denote by $v.\Theta$ the set of stars which are within $\sigma_s^v$ distance from $\Phi$. Also, **neighbor list** $v.N$ is stored with $v$, where being neighbors indicates that requests in corresponding nodes can be cloaked together. Two cloaking nodes $v_i$ and $v_j$ are considered to be neighbors iff: (i) Stars associated with each node are an element of the star-set of the other node, i.e., $\Phi_i \in v_j.\Theta$ and $\Phi_j \in v_i.\Theta$. (ii) Number of segments that cover both nodes is enough to satisfy their $l$-segment indistinguishability requirements, i.e., $|v_i.\Theta \cap v_j.\Theta| \geq max\{\delta_l^{v_i}, \delta_l^{v_j}\}$.

StarCloak performs two types of updates on the cloaking graph: add query to cloaking graph, remove query from cloaking graph. The function for adding queries to the cloaking graph is given in Algorithm 2. When the function is called to insert a new query, it checks all cloaking graph vertices associated with the corresponding star to add the new query (lines 4-14). If there is no possible vertex to add, a new vertex is created (lines 15-16). The new query can be added to an existing vertex only if its privacy profile does not conflict with the profile of the existing node. A conflict occurs when new spatial tolerance is not able to satisfy the new $l$-segment indistinguishability requirement. Thus, we need to perform the checks under lines 4-14 to avoid any conflicts.

**Algorithm 2:** Add Query to Cloaking Graph

```
Input: $q$: new query, $\Phi_n$: assigned star
Output: $v_u$: updated cloaking graph node
1  $v_u \leftarrow \varnothing$
2  $N \leftarrow M_S.get(\Phi_n)$
3  if $N \neq \emptyset$ then
4      forall $v_i \in N$ do
5          if $\sigma_s^q < \sigma_s^{v_i}$ then
6              $sc \leftarrow$ # of segments within $\sigma_s^q$ from $\Phi_n$
7              if $sc \geq max\{\delta_l^q, \delta_l^{v_i}\}$ then
8                  Add $q$ to the node $v_i$
9                  $v_u \leftarrow v_i$
10                 break
11         else if $v_i.sc \geq \delta_l^q$ then
12             Add $q$ to the node $v_i$
13             $v_u \leftarrow v_i$
14             break
15 if $v_u = \varnothing$ then
16     $v_u \leftarrow$ Create new node for query $q$
17 return $v_u$
```

The function for removing queries from the cloaking graph is described verbally here, and its formal version is given in the supplementary material. Say that $q_e$ is the expired query that should be removed. We first perform a look-up from $M_Q$ to find the cloaking graph node $v_u$ associated with $q_e$. If $|v_u.Q| > 1$, i.e., $v_u$ contains other queries as well, its information is updated based on remaining queries after deletion of $q_e$. The update is performed according to Equation 9 to re-compute $\delta_k^{v_u}$, $\delta_l^{v_u}$, and $\sigma_s^{v_u}$. If the updated $v_u$ now has either $\delta_l^{v_u} < \delta_l^{q_e}$ or $\sigma_s^{v_u} > \sigma_s^{q_e}$, then $v_u.\Theta$, $v_u.sc$ and $v_u.N$ are also re-computed. Note that the latter is only necessary if segment indistinguishability or spatial tolerance requirements are relaxed. On the other hand, if $|v_u.Q| = 1$, i.e., $q_e$ was the only query associated with $v_u$, then $v_u$ is removed from $G_C$, and $M_S$ is updated. The return value of the function is $v_u$, which is an input for the next step (candidate star-set selection).

## 4.6 Candidate Star-Set Selection

The goal of this step is to discover a set of stars, called *candidate star-set*, which constitutes a possible anonymized sub-graph for certain queries. In order to find such star-set, StarCloak searches over the cloaking graph and identifies a set of nodes, denoted by $NS$, that satisfy the privacy requirements of all queries associated with each node. Formally, let $\vartheta$ denote a candidate star-set, and let $seg(\vartheta)$ be a function that returns all segments associated with input stars. $NS$ meets $k$-user anonymity and $l$-segment indistinguishability if and only if:

$$\forall v \in NS : \ \left[\delta_k^v \leq \sum_{\hat{v} \in NS} |\hat{v}.Q|\right] \ \wedge \ \left[\delta_l^v \leq |seg(\bigcap_{\hat{v} \in NS} \hat{v}.\Theta)|\right] \tag{10}$$

Such $NS$ forms candidate star-set $\vartheta$ with all stars shared within the covered star-set of each node in $NS$:

$$\vartheta = \bigcap_{v \in NS} v.\Theta \tag{11}$$

**Algorithm 3:** Search Candidate Star-Set

```
Input: $v_u$: updated vertex
Output: $\vartheta$: candidate star-set
1  $NS \leftarrow v_u$
2  if $(\vartheta \leftarrow checkReqs(NS)) \neq \emptyset$ then
3      return $\vartheta$
4  $Q_{Comb} \leftarrow \emptyset$
5  forall $v \in v_u.N$ do
6      $NS \leftarrow v_u \cup v$
7      if $(\vartheta \leftarrow checkReqs(NS)) \neq \emptyset$ then
8          return $\vartheta$
9      $Q_{Temp} \leftarrow Q_{Comb}$
10     forall $C \in Q_{Comb}$ do
11         if $\forall v_c \in C, v_c \in v.N$ then
12             $NS \leftarrow v_u \cup v \cup C$
13             if $(\vartheta \leftarrow checkReqs(NS)) \neq \emptyset$ then
14                 return $\vartheta$
15             else
16                 $Q_{Temp} \leftarrow Q_{Temp} \cup C \cup v$
17     $Q_{Comb} \leftarrow Q_{Temp}$
```

We assume the existence of a procedure named *checkReqs(NS)*, which takes as input a set of nodes $NS$, performs the privacy check given in Equation 10, and returns either the $\vartheta$ built in Equation 11 if $NS$ passes the privacy check or an empty set $\emptyset$ otherwise. We use the *checkReqs* procedure in Algorithm 3 for candidate star-set selection.

