Location-based Services Routing in Networks

[1]M.Krishnamoorthy, [2]P.Ramyashri, [3]M.S. Srinidhi
[1]Assitant Professor, Department of M.C.A, Panimalar Engineering College, Chennai
[2]PG Scholar, Department of M.C.A, Panimalar Engineering College, Chennai
[3]Faculty, Department of M.C.A, Panimalar Engineering College, Chennai

1 ABSTRACT

Due to the nature of spatial queries, an LBS needs the user position in order to pro- cess her requests. On the other hand, revealing exact user locations to a (potentially untrusted) LBS may pinpoint their identities and breach their privacy. To address this issue, spatial anonymity techniques obfuscate user locations, forwarding to the LBS a sufficiently large region instead. Existing methods explicitly target processing in the Euclidean space, and do not apply when proximity to the users is defined according to network distance (e.g., driving time through the roads of a city). In this paper, we propose a framework for location based services. We design location obfuscation techniques that (i) provide anonymous LBS access to the users, and (ii) allow efficient query processing at the LBS side. Our techniques exploit existing network database infrastructure, requiring no specialized storage schemes or functionalities. We experimentally compare alternative designs in real road networks and demonstrate the effectiveness of our techniques. We will find out the best location based route.

2. INTRODUCTION

An LBS makes spatial data available to the users through one or more location servers (LS) that index and answer user queries on them. Examples of spatial queries could be "Where is the closest hospital to

There exist many algorithms for efficient spatial query processing, but the main challenge in the LBS industry is of a different nature. In particular, users are reluctant to use LBSs, since revealing their position may link to their identity. Even though a user may create a fake ID to interests and shopping needs, resulting in a flood of unsolicited advertisements through e-coupons and personal messages.

The latter obfuscates her location, replacing it with an anonymizing spatial region (ASR) that encloses u. The ASR is then forwarded to the LS. Ignoring where exactly u is, the LS retrieves (and reports to the AZ) a candidate set (CS) that is guaranteed to contain the query results for any possible user location inside the ASR. The AZ receives the CS and reports to u the subset of candidates that corresponds to her original query. In order for the AZ to produce valid ASRs, the users send location updates whenever they move (through their secure connection). The described model is shown in Figure 1.

An LBS makes spatial data available to the users through one or more location servers (LS) that index and answer user queries on them. Examples of spatial queries could be "Where is the closest hospital to my current location?" or "Which pharmacies are open within a 1 km radius?". In order for the LS to be able to answer such questions, it needs to know the position of the querying user.

Our contributions can be summarized as follows:

*We propose an edge ordering anonymization approach for users in road networks, which guarantees K -anonymity under the strict reciprocity requirement (described later).

* We identify the crucial concept of border nodes, an important indicator of the CS size and of the query processing cost at the LS.

* We consider various edge orderings, and qualitatively assess their query performance based on border nodes.

possible by various means, such as publicly available information (e.g., city maps and telephone directories),.

access the service, her location alone may disclose her actual identity. Linking a position to an individual is

3 RELATED WORK

3.1 Location based Services in Networks

Basic Notation. In general, a road network can be modeled as a weighted graph $G = (N, E)$. N contains the network nodes, while E is the set of edges. Nodes n in N model road intersections, locations of road turns, or positions where traffic conditions change (e.g., a street gets narrower). On the other hand, every edge e connects two nodes and is associated with a non-negative weight $w(e)$. Weight $w(e)$ may represent, for instance, the traveling time from one node to the other. Figure 2 shows an example of a road network.

Query Processing by Network Expansion. Users are often interested in location-based queries such as r-range and kNN queries, in the context of a road network. Given a distance threshold r and a user location u, the r-range query returns all objects within (network) distance r from u. On the other hand, the kNN query retrieves the k objects that are closest to u. In the rest of the paper, the term distance refers to the network distance, and the r-range and kNN queries refer to their network versions (unless otherwise specified).

