A New Trajectory Indexing Scheme for Moving Objects on Road Networks*

Jae-Woo Chang[1], Jung-Ho Um[1], and Wang-Chien Lee[2]

[1] Dept. of Computer Eng., Chonbuk National Univ., Chonju, Chonbuk 561-756, Korea
jwchang@chonbuk.ac.kr, jhum@dblab.chonbuk.ac.kr
[2] Dept. of CS&E., Pennsylvania State Univ., University Park, PA 16802

Abstract. In this paper, we propose an efficient signature-based indexing scheme for efficiently dealing with the trajectories of current moving objects on road networks. We show that our indexing scheme achieves much better trajectory retrieval performance than the existing trajectory indexing schemes, such as TB-tree, FNR-tree and MON-tree.

1 Introduction

Even though most of the existing work on spatial databases considers Euclidean spaces, objects in practice can usually move on road networks, where the network distance is determined by the length of the real shortest path on the network. For example, a gas station nearest to a given point in Euclidean spaces may be more distant in a road network than another gas station. Therefore, the network distance is an important measure in spatial network databases (SNDB). Meanwhile, there have been a couple of studies on trajectory indexing schemes for both Euclidean spaces and spatial networks (i.e., roads) [PJT00, F03, AG05]. First, Pfoser et al. [PJT00] proposed a hybrid index structure which preserves trajectories as well as allows for R-tree typical range search in Euclidean spaces, called TB-tree (Trajectory-Bundle tree). The TB-tree has fast accesses to the trajectory information of moving objects, but it has a couple of problems in SNDB. First, because moving objects move on a predefined spatial network in SNDB, the paths of moving objects are overlapped due to frequently used segments, like downtown streets. Secondly, because the TB-tree constructs a three-dimensional MBR including time, the dead space for the moving object trajectory can be highly increased. Next, Frentzos [F03] proposed a new indexing technique, called FNR-tree (Fixed Network R-tree), for objects constrained to move on fixed networks in two-dimensional space. Its general idea is to construct a forest of 1-dimensional (1D) R-trees on top of a 2-dimensional (2D) R-tree. The 2D R-tree is used to index the spatial data of the network while the 1D R-trees are used to index the time interval of each object movement inside a given link of the network. The FNR-tree outperforms the R-tree in most cases, but it has a critical drawback that the FNR-tree has to maintain a tremendously large number of R-trees. This is because it constructs as large number of R-trees as the total number of segments in the networks. Finally, Almeida and Guting proposed a new index structure for moving objects on

* This work is financially supported by the Ministry of Education and Human Resources Development(MOE), the Ministry of Commerce, Industry and Energy(MOCIE) and the Ministry of Labor(MOLAB) though the fostering project of the Lab of Excellency.

D. Bell and J. Hong (Eds.): BNCOD 2006, LNCS 4042, pp. 291 – 294, 2006.

network, called MON-tree, for both edge-oriented and route-oriented models. The MON-tree outperforms the FNR-tree in both updating and querying, but it has a drawback that the MON-tree should still maintain a very large number of R-trees, like the FNR-tree.

2 Trajectory Indexing Scheme for Current Moving Objects

To overcome the weaknesses of the existing schemes, we propose a new trajectory indexing scheme for moving objects on road networks, which is based on a signature file technique for efficiently dealing with the trajectories of moving objects. Figure 1 shows the structure of our trajectory indexing scheme. Our main idea is to create a signature of a moving object trajectory and maintain partitions which store the fixed number of moving object trajectories and their signatures together in the order of their start time. The main reason to use partitions is that because a partition is created and maintained depending on its start time, it is possible to efficiently retrieve the trajectories of moving objects on a given time. So, our trajectory indexing scheme has the following advantages. First, our indexing scheme is not affected by the overlap of moving objects' paths and never causes the dead space problem because it is not a

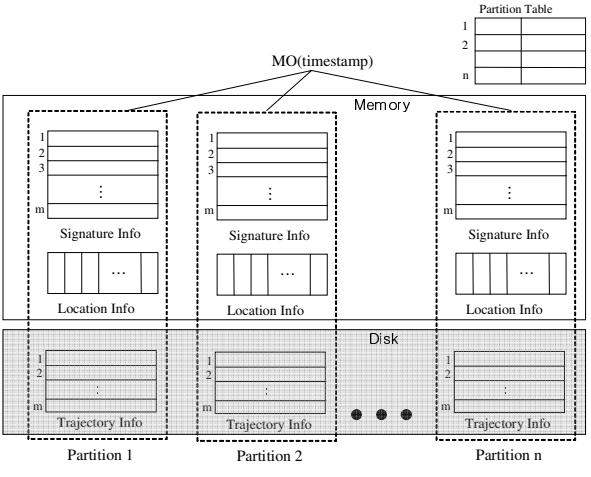

Fig. 1. Structure of our trajectory indexing scheme

tree-based structure like TB-tree. Secondly, our indexing scheme well supports a complex query containing a partial trajectory condition since it generates signatures using a superimposed coding. Finally, our indexing scheme can achieve very good insertion performance because it does not maintain a large number of trees, like FNR-tree and MON-tree. Our trajectory indexing scheme consists of a partition table and a set of partitions. A partition can be divided into three areas; trajectory information, location information, and signature information. A partition table for maintaining a set of partitions to store trajectories can be resided in a main memory due to its small size. To achieve good retrieval performance, we also store both the signature and the

location information in a main memory because of their relatively small size. To answer a user query, we find partitions to be accessed by searching the partition table. The trajectory information area maintains moving object trajectories which consist of a set of segments (or edges). The location information area contains the location of an object trajectory stored in the trajectory information area. This allows for accessing the actual object trajectories corresponding to potential matches to satisfy a query trajectory in the signature information area. To construct our trajectory indexing scheme in an efficient manner, we make use of a superimposed coding because it is very suitable to SNDB applications where the number of segments for an object trajectory is variable [ZMR98].