Algorithm 3 gives the technical details of candidate star-set selection. Searching over the cloaking graph for finding a candidate star-set starts with the updated vertex $v_u$. If the number of queries assigned to this vertex is fewer than the $k$-user anonymity requirement of the vertex, then the algorithm continues the search process over the neighboring nodes, ordered by the hop distance between their associated star and the star of the starting vertex. For each neighbor node, it applies *checkReqs* to $v_u$ and the neighbor node combined (lines 6-8). If a candidate star-set still cannot be found, neighbor node is evaluated with all possible node combinations generated with the previously processed neighbor nodes (lines 10-16). On line 10, we denote by $C$ a clique in $Q_{Comb}$, and line 11 checks if the clique satisfies the $l$-segment indistinguishability requirement. The possible node combinations are tracked by variable $Q_{Comb}$, which is enlarged in each iteration so that newly visited nodes are added (lines 16-17). The output of the algorithm is $\vartheta$, a candidate a star-set.

### 4.7 Star-Set Pruning

Final component of StarCloak is star-set pruning: pruning of extra segments from candidate star-set. As specified in Algorithm 1, candidate star-sets found by Algorithm 3 are added to the candidate star-set queue denoted $\mathcal{Q}_C$, and then they are pruned by the star-set pruning component. Star-set pruning plays an important role in cost reduction and improvement of attack-resilience by randomizing the star selection from outer to center of the candidate star-set. Note that star-set pruning is highly parallelizable, i.e., it is possible to implement one or more pruning processes running in parallel (each popping from $\mathcal{Q}_C$) while a separate process performs remaining tasks and adds to $\mathcal{Q}_C$.

The pruning starts by popping a candidate star-set from queue $\mathcal{Q}_C$. Let $\vartheta$ denote the popped star-set. We find the set of boundary stars $BS$ of $\vartheta$, which are the stars that have at least one neighbor star not in $\vartheta$, as well as the set of active stars $AS$ of $\vartheta$ which cannot be removed from the star-set. Let $l^{max}$ denote the maximum $l$-segment indistinguishability requirement in the star-set. We run multiple iterations, and within each iteration, the following are performed. First, a random star denoted $\Phi_r$ is selected from $BS \setminus AS$. If $\vartheta$ still satisfies $l^{max}$-segment indistinguishability after removing $\Phi_r$ from $\vartheta$; then $\Phi_r$ is removed from $\vartheta$, $BS$ is updated by removing $\Phi_r$ from $BS$, and we proceed to the next iteration. However, if $l^{max}$-segment indistinguishability is violated after removing $\Phi_r$ from $\vartheta$, then the pruning stops here and the current $\vartheta$ (without removing $\Phi_r$) is produced as the final output of the pruning process. An equivalent technical description of the pruning algorithm, along with a visual example of star-set pruning, can be found in the supplementary material.

## 5 Variants and Optimizations

In this section, we introduce two variants of StarCloak for finding cloaking regions with lower cost, query processing time, and network bandwidth usage without sacrificing privacy.

### 5.1 Spatially Bounded StarCloak

Basic StarCloak generates cloaking regions whenever it finds a star-set that satisfies all queries' privacy requirements. However, this approach may cause cloaking regions that consist of stars that are far from each other and scattered across the road network. An example scenario is given in supplementary material.

We propose **spatially bounded StarCloak** to generate more compact cloaking regions. Its essence is to sacrifice anonymization time in favor of lower query processing and communication costs. We define a system parameter $\lambda \geq 1$ called the **compactness factor**, that controls the maximum hop distance between selected vertices in the candidate star-set. To generate more compact cloaking regions, we make some modifications to the candidate star-set selection algorithm. First, we group neighbors by their distance $d$ to the starting node. $\lfloor d/\lambda \rfloor$ determines the level of each group element. At each level, the algorithm only considers neighbor nodes which can be cloaked with the node combinations generated in the previous level. The algorithm searches level by level iteratively in top-down manner. Then, spatially bounded StarCloak enforces compactness by selecting active stars that, for each star in the star-set there is at least one other star which is no further than $2\lambda - 1$ hop distance.

### 5.2 Hybrid StarCloak

The main difficulty in spatially bounded StarCloak is the choice of $\lambda$. At first sight, $\lambda$ can be determined by the query density of a general area. However, query density is often highly dynamic and changes street-by-street or star-by-star. Even neighboring segments may have different densities. Thus, the $\lambda$ determined based on query density of a general area may be undesirable for sparse sub-areas, and it is not possible to define an optimal compactness factor for each individual star at each time. To overcome this problem, we propose **hybrid StarCloak**, which leverages advantages from both basic StarCloak and spatially bounded StarCloak. In hybrid StarCloak, we first try to generate cloaking regions with spatially bounded StarCloak, and then for queries which could not be cloaked yet and are close to their expiration time, we apply basic StarCloak. We use a **consideration factor** denoted by $\alpha$ as the system parameter to decide when to apply basic StarCloak. Hybrid StarCloak periodically checks the expiration heap $H$ to see if any query is closer than $\alpha$ to their expiration time.

