SI-TM: Reducing Aborts in Transactional Memory

Heiner Litz and David Cheriton

heiner.litz@stanford.edu, david.cheriton@cs.stanford.edu

Transactional Memory

- Alternative synchronization mechanism to locks
- Optimistic, multiple threads can enter critical region
- Improves programmability
- Tracks memory accesses, aborts on conflict

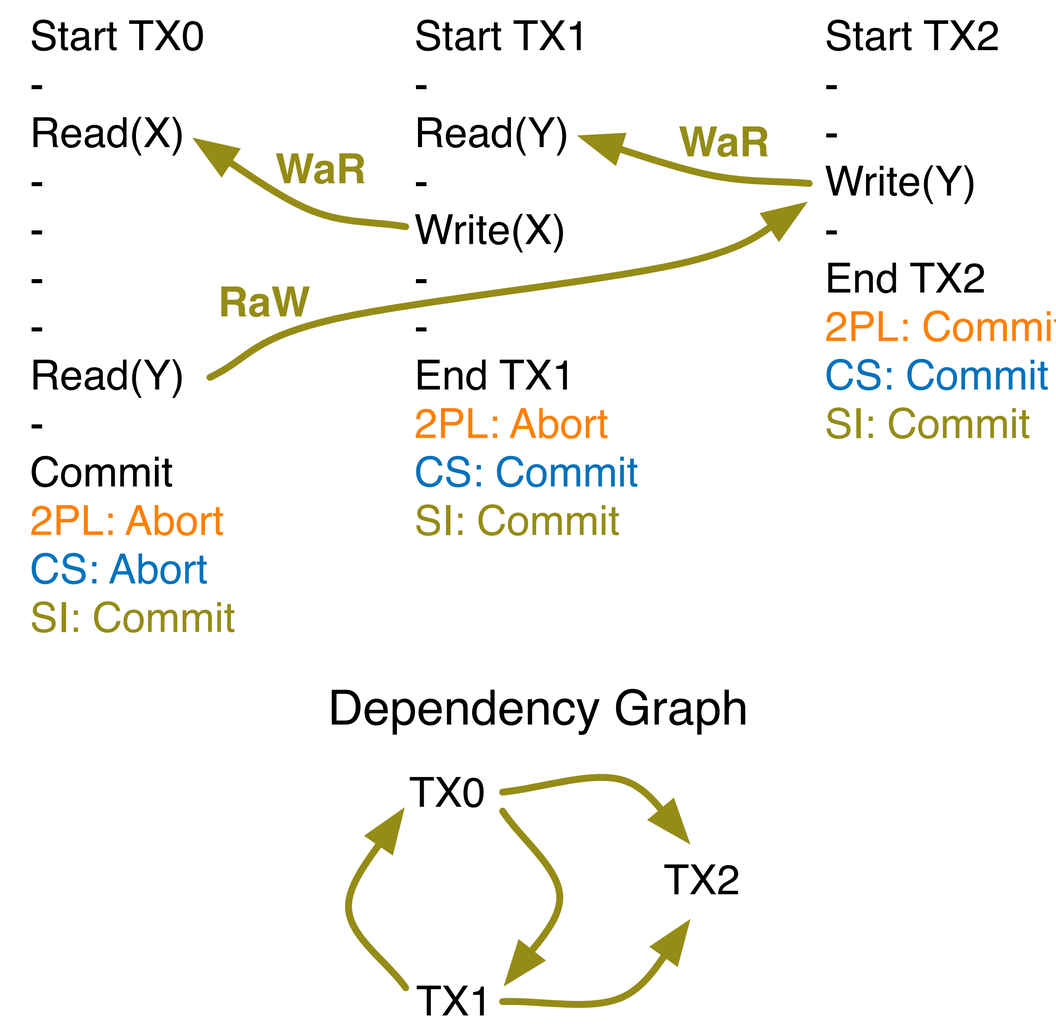
2-Phase Locking (2PL)

- Simple to implement
- Tracks every transactional read and write access
- Aborts on every RW, WR, WW conflict

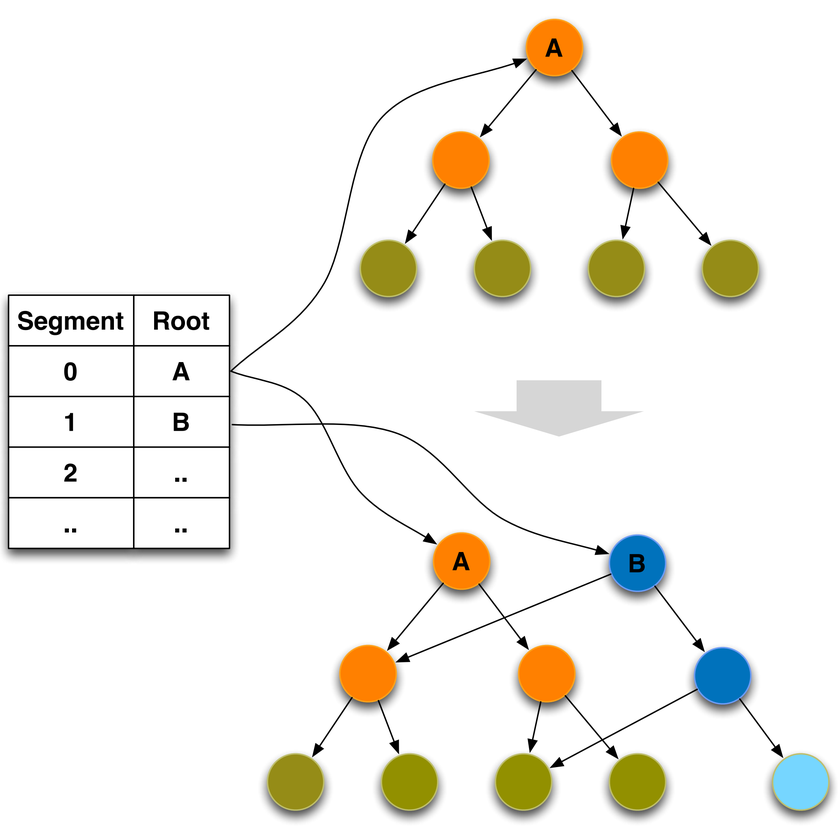
Conflict Serializability (CS)

- Permits certain conflicts
- Tracks dependencies between transactions
- Reorders transactions if necessary
- Aborts on cyclic dependencies

Transaction Example



HICAMP DAGs



SI-TM

- Based on Snapshot Isolation, MVCC
- Reads always return consistent data
- Aborts only on write-write conflicts
- Ignores all read-write, write-read conflicts
- Read only transactions always commit
- Requires efficient snapshotting capability

HICAMP Memory System

- Stores memory objects as segments
- Segment = directed acyclic graph (DAG)
- Segment uniquely identified by root
- Efficient copy on write

Implementation

- Implemented SI-TM in Zsim hardware simulator
- Embedded into RSTM framework
- Patched malloc() to allocate in HICAMP memory
- Ran STAMP benchmark suite
- High contention, 1-64 threads
- Outperforms both 2PL and CS

Results