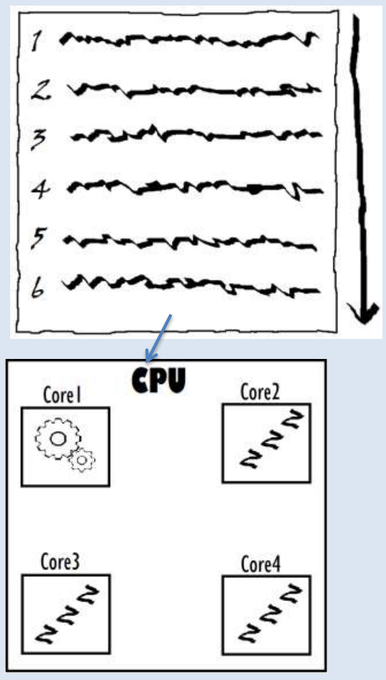
A Universal Construction for transaction based multi-process programs

Tyler Crain - Université de Rennes 1

It is now very clear that multi-core and multi-processor machines are the present and future of computing. It is also no secret that writing multi-process programs is very difficult. We try to address this problem by providing efficient abstractions to make multi-process programming easier.

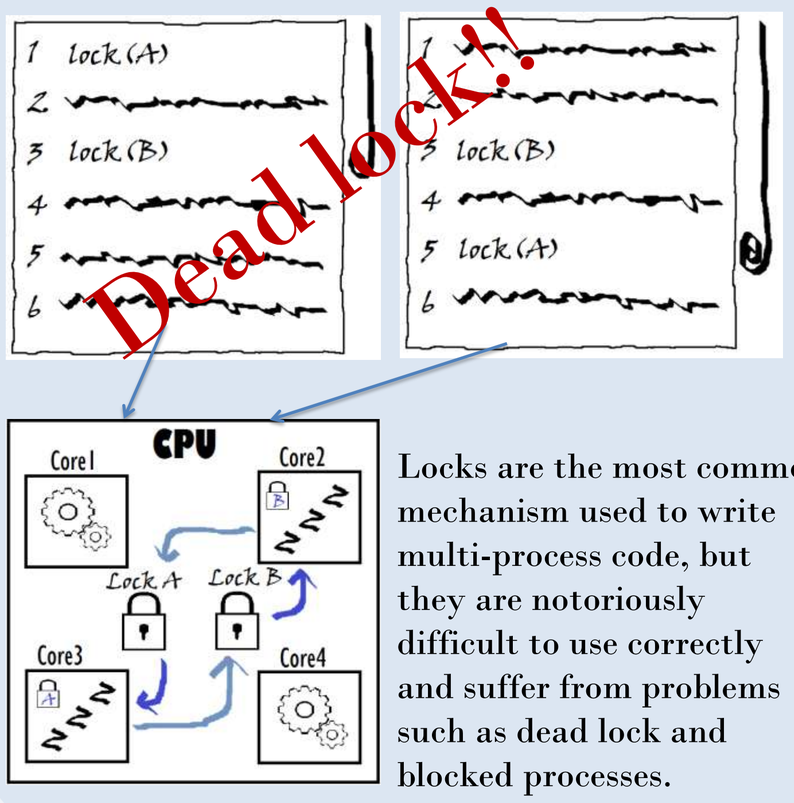
The problem → The solution?

Sequential Code



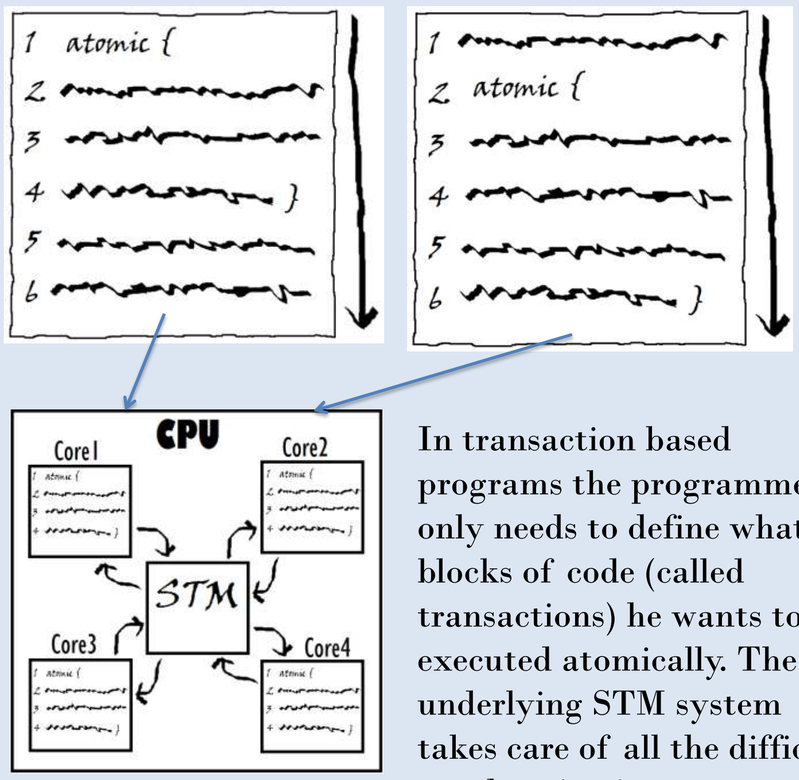
Running a sequential program on a multi-core machine leaves unused hardware.