TABLE 1: Default parameter settings used in our experiments

| Parameter | $k$ | $\delta_k$ | $\delta_l$ | $\sigma_s$ | $\sigma_t$ | $\gamma$ | $\lambda$ | $\alpha$ |
|---|---|---|---|---|---|---|---|---|
| Mean | 5 | 5 | 5 | 4 | 10 | 20 | 1 | 2 |
| Deviation | 1 | 1.5 | 1.5 | 1 | 2 | 2 | 0 | 0 |

## 6 Experimental Evaluation

### 6.1 Experimental Setup

We performed extensive experiments to evaluate the performance, utility and attack-resilience of StarCloak using real-world road network datasets. We implemented StarCloak and its variants in Java. We deployed them on a Windows 7 computer with 4.00 GHz Intel CPU and 16GB memory. Then, we simulated the movements of 10,000 users on two real-world road network datasets using the Brinkhoff simulator[1], which is a popular and commonly used simulator in the literature. Considering that users can have different types of vehicles and driving characteristics, instead of having a single type of vehicle in the simulations, we defined two types of vehicles: slow vehicles (e.g., trucks) and fast vehicles (e.g., passenger cars). As users are moving on the road network, they send $k$-nearest neighbor ($k$-NN) queries to retrieve information of $k$ nearest points of interest to their current location. These queries need to be anonymized to protect privacy.

The two real-world road network datasets used in the experimentation are California[2] and Georgia[3]. California road network contains only highways with 21,693 edges and 21,048 nodes. 87,635 points of interest from 62 different classes (e.g., hospital, school, etc.) are associated with the road network. Georgia is the larger road network dataset, which contains primary and secondary roads with 430,849 edges and 428,708 nodes. Notice that the Georgia road network has substantially larger number of nodes and edges compared to California. By using two road network datasets that have substantially different size, we aimed to observe the effect of map density on the effectiveness of StarCloak.

Default values of the parameters used in our experiments are listed in Table 1. To mimic users' varying query times, privacy and utility constraints; instead of using fixed values for all queries, the values for each query are drawn independently from Gaussian distributions with mean and standard deviation listed in Table 1.

- $k$: As noted earlier, as users are moving on the road network, they send $k$-NN queries. Here, $k$ is the number of points of interest requested.
- $\delta_k$, $\delta_l$, $\sigma_s$, $\sigma_t$: Recall from Definition 6 that the combination of these parameters make up the query profile. $\delta_k$ is the $k$-user anonymity parameter, $\delta_l$ is the $l$-segment indistinguishability parameter, $\sigma_s$ is the spatial constraint, and $\sigma_t$ is the temporal constraint.
- $\gamma$: Each user waits for a certain amount of time before issuing his/her next query. $\gamma$ captures the waiting time.
- $\lambda$ is the compactness factor used in spatially bounded StarCloak.
- $\alpha$ is the consideration factor used in hybrid StarCloak.

### 6.2 Compared Approaches

In our evaluation, we compare multiple approaches. Random sampling and network expansion serve as two baseline anonymization approaches. XStar [5] is the most relevant system to StarCloak. We also include the three variants of StarCloak in our comparison: basic, spatially bounded, and hybrid.

**Random Sampling:** Given an incoming query with profile $(\delta_k^q, \delta_l^q, \sigma_s^q, \sigma_t^q)$, this approach iteratively samples segments randomly from the spatial region within $\sigma_s^q$ one-by-one, and adds them to the anonymized location. It terminates when $(\delta_k^q, \delta_l^q)$ privacy requirements are satisfied. Since random sampling picks segments randomly, it is highly attack-resilient by nature. When measuring attack-resilience it serves as an "upper bound", since we would not expect any other approach to beat random sampling in terms of attack-resilience. However, the random selection of segments hurts utility and query processing cost; therefore, random sampling is typically not ideal in practice.

**Network Expansion:** Network expansion-based approaches have been popularly used for query processing on road networks [50]. For incoming query $q$, this approach starts from the actual segment of the query and incrementally adds a neighboring segment using Dijkstra's deterministic network expansion algorithm. The order of expansion is based on the distance between $q$'s focal position and neighboring segments' midpoints. The approach terminates when $(\delta_k^q, \delta_l^q)$ privacy requirements are satisfied. Network expansion results in an anonymized location that is a densely connected, compact subgraph. Its advantage is low query processing cost. Its main weakness is vulnerability to attack since the expansion follows a *deterministic* best-first search, which can be inferred and reverse-engineered by an adversary who knows the road network structure.

**XStar:** The most related work to StarCloak in the literature is XStar [5], which performs road network anonymization under utility and privacy constraints. We include it in our comparison and show the superiority of StarCloak over XStar.

**StarCloak and Variants:** We denote by StarCloak the basic version of StarCloak. We also include its two variants, which are spatially bounded StarCloak and hybrid StarCloak, in our experimental comparison.

### 6.3 Evaluation Metrics

To evaluate the performance of different approaches, we use multiple metrics: success rate in anonymization, anonymization time, query processing time, size of candidate result set, successful throughput, and entropy against inference attacks.

