

Real-Time Database Systems: Present and Future

Sang H. Son

Department of Computer Science
University of Virginia
Charlottesville, Virginia 22903, USA
son@cs.virginia.edu

Abstract

The design and implementation of real-time database systems presents many new and challenging problems. Compared with traditional databases, real-time database systems have a distinct feature: they must maintain temporally coherent data while satisfy timing constraints associated with transactions. In addition, a real-time database system must adapt to changes in the operating environment and guarantee the completion of critical transactions. In this paper we address the issues associated with temporal nature of transactions and data, and research areas for advanced real-time database systems.

1. Introduction

A real-time system is one whose basic specification and design correctness arguments must include its ability to meet its timing constraints. This implies that its correctness depends not only on the logical correctness, but also on the timeliness of its actions. To function correctly, it must produce a correct result within a specified time, called deadline. In these systems, an action performed too late (or even too early) may be useless or even harmful, even if it is functionally correct. If timing requirements coming from certain essential safety-critical applications would be violated, the results could be catastrophic. They may cause serious damage to the system or to its environment, including injury or even death of people involved. These are called hard real-time systems. By contrast, there are applications that also have deadlines but are noncritical. For example, one can define a transmission system to have failed if voice packets are not delivered within a certain deadline during a teleconference session. However, in those applications, such failures will not be catastrophic. They are called soft real-time systems.

Many real-world applications involve time-constrained access to data as well as access to data that has temporal validity. For example, consider telephone

switching systems, network management, program stock trading, managing automated factories, and command and control systems. All of these involve gathering data from the environment, processing the information gathered in the context of information acquired in the past, and providing a timely and temporally correct response.

Traditionally, real-time systems manage their data (e.g. chamber temperature, aircraft locations) in application dependent structures. As real-time systems evolve, their applications become more complex and require access to more data. It thus becomes necessary to manage the data in a systematic and organized fashion. Database management systems provide tools for such organization. The resulting integrated system, which provides database operations with real-time constraints is generally called a real-time database system. Conventional databases support several features that facilitate (1) the description of data, (2) the maintenance of correctness and integrity of data, (3) efficient access to data, and (4) the correct execution of queries and transactions in the context of concurrency and failures. However, several issues that are crucial in real-time database systems need to be addressed. They include the timeliness and predictability of transactions, and the temporal nature of data and transactions.

Much of the data in a real-time database is temporal in that it models a rapidly changing environment. Furthermore, transactions need to view a temporally consistent state of the database when accessing the database. The temporal nature of data and transactions has several implications:

- Temporal data becomes *invalid* (outdated) with the passage of time. Their validity is independent of the database state and transaction activities.
- There is a time lag between the monitoring and recording of temporal data, and hence a real-time database may not be completely *correct* all the time.
- Temporal data is not necessarily updated simultaneously or at the same rate, resulting in a dispersion of ages of temporal data. The values of a set of temporal data may not co-exist in any snapshot of the real-world.

This work was supported in part by ONR.

- Schedules must be serializable and temporally-correct. The correctness of a real-time transaction may depend on the ages and dispersion of ages of the data read, as well as the completion time.

Furthermore, since timeliness is so important in real-time applications, correctness may be traded for timeliness in certain situations, giving rise to various notions of "approximate" computation.

While many of the techniques used in real-time systems and databases systems may be combined and applied to problems arising in real-time database systems, it appears that new approaches and modifications are necessary to satisfy the requirements imposed by complex real-time applications. Meeting timing constraints demands new approaches to data and transaction management, some of which can be derived by tailoring, adapting, and extending solutions proposed for real-time systems and database systems. Meeting the predictability requirements of real-time databases requires a more rigorous framework that incorporates semantic information of real-time data and transactions. It also demands a deterministic subsystem support from underlying architecture and well-defined interactions between OS and the database system.

2. Research Issues

During the last several years, the area of real-time databases has attracted the attention of researchers in both real-time systems and database systems. New approaches to resolving contention over data and processing resources have been developed and evaluated. For applications that only require real-time transactions with soft deadlines, it appears that conventional database protocols modified to provide preferential treatment to high priority transactions may be sufficient. A primary goal of such database systems is to minimize the number of transactions that miss their deadlines; this can be achieved by adopting best-effort scheduling protocols. However, for applications that require critical transactions with hard deadlines, the system must provide a guarantee that transactions will complete by their deadlines. For such transactions, knowledge concerning the resource and data requirements of transactions is necessary to be utilized by the database protocols. For example, by using priority-based scheduling algorithms with predefined resource access patterns, the database protocols can guarantee that all transactions with shared resources can always meet their deadline as long as some well-defined schedulability conditions are satisfied.

Some initial progress has also been made in defining notions of approximation, and in defining the temporal nature of data and transactions. New notions of "correct" transaction processing in light of transactions' timing requirements are also appearing. However, research in these and many other aspects of real-time

databases is still in its early stages, and many problems relating to data modeling, predictable transaction executions, recovery, managing I/O, handling overloads, operating system support, tool support as well as distributed system support still remain.

