EFFICIENT MANAGEMENT OF VERY LARGE MOVING OBJECTS DATABASE

Seong-Ho Lee, Jae-Ho Lee, Kyoung-Hwan An, and Jong-Hyun Park

LBS/Telematics Convergence Research Team, Telamatics USN Research Division
Electronics and Telecommunications Research Institute, Daejeon, 205-350, KOREA
{sholee, snoopy, mobileguru, jhp}@etri.re.kr

ABSTRACT The development of GIS and Location-Based Services requires a high-level database that will be able to allow real-time access to moving objects for spatial and temporal operations. *MODB.MM* is able to meet these requirements quite adequately, providing operations with the abilities of acquiring, storing, and querying large-scale moving objects. It enables a dynamic and diverse query mechanism, including searches by region, trajectory, and temporal location of a large number of moving objects that may change their locations with time variation. Furthermore, *MODB.MM* is designed to allow for performance upon main memory and the system supports the migration on out-of-date data from main memory to disk. We define the particular query for truncation of moving objects data and design two migration methods so as to operate the main memory moving objects database system and file-based location storage system with.

KEY WORDS: Moving Objects, Main Memory Database, Migration

1. INTRODUCTION

The research on moving objects has attracted increasing attention in the community[2,5,6] in recent years. And, the location of the moving objects has been acquired by the various methods, such as GPS, RFID, USN, telecommunication, and so on. Accordingly, the demand of location-based services is gradually increased[1]. It is necessary to have to rapidly process very large moving objects data in order to be satisfied with this demand. The main memory moving objects database system can meet the demand than the existing disk-resident database system about moving objects.

However, as the time flows away because moving objects data is continuously generated and accumulated, there is a limit to store all data in the memory. It is necessary to drop historical data to the disk which is not in the main memory and can guarantee the reliability and persistence of data.

2. RELATED WORK

2.1 A Main Memory Moving Objects DBMS

We have developed the main memory moving objects database system which is one of the core technologies of the ETRI location-based service technology project[4]. This system manages massive data, such as the location of the current and past of the moving objects and geographic information.

Figure 1 show the architecture of the main memory moving objects database system(MODB.MM). This system includes the main elements as follows.

- moving object classes component : a set of temporal, geometry, and moving object classes that implement data model and operations.
- moving object SQL processor : processes and executes various and powerful moving objects queries.
- main memory storage component : transaction manager, lock manager, recovery manager, deadlock detector, index manager, and etc.

The complete explanation and specification of above elements are provided in an earlier paper[4].

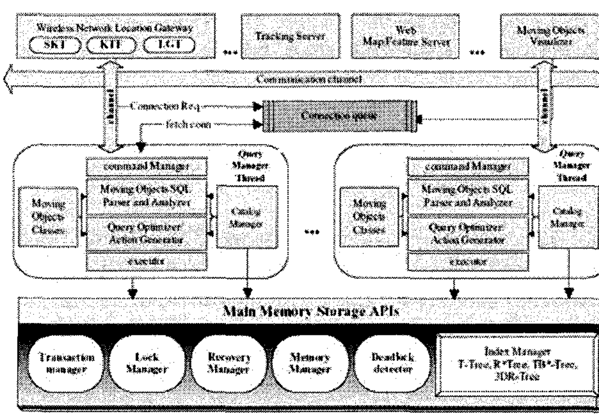

Figure 1. Architecture of Main Memory Moving Objects Database System.

2.2 A Location Information Storage System Based on File

We have developed not only a main memory moving objects database system but also a file-based location storage system[3](MODB.NET). This system provides the various *functions, such as the location* storage/retrieval, fast time query using hash and cluster, location trigger, OLEDB interface, multi user, location database import/export facilities, and so on. Figure 2 shows the architecture of file-based location storage system. This system consists of Connection Manager, Simple Query Processor, Buffer Manager, Log Manager, Location Manager, Index Manager, Meta Manager, Admin-Tool, Convert-Tool and OLE-DB Data Provider.

