Where To Download Concurrency Control And Recovery In Database Systems

**Concurrency Control And Recovery In Database Systems**

We report on initial research on the concurrency control issue of compiled database applications. Such applications have a repository style of architecture in which a collection of software modules operate on a common database in terms of a set of predefined transaction types, an architectural view that is useful for the deployment of database technology to embedded control programs. We focus on decoupling concurrency control from any functionality relating to recovery. Such decoupling facilitates the compile-time query optimization. Because it is the possibility of transaction aborts for deadlock resolution that makes the recovery subsystem necessary, we choose the deadlock-free tree locking (TL) scheme for our purpose. With the knowledge of transaction workload, efficacious lock trees for runtime control can be determined at compile-time. We have designed compile-time algorithms to generate the lock tree and other relevant data structures, and runtime locking/unlocking algorithms based on such structures. We have further explored how to insert the lock steps into the transaction types at compile time. To conduct our simulation experiments to evaluate the performance of TL, we have designed two workloads. The first one is from the OLTP benchmark TPC-C. The second is from the open-source operating system MINIX. Our experimental results show TL produces better throughput than the traditional two-phase locking (2PL) when the transactions are write-only; and for main-memory data, TL performs comparably to 2PL even in workloads with many reads.

Transaction processing is fundamental for many modern applications. These applications require the backend transaction processing engines to be available at all times as well as provide a massive horizontal scale for intensive transaction requests. Concurrency Control and Recovery features recent progress in research in online transaction processing. The book also showcases the authors' research on a highly scalable OLTP system. Its contents include the designs of an efficient multiple version storage engine, a scalable range optimistic concurrency control, high-performance Paxos-based log replication, global snapshot isolation, and fast follower recovery. This book is written for professionals, researchers, and graduate students specialising in database systems and its related fields.

**Abstract:** "Shared storage arrays enable thousands of storage devices to be shared and directly accessed by end hosts over switched system-area networks, promising databases and filesystems highly scalable, reliable storage. In such systems, however, concurrent host I/Os can span multiple shared devices and access overlapping ranges potentially leading to inconsistencies for redundancy codes and for data read by end hosts. In order to enable existing applications to run unmodified and simplify the development of future ones, we desire a shared storage array to provide the illusion of a single controller without the scalability bottleneck and single point of failure of an actual single controller. In this paper, we show how rapidly increasing storage device intelligence coupled with storage's special characteristics can be successfully exploited to arrive at a high performance solution to this storage management problem. In particular, we examine four concurrency control schemes and specialize them to shared storage arrays; two centralized ones: simple server locking, and server locking with leased callbacks; and two distributed ones..."
based on device participation: distributed locking using storage-device-embedded lock servers and timestamp ordering using loosely synchronized clocks. Simulations results show that both centralized locking schemes suffer from scalability limitations. Moreover, callback locking is particularly suspect if applications do not have much inherent locality and if the storage system introduces false sharing. Distributed concurrency control with device support is attractive as it scales control capacity with storage and performance capacity and offers the opportunity to piggyback lock/ordering messages on operation requests, eliminating message latency costs. Simulations show that both storage-optimized device-based protocols exhibit close to ideal scaling achieving 90-95% of the throughput possible under totally unprotected operation. Furthermore, timestamp ordering uses less network resources, is free from deadlocks and has performance advantages under high load. We show how timestamp ordering can be extended with careful operation history recording to ensure efficient failure recovery without inducing I/Os under normal operation. This brings the overhead of concurrency control and recovery to a negligible few percent thereby realizing the scalability potential of the shared array I/O architecture."

Abstract: "This paper addresses the problem of a transaction reading and writing data at multiple classification levels in a Multilevel Secure (MLS) database system. We refer to such transactions as multilevel update transactions and show that no secure scheduler can ensure atomicity of multilevel update transactions in the presence of transaction aborts. We then determine the covert channel capacity of various scheduling schemes. There are essentially two ways of scheduling multilevel update transactions. The first, which ensures strong atomicity, involves delaying the commit step of a low-level subtransaction until the fates of all siblings are known. The second scheme, which ensures only semantic atomicity, allows each subtransaction to commit or abort independently and compensates for committed transactions when necessary. Analysis of these schemes indicate that the compensation approach leads to lower covert channel bandwidths. A concurrently control and recovery protocol based on compensation is proposed for scheduling multilevel update transactions. The correctness of the protocol is demonstrated and security issues are discussed."
based concurrency control protocol that produces strict schedules. We illustrate our ideas with several examples of abstract data types."

Abstract: "We focus on the update-in-place recovery mechanism for concurrency control of arbitrary operations on abstract data types. In Part I of this technical report, we consider three notions of correctness - serial correctness, view serializability and conflict serializability. We give the definitions for recoverable, cascadefree and strict histories for arbitrary operations on objects. In Part II of this report, we study the relationship among conflict serializability, view serializability and serial correctness. For arbitrary operations on objects, we show that a sufficient condition for a history to be serially correct is that it is conflict serializable and recoverable. In Part III, we present an ordered- shared relation called independence which goes beyond commutativity. Based on independence, we propose a concurrency control and recovery protocol that produces conflict serializable and strict histories. We have implemented our protocol and compared its performance with that of 2PL for different data contentions and resources. Our protocol performed better than 2PL for multiple disks and for medium to high data contentions."

This is the first of the three volumes of the final technical report for the project Distributed Database Control and Allocation. This volume describes frameworks for understanding concurrency control and recovery algorithms for centralized and distributed database systems.

It is widely recognized by practitioners that concurrency control and recovery for transaction systems interact in subtle ways. In most theoretical work, however, concurrency control and recovery are treated as separate, largely independent problems. In this paper we investigate the interactions between concurrency control and recovery. We consider two general recovery methods for abstract data types, update-in-place and deferred-update. While each requires operations to conflict if they do not "commute", the two recovery methods require subtly different notions of commutativity. We give a precise characterization of the conflict relations that work with each recovery method, and show that each permits conflict relations that the other does not. Thus, the two recovery methods place incomparable constraints on concurrency control. Our analysis applies to arbitrary abstract data types, including those with operations that may be partial or non-deterministic.

Advances in Concurrency Control and Transaction Processing addresses developments in transaction processing, and the motivation for transactions and basic transaction concepts. The briefing begins with a thorough discussion of traditional transaction processing and leads into more advanced techniques that depend on the programmer providing additional input parameters enabling the system to process transactions in a more sophisticated way. The briefing examines current transaction advances in terms of their relative strengths, weaknesses, and appropriateness. It also concentrates on techniques based on sophisticated extensions to traditional concurrency control, recovery, and processing protocols that do not affect the way transactions are programmed. It reviews techniques based on the semantics of the data and operations, and improvements in the performance of traditional transactions. New correctness criteria are also discussed and the briefing concludes with a discussion of transaction processing trends.
The major objective of a distributed system is to provide low cost availability of the resources of the system by localizing access and providing insulation against failures of individual components. Since many users can be concurrently accessing the system, it is essential that a distributed system also provide a high degree of concurrency. Research into algorithms has been focused on concurrency, consistency, failure detection, management of replicated copy, and commitment and termination of transactions. This book is a compilation of a subset of research contributions in the area of concurrency control and reliability in distributed systems, with brief explorations of interesting areas, including theoretical and experimental efforts.

Copyright: 7d1f3a041df3d3de29b1d1a1cca581fe