Current Research on Real-Time Databases

Özgür Ulusoy
Department of Computer Science
University of Illinois at Urbana–Champaign
1304 W.Springfield, Urbana, IL 61801
e-mail: ulusoy@cs.uiuc.edu

1. Introduction

Real-time systems is becoming one of many application areas whose development requires the involvement of database system concepts. Due to ever increasing volume of information being handled by data-intensive real-time applications, a need has arisen for applying database technology to real-time systems. Examples of data-intensive real-time applications include stock market, computer-integrated manufacturing, telephone switching systems, network management, and command and control systems [Ram92].

A real-time database system (RTDBS) can be defined as a database system where transactions are associated with real-time constraints typically in the form of deadlines. The system attempts to execute transactions so as to both meet the deadlines and maintain the database consistency. Traditional database systems are designed to provide functionally correct information. The correctness of a database operation is hardly affected by the slowness of a transaction [Lin89]. Maintaining data consistency is the primary consideration in scheduling transactions. On the other hand, the basic issue considered in traditional real-time systems is the satisfaction of timing constraints associated with transactions. The problem of maintaining the consistency of shared data is usually not addressed. Real-time systems are typically categorized into two groups as *hard* and *soft* real-time systems [Sta88]. Hard real-time systems incorporate strict deadlines and the correctness of transaction operations depends on the time at which the results are produced. In soft real-time systems, satisfaction of timing constraints is still a key issue in scheduling transactions, however, in this case there is no guarantee that all transaction deadlines will be met.

Design of a RTDBS requires the integration of concepts from both real-time systems and database systems to handle the timing and consistency requirements together. The following section provides a discussion of various design approaches to RTDBS's. It is useful to study those approaches in two separate categories based on the type of timing constraints (i.e., hard or soft) issued by the underlying application.

2. Approaches to RTDBS Design

Implementation of RTDBS's is difficult due to the conflicting requirements of meeting deadlines and maintaining data consistency. In many applications, it is not possible to satisfy both requirements [Lin89]. For some applications the deadline requirement cannot be relaxed (i.e., timing constraints are hard); thus, the data consistency requirement has to be modified. In those systems getting timely but partially incorrect information is preferable to getting correct but late information [Sin88]. On the other hand, in some RTDBS applications maintaining data consistency can be more crucial than satisfying deadlines. Schedulers should not violate the data consistency requirement while observing the timing constraints of transactions. In the following subsections, various design alternatives are discussed for both types of RTDBS's.

2.1. RTDBS's with Hard Timing Constraints

With the current database technology it is extremely difficult to provide schedules guaranteeing the deadlines of RTDB transactions. This difficulty comes from the unpredictability of transaction response times. Each transaction operation accessing a data item takes a variable amount of time due to concurrency control and disk IO [Sta88].

Serializability is a widely accepted correctness criterion for concurrency control in database systems. Serializable schedules provide correct results and leave the database consistent. However, serializability is not a suitable technique to implement in hard real-time systems because of the limitation of concurrency allowed by serializable executions. Existing concurrency control protocols ensuring serializability are based on either one of two techniques: blocking transactions and restarting transactions. Both techniques are inappropriate for time-critical scheduling. Blocking can cause priority inversion; i.e., a high priority transaction (e.g., with an urgent deadline) can be blocked by a lower priority transaction [Sha88]. Restarting a transaction, on the other hand, causes a waste of processing time and other system resources already used by that transaction.

Although a general purpose consistency criterion that is less stringent than serializability has not yet been proposed, some possible solutions to the data consistency problem in hard RTDBS's have been provided. The consistency model proposed in [Vrb88, Lin89] is an attempt for the relaxation of strict serializability rules. The model satisfies timing constraints by sacrificing database consistency temporarily to some degree. *External* data consistency is defined in contrast to *internal* data consistency as maintained by traditional database systems. The external consistency constraint requires that the data used by a transaction reflect the physical environment at the time; this is in contrast to internal consistency which requires that all data must meet some predefined constraints in the database. The model is based on the assumption that for most RTDBS applications a timely and externally consistent result is more desirable than an out-of-date though internally consistent response. For instance, the trace of an unidentified object detected by an on-board system is externally consistent but may not be internally consistent before it is interpreted and filtered by the system.

Another approach to designing a RTDBS with extremely fast response requirements is to redesign conventional database systems or their components. Main memory database systems are capable of providing fast response for RTDBS applications. As we mentioned above, one of the factors leading to unpredictable transaction response times is disk IO. Main memory database systems can eliminate disk access delays from database access. However, main memory databases introduce some problems and design issues of their own [Sin88].