**Success Rate:** An effective anonymization approach should be successful in anonymizing as many of the received queries as possible. We say that a query has been successfully anonymized if and only if: (i) an anonymized (cloaked) location was generated for it by the location anonymization engine, (ii) it was sent to the LBS provider with the anonymized location and its result was received back, (iii) final result was successfully delivered to the end user. Then, *Success Rate* is measured as:

$$\text{Success Rate} = \frac{\text{Number of successfully anonymized queries}}{\text{Total number of queries}}$$

**Successful Throughput:** We use this metric to evaluate the scalability of the anonymization approaches. Let *NQPS* denote the *Number of Queries Processed per Second* by an anonymization engine. Then, *Successful Throughput* is the multiplication of the previously defined *Success Rate* metric with *NQPS*:

$$\text{Successful Throughput} = \text{Success Rate} \times \text{NQPS}$$

**Anonymization Time:** When users issue queries, they want fast answers. However, an anonymization engine needs a certain

1. http://iapg.jade-hs.de/personen/brinkhoff/generator/
2. http://www.cs.utah.edu/~lifeifei/SpatialDataset.htm
3. https://www.census.gov/geo/maps-data/data/tiger-geodatabases.html

amount of time to perform the anonymization. The *Anonymization Time* metric measures the average time elapsed from the query issue time until successful anonymization. Let $q_1$, $q_2$, ..., $q_N$ denote successfully anonymized queries. Recall that $q_i.t$ denotes the timestamp at which query $q_i$ was issued by the user. Let $q_i.t^*$ denote the timestamp at which the anonymization engine completes anonymizing $q_i$. Then:

$$\text{Anonymization Time} = \frac{\sum_{i=1}^{N} (q_i.t - q_i.t^*)}{N}$$

**Query Processing Time:** This metric measures the query processing time at the LBS provider. Without anonymization (i.e., without any privacy protection), the LBS provider receives queries' exact locations and computes the response using the exact location. For query $q_i$, let $\varphi(q_i)$ denote the time it takes for the LBS provider to compute the response using $q_i$'s exact location. However, with anonymization, $q_i$ is sent to the LBS provider not with the exact location but with an anonymized location. Let $\psi(q_i)$ denote the time it takes for the LBS provider to compute the response using $q_i$'s anonymized location. Then:

$$\text{Query Processing Time} = \frac{\sum_{i=1}^{N} (\psi(q_i) - \varphi(q_i))}{N}$$

**Candidate Result Size:** Recall from Figure 1 that the LBS provider sends a *candidate result* to the location anonymization engine after processing a query. As explained at the beginning of Section 3, due to the locations being anonymized, the candidate result may contain false positives. To measure the impact of this, the *Candidate Result Size* metric computes the cardinality of the candidate result. Larger the cardinality, higher the network communication cost.

**Entropy:** We use entropy as a quantitative measure of adversarial uncertainty achieved by anonymization, where higher entropy means higher attack-resilience. Given an anonymized location $S$ for user $u$ as a subgraph consisting of multiple segments, the entropy of $S$ can be calculated by:

$$H(S) = -\sum_{s \in S} \mathit{link}[u \leftarrow s] \cdot \log_2(\mathit{link}[u \leftarrow s])$$

Here, linkability (*link*) is calculated as explained in the attack models (Section 3.3). Since the size of $S$ can change, we normalize $H(S)$ similar to [51], and report normalized entropies calculated as: $H(S)/\log_2(|S|)$.

## 6.4 Experiment Results

**Results on Success Rate:** Figure 5 shows the percentage success rates of compared approaches with respect to varying $\delta_k$, $\delta_l$, $\sigma_s$, and $\sigma_t$ for California and Georgia maps. Generally, StarCloak and its optimized variants have high success rates because of their ability in handling different user requirements effectively. Results show that increasing the privacy requirements often decreases success rate, but the effects are different for different maps and different privacy requirements. For example, when $\delta_k$ increases, success rate decreases faster on the Georgia compared to California. The reason for this is the query density of the maps. Keeping the number of queries constant across maps, since the Georgia map is more detailed than California, the distribution of query density on Georgia is sparser. Thus, on Georgia, the chance of finding enough queries to cloak together under the same spatial constraint is smaller, causing lowered success rate. Yet, for XStar, increasing $\delta_l$ impacts success rate on California more than Georgia, unlike StarCloak. This is because XStar anonymizes queries on the same star together, whereas in StarCloak if there is a conflict between two queries' $l$-segment indistinguishability and $\sigma_s$ spatial tolerance, they are cloaked on different vertices of the cloaking graph. This allows StarCloak to maintain high success rates despite increasing $\delta_l$.

Comparing the three StarCloak variants, the basic version achieves highest success rate, followed by hybrid StarCloak and then spatially bounded StarCloak. This is expected because spatially bounded StarCloak aims at finding compact cloak regions, whereas basic StarCloak allows suboptimal regions for higher success rate. Hybrid StarCloak achieves a trade-off between success rate and compactness of a cloak region. The rightmost two graphs within Figure 5 display the impact of changing spatial and temporal tolerance constraints on success rate. When users have higher tolerance, their queries are anonymized with higher success rate. We see clearly that lower spatial tolerance affects XStar's success rate negatively far more than it affects StarCloak, once again showing StarCloak's superiority.

**Results on Successful Throughput:** The throughputs of compared approaches with respect to varying $\delta_k$, $\delta_l$, $\sigma_s$ and $\sigma_t$ are shown in Figure 6 for California and Georgia maps. We observe that throughputs of the baseline approaches (random sampling and network expansion) are often significantly lower than XStar and StarCloak. While XStar and StarCloak maintain high throughput despite increasing $\delta_k$ (stricter privacy), the throughput of XStar drops when $\delta_l$ is increased. Hence, we find that StarCloak is much more capable of satisfying challenging $l$-segment indistinguishability requirements than other approaches.

With respect to varying spatial and temporal tolerances, we observe that StarCloak variants are capable of handling a variety of tolerance values without significant degradation in throughput. In contrast, the throughputs of the baseline approaches are often 5-6 times smaller than StarCloak. Furthermore, while XStar's throughput is comparable to StarCloak when $\sigma_s$ is high, XStar may perform even worse than the baselines for small $\sigma_s$ (see Figure 6). Collectively, these results show the superiority of StarCloak and its variants in query service and scalability compared to both XStar and baseline approaches, under varying privacy and utility settings.

