Performance Studies of Locking Protocols for Real-time Databases With Earliest Deadline First

Kam-yiu Lam
Sheung-lun Hung
Ken Chee-keung Law
City University of Hong Kong

Transaction scheduling in real-time database systems (RTDBS) is complicated by the requirements to satisfy the timing constraints of transactions and at the same time to maintain database consistency. These two goals may not be easily achieved at the same time because of the incompatibility between the two schedulings, namely data scheduling and resource scheduling. In this paper, a new locking protocol, called Hybrid 2-Phase Locking (Hb2PL), is proposed for concurrency control in RTDBS with the objectives to maintain database consistency and to minimize its impact on the adopted resource scheduling algorithm. The performance of Hb2PL has been compared with two other well-known real-time locking protocols, High Priority 2-Phase Locking (H2PL) and Conditional High Priority 2-Phase Locking (CH2PL). Amongst the three protocols, the performance of Hb2PL is the best for different degrees of system workloads and number of CPUs. CH2PL is most sensitive towards changes in data contention and CPU preemption. The results also indicate that the effect of the protocols on the system performance is more significant under light workloads than under heavy workloads in which case the effectiveness of the resource scheduling algorithm is the dominant factor on system performance. Relationship between the optimal resource scheduling algorithm, Earliest Deadline First (EDF), and the locking protocols has also been studied. It has been found that their effect on the performance of RTDBS is significantly affected by the distribution of transaction deadlines and the deadline variability.

Real-time systems are defined as those systems in which the correctness of them not only depends on the logical result of computation but also on the time at which the results are generated (Stankovic, 1988). With the advancement in computer technology and more sophisticated requirements on real-time applications, it has become possible and necessary to support real-time applications with the use of database technology (Graham, 1992). Thus, real-time database systems (RTDBS) are becoming more and more common. They are defined as the database systems in which the transactions have their own timing constraints, typically in the form of deadlines. Satisfying these timing constraints are critical to the correctness of the system (Graham, 1992; Ramamritham, 1993). Based on the effect of missing transaction deadlines, RTDBS can be categorized into soft, firm and hard RTDBS. Missing deadline in soft RTDBS significantly affects the usefulness of the transaction. Completing transactions after their deadlines in firm RTDBS may produce harmful effects on the system. For hard RTDBS (Ulusoy, 1992), failing to complete the transactions before their deadlines may even result in disasters. Some of the common applications of RTDBS include computer integrated manufacturing, power plant system control, battle-field management, real-time network management and multi-media conferencing systems.

There are many new and challenging problems associated with the design of RTDBS. One of them is the scheduling of real-time transactions (Graham, 1992; Ramamritham, 1993; Son, 1990; Ulusoy, 1892) the objectives of which are to minimize the number of deadline missing transactions and to maintain the database consistency. In contrast with conventional real-time systems, there are two types of scheduling in
RTDBS: resource scheduling and data scheduling. The primary objective of resource scheduling is to minimize the number of deadline missing transactions. Some of the optimal resource scheduling algorithms are Earliest Deadline First (EDF), Least Slack Time (LST) and Rate Monotonic (RM) (Liu, 1976; Zhao, 1987). Although these algorithms have considered the scheduling of shared data, they cannot handle the issue in a more complex database environment. In conventional real-time systems, most tasks are independent of predictable data and resources requirements. The time for a task to access shared data is usually negligible compared with its total processing time. On the contrary, transactions in typical RTDBS are usually much more complex with many data manipulating operations. Their data access patterns and resource requirements are comparatively less predictable. More importantly, database consistency and atomicity of transactions have to be maintained (Bernstein, 1987).

