QUERYING SPATIAL-TEMPORAL DATA IN LOCATION-BASED SERVICES

Z.W. Yuan[a] , M. Cheng[a], H.S. Kim[a,b] , H.Y. Bae[a,b], J. W. Ge[a]

[a] Sino-Korea Chongqing GIS Research Center, College of Computer Science, Chongqing University of Posts and Telecommunications, Chongqing, 400065, P.R. China – (yuanzw, gejw)@cqupt.edu.cn, cheng_miao@hotmail.com
[b] Department of Computer Science and Engeering, Inha University, Incheon, 402-751, Korea – (hskim, hybae)@dblab.inha.ac.kr

KEY WORDS: mobile objects, Location-based Services (LBS), location server, MOD, spatial-temporal data, key technology

ABSTRACT:

With the growing advances of positioning technologies like GPS, tracking the location of mobile objects to push Location-based Services (LBS) has been applicable. Generally, the location server is constructed as a moving objects database (MOD). In this paper, we focus on the application of spatial-temporal data in MOD including its key technologies, say data model, index structure, under the LBS environment. These key technologies are discussed and analyzed in detail, and the relative proposal is given.

1. INTRODUCTION

With development of wireless communications and positioning techniques, tracking and recording the changing positions of objects capable of continuous movement is becoming increasingly feasible and necessary. And as being close to the scheduled launch of the third-generation mobile telecommunication networks, new classes of mobile information services will be possible. One type of such services is the Location-based Services (LBS). For the LBS systems to become reality, database systems should be capable of storing, updating, and querying the positions of large amounts of continuously moving object, which poses new challenges to database technology.

The tendency of tracking movement of moving objects in database has lead to the combination of temporal and spatial objects into spatial-temporal objects in database. However, in the past, research in spatial and temporal data models and databases has largely developed independently. Spatial database (R. H. Güting, 1994) has focused on modeling, querying and integrating geometric and topological information in databases. For the modeling of spatial objects the well known concept of abstract data types (ADT) has been proved very useful. Temporal database has concentrated on modeling, querying, and recording the temporal evolution of facts under different notions of time (valid time, transaction time) and thus on extending the knowledge stored in databases about the current and past states of the real world. The need for integrated handling of time and space dimension for scientific data has long been recognized.

Moving Objects refer to such entities as automobiles, cellular phones users, and air planes change their locations continuously. The goal of a spatial-temporal database is to model the real world accurately. The conventional assumption is that data remains constant unless it is explicitly changed. With this assumption, capturing continuous movement accurately would entail either performing very frequent updates or recording outdated, inaccurate data, which have received significant interest in efficient storage and retrieval of moving objects in database management systems. Location monitoring is an important issue for real time querying and management of mobile object positions. Signification research works have been dedicated to key technologies for efficient processing of queries on moving objects in spatial-temporal database, say spatial-temporal data modeling, index structure and so on. In this paper, we introduce such issue research works on spatial-temporal database that have been developed before firstly. Then we concentrate on query approach of tracking server and give a propositional framework of such related technique that could be used in LBS scenario.

The reminder of this paper is organized as follows. In section 2, the motivation is put forward. In section 3, we give an overview of the system architecture of tracking server in LBS. In section 4, we introduce the related works of spatial-temporal data model and give an appropriate proposal. In section 5, an investigation on index technologies will be shown, where we propose a powerful index structure to support spatial-temporal queries in MOD. Finally, conclusions are given in section 6.

2. MOTIVATION

In recent years, wireless technique such as WAP and Bluetooch has found widespread application. Some observes point out that there will be billions of wireless application within few years, which will lead to huge profit from personal wireless communication and related value-added services. With advances in wireless Internet and mobile computing, location-based services (LBS) are emerging as key value-added services for telecom operators to deliver.

