ABSTRACT

In this study, we present concurrency control algorithms based on data flow graphs for distributed and replicated databases.

Graphical representations such as token models and structure models are widely applied to represent data flow programs. Data flow programs are represented by directed graphs called data flow graphs where nodes represent the operations to be performed and arcs represent the data dependencies between them. The scheduling of operations is constrained by the data dependencies identified by the graph. So, data flow scheduling mechanism could execute operations based upon the dependency constraints defined by the data flow graph. In this way, concurrency is achieved through graphs.

In this study we explore the use of data flow graphs to design concurrency control algorithms. Precedence graphs are often used to determine serializability, and wait-for-graphs are often used to detect deadlocks. Among these, acyclic precedence graphs indicate serializability, and acyclic wait-for-graphs imply freedom from deadlocks. The tools for serializability theory, and transaction management coupled with data flow graphs will provide solutions that are more efficient than conventional concurrency control techniques, such as, two-phase locking. In the same light, we apply data flow graphs for concurrency control for distributed, and replicated databases.

At first, we propose a concurrency control algorithm based on data flow graphs for a distributed database system. The concurrency control approaches based on locking, require additional efforts in deadlock detection and its elimination. The deadlock resolution protocols associated with deadlock detection algorithms abort (restart) some transactions in the deadlock
cycle. The restarted transaction must release all its locks and send out the requests again. The possibility of a deadlock is also connected to existence of unpredictable delays and repeated restarts of transactions in a deadlock cycle. In the proposed approach, a technique for construction of local access graphs at various data sites is presented and analyzed. The deadlocks are removed instantly without rejecting the transactions. The local access graph based approach is a fully distributed approach. The proposed technique is free from deadlocks and does not lead to abort or rejection of any transaction.

Apart from this, we extend the application of data flow graphs to coordinate transactions for replicated databases. Replicated data management systems adopt the one-copy serializability criteria for processing transactions. In order to achieve this goal, many approaches rely on obtaining votes from other sites, for processing transactions. In the voting based approaches, if there is a conflict among a group of transactions, then one of the transactions is accepted and remaining transactions are rejected. The rejected transactions are resubmitted again, and by this, these incur additional processing delays and overheads. In the proposed approach, a technique for generation of transaction data flow graph for each transaction execution is analyzed. The proposed approach is also a fully distributed approach. In this, the transactions are ordered with the help of transaction data flow graphs. The technique is free from deadlocks, and avoids resubmission of transactions.

The commit processing in the new environment has also been discussed. The behavior of the proposed approaches in the event of site and partition failures has been examined.