2.2. RTDBS's with Soft Timing Constraints

In many RTDBS applications consistency of data is as important as the timeliness of transaction response. Banking, stock market, and airline reservation systems are several examples of such application areas where the data consistency requirement cannot be relaxed. While maintaining the consistency of underlying database, the management and scheduling of system resources should also take the timing constraints into account; i.e., CPU, main memory, IO devices and access to shared data all should be managed to make a best effort to satisfy transaction deadlines [Abb90].

The general approach to the scheduling problem in soft RTDBS's is using existing techniques in CPU scheduling, buffer management, IO scheduling, and concurrency control, and to apply time-critical scheduling methods to observe the timing requirements of transactions. The performance goal in satisfying timing constraints can change depending on the application environ-

ment. If the only real-time parameter associated with each transaction is the assigned deadline, the goal is to minimize the number of transactions that miss their deadlines. A priority order is established among transactions based on their deadlines. There are RTDB applications where transactions may be assigned different values, where the value of a transaction reflects the return the application expects to receive if the transaction commits within its deadline [Biy88, Hua89, Har91]. For such applications, the performance goal is to maximize the value realized by the in-time transactions. Transactions are assigned priorities which are functions of both their values and deadlines.

Most of the recent research in soft RTDB transaction scheduling has concentrated on development and evaluation of concurrency control protocols. Each locking protocol proposed for RTDBS's is based on either one of the following two schemes: *priority inheritance* and *priority abort*. The priority inheritance scheme allows a low priority transaction to execute at the highest priority of all the higher priority transactions it blocks [Sha88]. The priority abort scheme is based on aborting the lower priority transaction when priority inversion occurs. Development and evaluation of various locking-based concurrency control protocols are reported in [Abb88, Abb89, Hua89, Hua91a, Lin90, Sha91, Son90a, Son92, Agr92, and Ulu92a]. One well-known locking-based concurrency control protocol is the *priority ceiling* protocol which extends priority inheritance by eliminating deadlocks and bounding the blocking time of high priority transactions to no more than one transaction execution time [Sha88, Sha90]. Another class of proposed protocols is based on the optimistic method of concurrency control. [Har90a, Har90b, and Hua91b] present a set of optimistic protocols and provide the comparison of those protocols with locking. These performance comparisons do not completely agree due to the different types of systems used and the assumptions made in evaluating the protocols. Timestamp-based concurrency control protocols, which involve real-time priorities in constructing a timestamp order among transactions, are studied in [Son90b and Ulu92b].

Other recent research addressing the scheduling problem in soft RTDBS's can be summarized as follows. [Car89, Abb90, Che91, and Kim91] provide some new approaches to priority-based IO scheduling. Development and evaluation of several real-time policies for handling CPU scheduling is reported in [Hua89]. [Özs90] introduces new techniques to process database queries within fixed time quotas. Different degrees of accuracy of the responses to the queries can be achieved by using those techniques. Priority-based buffer management policies are discussed in [Car89 and Hua90].

3. Active RTDBS's

In an active database system conditions are defined on states of the database and when these conditions occur, some prespecified actions are invoked. The HIPAC project [Day88] introduces an active RTDBS that attempts to provide timely response to specified conditions. When conditions become true, transactions are scheduled both to perform corresponding actions and to meet certain timing constraints.

In [Kor90], a new approach is proposed to the modeling of an active RTDBS. In the proposed model timing constraints are associated with the states of the database system rather than directly with transactions. Timing constraints defined on system states enforce similar constraints on transactions that are triggered by those states.

4. Distributed RTDBS's

Distributed database systems are of great importance to real-time applications with severe performance requirements such as fast response and continued operation in the face of catastrophic failures. Two basic advantages provided by distributed database systems are high data availability and potentially improved read performance [Sin88]. Replicated copies of data can be stored redundantly at multiple sites, providing data access even if some of the copies are not

available due to failures. Improved read performance is due to access to local copies.

Although data replication provides some advantages, it also introduces some problems [Gar87]. Access to a data item is no longer controlled exclusively by a single site; instead the access control is distributed across the sites storing the copies of the data item. It is necessary to ensure that *mutual consistency* of the replicated data is provided; in other words, replicated copies must behave like a single copy. This is possible by preventing conflicting accesses on the different copies of the same data item, and by making sure that all data sites eventually receive all updates. Multiple copy updates lead to a considerable overhead due to the communication required among the data sites holding the copies. [Ulu92b] investigates the impact of storing multiple copies of data on satisfying the timing constraints of transactions under various application environments.