Lock based Code



Locks are the most common mechanism used to write multi-process code, but they are notoriously difficult to use correctly and suffer from problems such as dead lock and blocked processes.

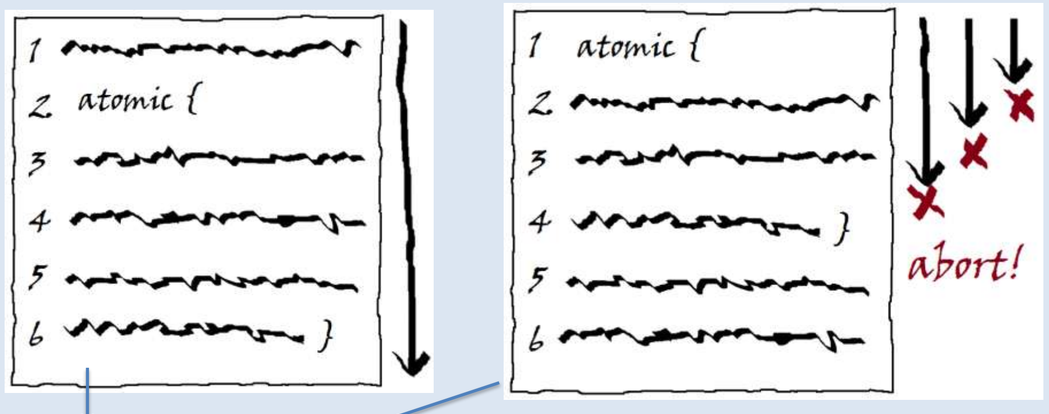
Software Transactional Memory (STM)



In transaction based programs the programmer only needs to define what blocks of code (called transactions) he wants to be executed atomically. The underlying STM system takes care of all the difficult synchronization.

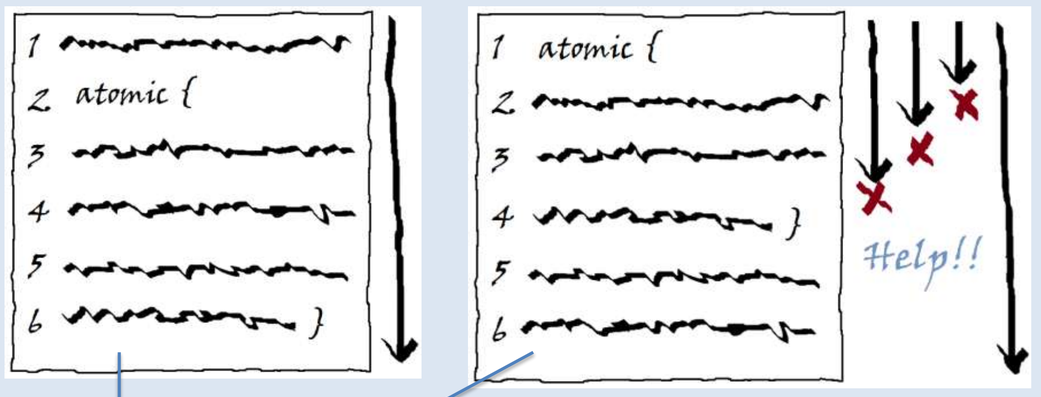
Still some problems… → Another solution

STM problem of aborts and blocking



Unfortunately it is not that simple, most STM implementations do not guarantee progress. A process might be blocked or a transaction might be aborted infinitely due to conflicts with concurrent transactions. The programmer must be aware of and able to deal with this.

A STM universal construction



The proposed Universal Construction guarantees that every transaction executes exactly once. The ideas of aborts and blocked transactions are unknown to the programmer. The main implementation idea is that cores can help other cores commit their transactions, ensuring progress.

A Universal Construction for transaction based multi-process programs is another step towards the goal of making concurrent programming more accessible. Synchronization is done using atomic read/write, fetch&increment, and compare&swap registers (no locks are used). A proof of correctness is given in the technical report. Ongoing work includes developing an efficient implementation of the algorithm.

Research funded by

TRANSFORM
Theoretical Foundations of Transactional Memory