Network expansion [30] is a well-known technique for evaluating r-range and kNN queries. Starting from the user location u, it discovers objects on encountered edges while traversing the network like Dijkstra's algorithm, until the query results (i.e., data objects of interest) are found. Suppose that, in Figure 2, user u issues a range query with $r = 9$. First, we access the adjacency index to identify edges within the query range, following the steps in Table 1. A min-heap H is employed for organizing entries of the form $(n_i, d_N(u, n_i))$ (for encountered nodes n_i) in ascending order of distance $d_N(u, n_i)$. In our example, the edge $n_2 n_3$ containing u is initially identified, and its end-nodes n_2 and n_3 (both with distance 2) are inserted into H. In each iteration, the node n_i with the minimum distance is de-heaped from H, its incident edges $n_i n_j$ are recorded, and its adjacent unvisited nodes n_j (having $d_N(u, n_j)$ within the range) are inserted into H.

3.2 Location-based Queries

Recently, considerable research interest has focused on pre- venting identity inference in location-based services. Stud- ies in this area [17], [14], [28], [23] typically assume the model described in Section 1, proposing spatial cloaking (i.e., location obfuscation) techniques. In the following, we describe existing techniques for ASR computation (at the AZ) and query processing (at the LS). At the end, we cover alternative location privacy approaches and discuss why they are inappropriate to our problem setting.

Spatial Cloaking at the AZ. In general, the AZ applies the concept of K-anonymity [33] to hide the querying user loca- tion u. The idea is to compute an anonymizing spatial region (ASR), containing u and at least $K - 1$ other user locations. This offers privacy protection in the sense that the actual user position u cannot be distinguished from others in the ASR, even when a malicious LS is equipped/advanced enough to possess all user locations. This spatial K-anonymity model is most widely used in location privacy research/applications, even though alternative models are emerging.

Casper [28] is the first work on efficient and scalable AZ implementation for ASR computation. A quad-tree is utilized for indexing user locations and deriving ASRs. Suppose that the AZ needs to compute a 2-anonymous region (i.e., $K = 2$) for querying user u_1 in Figure 3(a). The AZ first locates the leaf quad that contains u_1 and traverses the tree upwards until it identifies a region covering at least K users (including u_1). In this case, the AZ derives rectangle $R_{1,2,3}$ (containing three users) as the 2-anonymous region of u_1.

Alternative Location Privacy Approaches. There exist location privacy approaches other than spatial anonymity. For instance, [20], [15] apply private information retrieval to process nearest neighbor queries. Based on cryptographic tech- niques, they guarantee that an adversary cannot infer the user's location u within polynomial time of a security parameter (e.g., key length). [20] is theoretical in nature, whereas [15] proposes a practical algorithm with $O(\sqrt{n})$ communication cost. However, query processing is particularly slow, and the method is inapplicable to range and kNN queries (with $k > 1$).

4 PROBLEMDEFINITION

We propose an location based services framework, targeted at road network databases. We adopt the trusted anonymizer model (i.e., the use of the AZ as a mediator between users and the LS), as illustrated in Figure 1. This choice is due to

(i) the proliferation of this model in existing

We assume that the users $u \in U$ and the data objects $o \in O$ lie/move in a road network, as described in Section 2.1. We consider the generic network distance definition where the edge weights are non-negative and they do not have to be proportional to the Euclidean length of the corresponding road segment. For instance, they could be traveling times from endpoint to endpoint (i.e., depending on each road's congestion level, speed limit, number of lanes, etc). To simplify our discussion, we consider undirected (i.e., two-way) edges, but our techniques apply with trivial modifications to directed ones (e.g., one-way roads). We focus on static networks, with fixed edge weights. However, edge updates (e.g., weight/traveling time increase due to an accident) can be dealt with easily, provided that the AZ and the LS are informed of the network changes.

In this paper, we focus on two important location-based queries on road networks; the r-range and the kNN query described previously. For the latter type, we denote by $kNN\ dist(u)$ the network distance of the k-th NN. We consider snapshot queries (i.e., queries that are evaluated once and terminate), as opposed to queries that require constant update of their results (e.g., [9]).

5 NETWORK-BASED ANONYMIZATION

In this section, we present the cloaking algorithm of our NAP framework. Our primary objective is to guarantee reciprocity- based anonymity. In NAP, the AZ anonymizes u with a set of line segments/edges instead of a spatial region (ASR).

