Adaptive Data Management for In-memory Database Clusters

L. Camargos◊, F. Pedone◊, R. Schmidt◊, and M. Wieloch◊
◊University of Lugano (USI)
◊Ecole Polytechnique Federale de Lausanne (EPFL)
◊State University of Campinas (Unicamp)

Motivation

Sprint is an adaptive data management system that delivers high performance and availability by efficiently orchestrating in-memory database engines running in the individual servers, and by removing disk I/O from the critical path of transaction processing. Yet, Sprint does not give up strong data consistency.

Anatomy

Sprint runs on clusters of commodity computers with large main memories, powerful processors, and low latency network. Three kinds of logical servers may run on each node of the cluster:

- **Edge Servers**: receive transactional requests from external clients and execute them against the data servers.
- **Data Servers**: keep portions of the database into local in-memory database engines.
- **Durability Servers**: efficiently store durable information about completed transactions.

![Diagram of Sprint Clusters](image)

Physiology

Local transactions execute in a single data server only; global transactions execute in multiple data servers. Distributed deadlocks are solved with an optimistic resolution mechanism.

Sprint uses Paxos Commit to terminate transactions. Paxos Commit is a generalization of two-phase commit (2PC). It is **non-blocking, fault-tolerant**, and **always safe**.

Features

- **Simple execution model** - Local transactions have minimal overhead. Resolving distributed deadlocks requires neither timeouts nor complex data structures (e.g., wait-for graphs).
- **In-memory transaction execution** - Transactions are not limited by the memory capacity of a single data server. Transactions that would read from disk in on-disk databases access multiple data servers in Sprint.
- **Low-latency termination** - Update transactions rely on Paxos Commit to manage atomic commitment. Paxos Commit generalizes two-phase commit (2PC) and has comparable latency without its blocking shortcoming.
- **Decoupling of replication for performance from replication for fault tolerance** - Data servers use replication to allow parallel reads. Durability servers use replication to improve availability.
- **Efficient recovery** - Recovering a data server requires installing an image in some operational server, and applying new updates; it does not block the system.

Do you want to know more?
Visit the project’s web site at:
www.inf.unisi.ch/projects/sprint/