A High Performance Multiversion Concurrency Control Protocol for Object Databases

Craig Harris, ONTOS Inc.
Madhu Reddy, Ms., ONTOS Inc.
Carl Woolf, Ph.D., ONTOS Inc.

Abstract

Commercial object databases (ODBs) generally claim to have very fast data access. Some benchmarks have shown that ODBs perform very well in certain single-user application areas. But many commercial database applications have challenging concurrent, multi-user performance requirements. Traditional concurrency control protocols (CCPs) can slow down the overall system performance of an ODB to the point where it has no real performance advantages over non-ODB products. In the development of ONTOS, a commercial ODB, we felt it was important to develop new CCPs which were appropriate for our object orientation and performance goals.

The ONTOS approach to concurrency control is based on the notion of an explicit serialization graph. Traditional CCPs serialize transactions implicitly. Pessimistic lock-based schemes implicitly serialize in commitment order. Time-based schemes implicitly serialize in the order in which transactions start. In both cases, transactions are serialized on the basis of real-time ordering of special events: starting and ending transactions. In ONTOS, it is possible for a transaction which both starts and commits after another transaction to precede the other transaction in serialization order - if the operations it performs are compatible with such a serialization order. In the absence of semantic information, ONTOS assumes that a transaction which reads an object will serialize before those that update the object; and will make this determination at the time the object is manipulated. This allows readers and writers of the same object to proceed concurrently. In the presence of semantic information, ONTOS can sometimes allow multiple writers of the same object to proceed concurrently, performing semantic concurrency conflict detection.

This talk describes the serialization-based concurrency control algorithms in use in ONTOS. We argue that the technique permits greater concurrency over a well defined class of transaction histories, and is particularly well suited for use in distributed environments. We support this position with benchmarks which compare the throughput of certain transaction streams under various CCPs.