No compromises: distributed transactions with consistency, availability, and performance

Aleksandar Dragojević, Dushyanth Narayanan, Edmund B. Nightingale, Matthew Renzelmann, Alex Shamis, Anirudh Badam, Miguel Castro
FaRM

• A main memory distributed computing platform that provides distributed ACID
  – Serializability
  – High availability
  – High performance
• Two hardware trends to eliminate storage and network bottlenecks
  – Fast commodity networks with RDMA
  – Inexpensive approach to provide non-volatile DRAM
• Primary-backup replication and unreplicated coordinators, reducing message counts compared with Paxos
• One-side RDMA, parallel recovery...
Non-volatile DRAM

• Distributed UPS makes DRAM durable
  – Lithium-ion batteries
  – Saves contents of memory to SSD using energy from batteries

• Cost
  – Energy cost $0.55/GB
  – Storage cost (reserving SSD) $0.9/GB
  – ~15% of DRAM cost (NVDIMM costs 3-5x more)
Programming Model and Architecture

• Abstraction of a global address space that spans machines in a cluster
• FaRM API provides transparent access to local and remote objects within transactions
Figure 3. FaRM architecture
Architecture

• Configuration <i, S, F, CM>
  – i: 64-bit unique configuration identifier
  – S: set of machines
  – F: mapping to failure domains
  – CM: configuration manager
• Zookeeper ensures machines agree on the current configuration and stores it (not for managing leases, detecting failures, etc.)
• Fault tolerance
  – One primary and f replicas
• CM allocates new region (GB) in primary and replicas
  – Commit allocation only all replicas succeed
• Ring-buffer based send receive pairs
  – The sender appends records to the log using one-sided RDMA writes
  – The receiver periodically polls the head of the log
Distributed Transactions and Replication

- Lock
- Validate
- Commit backups
- Commit Primaries
- Truncate

**Figure 4.** FaRM commit protocol with a coordinator C, primaries on $P_1, P_2, P_3$, and backups on $B_1, B_2, B_3$. $P_1$ and $P_2$ are read and written. $P_3$ is only read. We use dashed lines for RDMA reads, solid ones for RDMA writes, dotted ones for hardware acks, and rectangles for object data.
Correctness and Performance

• Correctness
  – Locking ensures serialization of write and validation ensures serialization of read
  – Serializablity across failures: wait for hardware acks from all backups before writing COMMIT-PRIMARY
  – The coordinator reserves log space at all participants to avoid involving he backups’ CPUs
Correctness and Performance

• Performance
  – Two-phase commit (Spanner)
    • requires 2f+1 replicas to tolerate f failures
    • Each state machine operation requires 2f+1 round trip messages (4P(2f+1) messages)
  – FaRM
    • Use primary –backup replication instead of Paxos state machine replication
    • f+1 copies
    • Coordinator state is not replicated
    • Commit phase uses Pw(f+3) one-side RDMA writes
Failure Recovery

• Durability and high availability by replication
• Machines can fail by crashing but can recover the data by using non-volatile memory
• Durability for all committed transactions even the entire cluster fails or loss power as data are persisted in non-volatile DRAM
• Tolerant f non-volatile DRAM failures
Failure Detection

• Each machine holds a lease at the CM and the CM holds a lease at every other machine
• Expiration of any lease triggers failure recovery
• 5ms short lease to guarantee high availability
  – Dedicated queue pairs for leases
  – Lease manager uses Infiniband with connectionless unreliable datagram transport
  – Dedicated lease manager thread that runs at the highest user-space priority
  – Preallocate memory for the lease manager
• Suspect
• Probe
• Update configuration
• Remap regions
• Send new configuration
• Apply new configuration
• Commit new configuration

Figure 5. Reconfiguration
Transaction State Recovery

• Block access to recovering regions
• Drain logs
• Find recovering reansactions
• Lock recovery
• Replicate log records
• Vote
• Decide
Evaluation

• Setup
  – 90 machines for FaRM cluster and 5 machines for replicated Zookeepers
  – 256GB DRAM and two 8-core Intel E5 CPUs
  – 56Gbps Infiniband NICs

• Benchmarks
  – Telecommunication Application Transaction Processing (TATP)
  – TCP-C a well-known database benchmark with complex transactions
Performance

Figure 7. TATP performance

Figure 8. TPC-C performance
Figure 9. TATP performance timeline with failure
CM Failure

Figure 11. TATP performance timeline with CM failure
Conclusion

• FaRM, a memory distributed computing platform
  – Distributed transactions and replication
  – Strict serializability and high performance
• Primary-backup replication, not coordinator replication
• High throughput and low latency, fast recovery