Lecture Notes: Wednesday, May 21, 2008

Hybrid Transactional Memory

An alternate approach to Hardware Transactional Memory is to use Software Transactional Memory as the base case, and imrpove its performance with hardware.

Definitions

Hybrid Transactional Memory: The transaction first runs completely in hardware (best-effort), and if unsuccessful it restarts and executes as a software transaction.

Hardware-Assisted Software Transactional Memory: The transaction always runs as a software transaction, but it uses hardware support to accellerate some, if not all of the overheads.

The definitions above aren't universal, and the two methods could be combined.

Why Hybrid/Hardware Assisted?

- Full hardware support is not needed to get some benefits, best-effort will start paying off.
- Unbounded.
- Can run on systems without hardware support, and runs fasters on systems with support.
- Implementing contention management and other policy issues is easier to do in software than in hardware.
- Many contentious issues that are being debated now (i.e. nesting) don't have to be solved in hardware, and software could evolve as the needs change.
- Solves the chicken and the egg problem.

Hybrid Transactional Memory Design Drivers

- Assume (hope) that the hardware case is the common case
- Minimize the overhead for the common (hardware) case
- Minimize the effect the software case will have on the hardware case
- What kind of STM to choose
- What to do when a hardware transaction fails? Try again? Backoff? Immediately fallback onto software.
- Conflict detection and resolution between hardware and software transactions. Keep in mind that hardware transactions are normally invisible.

Hardware Accelerated Transactional Memory Design Drivers

- Simplify the algorithm by transactionalizing parts of it
- Reduce the overheads:
  - Logging/taking a backup copy [memory management]
  - Detecting conflicts between readers and writers (especially for invisible reads) [validation]
  - Simpler algorithms by having some nonblocking guarantees [metadata management]

Question

Is is possible to design an HyTM with no (or negligible) overhead compared to HTM?

References

Larus and Rajwar: Transactional Memory (Chapter 4.6)
Tabba, Wang, Goodman and Moir: NZTM: Nonblocking Zero-indirection Transactional Memory