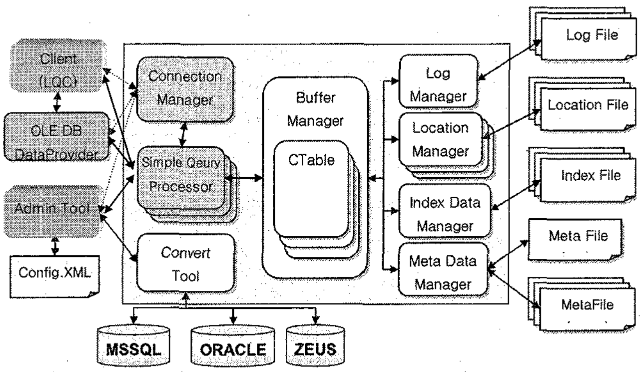

Figure 2. Architecture of File-Based Location Storage System.

2.3 Problems

When the number of moving objects is increased or the time to get location data gets longer, location data of the moving objects is very massive. Therefore, as to this system, it is necessary to have to guarantee the scalability which high-capacity moving objects data can be stored.

Of course, this system provides the backup function. But data migration function which is not in the simple backup is required. In detail, in order that we effectively manage the very large moving objects database, this system should migrate the moving objects database to the disk-resident system[3].

We added the migration function as follows in this system.

- Query language for truncation of moving objects data
- Two migration methods

The rest of this paper is organized as follows. Section 3 contains query languages for truncation of out-of-date moving objects database. In section 4, we describe two migration methods. The final section summarizes the results.

3. QUERY LANGUAGE FOR TRUNCATION

According to the flow of the time, the size of moving objects data continuously increased. In case this system continuously increases because of being the main memory based database, there is the concern in which data are lost due to the deficit of the storage.

We defined a DML in order to handle the moving objects. In this DML, insert, delete, update, and select statement are included. And the particular statement for truncation is added.

ALTER MOTABLE [*table_name*]
COMPACT UNTIL '*yyyy-mm-dd hh:mm:ss*';

Truncation is the function of being used in order to prevent the memory deficiency phenomenon of this main memory-resident system. It is the function of altogether burring historical data based on the given specific duration. It can apply to the whole moving objects table within a database and perform about the designated table. Also, this statement is processed and defined at the main memory moving objects database system.

Example 1. Truncate all moving objects tables until *2005-01-01 12:00:07.*

ALTER MOTABLE
COMPACT UNTIL '2005-01-01 12:00:07';

Example 2. Truncate the *mp2* table until *2005-01-01 12:00:07.*

ALTER MOTABLE mp2
COMPACT UNTIL '2005-01-01 12:00:07';

The truncated moving objects data from main memory moving objects database emigrates to the file-based location storage system.

4. MIGRATION METHODS

The migration module of MODB.MM consists of a *QueryChecker* and *UpdatePropagator*. Figure 3 shows the structure of the MODB.NET-linked MODB.MM through a migration function.

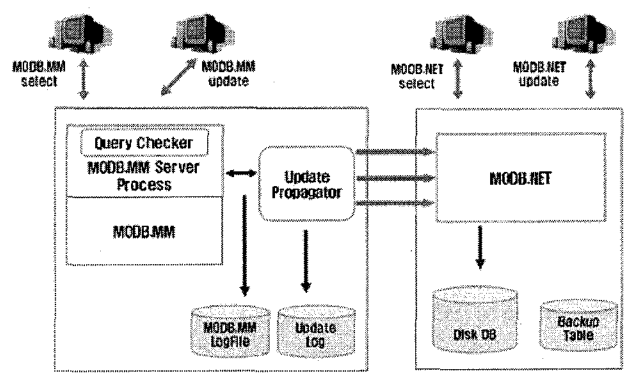

Figure 3. Structure of the link between moving objects database systems.

The update query about the duplex table in MODB.MM is delivered to MODB.MM. While a query is processed at MODB.MM, the update related query is selected with the *QueryChecker* and it is delivered to the *UpdatePropagator*. The *UpdatePropagator* delivers the delivered update related query to MODB.NET and the update about the same data is performed in MODB.NET.