Some recent RTDBS research has concentrated on investigating the possibility of improving the performance in distributed RTDBS's. In [Lin88] some techniques are proposed to increase the availability in a partitioned RTDBS. [Kim91] proposes new multiversion concurrency control protocols to increase concurrency in RTDBS's. In [Sop92] an adaptive commit protocol is introduced for distributed RTDB transactions. The approach taken in that work is to identify ways in which a commit protocol can be made adaptive in the sense that under different situations the system can dynamically change to a different commitment strategy.

5. Conclusions

A real-time database system (RTDBS) is designed to provide real-time information to the executing transactions. The system attempts to satisfy the properties of both traditional database systems and real-time systems; however, the integration of those two systems is not trivial. Traditional database systems focus on maintaining database consistency in scheduling transactions. The users are not allowed to specify timing constraints. In real-time system scheduling, on the other hand, the emphasis is on satisfying transaction deadlines and the problem of maintaining the consistency of shared data is usually not addressed. In this paper, we have discussed briefly the main problems and the basic approaches taken in designing a RTDBS. The design approaches have been categorized into two groups based on the strictness of the timing constraints associated with transactions. We have also summarized recent research in active and distributed RTDBS's.

Acknowledgement

I would like to thank Prof. Geneva G. Belford for her helpful comments on earlier versions of this paper.

References

[Abb88]: R. Abbott, H. Garcia-Molina 'Scheduling Real-Time Transactions: A Performance Evaluation', *14th International Conference on Very Large Data Bases*, 1988, pp.1-12.

[Abb89]: R. Abbott, H. Garcia-Molina 'Scheduling Real-Time Transactions with Disk Resident Data', *15th International Conference on Very Large Data Bases*, 1989, pp.385-396.

[Abb90]: R. Abbott, H. Garcia-Molina 'Scheduling I/O Requests with Deadlines: A Performance Evaluation', *11th Real-Time Systems Symposium*, 1990, pp.113-124.

[Agr92]: D. Agrawal, A. El Abbadi, R. Jeffers 'Using Delayed Commitment in Locking Protocols for Real-Time Databases', *ACM SIGMOD Conference*, 1992, pp.104-113.

[Biy88]: S.R. Biyabani, J.A. Stankovic, K. Ramamritham 'The Integration of Deadline and Criticalness in Hard Real-Time Scheduling', *9th Real-Time Systems Symposium*, 1988, pp.152-160.

[Car89]: M.J. Carey, R. Jauhari, M. Livny 'Priority in DBMS Resource Scheduling', *15th International Conference on Very Large Data Bases*,

1989, pp.397-410.

[Che91]: S. Chen, J.A. Stankovic, J. Kurose, D. Townley 'Performance Evaluation of Two New Disk Scheduling Algorithms for Real-Time Systems', *Real-Time Systems Journal*, vol.3, no.3, 1991, pp.307-336.

[Day88]: U. Dayal et al. 'The HiPAC Project: Combining Active Database and Timing Constraints', *ACM SIGMOD Record*, March 1988, pp.51-70.

[Gar87]: H. Garcia-Molina, R.K. Abbott 'Reliable Distributed Database Management', *Proceedings of the IEEE*, vol.75, no.5, 1987, pp.601-620.

[Har90a]: J.R. Haritsa, M.J. Carey, M. Livny 'On Being Optimistic About Real-Time Constraints', *ACM SIGACT-SIGMOD-SIGART*, 1990, pp.331-343.

[Har90b]: J.R. Haritsa, M.J. Carey, M. Livny 'Dynamic Real-Time Optimistic Concurrency Control', *11th Real-Time Systems Symposium*, 1990, pp.94-103.

[Har91]: J.R. Haritsa, M.J. Carey, M. Livny *Value-Based Scheduling in Real-Time Database Systems*, Technical Report No.1204, Dept. of Computer Science, University of Wisconsin-Madison, 1991.

[Hua89]: J. Huang, J.A. Stankovic, D. Towsley, K. Ramamritham 'Experimental Evaluation of Real-Time Transaction Processing', *10th Real-Time Systems Symposium*, 1989, pp.144-153.

[Hua90]: J. Huang, J.A. Stankovic *Real-Time Buffer Management*, COINS TR 90-65, Dept. of Computer and Information Science, University of Massachusetts, Amherst, 1990.