The crux of our cloaking method is to utilize a global edge ordering; i.e., an ordered sequence that contains all network edges exactly once. The edge ordering is setting-sensitive, i.e., it specifies which end-node of the edge precedes the other. We refer to the position and setting of an edge in the ordering as the edge order and the edge setting, respectively. To avoid confusion, the setting of an edge depends solely on the ordered sequence, and has nothing to do with the direction (in the case of directed networks) of the road segment it models. Figure 5(a) shows a road network, and an ordering of its edges. The number next to each edge indicates its order and the arrow its setting.

We follow the above global edge/user ordering approach, because a local one could lead to privacy breach in a way similar to Figure 3(a) for Casper. Specifically, upon intercep- tion of an AEL generated by our method, one cannot infer who among the users in the corresponding bucket was the querying one. In other words, the AEL for any user in the same bucket is identical, and therefore an adversary cannot pinpoint the query originator with a probability higher than $1/K$ (recall that each bucket contains K or more users). Hence, our cloaking method satisfies reciprocity. Reciprocity, in turn, is a sufficient condition for anonymity [23] and, thus, NAP guarantees K - anonymity to the querying users.

In the rest of this section, we describe edge ordering strate- gies (Section 4.1). Then, we present particular techniques for the anonymization procedure (Section 4.2). Finally, we analyze the properties of the proposed edge orderings

5.1 Border Nodes and Edge Ordering

To ensure reciprocity we can use any global edge ordering; e.g., we could use a random permutation of network edges. both of which are inside the AEL. Hence, n_8 is not a border node.

Border nodes have an important property. It follows from Definition 1 that any shortest path from a point inside the AEL to some object outside the AEL passes through some border node. Based on this property, Theorem 1 below constitutes the foundation of CS computation in NAP.

Theorem 1: Assume that the LS receives an AEL for a kNN (r-range) query with parameter k (r, respectively). An inclusive and minimal CS is formed as the union of

- the objects that fall on the AEL edges
- the kNN objects of all border nodes (the objects within distance r from any border node, respectively).

Proof: We focus on the kNN case, as the proof for r- range queries follows the same lines. We prove inclusiveness by contradiction. Suppose that there is some object o among the k NNs of u that is not in the CS. The CS includes all objects on AEL edges, so o must fall outside the AEL. User u lies inside the AEL and, thus, the shortest path from u to o passes through some border node n; i.e., it holds that $d_N(u, o) = d_N(u, n) + d_N(n, o)$ (A). The CS includes the k NNs of n, so (i) o must lie further from n than its k-th NN, i.e., $d_N(n, o) > kNN\ dist(n)$ (B), and (ii) there are at least k objects o^0 (where $o^0 = o$) within distance $d_N(u, n) + kNN\ dist(n)$ from u in the CS, i.e., $kNN\ dist(u) \leq d_N(u, n) + kNN\ dist(n)$ (C). By adding factor $d_N(u, n)$ to both sides of inequality (B), we derive $d_N(u, n) + d_N(n, o) > d_N(u, n) + kNN\ dist(n)$, which ac- cording to (A) and (C) (applied to the left and right side of the inequality, respectively) leads to

d_N (u, o) > kN N dist(u). The latter contradicts the original hypothesis that o is one of the k NNs of u. Thus, CS is inclusive.
From Theorem 1 it follows that the CS size and the (AEL- based) processing cost at the LS increase with the number of border nodes in the AEL. To compute an optimal ordering (i.e., an ordering that achieves the minimum number of border nodes for any user) is impossible, since the same ordering must serve different user-specific anonymity degrees K . Therefore, we use heuristic-based algorithms.[1]

Random orderings. We describe two naïve orderings, RE and RN, used as baseline approaches. RE (random edge ordering) is a random permutation of the network edges; their settings are also decided at random. On the other hand, RN (random node ordering) first forms a random permutation of network nodes. Then, it scans the permutation and for each considered node n_i , it includes its incident edges $n_i n_j$ (with setting from n_i to n_j) into the produced ordering; previously encountered edges are ignored to avoid duplicates.

