

Start Time: 11:25 a.m  
Finish Time: 11:35 a.m

Anil Ust

## PARELLEL PROGRAMMING with TRANSACTIONAL MEMORY

Speaker: Pratibha Kona

Scribe: Anil Ust

- Increasing the speed of execution
  - It uses more than one core but program must have parallelized, it divides the program work between the core
- Amdahl's Law
  - Shows the formula for speeding up which is
$$\frac{1}{(1-P) + \frac{P}{S}}$$

P=Fraction of the program  
S=Core
- Synchronization problems
  - Data memory needs to be shared, write access has to be controlled
  - Mutex technique can help to improve these problems.
  - If Mutex gets locked, it will cause new sets of problems.
- Problems with Locking Mutex
  - Cause of performance due to portion of program running in parallel.
  - Multiple mutex will increase the fraction (P) on Amdahl's Law, but also will increase the overhead locking and unlocking the mutex
- Deadlock
  - If overlapping Mutexes are locked by multiple threads in a different order
  - Livelock also problem with some of the algorithms
  - Deadlock detection will trigger more than once if multiple threads are executing
- The concept of transactional memory(TM) to solve the consistency
  - Allow sequences of concurrent operations to be combined into atomic transactions
  - A transaction is a piece of code that executes a series of reads and writes to shared memory
- Transaction
  - Checks the memory, if same memory location is used aborts the current transaction, if not records the memory location and marks it as its in use. Depending on the read or write it will communicate with the storage and do its work.
  - If transaction is aborted it will reset all the internal signals, stores the new values so memory location will be fresh to be reached.
  - Resets the information
- Issues with TM
  - May cause overheads.

Start Time: 11:25 a.m  
Finish Time: 11:35 a.m

Anil Ust

- If different variables depends on the same set of block, then both transactions will be aborted. False Sharing.
- Handling Aborts
  - There is no such way that one or the other algorithm is efficient than the other one.
  - Compilers will implement different each time and try and see if its seem to be more advantaged from the other.
- Conclusion
  - TM concept solves problems associated with locking system, there is still a lot of research is to be carried out to take the full advantage of the TM.

"Parallel Programming with Transactional Memory," U. Drepper, Comm. ACM, vol. 52, no. 2, pp. 38-43, February 2009.

[http://research.sun.com/spotlight/2007/2007-08-13\\_transactional\\_memory.html](http://research.sun.com/spotlight/2007/2007-08-13_transactional_memory.html)