A Network-Based Indexing Method for Trajectories of Moving Objects

Kyoung-Sook Kim[1], Mario A. Lopez[2], Scott Leutenegger[2], and Ki-Joune Li[1]

[1] Department of Computer Science and Engineering
Pusan National University, Pusan 609-735, South Korea
{ksookim,lik}@pnu.edu
[2] Department of Computer Science
University of Denver, 2360 S. Gaylord St., Denver, CO 80208-0183, U.S.A
{mlopez,leut}@cs.du.edu

Abstract. Recently many researchers have focused on management of historical trajectories of moving objects due to numerous size of accumulated data over time. However, most of them are concentrated in Euclidean spaces with (x, y, t). In real world, moving objects like vehicles on transportation networks have constraints on their movements, and some of applications need to manage and query them. Previous work based on Euclidean is inefficient to process trajectories on road networks. In this paper, we propose a indexing method for trajectories of moving objects on road networks. While some work has been done for indexing the trajectory in spatial networks, little indexing method support the network-based spatiotemporal range query processing. Our method consists of multiple R-trees and graph structures to process the network-based spatiotemporal range query defined by the network distance instead of Euclidean distance. Consequently, we show that our method takes about 30% less in node accesses for the network-based range query processing than other methods based on the Euclidean distance by experiments.

1 Introduction

As a development of wireless communications and position-based technologies, the geographical location information of moving objects such as human or vehicles becomes applied broadly. Moving objects can be divided into two categories depending on the domain where they move: one is Euclidean spaces, and the other is the space with some constraints such as transportation networks or terrain data. Although many researchers have focused on the management of moving objects in Euclidean spaces, the moving objects in the constraint space are more important on the specific application area, especially telematics. Telematics manages the geographical location of vehicles on transportation networks and renders various information about traffic, accident, routes, and so on. In order to improve the qualities and provide manifold services for this kind of applications, we need not only to track the continuous locations of vehicles on road networks, but also to store and to retrieve them in databases efficiently.

However, most of methods that have been developed for indexing and query processing deal with moving objects in Euclidean spaces. Therefore, they have some problems to handle moving objects on road networks.

The network space has three differences comparing to Euclidean:(1) the representation of coordinates; (2) the distance between two objects; (3) the dimensionality of domains. In this paper, we propose a new indexing method for trajectories of moving objects on transportation networks. First, we represent the trajectories using the information of road network instead of 3D Euclidean space with $(x,\ y,\ t)$, and introduce the network-based spatiotemporal range query. Second, we involve the network connectivity information in indexing structures to process the network-based spatiotemporal queries. Our method is based on multiple R-trees for trajectories and on a graph structure for connectivity relationship of road networks. For the network-based spatiotemporal query processing, we use the network distance instead of Euclidean distance. Using the network distance leads to a good performance because of reducing the unnecessary searching space in road networks.

The rest of this paper is organized as follows. Section 2 introduces the related works, Section 3 defines the problems in the network spaces, and Section 4 describes our data structures and associated algorithms. Section 5 evaluates the proposed method with some experiments. We conclude our paper in Section 6.

2 Related Work

Many researchers have investigated indexing methods for moving objects to improve the performance of updating data and query processing. Generally, there are two groups of the indexing method for moving objects. The first group of the indexing method tries to both handle current positions continuously changed over time and anticipate their positions in near future. This kind of methods focuses on reducing the cost of query processing as well as the cost of frequent updating for the real-time location-aware services. The other group deals with the historical trajectories of moving objects from past to present. Because of the massive size of trajectory data, we have high retrieval cost in the database into which we store their past positions. Therefore, this kind of indexing methods aims to improve the spatiotemporal query performance.