Hilbert-based orderings. The rationale behind this approach is that nodes/edges close in Euclidean space are likely to be close in terms of network distance. In turn, this means that consecutive edges in the ordering are likely to share end- nodes. HE (Hilbert edge ordering) and HN (Hilbert node ordering), work in the same way as RE and RN, respectively, the difference being that edges and nodes are considered in Hilbert order (in HE, the Hilbert values of the edge midpoints are used as their sorting keys). We establish the convention that the settings are from left to right and, in case of vertical edges, from down to up. Intuitively, HE/HN are adaptations of the HilbASR (Euclidean) cloaking to road networks[2] . Hilbert-based orderings (as opposed to the previous two categories) are inapplicable to environments where the Cartesian coordinates of nodes/users are unknown or undefined (see Section 5.3).
Edge ordering is performed once when the AZ is set up for the first time. Upon termination, the computed orders/settings

5.2 Anonymization Procedure

Given an edge ordering, the next question is how AEL com- putation can be implemented efficiently at the AZ. Parameter K is not known in advance and varies, since different users have different anonymity requirements, and even queries by the same user may specify different K , depending on the nature of the queried data. As buckets are defined according to K , they cannot be explicitly materialized. Instead, the AZ employs an index that keeps the users sorted on their order and allows efficient AEL computation for arbitrary K . The index is an aggregate B-tree (similar to an aggregate R-tree [29]), whose internal nodes keep for each child the number of users in the corresponding sub-tree. Figure 6 shows this tree in the example of Figure 5(a). For each user (e.g., u_6) we store the ID of the edge it falls on ($n_9 n_2$), the edge's order (7), and its distance from the edge's first end-node ($|n_9 u_6|$). The latter two values are used (primarily the edge order and secondarily the distance from the first end-node) as the sorting key of the tree. In Figure 6 the numbers in the shaded boxes correspond to the aggregate information maintained, i.e., the cardinalities of the sub-trees rooted thereof. Note that we use a B-tree instead of a B^+-tree (i.e., user information is also stored in internal nodes), because it is faster for in-memory indexing [27].

5.3 Analysis of Edge Orderings

The number of border nodes is an important indicator of the CS size and of the LS processing cost. To provide an insight on the behavior of our proposed edge orderings, we analyze their numbers of border nodes in a simple road network. A typical road network branches from the city center (e.g., the root) and exhibits a self-similar structure. We decompose the network into junctions and road sections. The junctions are nodes of degree higher than 2. The sections are paths (i.e., sets of connected edges) between two consecutive junctions that pass strictly through non-junction nodes. We treat the decomposed network as a tree with parameters: the fanout f , and the average number of edges per road section l. Figure
7 depicts a road network with $f = 2$ and $l = 2$.
An edge is said to be at level m if its path to the root passes through $m - 1$ junctions; there are $l \cdot f^m$ edges at level m. To simplify our analysis, we assume that there is exactly one user per edge (the general case is obtained by scaling K accordingly). Thus, the AEL of a query contains K edges for any edge ordering.

6.1 Single Query Processing

Range query optimizations. If the query type is r-range, border node expansion (in step 4) does not need to proceed to AEL edges, because the corresponding objects are either on some AEL edge (and, thus, retrieved by step 2) or, if they are outside the AEL, they are discovered by the expansion of another border node. An additional optimization particular to the range query type is to combine steps 2 and 4 so that CS objects are collected by querying the ORT only once. Specifically, after step 1, we expand the network (using only the adjacency index) for every border node up to distance r, and then query the ORT to retrieve the objects that fall in some of the acquired edges or inside the AEL. An additional optimization is that when the expansion of a border node n visits (i.e., de-heaps) a previously expanded node n^0, then expansion needs not proceed to (i.e., en-heap) the adjacent nodes of n^0, since the objects reachable through n^0 are inserted into the CS by n^0's expansion.

6.2 Batch Query Processing

In general, the LS processes queries in discrete timestamps, and multiple AEL-based queries may be arriving in the same timestamp. In this case, the queries are evaluated in a batch.

Rule 1. Among queries in Q_r (Q_{NN}) only the one with the largest parameter r_{max} (k_{max}, respectively) needs to be processed, since its result is by definition a superset of that for any other query in Q_r (Q_{NN}).