**Results on Anonymization Time:** In Figure 7, we report the average anonymization times for the California map. Results show that StarCloak variants have significantly better anonymization time than compared approaches under various settings. XStar often has the highest anonymization time. Among StarCloak variants, hybrid StarCloak and spatially bounded StarCloak are similar, whereas basic StarCloak has lowest anonymization time. This is because basic StarCloak has no preference towards "waiting for a better opportunity" to generate cloaked regions for incoming queries, whereas the other two variants can wait closer until the query expiration time before anonymization.

**Results on Query Processing Time:** The results are reported in Figure 8. Since each compared anonymization approach may have different success rate, in order to ensure a fair comparison, we pick the same number of anonymized locations across all approaches in this set of experiments and those experiments reported for the next metric. First, we observe that StarCloak's results are often significantly better than its main competitor XStar. Second, among the three StarCloak variants, basic Star-

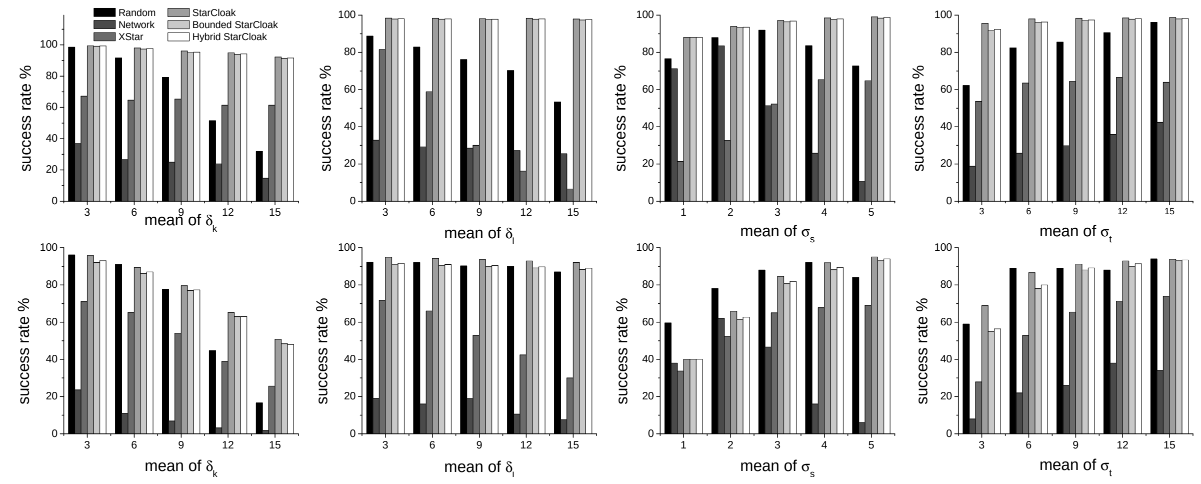

Fig. 5: Success rate for California map (four graphs in top row) and Georgia map (four graphs in bottom row)

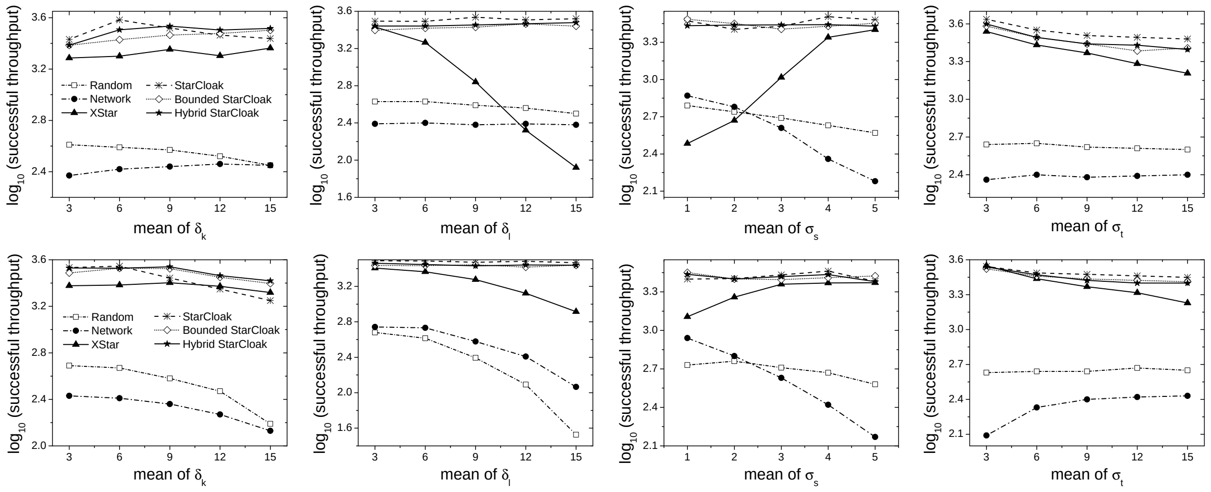

Fig. 6: Successful throughput for California map (four graphs in top row) and Georgia map (four graphs in bottom row)