In this paper, we focus on processing trajectories of moving objects on road networks. Many spatiotemporal access methods are based on R-tree[1]. For example, 3D R-tree[2], HR-tree[3], and MV3R-tree[6] are spatiotemporal access methods to be able to process temporal predicates such as timestamp(or time slice) and time interval queries; but, they have some problems to access to part or whole trajectory for trajectory-based queries or navigational queries due to the absence of the location information of itself. For them, Pfoser et al. have suggested spatiotemporal R-tree(STR-tree) and Trajectory-Bundle tree(TB-tree) in [4]. Besides, some indexing methods like SETI and SEB-tree[10], which have been proposed by Prasad Chakka et al. and Song and Roussopoulos respectively, are for the efficient insertion of trajectory segments using the space partition mechanism. These access methods deal with trajectories of moving objects in (x, y, and t) Euclidean space. However, they overlook moving objects whose movement has some constraints; for instance, a vehicle or a train respectively stays a road or a rail, a human just moves a predefined area. In particular, as increasing the interest of the applications like telematic, many researchers have exploited road networks for data modelling and query processing of moving objects[8, 11, 12, 16]. Furthermore, the access methods for trajectory on road networks have been proposed in [13–15]. Pfoser and Jensen use two 2D R-trees in [13]: one is for indexing road segments and the other is for accessing trajectory segments whose coordinates are transformed from 3 dimensional (x, y, t) space to 2 dimensional (x, t) space by Hilbert-Mapping. Also, FNR-tree[14] and MON-tree[15] use a 2D R-trees to index the spatial road segments, and have respectively several 1D R-trees and 2D R-trees to index trajectories on specific road segments. Namely, these three methods retrieve trajectories after finding road segments where are satisfied with a spatial condition using Euclidean distance in 2D R-tree. However, according to the literature [8, 9, 12], the network distance between two objects on road networks is different from Euclidean one and it may lead to different results for query processing. In order to compute the network distance, we need network connectivity information. Therefore, we consider the connectivity of networks in indexing structures to be able to process the queries based on the network distance.

3 Problem Definition on Networks

In this section, under the assumption that all objects move along the road networks, we present trajectories of moving objects and redefine spatiotemporal range query regarding network spaces. The road network domain has differences from Euclidean domains according to several properties. First, positions of a moving object on the network space can be expressed by the specific road information such as a road identifier and its relative position instead of the coordinates formatted (x, y) in two dimensional Euclidean space. Second, while Euclidean space is represented by generally two or three dimensional space, the network space lies on between one and two dimensional space [13, 15]. Further, the distance between two moving objects p and q on road networks is computed by the length of the shortest path from p to q unlike the straight line length in 2D Euclidean space because moving objects can only travel along connected roads. This different distance measure brings out different results of the query processing. For instance, suppose that we process a query of the form "*find all objects which are contained within the radius r from object A on the road*" in figure 1. If we use the Euclidean distance, object B, C, and D are the results of the range query Q; nevertheless, the real result that we want to get is only object B under the constrained movement in road networks. Therefore, we should take into account network properties for the trajectory representation and query processing.

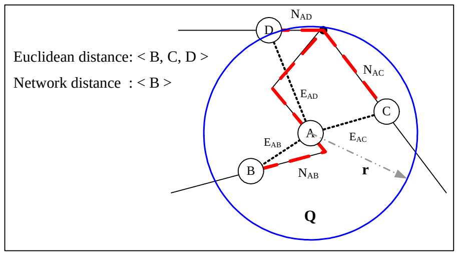


Fig. 1. The difference between Euclidean distance and Network distance

First of all, in this paper a position of a moving object is expressed as a form (rid, d) given in figure 2, where rid is an identifier of a road, d is a displacement, and l is the length of road rid. The displacement is expressed by a relative distance from start position of the road where a objects is located. Therefore, a trajectory of a moving object in road network space is defined as

$$Trajectory = \{(rid, d, t) * | rid \in int, d \in real(0 \leq d \leq l), t \in time\}$$

In other words, the trajectory, denoted T, is represented as a sequence of positions (rid_i, d_i, t_i), (r_1, d_1, t_1),...,(r_n, d_n, t_n). However, in indexing aspect preserving whole trajectory is inefficient due to large amount of dead space. The dead space declines the performance of query processing in the index structure based on R-tree. Therefore, we divide a trajectory into several trajectory segments, whose information is used for constructing our indexing method. Let a trajectory segment denote TS, each trajectory segment, a form [rid_i, (d_i,t_i), (d_{i+1},t_{i+1})]($t_i < t_{i+1}$), is defined with respect to each road segment.

