Real-Time Databases and Multimedia Systems

Multi-Versions Data for Improvement of Quality of Service in RTDB.

Claude Duvallet

University of Havre
Faculty of Sciences and Technology
25 rue Philippe Lebon - BP 540
76058 Le Havre Cedex, FRANCE
Claude.Duvallet@gmail.com
http://litis.univ-lehavre.fr/∼duvallet/index-en.php

Outline

Introduction and context

Real-Time Database Model

Feedback Control Scheduling Architecture

Multi-Versions Data - Feedback Control Scheduling Architecture

Conclusion and future work

Introduction and context

Due to:

In many applications, the demand for Real-Time Databases services has increased.

The workload of Real-Time Databases Systems is unpredictible.

Stringent timing/data freshness constraints.

Consequently:

Real-Time Databases Systems may become overloaded.

Real-Time Transactions may miss their deadlines.

Solutions:

Some techniques based on QoS.

Feedback Control Real-Time Scheduling (FCS).

Data model

Data objects are classified:

- Non Real-Time Data: classical data
- Real-Time Data:
  - change continuously to reflect the real world state
  - have an absolute validity interval

Quality of Data (QoD):

- DE (Data Error): deviation between the current value and the updated value.
- MDE (Maximum Data Error).

Transactions model

Real-Time Transactions:

Update Transactions

arrive periodically and are executed

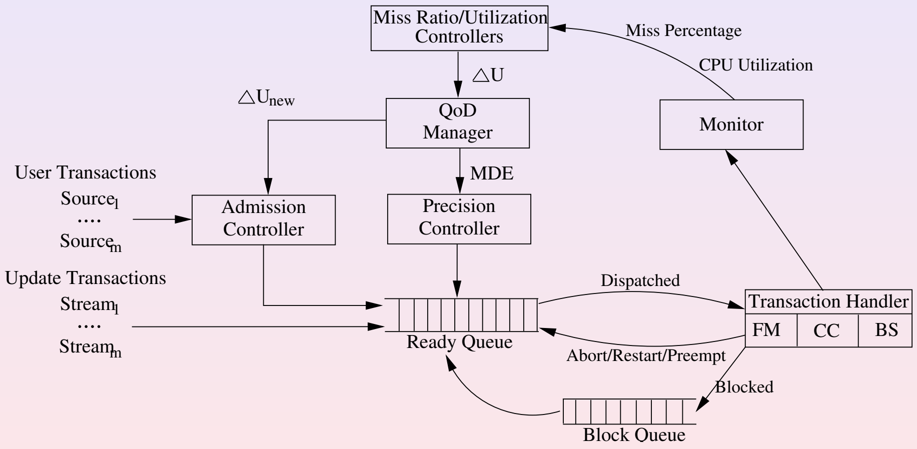
have only to write real-time data

User Transactions

arrive aperiodically and are executed

read real-time data, and read or write non real-time data

Feedback Control Scheduling Architecture



The transaction manager (1/4)

- The transaction handler is composed of:
  - a freshness manager (FM) which check the freshness of the real-time that will be acceded using the timestamp of the data and absolute validy interval: it blocks the transactions which want to access to non fresh data,
  - a concurrency control (CC) protocol which is most of the time 2PL-HP,
  - a basic scheduler (BS) which is most of the time EDF.
- Two queues for the transactions:
  - update transactions and mandatory (users) sub-transactions are placed in the highest queue priority,
  - optional (users) sub-transactions (users) are placed in the lower queue priority,
  - taking into account the transactions of these two queue is decided at the transactions handler level.

The transaction manager (2/4)

- The admission controller (AC):
  - it controls the flow of input transactions,
  - it decides whether a transaction can be accepted or not in the system,
  - it uses parameters such as the importance of transactions (priority), the load of the system (resource use).
- The precision manager:
  - it eliminates update transactions which try to write data (d_i) with an error $DE_i \leq MDE$,
  - otherwise the new value of d_i is updated,
  - in all cases the timestamp of d_i is updated,
  - its goal is to reduce the load of the system in terms of execution of update transactions,
  - it increases or decreases the value of MDE depending on the ΔU returned by the controller use.