Positioning system based on satellite technique, say Global Positioning System (GPS), which could only be used for martial application in the past. Recently, such tendency has been met that the huge civil use of GPS to carry out LBS. LBS enables communication working builders to provide personal location-aware content to subscribers using their wireless network infrastructure. To our best knowledge, moving objects database (MOD) plays an important role in tracking server in LBS. Moving objects use e-services that involve location information. The objects disclose their positional information to the services, which in turn use this and other information to provide specific functionality. When queries on moving objects is put forward in tracking server, the moving objects database will execute related processing to give the answer to such queries. However,

it may be difficult to construct such system because of such special dynamic environment and complex operation. To get a blue point of tracking server in point, such technologies as data models, index structure should be thought over firstly. In this paper, we present some propositional ideas for such investigative fields, which may be used to construct a powerful server side for positioning system.

3. SYSTEM ARCHITECTURE

The crucial issue in LBS environments is how to design LBS application developing platform (LBS-ADP) and organize the location servers. For LBS-ADP, a system architecture that is easy implemented and adapt to all kinds of mobile network system has been proposed in (Y. Xia, 2004). Simplest approach to location database organization is to store all location information in a single physical storage site. However, such a solution results in a single point is impractical and may be not able to meet the demand for large numbers of moving objects. Instead, there is an architectural alternative in which the whole space is decomposed into regions or cells; for each one, there is a tracking server to supply LBS as shown in Figure 1.

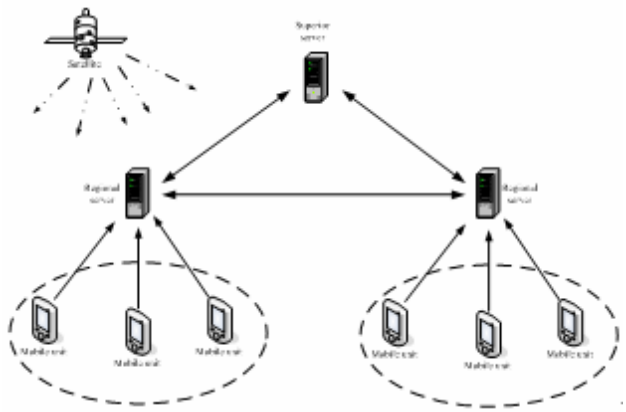

Figure 1. System architecture of Location Server

Thus, this approach may not only be able to overcome the above-mentioned drawbacks of a centralized database, but also enhance the availability and scalability of various location services.

We assume that each mobile phone is equipped with a device like GPS to locate its position and moving objects move in a 2-d space. Moving objects connect directly to regional servers and regional servers can communicate with each other. Regional servers handle with the incoming data including location information and queries from mobile users. And each group of regional servers also connects with the superior server. Note that the regional servers only store the position information of mobile units that move in their monitoring region. When the regional server receives a query request, it will execute relative query processing to give the answer. And if the query refers to some spatial-temporal data that is stored in other regional servers, the regional server will send such query request to the superior server. The superior server analyses such request, and decomposes it into several parts based on the regional servers that are involved. Then each request part will be send to the relevant regional server to perform such query operation. Finally, the query answer will be send back to the regional server that put forward the request through the superior server. So the superior server only plays the roll of coordinator and controlling of the regional servers, all the query operations are executed in regional servers. In the rest of this paper, we will concentrate on the query processing in regional server.

4. DATA MODEL

In the server side, there is still spatial-temporal data stored and retrieved. Until now, many studies also have been done on the modeling and representation of moving objects. In (M. Worboys, 1994), Worboys generalized earlier work on spatial data and present the first unified model for spatial and temporal information by extending a spatial data model to a spatial-temporal model. Wolfson et al. presented the famous data model called Moving Object Spatial-Temporal (MOST) for representing moving objects (A. P. Sistla and O. Wolfson, 1997). In the MOST model, the location of a moving object is simply given as a linear function of time, which is specified by two parameters: the position and velocity vector of the object. But this data model can only support future query well.

Another work proposed an extended SQL exploiting the concept of ADTs, which described an approach to modeling moving and evolving spatial objects to generalize new data types (M. Erwig, 1999; R. H. Güting, 2000). Two new spatial-temporal data types: moving point and moving region are introduced to represent moving objects, which are implemented based on discrete data models (L. Forlizzi, 2000) (see Figure 2).

