Design of Memory-Resident GIS Database Systems

J. H. Lee
ETRI
161, Kajeongdong, Yuseonggu, Daejeon, Korea, 305-350
snoopy@etri.re.kr

K. W. Nam
ETRI
161, Kajeongdong, Yuseonggu, Daejeon, Korea, 305-350
kwnam@etri.re.kr

S. H. Lee
ETRI
161, Kajeongdong, Yuseonggu, Daejeon, Korea, 305-350
sholee@etri.re.kr

J. H. Park
ETRI
161, Kajeongdong, Yuseonggu, Daejeon, Korea, 305-350
jhp@etri.re.kr

Abstract: As semiconductor memory becomes cheaper, the memory capacity of computer system is increasing. Therefore computer system has sufficient memory for a plentiful spatial data. With emerging spatial application required high performance, this paper presents a GIS database system in main memory. Memory residence can provide both functionality and performance for a database management system.

This paper describes design of DBMS for storing, querying, managing and analyzing for spatial and non-spatial data in main-memory. This memory resident GIS DBMS supports SQL for spatial query, spatial data model, spatial index and interface for GIS tool or applications.

Keywords: MMDB, GIS.

1. Introduction

Recently, new applications using spatial data such as location based services(LBS) or intelligent transportation system(ITS) require real-time access to database as well as spatial operation. For example, if we provide location and path of restaurant to a man who look for restaurant so as to eat lunch, we need to continuously update current location of man through wireless networks and are required operation of finding path in order to provide path for him. As before, high-performance access to data is necessary so as to providing frequent update operations and spatial operations simultaneously. And the increasing availability of large and relatively cheap memory suggests that spatial database could reside entirely or almost entirely in main memory. So, in this paper, we propose the GISDB.MM, which is main memory based spatial database system, for high-performance access and manage to spatial data.

The outline of the paper is as follows. Section 2 reviews past works for main memory DBMS and spatial DBMS. Section 3 presents the architecture of GISDB.MM. Section 4 describes spatial data model. Section 5 describes method of storing variable length spatial data. Finally, section 6 summarizes the paper with future work.

2. Related Work

We briefly describe difference of between main memory database system and conventional database system (DRDB) it is disk resident [1]. Disks have a high, fixed cost per access that does not depend on the amount of data that is retrieved during the access. For this reason, disks are block-oriented storage devices. Main memory is not. The layout of data on a disk is much more critical than layout of data in main memory, since sequential access to a disk is faster than random access. Sequential access is not as important in main memories. Main memory is normally directly accessibly by the processor. With steadily increasing availability and cheap memory, MMDB have been designed or implemented in consideration of memory properties. Therefore, many research results come out in order to improve performance, such as index, currency control, recovery and so on.

Many researches study spatial database system and we can use many commercial spatial database systems for example Oracle Spatial, ESRI ArcSDE, DB2 Spatial Extender, Zeus, Informix Spatial database, GeoToolKit. However, most of them are based on the traditional disk-based database architecture. A high-performance spatial storage system based on the main-memory database system architecture appeared like Xmas-SX[2]. Xmas-SX is a spatial extension of an extensible main-memory storage system named Xmas[3] and provide spatial data type following the OpenGIS geometry model[6,7], spatial operators, spatial indexes and client programming interfaces.

3. GISDB.MM Architecture

Fig. 1 shows the overall architecture of the GISDB.MM that is designed in consideration of extending to treat moving objects. And GISDB.MM server process is multi-threaded to run multiple transactions concurrently. The functions of each module are as follows.

1) User interface module

The GISDB.MM provides a few of standard interfaces such as ODBC, JDBC, CLI. Specially, it provides data provider following Open GIS OLE/COM standard. Therefore variable applications access it through same standard interfaces.

2) Query processing module

The Query processing module process SQL and take OpenGIS Simple Feature Specification for SQL[7]. The Query processing module consists of Query Dispatcher, Parser, Query Analyzer, Execution Planner, Query Optimizer and Catalog Manager.