[Hua91a]: J. Huang, J.A. Stankovic, K. Ramamritham, D. Towsley 'On Using Priority Inheritance In Real-Time Databases', *12th Real-Time Systems Symposium*, 1991, pp.210-221.

[Hua91b]: J. Huang, J.A. Stankovic, K. Ramamritham, D. Towsley 'Experimental Evaluation of Real-Time Optimistic Concurrency Control Schemes', *17th International Conference on Very Large Data Bases*, 1991, pp.35-46.

[Kim91]: W. Kim, J. Srivastava 'Enhancing Real-Time DBMS Performance with Multiversion Data and Priority Based Disk Scheduling', *12th Real-Time Systems Symposium*, 1991, pp.222-231.

[Kor90]: H.F. Korth, N. Soparkar, A. Silberschatz 'Triggered Real-Time Databases with Consistency Constraints', *16th International Conference on Very Large Data Bases*, 1990, pp.71-82.

[Lin88]: K.J. Lin, M.J. Lin 'Enhancing Availability in Distributed Real-Time Databases', *ACM SIGMOD Record*, March 1988, pp.34-43.

[Lin89]: K.J. Lin 'Consistency Issues in Real-Time Database Systems', *22nd Hawaii International Conference on Systems Sciences*, 1989, pp.654-661.

[Lin90]: Y. Lin, S.H. Son 'Concurrency Control in Real-Time Databases by Dynamic Adjustment of Serialization Order', *11th Real-Time Systems Symposium*, 1990, pp.104-112.

[Özs90]: G. Özsoyoğlu, Z.M. Özsoyoğlu, W.C. Hou 'Research in Time and Error-Constrained Database Query Processing', *7th IEEE Workshop on Real-Time Operating systems and Software*, 1990, pp.32-38.

[Ram92]: K. Ramamritham 'Real-Time Databases', to appear in *International Journal of Distributed and Parallel Databases*, 1992.

[Sha88]: L. Sha, R. Rajkumar, J. Lehoczky 'Concurrency Control for Distributed Real-Time Databases', *ACM SIGMOD Record*, March 1988, pp.82-98.

[Sha90]: L. Sha, R. Rajkumar, J. Lehoczky 'Priority Inheritance Protocols: An Approach to Real-Time Synchronization', *IEEE Transaction on Computers*, vol.39, no.9, 1990, pp.1175-1185.

[Sha91]: L. Sha, R. Rajkumar, S.H. Son, C.H. Chang 'A Real-Time Locking Protocol', *IEEE*

Transactions on Computers, vol.40, no.7, 1991, pp.793-800.

[Sin88]: M. Singhal 'Issues and Approaches to Design of Real-time Database Systems', *ACM SIGMOD Record*, March 1988, pp.19-33.

[Son90a]: S.H. Son, C.H. Chang 'Performance Evaluation of Real-Time Locking Protocols Using a Distributed Software Prototyping Environment', *10th International Conference on Distributed Computing Systems*, 1990, pp.124-131.

[Son90b]: S.H. Son, J. Lee 'Scheduling Real-Time Transactions in Distributed Database Systems', *7th IEEE Workshop on Real-Time Operating Systems and Software*, 1990, pp.39-43.

[Son92]: S.H. Son, S. Park, Y.Lin 'An Integrated Real-Time Locking Protocol', *8th International Conference on Data Engineering*, 1992, pp.527-534.

[Sop92]: N. Soparkar, E. Levy, H.F. Korth, A. Silberschatz *Adaptive Commitment for Real-Time Distributed Transactions*, TR-92-15, Department of Computer Science, University of Texas at Austin, 1992.

[Sta88]: J.A. Stankovic, W. Zhao 'On Real-Time Transactions', *ACM SIGMOD Record*, March 1988, pp.4-18.

[Ulu92a]: Ö. Ulusoy, G.G. Belford 'Real-Time Lock Based Concurrency Control in a Distributed Database System', *12th International Conference on Distributed Computing Systems*, 1992, pp.136-143.

[Ulu92b]: Ö. Ulusoy *Concurrency Control in Real-Time Database Systems*, UIUCDCS-R-92-1762, Department of Computer Science, University of Illinois at Urbana-Champaign, 1992.

[Vrb88]: S.V. Vrbsky, K.J. Lin 'Recovering Imprecise Transactions with Real-Time Constraints', *7th Symposium on Reliable Distributed Systems*, 1988, pp.185-193.