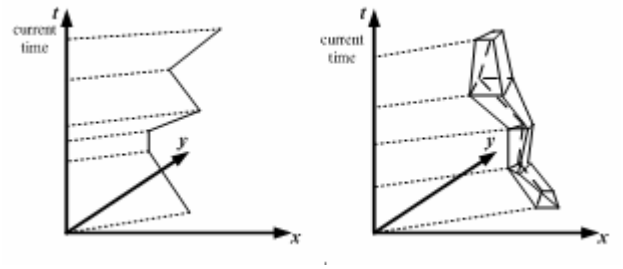

Figure 2. Discrete representation of moving point and moving region

Then the movement of moving objects can be represented as attribute timestamped models and described as function of time. The shortcoming of such process is that, the ADT data model and related operations are too complex to implement and unpractical in real-life application environment. Moreover, only historical spatial-temporal data is considered in this data model. In (G. Faria, 1998), a lot of useful operators are raised to build an extensible framework for spatial-temoral database applications. C.X. Chen and C. Zaniolo (C. X. Chen, 2000) developed a spatial-temporal data model and query language based on the model proposed by Worboys where each state of a spatial object is captured as a snapshot of time. They use a triangulation model to represent spatial data, and a point-based model to represent time at the conceptual level. Complex spatial data is decomposed into simple triangulations to store and retrieve. However, there are also only historical spatial-temporal queries that could be supported in such model.

To build a vigorous Location-based Service, the location server should support not only current and historical queries but also

future queries (M. F. Mokbel, 2003). Until now, there isn't any appropriate data model to support all spatial-temporal queries types. The most reason is that, as shown in Figure 3, the different queries are based on different spatial-temporal data. And also, the current spatial-temporal data is still under frequent dynamic update environment.

Figure 3. The classification of spatial-temporal queries

An appropriate way is to build and query the historical and present data model respectively. For the historical spatial-temporal data, usually called trajectory data, a line segment trajectory data model may be constructed. A trajectory segment is represented as $<P_{i-1}, P_i>$, where P_{i-1} and P_i are two spatial-temporal point parts. Each point P_i is a three-tuple $<x_i, y_i, t_i>$, where x_i and y_i are the spatial coordinates of the object at time t_i. Then the tuples have the following format (O_{id}, N_i, P_{i-1}, P_i) where O_{id} is a unique id of moving object, N_i is a unique segment number for this segment of the trajectory. All the trajectory segments belonging to one moving object represent the moving object's trajectory. For the present spatial-temporal data, such data model can be regarded as a trajectory segment that don't have end point. Then a moving object is described as (O_{id}, N_i, P_i, Vel>, where the O_{id} and N_i have the same signification as described above, the P_i also is a three-tuple that represents the latest update position information, and the Vel is the latest update velocity of moving object, which used to calculate the future position of moving objects. If new update information is received by the server, the update data will replace the latest data to represent the current state of the moving object.

5. INDEX TECHNOLOGY

5.1. Main Idea

In the location-based services, mobile objects reveal their positional information to the location server to provide general application. Our focus is on location-based services that access and query positions of moving objects. As discussed above, there queries in MOD (M. F. Mokbel, 2003) can be classified into three categories (see Figure 3). To support historical queries, the tracking server stores only the locations of the moving objects at different times via sampling. Once a location of a moving object is updated, the old location is invalid and to become historical data. Current queries are interested on the location of moving objects at present. To answer current queries, a location-aware server keeps track of the latest locations of all moving object. Future queries are interested in predicting the locations of moving objects. Usually, the future queries depend on some additional information (e.g., the velocity or destination). Until now, there is few access methods that combine both characters to index from past to future positions of moving objects. In (J. Sun, 2004), two respective indices are used to support both query operations. Especially, an incrementally updateable, multi-dimensional histogram for present-time queries and a stochastic approach for predicating the future queries are proposed. Based on the idea of TPR-tree (S. Saltenis, 2000), Pelanis *et al.* present the framework of partial persistence to accurately record the history of movement. They modified TPR-tree with a time-parameterized bounding rectangle (TPBR) and employed a PP-structure to index past data. The BB^x-index (D. Lin, 2005) structure also can index the positions of moving objects at any time through linear functions of time, which depends on the B^x-index (C. S. Jensen, 2004) structure.