The Query Dispatcher accepts connection and query from user and sends other module in order to processing query.

The Parser creates a parse tree after perform semantic and syntax analysis about accepted query language.

The Query Analyzer analyzes the parse tree created by the parser module, confirms catalog information about each query elements and judges to go on query processing by confirming consistency.

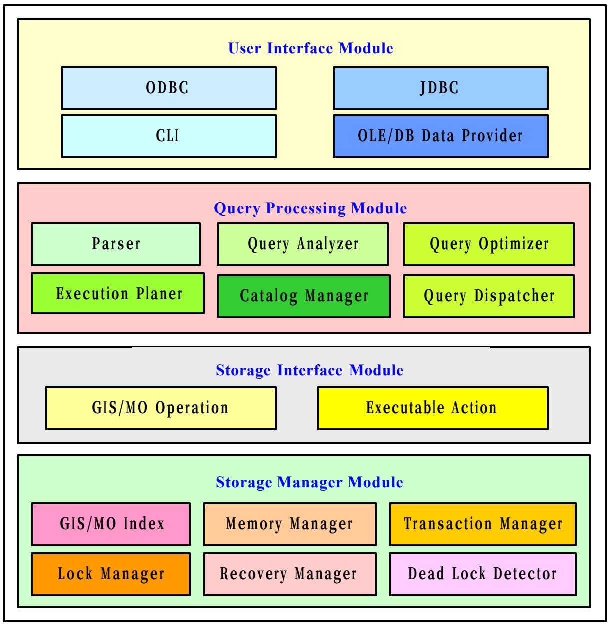

Fig. 1. GISDB.MM Architecture.

The Execution planner alters each query elements into procedures that the storage manager can execute and makes a decision execution order of procedures.

The Query Optimizer makes an optimized execution path. It selects a path by selection standard such as cost-based or heuristic.

The Catalog manager manages metadata of database object (ex. table, index, restriction) in database.

3) Storage interface module

The storage interface module has the GIS/MO operation module and the Execution Action module.

The GIS/MO operation module defines spatial operator about spatial data type and provides interface called by query processing module. Now we handle spatial operator about spatial object, but later we will add spatio-temporal operator about moving object. We will deal with spatial object model and operator in next section.

The Executable action creates jobs of action unit that executed in storage manager module.

3) Storage manager module

The Storage manager module consists of the GIS/MO index, the Memory manager, the Transaction manager, the Recovery manager, the lock manager, and Dead lock detector.

The GIS/MO index deals with spatial index and later will be added spatio-temporal index. We implement R*-tree spatial index. The R*-tree is adjusted according to main memory environment. The index about non-spatial data is applied T-tree index and ECBH(Enhanced Chained Bucket Hashing).

The Memory manager manages physical memory and map to it.

The Transaction manager deals with processing and manager transaction.

The Recovery manager recovers from system error and guarantees data consistency.

The Lock manager is concurrency control module. The function of this module is concurrency control between tables or records in multi transaction environment. We apply 2-Phase Locking protocol.

The Dead lock Detector determines whether the system has entered a deadlock state and recovers from it.

4. Data Model

The GISDB.MM take OpenGIS geometry model in order to express the general spatial objects. Fig 2 shows spatial object type of GISDB.MM following OpenGIS geometry model.

The GISDB.MM defined abstract class such as SpatialRelation, SpatialOperator and WKS.

The SpatialRelation class provides the methods for spatial relation operation such as Contains, Crosses, Disjoint, Equals, Intersects, Overlaps, Touches and Within.