Second, we redefine the spatiotemporal range query that has both spatial conditions in the network spaces and temporal conditions. In this paper, the spatiotemporal range query is retrieving trajectory segments within some distance from a given point on road networks at some time during a given time interval. In addition, when we process the query, we use the network distance instead of the Euclidean distance. For definition of network-based spatiotemporal range query, let TS be a trajectory segment set and q be a range query given by

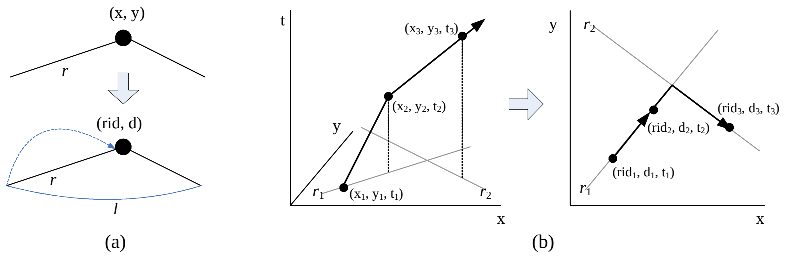


Fig. 2. The network-based trajectories representation

its position (rid, d) and some radius r with a time interval $ti = [t_s, t_e](t_s \le t_e$, in the timestamp case, $t_s = t_e)$. The network-based spatiotemporal range query retrieves all trajectory segments which satisfy following condition:

$$TS(q) = \{ts | \exists t \in T_I = [(q.ti \cap ts.ti) \neq \emptyset] \wedge ND_{q,ts}(t) \le q.r\}$$
$$\text{for } ts \in TS = \{ts_1, ts_2, ..., ts_m\}$$

where ts is a trajectory segment in TS, T_I is the temporal intersection interval between the query and a trajectory segment, and $ND_{q,ts}(t)$ is the network distance, the length of the shortest path from the query position to the position of trajectory segment at intersection timestamp t with linear interpolation. If trajectory segment has no temporal intersection with the query time interval, we regard ND as the infinitive real number(∞) to discard it.

4 Network-based Indexing Method

This section explore the data structures based on R-tree, a graph structure, and a hashtable for our method. Furthermore, we propose algorithms for the network-based range query processing through them. The connectivity relationship and the network distance play important roles in query processing to reduce the searching network space.

4.1 Data Structures

Roughly, our network-based indexing method consists of a hashtable, a B+-tree, and multiple 2D R-trees(see Figure 4). The road networks can be represented by a graph G=(V, E), with a node set V and a edge set E. In this paper, we regard intersections and road line-segments in road networks, respectively, as nodes and edges. To process the network-based rage query, a hashtable HT for edges is created by road segment-id, and each element in HT contains two bounded(start, end) node-ids, the displacement of start node, and a pointer of a R-tree RT with the 2D MBRs ([d_l,d_h],[t_l,t_h]) of trajectory segments. We use 2D R-tree for adapting both the spatial and the temporal range. This pointer allows to find trajectory segments which passed through this road segment. Also, we use B+-tree based on the node-id values for accessing the connectivity relationship of each node. A node in the data block of B+-tree has its id value, coordinate and adjacency list of following entry: $<$ *adjNode*, *adjEdge*, *weight* $>$, where *adjNode* is the connected node-id, *adjEdge* corresponds to the edge which is incident on these two nodes. $|weight|$ indicates the length of *adjEdge*. Nodes are sorted via Hilbert value and stored on node blocks in sequence

4.2 Insertion Trajectory Segments

As above mentioned, we have R-trees as many as a number of road segments and each trajectory segment is inserted into a corresponding R-tree. However, it is inefficient both for storage utilization and the query processing. Therefore,

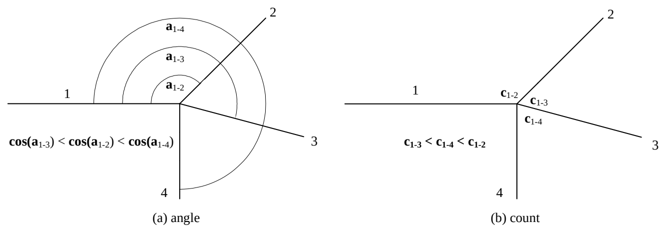


Fig. 3. The case of merging road segment in networks

we try to merge several adjacent road segments to a polyline in order to manage a reasonable number of R-trees. In this paper, we consider two heuristical information: one is an angle between two adjacent road segment, and the other is moving frequency of trajectories. For example, figure 3 shows a road segment is connected with three other segments. If we use the angle between two segment, we select a segment to make straight line along segments because of the observation that vehicles pass straightly when it meets intersection in common case. In the case using the frequency, we merge the segment whose data count is larger than other adjacency list. As a result, segment 1 in figure 3 (a) and (b) will be combined with segment 3 and segment 2, respectively. After merging, hashtable entries contained in the polyline have are assigned by the same R-tree pointer.