All the index structures introduced above can index past, recent and future position of moving objects. Unfortunately, they all lose applicable in realistic LBS environment simply. Firstly, in the LBS scenario, there must be millions of mobile objects that should be handled. These index structures can't meet such demand. Secondly, dealing with the large number of update, there may be highly dynamic environment in the server side. To solve the problem, an optional way is to build the index structure in the main memory to reduce the disk access frequency, which will outperform other disk based access methods. In (K. Kim, 2001), a cache-conscious version of the R-tree called CR-tree is introduced, which compresses the MBR through the quantized relative representation of MBR (QRMBR) and performs well for the main memory indexing. Moreover, some distance-based geolocation update scheme (W. J. Choi, 2003) also can be made use of to avoid frequently update through location network.

For efficiently querying moving objects, we index the past and present/future positions in the disk and the main memory separately. The index structure PCFI (Z. H. Liu, 2004) and $PCFI^+$ (Z. H. Liu, 2005) are developed to support complex spatial-temporal queries in tracking server, which consists of two parts: One part locates in memory, and the other locates on disk as shown in Figure 4.

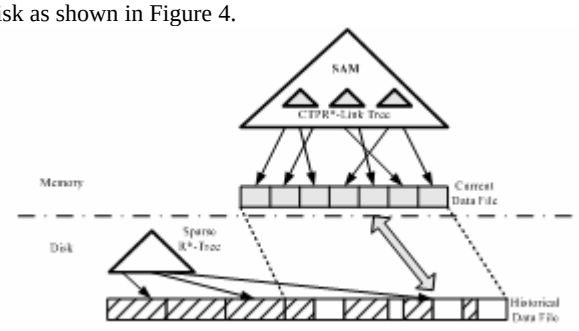

Figure 4. Data structure of PCFI

The in-memory component is called frontline. It consists of a current data file, a spatial index (SAM) (e.g., Grid-file) to index the non- overlapped partitions. Based on the technologies of TPR*-tree (Y. F. Tao, 2003) and CR-tree, a set of CTPR*-link trees (X. Zhou, 2005), is constructed for indexing current records in the current data file.

For the disk part, the historical data file will keep the segments of the moving objects. The disk index structure is based on the SETI (V. P. Chakka, 2003). One-dimensional sparse R*-tree is

used to index the life time of the historical data pages. A data page's life time (starttime, endtime) is mapped to one-dimension line (starttime, endtime), we call this the lifetime dimension. When a new entry is inserted in the sparse R*-tree with the lifetime's endtime is empty. When a data page is fully filled by the subsequence segments, the corresponding entry in sparse R*-tree will be updated to set the endtime. Entries in leaf nodes are consisted by pointer (PID) to data page, lifetime, and cellid (restriction: one page only contains the segments belong to one cell). Entries in internal nodes are pairs of a pointer to a subtree and a lifetime that bounds the lifetime in that subtree. The cellid in leaf page is used to reduce disk read cost of the candidate page while processing historical slice queries.