The transaction manager (3/4)

- The monitor:
  - it measures the number of transactions that ended before theirs deadline, ending prior to maturity or that fail to meet their deadline,
  - it take its measure from the transaction handler, Item it sends the measures it has done to utilization controller.
- The utilization controller:
  - available information provided by the instructor,
  - it makes computations on the use of the system that allows it to detect transients overload (too many transactions that fail to meet their deadline, for example)
  - it looks at CPU load of the system,
  - it makes a final computation to determine ΔU (the difference between the current utilization and the reference value) that will affect the quality of data manager.

The transaction manager (4/4)

- The quality of data manager:
  - it will increase or decrease the quality of the data based on the use of the system (in overload periods, it will decrease the quality of data)
  - it affects user transactions admitted in the system by the admission controller but also on the execution or not of the update transactions,
  - it recomputes MDE in order to decrease or increase the number of update transactions that will be executed,
  - it calculates a new ΔU from that's one provided by the utilization controller and its own internal changes,
  - the new value of ΔU is transmitted to the admission controller.

Resume about the Feedback Control Scheduling Architecture

- In input, we have parameters of quality of service specified by the DBA.
- Recomputing the parameters of the quality of service according to the runtime and the references parameters of the systems.

⇒ Creating a feedback loop to control the behavior of the RTDB during overload period of the systems.

⇒ It is not necessary to have a specific model of the load of the system over time.

⇒ It leads to a dynamic stabilization system according to the load and the available resources.

Advantage and Inconvenients of the global model

Advantage

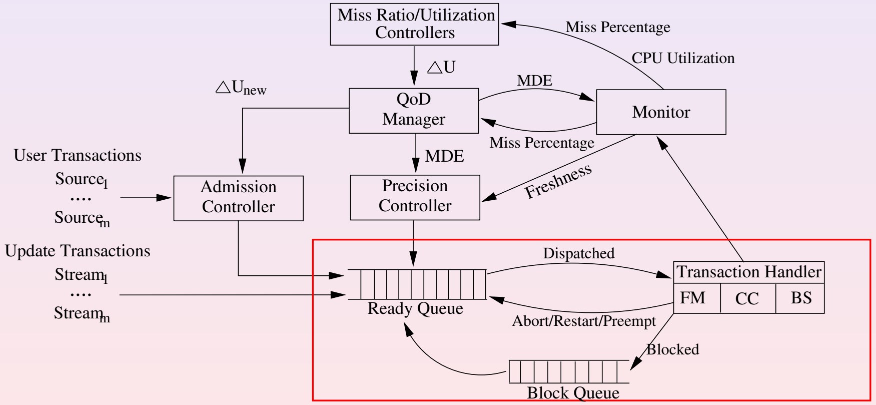
- guarantees a set of requirements on the RTDB behavior
- provides a QoS guarantee

Inconvenients

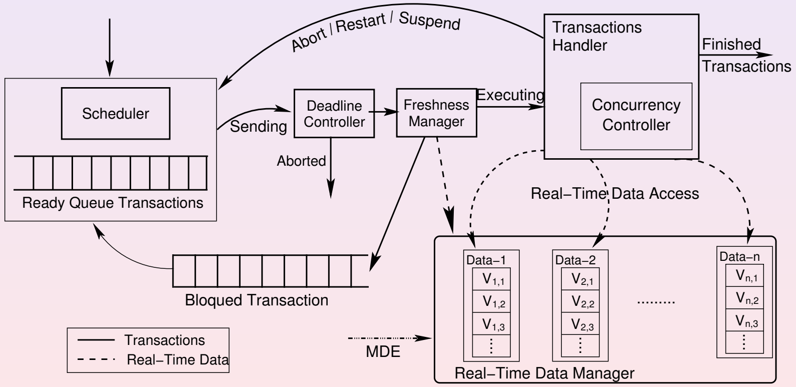
- when a higher priority transaction uses the data item, transactions with lower priority will be blocked
- FM blocks user transactions if the accessed data is stale
- this may lead transactions to miss their deadline

To address this problem and to aleviate this risk, we propose Multi-Versions Data Feedback Control Scheduling Architecture .

Feedback Control Scheduling Architecture



Multi-Versions Data - Feedback Control Scheduling Architecture



Multi-Versions Data - Feedback Control Scheduling Architecture: 3 approaches (1)

Approach n^{o}1: MVD with fixed number of data versions

- We keep all data values that correspond to different versions of the same data item.
- The maximum number of versions is limited and is fixed in advance by the DBA according to QoS requirement level.