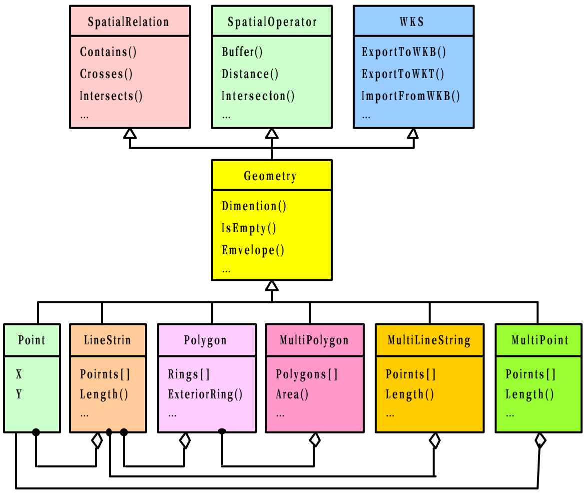

Fig. 2. Spatial object model

The SpatialOperator class is made of spatial operators. There are Buffer, convexHull, Difference, Distance, Intersection, Symdifference and Union.

The WKS deal with well-known expression of spatial objects. Methods of WKS class have methods such as ExportToWKB, ExportToWKT, ImportFromWKB, and ImportFromWKT.

Those surrounded by bold line such as Point, LineString, Polygon represents concrete types, which can be instantiated. We abbreviate description of spatial object because it follows OpenGIS geometry model.

6. Spatial Data Management

Generally, spatial data is variable-length. So the efficient handling of variable-length data is needed. Fig 3 shows how to store spatial object in GISDB.MM. Tables of database are made of fixed-length records, and records consist of fixed-length fields. Because length of field stored spatial objects is fixed, in case a field does not accommodate spatial object, we will store it in BLOB(binary large object) form at continuous memory space.

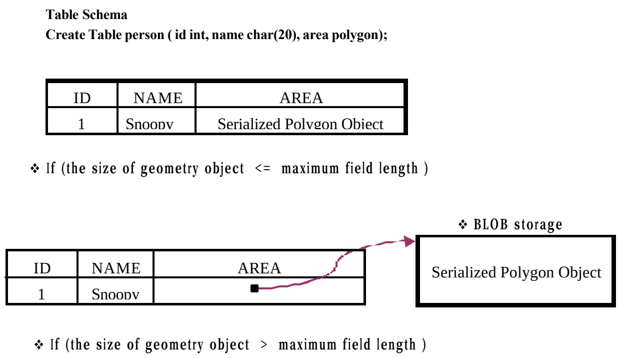

Fig. 3. Storage structure of spatial objects

6. Summary and Future Work

We designed high-performance spatial database system based main memory. We took consideration into extending to treat moving objects and are implementing this system. After we implement, we have to work about index, concurrency control and storage management of moving object in main memory environment.

References

[1] A Hector Garcia-Molina, Kenneth Salem: Main Memory Database Systems: An Overview. TKDE 4(6): 509-516 (1992)

[2] J.H. Park, K. Kim, S. K Cha, M.S. Song, S. Lee, and J. Lee, "A High-performance Spatial Storage System Based on Main-Memory Database Architecture", In Proceedings of DEXA Conference, 1999

[3] J. H. Park, Y. Sik Kwon, K. Kim, S. Lee, B. D. Park, and S. K. Cha, "Xmas: An Extensible Main-memroy Storage System for High-Performance Applications", In Proceedings of ACM SIGMOD Conference, 1998.

[4] H. V. Jagadish, D. Lieuwen, R. Rastogi, and A. Silberschartz, "Daili: A High Performance Main Memory Storage Manager", In Proceedings of VLDB Conference, 1998

[5] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, Bernhard Seeger: The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. SIGMOD Conference 1990: 322-331

[6] OpenGIS Implementaion Specifications : OpenGIS Simple Features Specification For OLE/COM Revision 1.1, OpenGIS Consortium, Inc. 1999

[7] OpenGIS Implementaion Specifications : OpenGIS® Simple Features Specification for SQL Revision 1.1, OpenGIS Consortium, Inc. 1999