Then, we insert a trajectory segment into R-tree via following steps. First, we get the corresponding entries from the hashtable with two rids of successive sampling positions $(rid_1, d_1, t_1), (rid_2, d_2, t_2)$. If two entries is the same, a root of R-tree is loaded with the pointer in the entry, and the 2D MBR ($[d_l$,$d_h]$,$[t_l$,$t_h]$) where d_l, d_h, t_l, and t_h are the minimum and maximum value of displacements and timestamps of two positions, is inserted into R-tree. Otherwise we find the shortest path from start position to end position. After finding the edge list of the path, a new trajectory segment consists of several trajectory segments of each edge considering that the object moves along this path with same speed for a sampling time interval. Each decomposed trajectory segment is inserted to R-trees by above steps.

4.3 Network-based Query Processing

Now, we introduce how to process the range query based on the network distance from a query point to a moving object. Generally, our algorithm for query processing consists of two parts: pruning the search space with the network distance, and accessing 2D(displacement and time) R-trees for trajectories.

Figure 4 shows the examples of a range query "find all trajectories which are contained within the radius 6.0 miles from the query position $(r_5, 3.48)$ during the given time interval $ti = [1P.M., 3P.M.]$." For processing the range query, first, we get the entry of hashtable for edges with a given rid r_5 like insertion algorithm, and we add edge r_5 into the edge list for accessing R-trees. Next, its two nodes n_4 and n_6 with remaining query radius are enqueued. Namely,

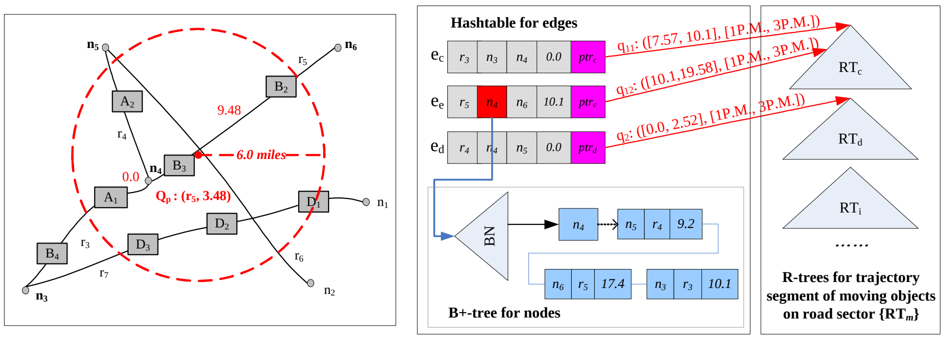


Fig. 4. The example of the range query on road networks

$< n_4, 2.52 >$ and $< n_6, -7.92 >$ are inserted into the priority query. The queue is arranged in an descending order of radius. The node which the radius is larger than others is visited to expanding the search area. Then, node n_4 is dequeued. As the remaining query radius is more than 0, its adjacent edge, r_3 and r_4, are inserted into the edge list, and node n_3 and n_5, the adjacency nodes of n_4, are enqueued in the priority query with the radius -6.68 and -7.58, respectively. This procedure repeats until the priority query becomes empty. If the remaining radius of a node is less than 0, the node is discarded because it does not need to expand the search area. After that, we compute the query set for each R-tree of road segments in order to avoid duplication of accessing the same R-tree shared by several road segments. We retrieve each R-tree with the query set.

5 Experimental Results

We performed experiments to evaluate the performance of the proposed method using synthetic moving objects on real road networks. For the test, we used two kinds of road network datasets: one is synthetic networks like mesh networks, and the other is real networks showed in figure 5. Then we prepared moving objects with the network-based generator developed by Brinkhoff[5] because massive real moving object trajectories are rarely spread wide. About 2M number of trajectory segments are generated during 500 time units using the network-based generator. The location (x, y, t) of generated moving objects is transformed into a network-based location formed (rid, d, t) on each road networks for our method.

