RESEARCH ON DATA CONSISTENCY IN SPATIAL DATABASE SYSTEM

ZHU Xinyan
WEN Yi
LI Deren
GONG Jianya

KEY WORDS data sharing; data consistency; lock; UNDO/REDO transaction; notification-reread method

ABSTRACT In this paper, two cases of data consistency problems are discussed in Multi-user Geographical Information System. When a user updates some spatial data that have been accessed by other users in database, or executes an UNDO/REDO operation, data consistency will probably be destroyed. To solve the problem caused by the first case, a "Notification-Reread Method" is introduced. As for the second case, there are many problems to be studied.

1 Introduction

Geographical Information System (GIS) is widely used in many fields. With the rapid development of computer network, GIS users care more about data sharing in networks. In traditional relational database, data consistency was controlled by consistency control mechanism when a data object is locked in a sharing mode, other transactions can only read it, but can not update it. This is appropriate in traditional relational databases that store attribute data and mainly deal with short transactions. In spatial databases, because of vast amount of data and complex topological relations, long transaction are met frequently. If the traditional consistency control method has been used yet, the system’s concurrency will be badly influenced. So there come many new requirements for the consistency control in the field of GIS. There are many aspects of data consistency problems in spatial databases, such as the inconsistency between attribute and geometry data; the inconsistency of topological relations after geometry objects has been modified. In this paper, other two cases of data consistency are discussed in Multi-user Geographical Information System.

In GIS, there are many forms of data, such as geometry data, attribute, image data, and DEM data. In this paper, we only discuss spatial geometry data.

2 Update consistency

Considering the case in Fig. 1, a group of users are manipulating spatial data in the same range. Suppose that there is a data object $D$, at one moment, one user wants to update $D$, but at this time, other users have read it.

![Fig. 1 A block of spatial data accessed by multi-users](image-url)
2.1 Traditional method in relational databases

In traditional relational databases, DBMS deals with this problem by concurrency control protocol\cite{1}. Transaction was introduced to implement this control protocol\cite{2}. In order to prevent any transaction from reading or updating data that is being updated by another transaction we require a locking mechanism, which guarantees a transaction exclusive access to an item of data while a lock is in force. There are two kinds of locks: read-lock and write-lock. A read-lock (sharing lock) gives read-only access to a data item and prevents any other transaction from updating the item. Any number of transaction may hold a read-lock (exclusive lock). A write-lock gives read/write access to a data item, while in force and prevents any other transaction from reading or writing for the data item. It gives a transaction exclusive access to a data item.

Using the traditional locking mechanism means if any one of the users read the object \( D \), other users can not update it. In traditional relational databases which mainly deal with attribute, and as the data items that a transaction concerned usually are not so many, long transactions are not frequently happened, thus the locking mechanism works well. In spatial database, because of vast volume of geometry data, usually, users manipulate such a large scope of data items and hold for such a long time that long transactions are met frequently, the traditional locking mechanism is no longer satisfying in spatial database management. For example, a user read 16 maps at 1:10 000 scale for display or spatial analysis, one way is holding a read-lock for these maps and keep them in client memory until the operations end, in this case, other users cannot update these 16 maps, so the concurrency of the system is reduced. Another way is to lock a spatial object when it is used, and the other objects are not kept in the client memory. In this case, we need to read object from the spatial database frequently and the system would be of low efficiency.

2.2 Notification-reread method

The locking system in the traditional database management system is conventional. The basic premise is that the inconsistency and imperfection of data is not allowed without considering how to solve the problem once it happens. To a certain extent, the traditional database management system is very passive. In fact, if DBMS is able to feed back the database update (adding, deletion, modification) of the committed transactions to some application programs of relative transactions, the problems may be solved to a large extent.

Suppose we do some modifications to the locking system as follows: data with a sharing lock may allow a transaction (at most one transaction) to add an exclusive lock. This kind of modification will certainly cause the data inconsistency. As Fig. 1 shows, user 1 reads data \( D \) for query, and holds them until client process ends, while user 2 reads data \( D \) for modification. Once the modification transaction is successfully committed, data in the database will be inconsistent with the data on the customer side of user 1, because user 1 does not know the modification transaction. In order to avoid this situation, after the modification has been successfully committed, the modified information must be feed back to user 1 in order to keep the data consistency.

The key point of the improvement is to inform client side of the database modification promptly. Send client sides related to the modified data a message so that client sides will reread the modified data after receiving the message. One method is to reread all data, which, however, will cause low efficiency of the system. The other method is to reread the modified part of the data. It is demanded that the notification message should include information of the modified data. In the object-oriented system, information of the modified data can be represented by object identification.

2.3 Realization of notification-reread method

The notification-reread method must be supported by both DBMS and the client side. When implementing the operation, DBMS and the client programs should be improved accordingly.

2.3.1 Improvement of DBMS

Under the general circumstance of multi-user environment, DBMS on the server end should maintain the computer number, user number and process number of client task. The key to realize the notification-reread method is to inform the client process