Current real-time database research has focused on database systems with flat, soft real-time transactions operating on centralized databases with read/write objects. In the near future, applications that need real-time databases will become large, complex, and distributed. They will need to operate in a highly dynamic environment, and yet will be required to be predictable. They need to access distributed multimedia objects, in addition to traditional alpha-numeric information. The design and implementation of real-time database systems for such applications introduces interesting challenges. Meeting those challenges imposed by the new requirements depends on a focused and coordinated research efforts in several areas listed below:

- Development of modeling techniques for real-time transactions and databases to specify timing properties and temporal consistency in a precise manner. Validity of stored data and relationships between consistency constraints and timing constraints need to be specified clearly.
- Development of methodologies to analyze timing properties of the system with a large number of interacting transactions and constraints. Given the timing properties of an application and a set of timing requirements expressed as assertions, the objective is to relate each assertion to the timing specification. If the assertion is a theorem derivable from the timing specification, then the system is safe with respect to the requirement denoted by the assertion, as long as the implementation is faithful to the timing specification. Otherwise, the system is inherently unsafe because the timing properties will cause the assertion to be violated.
- Development of time-driven priority-based scheduling protocols and concurrency control protocols that can, in an integrated and dynamic fashion, manage complex transactions with resource and precedence constraints, manage resources (e.g., communication resources and I/O devices), and manage timing constraints of varying granularity. In particular, time-driven resource allocation policies and distributed transaction management protocols (e.g., timed atomic commit protocols) need to be developed to meet the real-time scheduling requirements. Since recovery by "undoing" operations may not be applicable in many circumstances, an approach to achieving forward recovery might be necessary.

- Integration of operating system functions with data management in a highly integrated and cooperative, and fast and predictable manner. Since a database system must operate in the context of available operating system services, correct functioning and timing behavior of database management algorithms depends on the services of the underlying operating system. In many areas such as buffer management and consistency control, operating system facilities have to be duplicated by database systems because they are too slow or inappropriate. Research is needed to identify a set of efficient OS primitives required to support the database management protocols, especially addressing real-time constraints.
- Integration of artificial intelligence and active database techniques with real-time data management protocols to provide the capability to reason about time-constrained processes and to control those processes, and to adapt system behavior to dynamically changing situations. A key consideration in active real-time data management is to determine the best available execution sequence when several choices are available with different timing constraints and precedence requirements.
- Architectural support for fault-tolerance, for efficient data management, and for time-constrained communication. Due to advances in VLSI technology, it is possible to develop a new distributed architecture that is suitable for broader class of real-time database applications. Important issues in this new architecture include interconnection topology, interprocess communications, and support of fault-tolerant database operations. It is essential to have hardware support for fast error detection, reconfiguration and recovery. In addition, new architectures need to support real-time scheduling algorithms.

REFERENCES

- [Abb92] Abbott, R. and H. Garcia-Molina, "Scheduling Real-Time Transactions: A Performance Evaluation," *ACM Trans. on Database Systems*, vol. 17, no. 3, pp 513-560, Sept. 1992.
- [Buc89] Buchmann, A. et al., "Time-Critical Database Scheduling: A Framework for Integrating Real-Time Scheduling and Concurrency Control," *5th Data Engineering Conference*, Feb. 1989.
- [Har91] Haritsa, J., M. Livny, and M. Carey, "Earliest Deadline Scheduling for Real-Time Database Systems," *Real-time Systems Symposium*, pp 232-242, Dec. 1991.
- [Hua91] Huang, J., J. Stankovic, K. Ramamritham, and D. Towsley, "On Using Priority Inheritance in Real-Time Databases," *Real-time Systems Symposium*, pp 210-221, Dec. 1991.
- [Kor90] Korth, H., "Triggered Real-Time Databases with Consistency Constraints," *16th VLDB Conference*, Brisbane, Australia, Aug. 1990.
- [Kuo93] T. W. Kuo and A. Mok, "SSP: A Semantic-Based Protocol for Real-Time Data," *Real-Time Systems Symposium*, Raleigh-Durham, North Carolina, December 1993.
- [Lin90] Lin, Y. and S. H. Son, "Concurrency Control in Real-Time Databases by Dynamic Adjustment of Serialization Order," *Real-Time Systems Symposium*, Orlando, Florida, Dec. 1990.
- [Leh95] Lehr, M., Y. Kim, and S. H. Son, "Managing Contention and Timing Constraints in a Real-Time Database System," *IEEE Real-Time Systems Symposium*, Pisa, Italy, Dec. 1995.
- [Lee93] J. Lee and S. H. Son, "Using Dynamic Adjustment of Serialization Order for Real-Time Database Systems," *Real-Time Systems Symposium*, Raleigh-Durham, North Carolina, December 1993.
- [Ram93] Ramamritham, K., "Real-Time Databases," *Journal of Distributed and Parallel Databases*, vol. 1, no. 2, pp 199-226, April 1993.
- [Sha91] Sha, L., R. Rajkumar, S. H. Son, and C. Chang, "A Real-Time Locking Protocol," *IEEE Transactions on Computers*, vol. 40, no. 7, July 1991, pp 793-800.
- [Son88] Son, S. H., guest editor, *ACM SIGMOD Record* 17, 1, Special Issue on Real-Time Database Systems, March 1988.
- [Son92] Son, S. H., J. Lee, and Y. Lin, "Hybrid Protocols using Dynamic Adjustment of Serialization Order for Real-Time Concurrency Control," *Journal of Real-Time Systems*, vol. 4, Sept. 1992, pp 269-276.
- [Son93] Son, S. H. and S. Koloumbis, "A Token-Based Synchronization Scheme for Distributed Real-Time Databases," *Information Systems*, vol. 18, no. 6, December 1993, pp 375-389.
- [Yu94] Yu, P., K. Wu, K. Lin, and S. H. Son, "On Real-Time Databases: Concurrency Control and Scheduling," *Proceedings of IEEE, Special Issue on Real-Time Systems*, January 1994, pp 140-157.