In the experiments, we compared the average number of node accesses of R-trees for trajectories of our method and two other index structures for moving objects on road networks showed by literatures [15, 13]. We indicated each other index structures to MON-tree[15], Hibert-Mapping[13], and our method is called by NB-tree in the experiments. The page size for nodes was 4096 bytes, which could store at most 102 entries, and the fillfactor was 70%. 5000 queries were randomly generated with a range of 0.01%, 0.1%, 1%, 5%, 10%, 20%, 30%, and

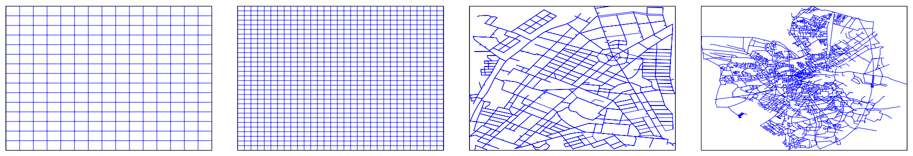

(a) Grid-16 (b) Grid-32 (c) Busan (d) Oldenburg

Fig. 5. The road networks

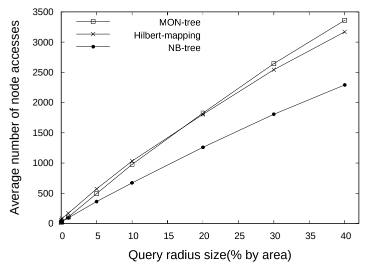
40% in spatial area of whole networks, and a constant range of 25% in temporal domain.

Figure 6 shows the average number of node accesses for the range queries between the network-based distance and Euclidean distance. Each indexing methods are created by the road segments. the indexing method using multiple R-trees, MON-tree and NB-tree, provide a good performance for smaller query size, while for larger query size MON-tree increases the node accesses more than Hilbert-Mapping, which uses one R-tree for trajectories. These indicate the large number of R-trees has a bad influence on the performance. However, we observe that NB-tree based on the query processing with the network distance reduce the search cost about 40% less than others. Namely, computing the distance using connectivity relationship of road networks restricts the searching space for the range queries comparing Euclidean distance.

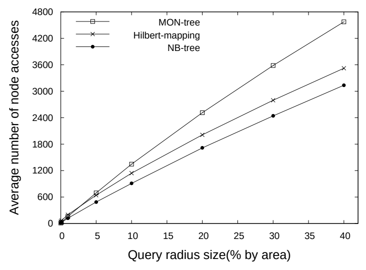
As we have already mentioned the large number of R-trees for road segments affects the performance of the network-based query processing. Therefore, in this paper, we introduced two heuristical merge mechanisms with statistical information. Figure 7 depicts the performance comparison of these merge mechanism. Although we could not improve the performance by merging in small network data set, in large network data set the performance of the range query was improved at most 25%.

6 Conclusion

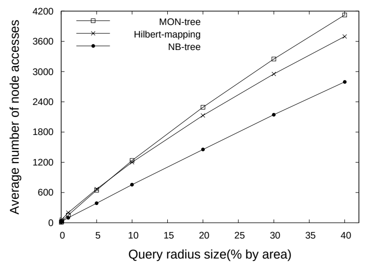
Although many indexing methods of trajectories have been proposed, they have dealt with the movement in Euclidean spaces. In this paper, we focused on the constraint movement of moving objects, especially like vehicles on the transportation networks. The idea was to represent trajectories and the spatial range queries in the road network space and to propose the indexing structure for handling them. Our network-based indexing method included the connectivity relationship of road networks to allow the network-based spatiotemporal range query. Moreover, we considered the merging way to share the R-trees for several road segments. The experiments showed lots of R-trees caused a performance degradation for query processing. In the future, we will try to apply other query types like k-nearest neighbor, and evolve analytical cost model for the performance of the network-based query processing.