Data scheduling is an important issue in database management systems and is achieved by the implementation of a well-formed concurrency control protocol (Eswaran, 1976). Although the research on concurrency control protocols for database systems has received considerable attentions in the last two decades (Agrawal, 1987; Barghouti, 1991; Franaszek, 1992; Hung, 1992; Hus, 1992), data scheduling in RTDBS is still a relatively new research area. There are many unresolved problems. The basic problem is that a concurrency control protocol may affect the schedulability of the adopted resource scheduling algorithm owing to their incompatibilities in scheduling methods and differences in objectives. The main goals of conventional concurrency control protocols are to maintain database consistency and to maximize system throughput. Most of them schedule data to transactions on a first-come-first-served basis or their past conflict history (Bernstein, 1987). They completely neglect the deadline constraints of the transactions. One serious consequence of these incompatibilities is priority inversion (Davari, 1992; Sha, 1988). It is a situation in which a higher priority transaction has to wait for a lower priority transaction due to data contention. The blocking time of the higher priority transaction is unbounded. It is because the lower priority transaction may be preempted from using the resource by some other intermediate priority transactions or may be blocked by other transactions. As most concurrency control protocols resolve data contention by blocking and restart of transactions, some other problems such as deadlock, unbounded blocking and restart may occur. All these further aggregate the degree of unpredictability in transaction response time. It becomes more difficult to ensure that all the transactions can be completed before their deadlines even with hardware upgrading.

To resolve the concurrency control problem in RTDBS, different real-time concurrency control protocols have been proposed (Abbott, 1992; Haritsa, 1990; Huang, 1992; Sha, 1991; Son, 1992). The previous work mainly focused on modifying some well-known concurrency control protocols with the incorporation of different real-time features. Most proposed protocols are based on Two Phase Locking (2PL). Two popular protocols of this type are High Priority 2-Phase Locking (H2PL) and Conditional High Priority 2-Phase Locking (CH2PL) (Abbott, 1992). These protocols have been shown to be suitable for both soft and firm RTDBS.

Although a number of real-time concurrency control protocols have been proposed, up to now, there is still a lack of detailed studies on many fundamental problems such as:

1. Past studies seldom separate the effect of concurrency control on system performance from resource scheduling algorithms. It is important and useful to identify under which conditions the roles of concurrency control protocols are most significant.

2. Although Earliest Deadline First (EDF) is the resource scheduling algorithm mostly adopted in the studies of transaction scheduling in RTDBS, the relationship between EDF and different concurrency control protocols is still unclear. Is it possible to minimize the impact of concurrency control protocols on system performance by adjusting the transaction deadlines?

3. The problem of high resource contention in real-time systems can be resolved by upgrading or increasing the number of the hardware resources. Are these methods still useful for RTDBS?

In this paper, a new real-time locking protocol, called Hybrid 2-Phase Locking (Hb2PL), is proposed for soft and firm RTDBS. Its performance is compared with H2PL and CH2PL. Special attention is paid to the above three problems with the three locking protocols in single processor and multiprocessors RTDBS.

**Real-Time Two Phase Locking Protocols**

Most concurrency control protocols proposed for RTDBS to date are based on Two Phase Locking (2PL). The blocking time due to priority inversion can be reduced by adopting the priority inheritance method (Rajkumar, 1991) whereby the lower priority lock holding transaction will inherit the priority of the highest priority transaction that is waiting for the lock seized by the lower priority transaction. Although priority inheritance has been shown to be effective for solving priority inversion in conventional real-time systems (Rajkumar, 1991), it is not suitable for RTDBS as the blocking time of the transaction can be very long. Lock holding transactions only release their locks upon their commitments. Abbott and Molina (Abbott, 1992) have proposed a restarted based method called High Priority 2-Phase Locking (H2PL) protocol. In H2PL, whenever a lock conflict is encountered, the priorities of the lock requesting and lock holding transactions will be checked as follows:
9 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product's webpage:
www.igi-global.com/article/performance-studies-locking-protocols-real/51148?camid=4v1

This title is available in InfoSci-Journals. Recommend this product to your librarian:
www.igi-global.com/e-resources/library-recommendation/?id=2

Related Content

Storing XML Documents in Databases
www.igi-global.com/chapter/storing-xml-documents-databases/11220?camid=4v1a

Categorizing Post-Deployment IT Changes: An Empirical Investigation
www.igi-global.com/article/categorizing-post-deployment-changes/3368?camid=4v1a

OOXKSearch: A Search Engine for Answering XML Keyword and Loosely Structured Queries Using OO Techniques
Kamal Taha and Ramez Elmasri (2009). Journal of Database Management (pp. 18-50).
www.igi-global.com/article/ooxksearch-search-engine-answering-xml/4123?camid=4v1a

Spatio-Temporal Prediction Using Data Mining Tools
www.igi-global.com/chapter/spatio-temporal-prediction-using-data/29667?camid=4v1a