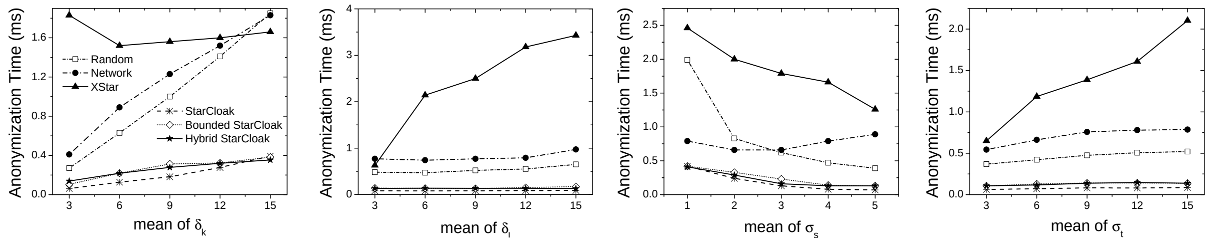

Fig. 7: Anonymization time for California map

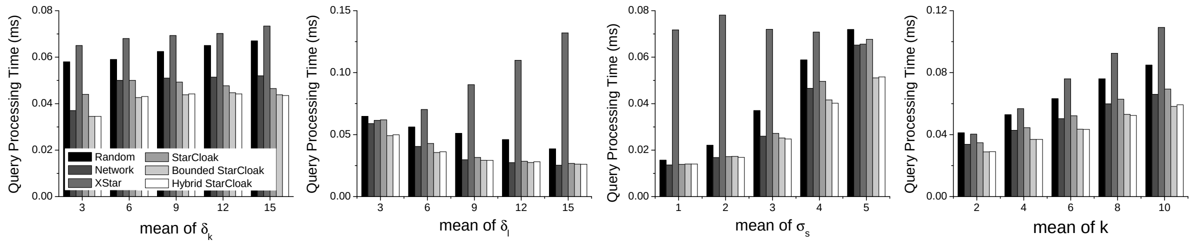

Fig. 8: Query processing time for California map

CLOAK has highest query processing time, whereas the hybrid and spatially bounded versions have similar processing time, because of their more compact cloaked regions. The improvement of spatially bounded STARCLOAK becomes significant particularly when $\sigma_s$ is increased. These findings confirm that cloaking regions with scattered segments cause higher query processing time.

**Results on Candidate Result Size:** We measure the candidate result set size under varying $\delta_k$, $\delta_l$, $\sigma_s$, and $k$ parameters, and report the results in Figure 9. Spatially bounded and hybrid STARCLOAK often provide the best results due to their compact output cloak regions. STARCLOAK's competitors are comparable when the $\delta_k$, $\delta_l$ privacy requirements are relaxed, but as we make

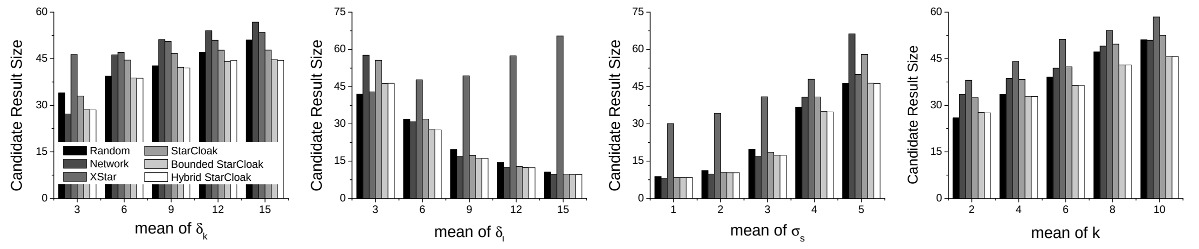

Fig. 9: Candidate result size for California map

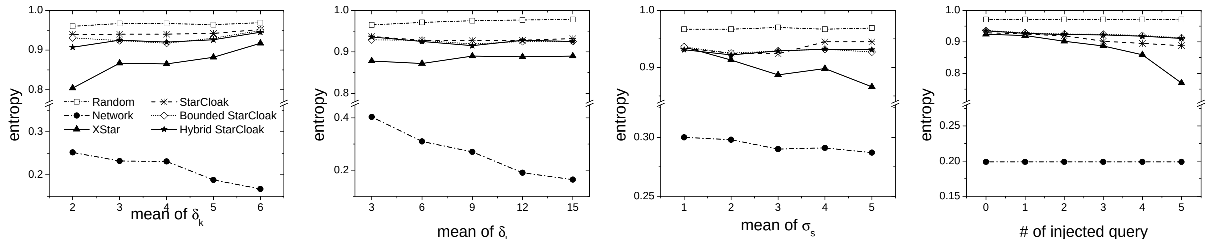

Fig. 10: Average entropy for California map

the privacy requirements stricter, the bandwidth cost of XStar in particular becomes significantly large. The increase in candidate result size caused by large $\sigma_s$ can be explained by the fact that relaxed spatial tolerance inevitably causes the StarCloak approaches to be more relaxed regarding the compactness of the output cloak regions, thus result sets are also more scattered and diverse. The increase in candidate result size due to increased $k$ is expected, since $k$ is the parameter controlling the number of nearest neighbors returned by the $k$-NN query. Naturally, with higher $k$, more candidates have to be returned, hence the candidate result set has larger size.

**Results on Entropy (Attack-Resilience):** We use the normalized entropy metric to measure attack-resilience, with higher entropy meaning higher attack-resilience. The results are shown in Figure 10. In this set of experiments, it is expected that random sampling will yield highest entropy, whereas network expansion will yield lowest entropy. The results in Figure 10 confirm these expectations, and show that the entropy of StarCloak variants and XStar are between random sampling and network expansion. An important note is that StarCloak has higher entropy than XStar. Furthermore, StarCloak's entropy values are similar to random sampling, showing that it achieves near-optimal attack resilience. As $\delta_k$ increases, since more users are cloaked together, entropy increases. The increase in entropy is more clear for spatially bounded StarCloak and hybrid StarCloak compared to basic StarCloak, as their output cloak regions are more compact (focused on the users' actual locations) with small $\delta_k$ in the first place. In the rightmost graph in Figure 10, we show the impact of the number of injected queries on entropy in the query injection attack. In XStar and StarCloak, while more injections generally cause a more successful attack, the vulnerability of XStar becomes significantly higher than StarCloak when 4 or more queries are injected. Unlike XStar and StarCloak, random sampling and network expansion do not consider nearby queries' locations during cloaking, thus their entropy remains unaffected by query injections.