Multi-Versions Data - Feedback Control Scheduling Architecture: 3 approaches (2)

Approach n^{o}2: MVD with dynamically adjusted number of data versions

- We adjust dynamically the number of data versions.
- For each data, a queue of versions is maintained.
- The queue is continually updated in order to limit the number of data versions by removing/adding versions, based on both data freshness and MDE criterion

Multi-Versions Data - Feedback Control Scheduling Architecture: 3 approaches (3)

Approach n^{o}3: Mixed approach

- We merge the two last approaches: the number of data versions is dynamically adjusted and does not have to exceed the fixed threshold representing the number of data versions.
- We have considered in the same time a threshold representing the database size.
- A data item will be added only if its version number is lower than the maximum database size.

Parameters of Simulation

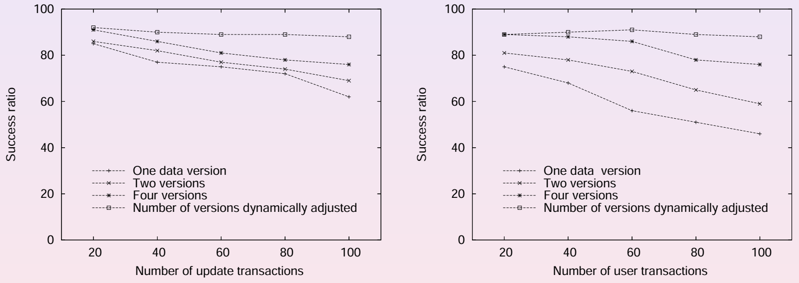
Parameter	Meaning	Value
NbOfOperations	Number of operations in a user transaction	[1, 5]
OpExecTime	Execution time of an operation	1s
$Period_i$	Periodicity of update transaction	[1000ms, 5000ms]
DBsize	Database size	300

Simulation Parameters

We have evaluated the behavior of the system by varying a set of parameters:

1. The threshold of data versions number
2. The threshold of database size
3. The number of transactions

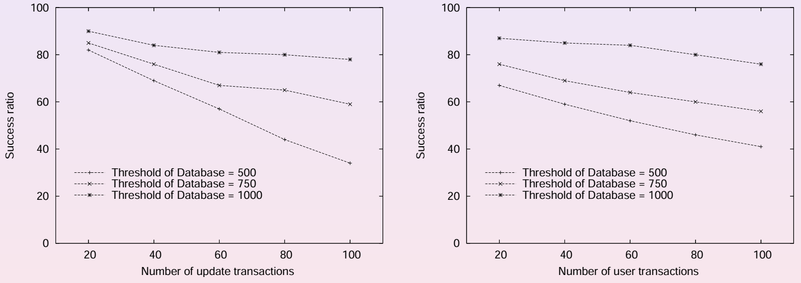
Experiment 1: Results of Multi-Versions Data - Feedback Control Scheduling Architecture



(a) For update transactions (b) For user transactions

Simulation results for the Multi-Versions Data - Feedback Control Scheduling Architecture.

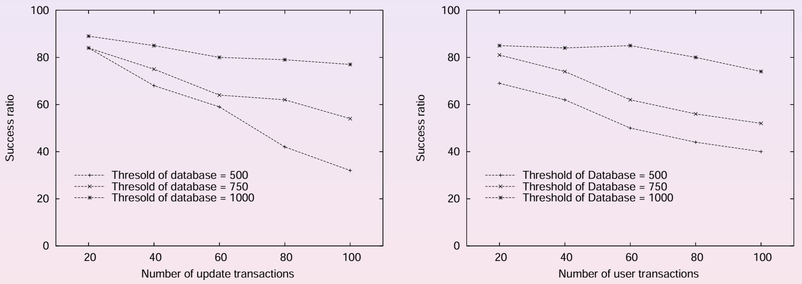
Experiment 2: Varying the threshold of database size using the mixed approach of MVD-FCSA (1)



(c) Update transactions (d) For user transactions

Simulation results when using the mixed approach of MVD-FCSA (maximum number of versions = 4) and varying the threshold of database size.