That is, *QueryChecker* 1) classifies update related query language and 2) maintains according to mode of update and 3) plays the role of delivering towards *UpdateProagator* in the time appropriate with.

And, it plays the role that *UpdatePropagator* 1) receives the update related query and 2) delivers the query to MODB.NET and 3) guarantees that the query is processed in MODB.NET.

Table 1. Comparison between migration methods.

	Lazy method	Eager method
Update Process	asynchronous update process	synchronous update process
Feature	fast response time	data consistency
MODB.NET's Role	backup system	link system

As shown in Table 1, the migration mode has the lazy and eager technique. In case of serving the fast response time about client, the migration module uses the lazy update method which later reflects this to MODB.NET with this firstly after the migration module renews MODB.MM. On the other hand, the eager update method is used about the application maintaining data consistency between two database systems in real-time.

Moreover, the lazy update method is used in the application when MODB.NET only does the backup role or all data processing for an update and search about the duplex tables are made through MODB.MM. In case an update is delivered to MODB.MM and is performed but a search can be generated in MODB.MM and MODB.NET both sides, the method that uses the eager technique is mainly applied.

Figure 4 shows the result on the query that retrieves the trajectories of taxies from MODB.MM for recent moving objects database and MODB.NET for old-fashioned one.

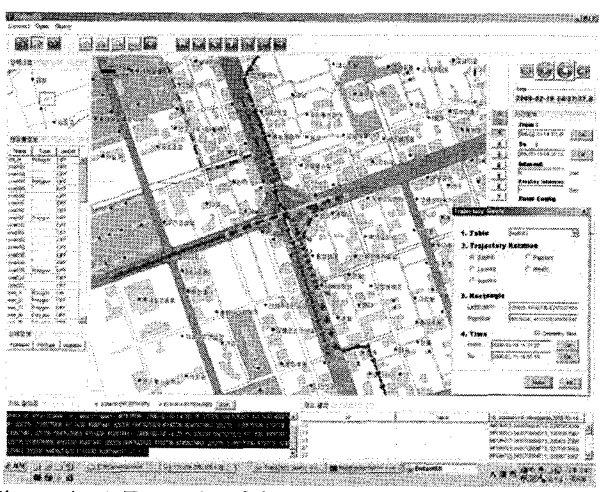

Figure 4. A Example of the trajectory query on Moving Objects.

5. CONCLUSION

In real world, the location data of moving objects very increases over varying time. We propose the methods for efficiently managing the high-capacity moving objects database. We apply the cooperate-technology of the already developed systems(MODB.MM and MODB.NET), the truncation SQL DML, and two migration methods.

We defined the particular query statement for truncation of out-of-date moving objects data. The truncated moving objects database emigrates to the file-based location storage system. And we designed two migration methods, that is, the lazy and eager update technologies. Thus we could operate the main memory moving objects database system and file-based location storage system with.

We expect that this system can be used for the various filed with the location-based services, data mining of moving objects as well as the telematics services, sensor network, and etc.

References

[1] Erwig, M., Guiting, R. H., Schneider, M., and Vazirgiannis, M., 1999, Spatio-Temporal Data Types : An Approach to Modeling and Querying Moving Obejct in Databases, *GeoInfomatica*, 3(3), pp.269-296.

[2] G.S. Iwerks, H.Samet, and K.Smith., 2003, Continuous k-nearest neighbour queries for continuously moving points with updates. *VLDB*, 31(2), pp.512-523.

[3] Jang, I.S. Cho,D.S., 2004, Lcoation Information Storage System Based On File. *ISPRS*, 35(4), pp.1307-3111.

[4] Nam, K. W. Lee, J. H. Lee, S. H. Lee, J. W. and Park, J. H., 2004, Developing a Main Memory Moving Objects DBMS for High-Performance Location-Based Services. LNCS(3007), pp.864-873.

[5] S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez., 2000, Indexing the positions of continuously moving objects. *SIGMOD*, pp.331-342.

[6] Y. Tao and D. Papadias., 2002, Time-parameterized queries in spatial-temporal databases. *SIGMOD*, pp.334-345.