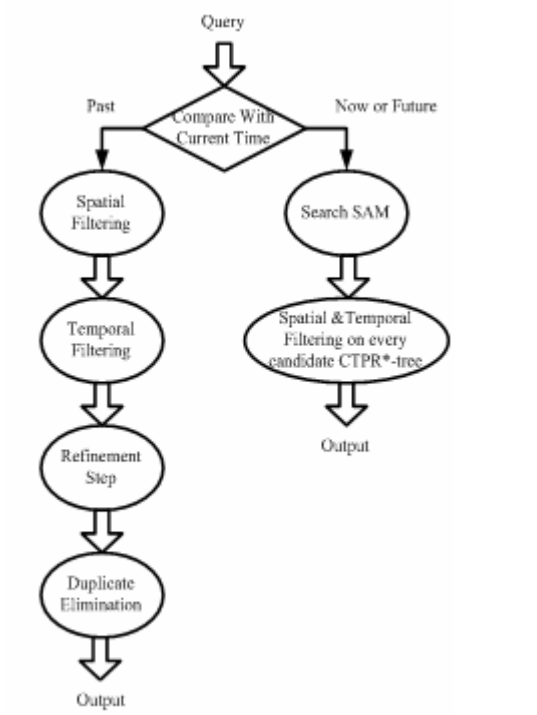

Figure 5. Diagram of Query Algorithm

5.2 Query Processing

Figure. 5 shows the steps in the query algorithm. The input to the search algorithm for a time interval query can be considered as a three dimensional query box, which consists of a temporal predicate range and a spatial predicate box.

The search algorithm executes the following steps:
Compare with the current time: In this step, the temporal predicate range is compared with the current time. There are three cases:

Case 1 if the temporal predicate range is fully fall into past, Step 1 to Step 4 is performed.

Case 2 if the temporal predicate range is fully fall into now or future, Step 5 is performed.

Case 3 if the temporal predicate range covers past, now and future, we separate it into two parties: one is the past party, the procedure in case 1 is adopted; another one is the now or future party, the procedure in case 2 is adopted.

Step 1: Spatial Filtering: In this step the spatial partitions that overlap with the spatial predicate box is computed and a candidate list of pair (cellid, full_overlap_or_not) is produced.

Step 2: Temporal Filtering: The sparse R*-tree index is searched with the temporal predicate range. Each entry in the sparse R*-tree leaf page is checked to insure whether the cell id falls into the cell list without fetching the corresponding data page into memory, if the cell is fully overlapped by the query region, all the segments belong to this cell can be putted to the result segment list directly without refinement, or, the page id is put to the candidate page list. This step generates a list of pages whose lifetimes overlap with the temporal predicate. For time-slice queries, those data pages whose lifetimes contain the predicate timestamp are fetched.

Step 3: Refinement Step: This procedure is as same as the original SETI-tree. A list of segments that overlap with query box is produced in this step.

Step 4: Duplicate Elimination: This procedure is the same as the original SETI-tree. It produces a list of moving object ids or a list of segments.

Step 5: Spatial & Temporal Filtering: This procedure is the same as the query algorithm of original TPR*-tree.

For the slice query, the temporal predicate range consists of only a single value, the query can be answered with sparse R*-tree (past time) or frontline (CTPR*-tree, current of future).

For the window query, there are three cases: 1) if both of [Ts, Te] are located in the past, the query can be answered via sparse R*-tree; 2) if both of [Ts, Te] are located in the present or future, we can process the query via frontline part; 3) if the Ts is located in the past and Te is located in the future, the [Ts, now) part is answered via sparse R*-tree part, and the [now, Te] part is answered via frontline part.

For the moving query, it is similar to the window query, except the query range should be divided into two parts according to the current time if the time range contains current time.

6. CONCLUSION

Location-based services are characterized as dealing with the large number of mobile objects. Generally, the location server is constructed as a moving objects database (MOD). In this paper, we investigate the relative research works of building a rigorous MOD firstly. Then the weakness of these ideas and an optional advice that we propose are put forward for realistic LBS environment. A dual data model is built to model the present/future and past position of moving objects. And a powerful index structure called PCFI$^+$ is proposed to index the moving objects at any time. The PCFI$^+$ consists of the memory part and the disk part, which is used to index the current and historical data file separately.

REFERENCES

A. P. Sistla, O. Wolfson, et al, 1997. Modeling and Querying Moving Objects. In *Proc. IEEE Intl. Conf. on Data Engineering (ICDE)*, pp. 422-432.

C. S. Jensen, D. Lin, and B. C. Ooi, 2004. Query and Update Efficient B+-Tree Based Indexing of Moving Objects. *Proc. of VLDB*, 2004, pp. 768-779.