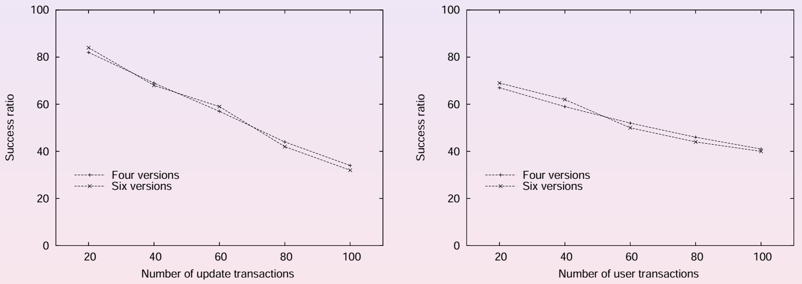
Experiment 2: Varying the threshold of database size using the mixed approach of MVD-FCSA (2)



(e) For update transactions (f) For user transactions

Simulation results when using the mixed approach of MVD-FCSA (maximum number of versions = 6) and varying the threshold of database size.

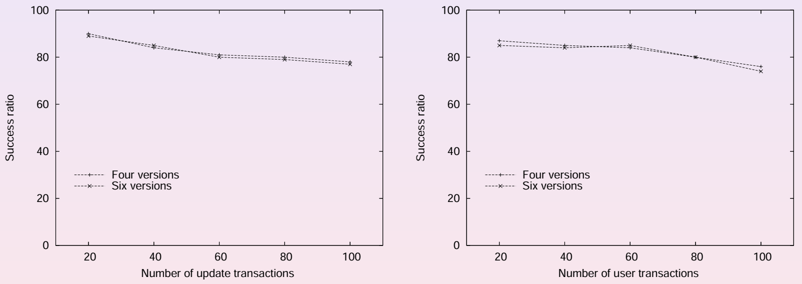
Experiment 3: Varying the threshold of data versions number (1)



(g) For update transactions (h) For user transactions

Simulation results of using the mixed approach of MVD-FCSA: varying the number of versions and the threshold of database size 500.

Experiment 3: Varying the threshold of data versions number (2)



(i) For update transactions (j) For user transactions

Simulation results of using the mixed approach of MVD-FCSA: varying the number of versions and the threshold of database size 1000.

Benefits of Multi-Versions Data - Feedback Control Scheduling Architecture

MVD-FCSA allows:

1. to decrease the deadline miss ratio,
2. to guarantee the accessed data freshness by timely transactions even in the presence of unpredictable workloads,
3. the data used by committed transactions to always be 100% fresh (at commit time),
4. to guarantee the QoD and the QoT: quality of data (precision and freshness) and quality of transaction are enhanced by alleviating the risk of transaction miss deadline, and therefore to enhance the QoS.

Conclusion

- We have presented 3 approaches of MVD-FCSA:
  - MVD with fixed number of data versions,
  - MVD with dynamically adjusted number of data versions,
  - MVD with a mixed approach and a threshold on database size.
- MVD-FCSA is used to minimize the number of conflicts by decreasing the number of aborted transactions.
- Simulations show that MVD-FCSA is more successful than FCSA.

Future work

We plan to:

- Consider other aspects to study different components of FCSA.
- Manage derived data in MVD-FCSA.
- Apply QoS approach and FCSA to distributed multimedia system.

References

- Krithi Ramamritham, Sang Hyuk Son, Lisa Cingiser DiPippo: Real-Time Databases and Data Services. Real-Time Systems 28(2-3): 179-215 (2004)
- Mehdi Amirijoo, Jörgen Hansson, Sang Hyuk Son: Specification and Management of QoS in Real-Time Databases Supporting Imprecise Computations. IEEE Trans. Computers 55(3): 304-319 (2006).
- Emna Bouazizi, Claude Duvallet and Bruno Sadeg. Improvement of QoD and QoS in RTDBS. Proceedings of 14th International Conference on Real-Time and Network System (RTNS'2006), Poitiers, France, May 30-31, pages 87-95, 2006.
- Emna Bouazizi, Claude Duvallet and Bruno Sadeg. Multi-Versions Data for improvement of QoS in RTDBS. Proceedings of 11th IEEE International Conference on Real-Time and Embedded Computing Systems and Applications (IEEE RTCSA'2005), Hong Kong, China, pages 293-296, August 17-19, 2005.

My current topics of research

- Quality of service in Real-Time Database.
  - Use of Multi-Version Data to improve Quality of Service in Real-Time Databases (with a PhD student: Emna Bouazizi).
  - Management of Real-Time Derived Data in Feedback Control Scheduling (currently a graduate student is developing a simulator).
- Quality of service in Multimedia Systems (with a PhD student: Bechir Alaya and a graduate student is developing a simulator).
- Structural Model for Real-Time Databases (with a PhD student: Nizar Idoudi).