3 Performance Analysis

We implement our trajectory indexing scheme under Pentium-IV 2.0GHz CPU with 1GB main memory. For our experiment, we use a road network consisting of 170,000 nodes and 220,000 edges [WMA]. We also generate 50,000 moving objects randomly on the road network by using Brinkhoff's algorithm [B02]. For performance analysis, we compare our indexing scheme with TB-tree, FNR-tree and MON-tree, in terms of insertion time, storage space, and retrieval time for moving object trajectories. First, Table 1 shows insertion times to store a moving object's trajectory. It is shown that our indexing scheme preserves nearly the same insertion performance as TB-tree, while it achieves about two orders of magnitude better insertion performance than FNR-tree and MON-tree. This is because both FNR-tree and MON-tree construct an extremely great number of R-trees. Secondly, we measure storage space for storing moving object trajectories, as shown in Table 1. It is shown that our indexing scheme requires nearly the same storage space as TB-tree, while it needs about one fifth of the storage space required for FNR-tree and MON-tree.

Table 1. Trajectory insertion time and storage space

	TB-tree	FNR-tree	MON-tree	Our indexing scheme
Trajectory insertion time(sec)	0.7488	344	260	0.6552
Storage space(MB)	4.42	20.2	23.75	4.87

Finally, we measure retrieval time for answering queries whose trajectory contains 2 to 20 segments, as shown in Figure 2. It is shown that our indexing scheme requires about 9 ms while MON-tree, FNR-tee and the TB-tree needs 15ms, 21ms, and 630ms, respectively, when the number of segments in a query is 2. It is shown that our indexing scheme outperforms the existing schemes when the number of segments in a query trajectory is small. The TB-tree achieves the worst retrieval performance due to a large extent of overlap in its internal nodes. As the number of segments in queries increase, the retrieval time is increased in the existing tree-based schemes; however, our indexing scheme requires constant retrieval time. The reason is why our indexing scheme creates a query signature combining all the segments in a query and it searches for potentially relevant trajectories of moving objects once by using the query signature as a filter. When the number of segments in a query is 20, it is shown that our indexing scheme requires about 9 ms while MON-tree, FNR-tree and TB-tree needs 108ms, 157ms and

1.3s, respectively. Thus our indexing scheme achieves at least one order of magnitude better retrieval performance than the existing schemes. This is because our indexing scheme constructs an efficient signature-based indexing structure by using a superimposed coding technique. On the contrary, because the TB-tree builds a MBR for each segment in a query and the number of range searches increases in proportion to the number of segments, the TB-tree dramatically degrades on trajectory retrieval performance when the number of segments is great. Similarly, because both MON-tree and FNR-tree search for an R-tree for each segment in a query, they degrade on retrieval performance as the number of segments in the query is increased.

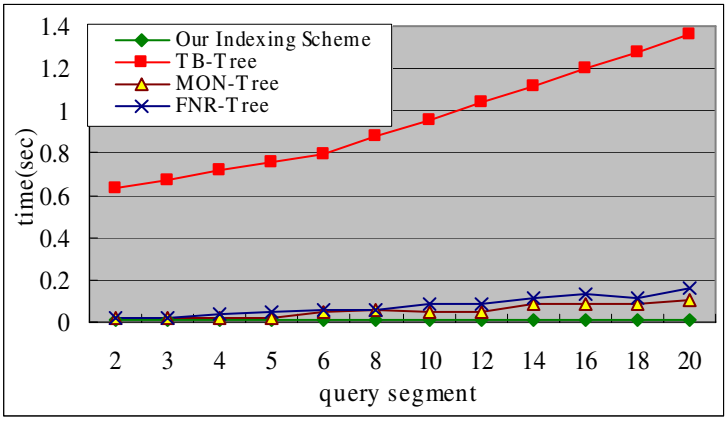

Fig. 2. Trajectory retrieval performance

4 Conclusions

We proposed an efficient signature-based indexing scheme for current moving objects' trajectories on road networks. We showed that our indexing scheme achieved at least one order of magnitude better retrieval performance than the existing trajectory indexing schemes, such as TB-tree, FNR-tree and MON-tree.

References

[AG05] V.T. Almeida and R.H. Guting, "Indexing the Trajectories of Moving Objects in Networks," GeoInformatica, Vol. 9, No. 1, pp 33-60, 2005.

[B02] T. Brinkhoff, "A Framework for Generating Network-Based Moving Objects," GeoInformatica, Vol. 6, No. 2, pp 153-180, 2002.

[F03] R. Frentzos, "Indexing Moving Objects on Fixed Networks," Proc. of Int'l Conf on Spatial and Temporal Databases (SSTD), pp 289-305, 2003.

[PJT00] D. Pfoser, C.S. Jensen, and Y. Theodoridis, "Novel Approach to the Indexing of Moving Object Trajectories," Proc. of VLDB, pp 395-406, 2000.

[WMA] http://www.maproom.psu.edu/dcw/

[ZMR98] J. Zobel, A. Moffat, and K. Ramamohanarao, "Inverted Files Versus Signature Files for Text Indexing," ACM Tran. on Database Systems, Vol. 23, No. 4, pp 453-490, 1998.