C. X. Chen, C. Zaniolo, 2000. SQLST: A Spatio-Temporal Data Model and Query Language. *Conceptual Modeling - ER 2000*, pp. 96-111.

D. Lin, C. S. Jensen, B. C. Ooi, and S. Saltenis, 2005. Efficient Indexing of the Historical, Present, and Future Positions of Moving Objects. In *Mobile Data Management, MDM 2005*, To appear.

G. Faria, C. B. Medeiros, and M. A. Nascimento, 1998. An Extensible Framework for Spatio-Temporal Database Applications. *SSDBM 1998*, pp. 202-205.

J. Sun, D. Papadias, Y. Tao, and B. Liu, 2004. Querying about the Past, the Present and the Future in Spatio-Temporal Databases. In *Proc. of ICDE*, 2004, pp. 202–213.

K. Kim, S. K. Cha, and K. Kwon, 2001. Optimizing Multidimensional Index Trees for Main Memory Access. *SIGMOD Conference 2001*.

L. Forlizzi, R. H. Güting, E. Nardelli, and M. Schneider, 2000. A Data Model and Data Structure Structures for Moving Objects Databases. *Proc. ACM SIGMOD Conf 2000*, pp. 319-330.

M. Erwig, R. H. Güting, M. Schneider, and M. Vazirgiannis, 1999. Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases. *GeoInformatica*, 3(3): pp. 269-296.

M. F. Mokbel, W. G. Aref, S. E. Hambrusch, and S. Prabhakar, 2003. Towards Scalable Location-aware Services: Requirements and Research Issues. *GIS 2003*, pp. 110-117.

M. Worboys, 1994. A Unified Model for Spatial and Temporal Information. *The Computer Journal*, 37(1): pp. 25–34.

R. H. Güting, 1994. An Introduction to Spatial Database Systems, *VLDB Journal*, 3, pp 357-399.

R. H. Güting, M. H. Böhlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, and M. Vazirgiannis, 2000. A Foundation for Representing and Querying Moving Objects. *ACM Transactions on Database Systems*, 25(1): pp. 1-42.
.
S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez, 2000. Indexing the Positions of Continuously Moving Objects. *Proc. of SIGMOD*, pp. 331–342.

V. P. Chakka, A. Everspaugh, and J. M. Patel, 2003. Indexing Large Trajectory Data Sets With SETI. In *Proc. of the Conf. on Innovative Data Systems Research, CIDR*, Asilomar, CA

W. J. Choi, S. Tekinay, 2003. Location Based Services for Next Generation Wireless Mobile Networks. *IEEE 2003*, pp. 1988-1992.

X. Zhou, Z. H. Liu, Y. Xia, J. W. Ge, and H. Y. Bae, 2005. CTPR*-link tree: A Cache-Conscious TPR*-tree for Spatio-Temporal Main Memory Databases. *In Proc. Of the 3rd Asian Symposium on GISs from Computer Science & Engineering View (2005), Japan*, To appear.

Y. F. Tao, D. Papadias, and J. Sun, 2003. The TPR*-Tree: An Optimized Spatio-Temporal Access Method for Predictive Queries. *Proc. of VLDB*, pp. 790–801.

Y. Xia, G. J. Wei, and H. Y. Bae, 2004. Design and analysis of generic LBS application developing platform. *In Proc. Of the 2rd Asian Symposium on GISs from Computer Science & Engineering View (2004)*, pp 112-114

Z. H. Liu, C. H. Lee, J. W. Ge and H. Y. Bae, 2004. Indexing Large Moving Objects from Past to Future with PCFI. *In Proc. Of the 2rd Asian Symposium on GISs from Computer Science & Engineering View (2004)*, pp 25-34.

Z. H. Liu, X. L. Liu, J. W. Ge, and H. Y. Bae, 2005. Indexing Large Moving Objects from Past to Future with PCFI+-Index. COMAD 2005, pp. 131-137.