Rule 2. The r_{max} range query is evaluated before the k_{max}NN one (recall that range search is simpler/faster than kNN). Let S_r be the set of objects retrieved by the r_{max} range search and $|S_r|$ be their number. If $|S_r| \geq k_{max}$, then no further processing is necessary for Q_{NN}, as S_r is a superset of any kNN result for Q_{NN}. Otherwise (i.e., $|S_r| < k_{max}$), we need to additionally retrieve $k_{max} - |S_r|$ more NNs; the expansion continues from the point where the r_{max} range search stopped, starting with S_r as the set of NNs found so far and reusing the search heap of the range search.

6.3 Alternative Storage Schemes

NAP processing can be applied to any network storage scheme, as long as it provides the weights and the connectivity of the edges, and allows retrieval of the objects that fall on a given edge. To exemplify this, we use as a case study an alternative, widely used storage scheme [36]

7 EXPERIMENTAL EVALUATION

In this section, we evaluate the robustness and scalability of our proposed methods on a real road network. Our algorithms were implemented in C++ and experiments were executed on a Pentium D 2.8GHz PC. We measured the average of the following performance values over a query workload of 100 queries: (i) anonymization time and refinement time at the anonymizer AZ, (ii) I/O time and CPU time for query pro- cessing at the location server LS, and (iii) the communication cost (in terms of transmitted points) for the anonymizing edge list AEL and the candidate set CS. Note that each edge in AEL is counted as two points.

7.1 Experiment Setup

We use the real road network of San Francisco, obtained from [7]. By default, our experiments use a subnetwork with 50000 edges. Weights of the edges are set to their lengths. We generate $|U|$ users and $|O|$ objects. The locations of users and objects follow either uniform distribution (by default) or Gaussian distribution[4]. At the LS, the disk page is 4KBytes and each index structure (edge R-tree, ORT, etc) is associated with a memory buffer with capacity set to 5% of its disk size. Table 3 summarizes the investigated parameters and their examined values, with the defaults shown bold. Parameter r is expressed as a multiple of the average edge weight.

7.2 Robustness Experiments

In this section, we illustrate the achieved anonymity and study the performance of our methods for different orderings, location privacy models, and user/object distributions.

Anonymity strength. NAP is theoretically guaranteed to honor reciprocity and provide K-anonymity. Figure 9 empir- ically demonstrates this fact, i.e., that no user in the AEL is more than $1/K$ likely to have issued the query. We generate 1000 random queries with $K = 40$ and record the position of the querying user within the AEL according to ord_u (we include results only for DF as those for other orderings are almost identical). Figure 9(a) plots the querying frequency per user position in the AEL. The dashed line, labeled “safe bound”, corresponds to probability $1/K = 0.025$. There are more than $K = 40$ positions (up to 48) because the AEL may contain over K users. Figure 9(b) provides another viewpoint, considering the median AEL position as slot 0.

Effect of the proposed orderings. Table 4 shows the cost of the orderings presented in Section 4.1 for kNN queries, by fixing the parameters to their default values. Observe that the AEL size accounts for only a small fraction of the communication cost and it is insensitive to the particular ordering used. In RE, the AEL contains edges that are widely distributed in the network, leading to high processing cost (at LS) and communication cost due to the large CS size. RN places close to each other edges that share a common end- node and, thus, its cost is much lower than RE. Between the traversal-based orderings, DF outperforms BF because edges along a path are arranged next to each other, so the AEL contains many connected edges.

7.3Scalabilit Experiments

In this section, we investigate the scalability of NAP with respect to various factors. To provide an indication of the space requirements, we note that for the largest tested data sizes (i.e., |U |=200000 and |O|=1024000), the AZ uses only 12.5 MBytes of main memory (including the network graph) and the LS needs a total of 23.5 MBytes hard disk storage. End-to-end time. Before a lower level study, we present an experiment on the overall response latency. Specifically, from the user's viewpoint, the end-to-end time captures the elapsed time between issuing a query and obtaining the results. It includes the processing time at AZ, the computation time at LS, and the communication time between AZ and LS. Figure
13 shows the end-to-end time as a function of the anonymity degree K , assuming a communication bandwidth of 10Mbps.