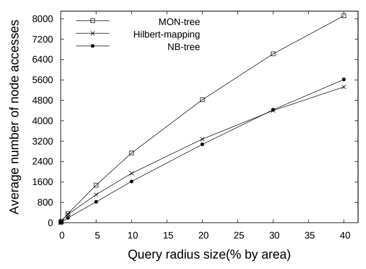
(a) Grid-16



(b) Grid-32



(c) Busan



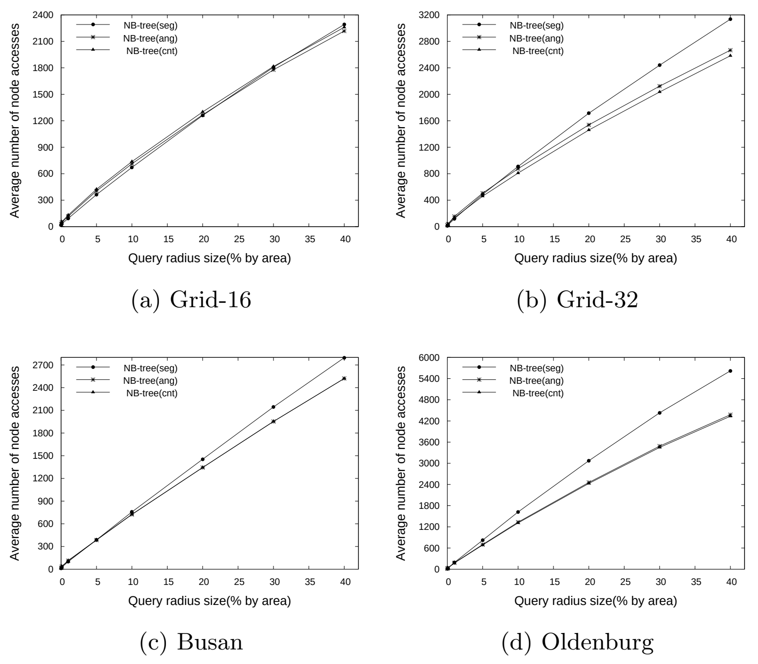
(d) Oldenburg

Fig. 6. The average number of node accesses for range queries between the network-based distance and Euclidean distance

Acknowledgment. This work was supported by the Korea Research Foundation Grant Funded by the Korea Government(MOEHRD)(KRF-2006-209-D00008), by IRC(Internet Information Retrieval Research Center) in Hankuk Aviation University. IRC is a Regional Research Center of Kyounggi Province, designated by ITEP and Ministry of Commerce, Industry and Energy.

References

1. A Guttman, R-trees a dynamic mdex structure for spatial searching, in Proc ACM SIGMOD Int Conf on Management of Data,pp. 47–57, 1984.
2. Y. Theodoridis, M. Vazirgiannis, and T. Sellis, Spatio-Temporal Indexing for Large Multi- media Applications, in Proc. IEEE Conf. On Multimedia Computing and System, pp. 441–448, 1996.
3. M. Nascimento and J. Silva, Towards historical R-trees, in Proc. ACM-SAC, pp. 235–240, 1998.
4. D. Pfoser, C. S. Jensen, and Y. Theodoridis, Novel Approaches in Query Processing for Moving Object Trajectories, in Proc. VLDB, pp. 395–406, 2000.
5. T. Brinkhoff, Generating Network-Based Moving Objects, in Proc. SSDBM, pp. 253–255, 2000.
6. Y. Tao and D. Papadias, MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries, in Proc. VLDB, pp. 431–440, 2001.



(a) Grid-16 (b) Grid-32

(c) Busan (d) Oldenburg

Fig. 7. The comparison of each merge mechanism for network-based range query processing on networks

7. M. Vazirgiannis and O. Wolfson, A Spatiotemporal Model and Language for Moving Objects on Road Networks, in Proc. SSTD, pp. 25–35, 2001.
8. C. Shahabi, M. R. Kolahdouzan, and M. Sharifzadeh, A Road Network Embedding Technique for K-Nearest Neighbor Search in Moving Object Databases, in Proc. ACM GIS, pp. 94–100, 2002.
9. D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, Query Processing in Spatial Network Databases, in Proc. VLDB, pp. 802–813, 2003.
10. Z. Song and N. Roussopoulos, SEB-tree: An approach to Index continuously moving ob- jects, in Proc. IEEE Conf. MDM, pp. 340–344, 2003.
11. L. Speicys, C. S. Jensen, and A. Kligys, Computational Data Modeling for Network-Constrained Moving Objects, in Proc. ACM-GIS, pp. 118–125, 2003.
12. C. S. Jensen, J. Kolar, T. B. Pedersen, and I. Timko, Nearest Neighbor Queries in Road Networks, in Proc. ACM GIS, pp. 1–8, 2003.
13. D. Pfoser and C. S. Jensen, Indexing of Network-Constrained Moving Objects, in Proc. ACM-GIS, pp. 25–32, 2003.
14. E. Frentzos, Indexing objects moving on Fixed networks, in Proc. SSTD, pp. 289–305, 2003.
15. V. de Almeida and R. Guting, Indexing the Trajectories of Moving Objects in Networks, in Proc. SSDBM, pp. 115–118, 2004.
16. R. H. Guting, V. T. de Almeida, and Z. Ding, Modeling and Querying Moving Objects in Networks, Tech. Rep. Informatik-Report 308, Fernuniversitat Hagen, April 2004.