## 7 Conclusion

In this paper, we designed and developed StarCloak, a location privacy protection service for mobile users. StarCloak has an array of desirable features, including utility-aware and personalized privacy protection, cost-aware star selection, and randomized star-set pruning for improved attack-resilience. The two optimized variants of StarCloak, namely spatially bounded StarCloak and hybrid StarCloak, improve network bandwidth usage and query processing time, with small sacrifice in success rate, throughput, and anonymization time. In comparison to XStar, StarCloak achieves reduced query processing and anonymization time, higher success rate in anonymization, and higher resilience against replay and query injection attacks.

## References

[1] K. Lee, L. Liu, B. Palanisamy, and E. Yigitoglu, "Road network-aware spatial alarms," *IEEE Transactions on Mobile Computing*, vol. 15, no. 1, pp. 188–201, 2016.

[2] X. Miao, Y. Gao, G. Mai, G. Chen, and Q. Li, "On efficiently monitoring continuous aggregate k nearest neighbors in road networks," *IEEE Transactions on Mobile Computing*, 2019.

[3] X. Miao, Y. Gao, S. Guo, and G. Chen, "On efficiently answering why-not range-based skyline queries in road networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1697–1711, 2018.

[4] S. Luo, B. Kao, G. Li, J. Hu, R. Cheng, and Y. Zheng, "Toain: a throughput optimizing adaptive index for answering dynamic k nn queries on road networks," *Proceedings of the VLDB Endowment*, vol. 11, no. 5, pp. 594–606, 2018.

[5] T. Wang and L. Liu, "Privacy-aware mobile services over road networks," *VLDB*, vol. 2, pp. 1042–1053, 2009.

[6] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*. ACM, 2003, pp. 31–42.

[7] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 1, pp. 1–36, 2021.

[8] G. Zhong and U. Hengartner, "A distributed k-anonymity protocol for location privacy," in *IEEE International Conference on Pervasive Computing and Communications*. IEEE, 2009, pp. 1–10.

[9] J. Li, H. Yan, Z. Liu, X. Chen, X. Huang, and D. S. Wong, "Location-sharing systems with enhanced privacy in mobile online social networks," *IEEE Systems Journal*, vol. 11, no. 2, pp. 439–448, 2015.

[10] C.-Y. Chow, M. F. Mokbel, and X. Liu, "Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments," *GeoInformatica*, vol. 15, no. 2, pp. 351–380, 2011.

[11] R. Gupta and U. P. Rao, "Achieving location privacy through cast in location based services," *Journal of Communications and Networks*, vol. 19, no. 3, pp. 239–249, 2017.

[12] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *25th IEEE International Conference on Distributed Computing Systems*. IEEE, 2005, pp. 620–629.

[13] B. Bamba, L. Liu, P. Pesti, and T. Wang, "Supporting anonymous location queries in mobile environments with privacygrid," in *Proceedings of the 17th International Conference on World Wide Web*, 2008, pp. 237–246.

[14] R.-H. Hwang, Y.-L. Hsueh, and H.-W. Chung, "A novel time-obfuscated algorithm for trajectory privacy protection," *IEEE Transactions on Services Computing*, vol. 7, no. 2, pp. 126–139, 2013.

[15] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2013, pp. 901–914.

[16] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 251–262.

[17] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux, and J.-Y. Le Boudec, "Protecting location privacy: optimal strategy against localization attacks," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. ACM, 2012, pp. 617–627.

[18] R. Shokri, "Privacy games: Optimal user-centric data obfuscation," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 2, pp. 299–315, 2015.

[19] L. Yu, L. Liu, and C. Pu, "Dynamic differential location privacy with personalized error bounds," in *Network and Distributed System Security Symposium (NDSS)*, 2017.

[20] B. Niu, Y. Chen, Z. Wang, F. Li, B. Wang, and H. Li, "Eclipse: Preserving differential location privacy against long-term observation attacks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 1, pp. 125–138, 2020.

[21] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *21st International Conference on Data Engineering Workshops (ICDEW'05)*.

[22] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," *International Conference on Pervasive Services*, pp. 88–97, 2005.

[23] H. Lu, C. S. Jensen, and M. L. Yiu, "Pad: privacy-area aware, dummy-based location privacy in mobile services," in *Proceedings of the Seventh ACM International Workshop on Data Engineering for Wireless and Mobile Access*, 2008, pp. 16–23.

[24] H. J. Do, Y.-S. Jeong, H.-J. Choi, and K. Kim, "Another dummy generation technique in location-based services," in *2016 International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2016.

[25] H. Liu, X. Li, H. Li, J. Ma, and X. Ma, "Spatiotemporal correlation-aware dummy-based privacy protection scheme for location-based services," in *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2017, pp. 1–9.

[26] T. Hara, A. Suzuki, M. Iwata, Y. Arase, and X. Xie, "Dummy-based user location anonymization under real-world constraints," *IEEE Access*, vol. 4, pp. 673–687, 2016.

