

Concurrency Control in Real-Time Database Systems

Piotr Krzyżagórski

Poznań University of Technology, Institute of Computing Science

Piotrowo 3A, 60-965 Poznań, Poland

Piotr.Krzyzagorski@cs.put.poznan.pl

Real-time systems (RTSs) form an important class of computer systems. They are characterized by existence of timing requirements concerning task execution times, and processing data items which values and validity change in time. We observe the growing importance of RTS applications which is accompanied with both the increase of data volumes processed by the systems and the requirements concerning durability, security and consistency of the processed data. So far, such requirements have been typical for traditional database systems (DBSs). Now, there is a requirement to create systems which could integrate the features of both RTSs and DBSs. Such systems are called real-time database systems (RTDBSs). A real-time database system (RTDBS) is a database system designed to process transactions having timing constraints associated with them, accessing data items which values and validity change in time, supporting data descriptions (which help avoiding data redundancy), data correctness and integrity maintenance methods, efficient data access and management techniques, and guarantee the correct execution of transactions in spite of concurrency and failures [1]. In RTDBSs three categories of transactions can be distinguished: hard deadline transactions for which missing deadlines is equivalent to a catastrophe, soft deadline transaction which are still valuable after missing deadline (although the value is reduced) and firm deadline transactions which have no value if the deadline is missed.

Modern applications of RTDBSs concern military command and control, aerospace systems, aircraft control, traffic control, telecommunication and computer network management, telephone directory service systems, computer integrated manufacturing (CIM), factory automation and robotics, workflow systems, medical monitoring, stock arbitrage systems, multimedia systems. All the above applications make substantial use of databases, but also require support for timeliness, which is not provided by traditional databases.

The wider application of RTDBSs in practice depends on solving several important problems concerning the performance requirements of such systems, and specially meeting transaction deadlines. In computer systems the performance can be increased by allowing concurrent execution of operations. The activity of coordinating concurrent access of transactions to shared data in DBSs is called concurrency control. Concurrency control algorithms should ensure consistency of the database and the correct completion of each transaction initiated in the system. Concurrency control protocols resolve data access conflicts in a manner that induces a serialization order among the conflicting transactions. The problem of concurrency control in traditional DBSs has been intensively studied and presented in literature [2], [3].

Concurrency control in RTDBSs is more complex than in traditional DBS. Apart from ensuring database consistency also transaction timing constraints (transaction deadlines) should be guaranteed. Moreover, validity and values of data items change in time.

Concurrency control algorithms for RTDBSs are in general some extensions or combinations of traditional concurrency control techniques, i.e. two phase locking (2PL), optimistic concurrency control (OCC) or timestamp ordering (TO), which guarantee a serialization order among conflicting transactions. Most of these algorithms were developed only for firm deadline RTDBS which is the simplest environment, and thus provide a foundation for more complex soft deadline systems [4].

The most important problem faced in RTDBSs concerns the degradation of system performance due to aborts and restarts of transactions. These restarts are caused by concurrency control protocols trying to resolve conflicts between transactions [5], [6].

The objective of this Ph.D. thesis is to present new, more efficient concurrency control algorithms for firm deadline RTDBSs. The analysis of currently used methods allows to propose two new research directions:

- dynamic serialization adjustment,
- partial abort.

The former direction assumes that it is not necessary to guarantee that the serialization order of concurrently executed transactions is the same as their execution order. Traditional concurrency control protocols are generally based on the serializability criterion. The criterion is easy to implement and obviously correct, however not sufficient for RTDBSs in which the correctness of the database operations depends also on the time the results are delivered. To satisfy the requirement of real-time scheduling it is sufficient that only the execution order reflects transaction priorities. So, if a serialization order which is compatible with the execution order can be found it means that the set of transactions has been executed correctly. We propose new real-time optimistic concurrency control algorithm with dynamic serialization adjustment which tries to minimize the number of aborted and restarted transactions and thus to improve the performance [7]. In our dynamic serialization adjustment method if the validation procedure cannot serialize conflicting transactions after committed transactions (according to the execution order) then an alternative serialization order is searched for to prove the correctness of the execution.

The latter research direction is based on the observation that although restarts caused by concurrency control algorithms trying to resolve conflicts between transactions are difficult to avoid, however their influence on the system performance may be reduced. In existing concurrency control algorithms the actual range of conflicts is not taken into account. We propose a new approach - called partial abort - which is aimed to minimize the costs of transaction restarts by saving the part of previously performed work [8]. Each conflicting transaction identifies the conflict-independent subset of previously performed operations, and resumes execution saving this subset.

It is obvious that the methods proposed above do not exhaust all possibilities of the performance improvement of firm deadline RTDBSs. They provide a foundation for more sophisticated investigations. Further investigations will migrate also towards significantly more complex soft deadline RTDBSs.

References

- [1] Ramamritham, K., "Real-Time Databases", Distributed and Parallel Databases Vol. 1, no. 2, 1993, pp. 199-226.
- [2] Bernstein, P.A., Hadzilacos, V., and Goodman, N., "Concurrency Control and Recovery in Database Systems", Reading, MA: Addison Wesley, 1987.

- [3] Cellary, W., Gelenbe, E., and Morzy, T., "Concurrency Control in Distributed Database Systems", North Holland Pub. Co., 1988.
- [4] Haritsa, J.R., Carey, M.J., and Livny, M., "Data Access Scheduling in Firm Real-Time Database Systems", *The Journal of Real-Time Systems*, no. 4, 1992, pp. 203-241.
- [5] Lee, J., and Son, S.H., "Performance of Concurrency Control Algorithms for Real-Time Database Systems", Kumar V. (ed.) *Performance of Concurrency Control Mechanisms in Centralized Database Systems*, Prentice Hall, 1995.
- [6] Özsoyoglu, G., and Sondgrass, R.T., "Temporal and Real-Time Databases: A Survey", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 7, no. 4, 1995. pp. 513-532.
- [7] Krzyżagórski, P., Morzy, T., "Optimistic concurrency control algorithm with dynamic serialization adjustment for firm deadline real-time database systems", *Proceedings of ADBIS'95*, Moscow, June 1995, also Springer Verlag, Series: Workshops in Computing, pp. 27-42.
- [8] Krzyżagórski, P., Morzy, T., "Two Phase Locking-Based Algorithm with Partial Abort for Firm Deadline Real-Time Database Systems", *Proceedings of the First East-European Symposium on Advances in Databases and Information Systems - ADBIS'97*, St. Petersburg, September 1997, pp. 40-46; also Springer Verlag electronic Workshops in Computing.