Batch processing experiment. In the next experiment, we investigate the processing of queries in batches (as opposed to one-by-one). Within the same batch, half of the queries are of kNN type and the others are range queries. Their selectivities (k and r) are randomly chosen among the corresponding values

8 CONCLUSION

NAP relies on a global user ordering and bucketization that satisfies reciprocity and guarantees K -anonymity. We identify the ordering characteristics that affect subsequent processing, and qualitatively compare alternatives. Then, we propose query evaluation techniques that exploit these characteristics. In addition to user privacy, NAP achieves low computational and communication costs, and quick responses overall. It is readily deployable, requiring only basic network operations.

In the traditional spatial anonymity model, the data owner (e.g., a location-based service) makes its data available using location server. It may, however, be the case that the owner is outsourcing its database to a third-party (and, thus, untrusted) location server. A challenge here is how to encrypt the owner's data so that they are hidden from the location server, while it can still process anonymous queries. Another interesting question is how (anonymous) users could verify that the location server did not tamper with the original owner data.

REFERENCES

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

[2] http://www.mailshell.com/.

[3] http://www.spamgourmet.com/.

[4] G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, N. Mishra, R. Motwani, U. Srivastava, D. Thomas, J. Widom, and Y. Xu. Vision Paper: Enabling Privacy for the Paranoids. In VLDB, 2004.

[5] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic Databases. In VLDB, 2002.

[6] A. R. Beresford. Location privacy in ubiquitous computing. PhD thesis, Computer Laboratory, University of Cambridge, 2005.

[7] T. Brinkhoff. A Framework for Generating Network-based Moving Objects. GeoInformatica, 6(2):153–180, 2002.

[8] A. R. Butz. Alternative Algorithm for Hilbert's Space-Filling Curve. IEEE Trans. Comput., C-20(4):424–426, 1971.

[9] C.-Y. Chow and M. F. Mokbel. Enabling private continuous queries for revealed user locations. In SSTD, 2007.

[10] C.-Y. Chow, M. F. Mokbel, and X. Liu. A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In GIS, 2006.

[11] E. W. Dijkstra. A Note on Two Problems in Connection with Graphs. Numerische Mathematik, 1:269–271, 1959.

[12] M. Duckham and L. Kulik. A Formal Model of Obfuscation and Negotiation for Location Privacy. In PERVASIVE, 2005.

[13] J. Edmonds and E. Johnson. Matching, Euler Tours and the Chinese Postman. Mathematical Programming, 5:88–124, 1973.

[14] B. Gedik and L. Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. In ICDCS, 2005.

[15] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private Queries in Location Based Services: Anonymizers are not Necessary. In SIGMOD, 2008.

[16] G. Ghinita, P. Kalnis, and S. Skiadopoulos. PRIVE: Anonymous Location-based Queries in Distributed Mobile Systems. In WWW, 2007.

[17] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In MobiSys, 2003. [18]
E. G. Hoel and H. Samet. Efficient Processing of Spatial Queries in Line Segment Databases. In SSD, 1991.

[19] X. Huang, C. S. Jensen, and S. Saltenis. The Islands Approach to Nearest Neighbor Querying in Spatial Networks. In SSTD, 2005.

[20] P. Indyk and D. Woodruff. Polylogarithmic Private Approximations and Efficient Matching. In Theory of Cryptography Conference, 2006.

[21] N. Jefferies, C. J. Mitchell, and M. Walker. A Proposed Architecture for Trusted Third Party Services. In CPA, 1995.

[22] N. Jing, Y. W. Huang, and E. A. Rundensteiner. Hierarchical Encoded Path Views for Path Query Processing: An Optimal Model and Its Performance Evaluation. IEEE TKDE, 10(3):409–432, 1998.

[23] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preserving Location-based Identity Inference in Anonymous Spatial Queries. IEEE TKDE, 19(12), 2007.

[24] A. Khoshgozaran and C. Shahabi. Blind Evaluation of Nearest Neighbor Queries Using Space Transformation to Preserve Location Privacy. In SSTD, 2007.

[25] H. Kido, Y. Yanagisawa, and T. Satoh. An Anonymous Communication Technique using Dummies for Location-based Services. In ICPS, 2005. [26]
M. R. Kolahdouzan and C. Shahabi. Voronoi-Based K Nearest Neighbor Search for Spatial Network Databases. In VLDB, 2004.

[27] T. J. Lehman and M. J. Carey. Query Processing in Main Memory Database Management Systems. In SIGMOD, 1986.

[28] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In VLDB, 2006.