[27] G. Sun, S. Cai, H. Yu, S. Maharjan, V. Chang, X. Du, and M. Guizani, "Location privacy preservation for mobile users in location-based services," *IEEE Access*, vol. 7, pp. 87 425–87 438, 2019.

[28] A. Gutscher, "Coordinate transformation-a solution for the privacy problem of location based services?" in *Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium*, 2006.

[29] A. Khoshgozaran, H. Shirani-Mehr, and C. Shahabi, "Blind evaluation of location based queries using space transformation to preserve location privacy," *GeoInformatica*, vol. 17, no. 4, pp. 599–634, 2013.

[30] R. Gupta and U. P. Rao, "Vic-pro: vicinity protection by concealing location coordinates using geometrical transformations in location based services," *Wireless Personal Communications*, vol. 107, no. 2, pp. 1041–1059, 2019.

[31] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner, "Android permissions: User attention, comprehension, and behavior," in *Proceedings of the 8th Symposium on Usable Privacy and Security*. ACM, 2012, p. 3.

[32] P. G. Kelley, L. F. Cranor, and N. Sadeh, "Privacy as part of the app decision-making process," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2013, pp. 3393–3402.

[33] B. Liu, J. Lin, and N. Sadeh, "Reconciling mobile app privacy and usability on smartphones: Could user privacy profiles help?" in *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 2014.

[34] K. Olejnik, I. I. Dacosta Petrocelli, J. C. Soares Machado, K. Huguenin, M. E. Khan, and J.-P. Hubaux, "Smarper: Context-aware and automatic runtime-permissions for mobile devices," in *Proceedings of the 38th IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017.

[35] E. Yigitoglu, M. E. Gursoy, L. Liu, M. Loper, B. Bamba, and K. Lee, "Privacyzone: A novel approach to protecting location privacy of mobile users," in *IEEE International Conference on Big Data*. IEEE, 2018.

[36] B. Palanisamy and L. Liu, "Mobimix: Protecting location privacy with mix-zones over road networks," in *27th IEEE International Conference on Data Engineering (ICDE)*, 2011.

[37] B. Palanisamy and L. Liu, "Effective mix-zone anonymization techniques for mobile travelers," *Geoinformatica*, vol. 18, no. 1, pp. 135–164, 2014.

[38] L. Zhang, "Otibaagka: a new security tool for cryptographic mix-zone establishment in vehicular ad hoc networks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 2998–3010, 2017.

[39] C. Vaas, M. Khodaei, P. Papadimitratos, and I. Martinovic, "Nowhere to hide? mix-zones for private pseudonym change using chaff vehicles," in *2018 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2018.

[40] K. Mouratidis and M. L. Yiu, "Anonymous query processing in road networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 1, pp. 2–15, 2009.

[41] C.-Y. Chow, M. F. Mokbel, J. Bao, and X. Liu, "Query-aware location anonymization for road networks," *GeoInformatica*, vol. 15, no. 3, pp. 571–607, 2011.

[42] C. Li and B. Palanisamy, "Reversecloak: Protecting multi-level location privacy over road networks," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 673–682.

[43] K.-T. Yang, G.-M. Chiu, H.-J. Lyu, D.-J. Huang, and W.-C. Teng, "Path privacy protection in continuous location-based services over road networks," in *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2012, pp. 435–442.

[44] I. Memon and Q. A. Arain, "Dynamic path privacy protection framework for continuous query service over road networks," *World Wide Web*, vol. 20, no. 4, pp. 639–672, 2017.

[45] E. Yigitoglu, M. L. Damiani, O. Abul, and C. Silvestri, "Privacy-preserving sharing of sensitive semantic locations under road-network constraints," in *2012 IEEE 13th International Conference on Mobile Data Management*. IEEE, 2012, pp. 186–195.

[46] Y. Li, Y. Yuan, G. Wang, L. Chen, and J. Li, "Semantic-aware location privacy preservation on road networks," in *International Conference on Database Systems for Advanced Applications*. Springer, 2016.

[47] C. Qiu, A. C. Squicciarini, C. Pang, N. Wang, and B. Wu, "Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability," *IEEE Transactions on Mobile Computing*, 2020.

[48] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, pp. 1–18, 2008.

[49] R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei, "Minimality attack in privacy preserving data publishing," in *Proceedings of the 33rd International Conference on Very Large Data Bases*, 2007, pp. 543–554.

[50] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query processing in spatial network databases," in *VLDB*, 2003, pp. 802–813.

[51] C. Díaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *PET*, 2003, pp. 54–68.

**Emre Yigitoglu** received his PhD in Computer Science from Georgia Tech, MS in Computer Engineering from TOBB University of Economics and Technology, and BS in Computer Science from Hacettepe University. His research interests lie in data-intensive distributed computing systems, mobile computing, and location-based services.

**M. Emre Gursoy** is an Assistant Professor of Computer Engineering at Koc University, Turkey. He received his PhD in Computer Science from Georgia Tech, his MS in Computer Science from UCLA, and his BS in Computer Science and Engineering from Sabanci University. His research interests include privacy, security, adversarial machine learning, Internet of Things, and big data analytics. He is the recipient of a Best Paper Award from EdgeSys in 2020 and a CAREER grant from TUBITAK (Turkish National Science Foundation) in 2021.

**Ling Liu** is a full Professor in the School of Computer Science at Georgia Tech. She is an elected IEEE Fellow, a recipient of the IEEE Computer Society Technical Achievement award in 2012, and a recipient of the best paper award from a dozen of top venues. She served as the general chair and PC chair of numerous IEEE and ACM conferences, as well as on the editorial boards of over a dozen international journals. Currently, she is the Editor-in-Chief of ACM Transactions on Internet Technology.