Correctness and Concurrency: Formal Verification within UPMARC

UPMARC
Uppsala Programming for Multicore Architectures Research Center

UPPSALA UNIVERSITET

1 Automatic Verification under Relaxed Memory Models

Results for the TSO Memory Model

	Size Proc./States/Var./Trans	Total time seconds (one fence set)		Fences necessary (smallest set)
		SB	PB	
Simple Dekker	2/8/2/10	0.02	0.04	1 per process
Full Dekker	2/14/3/18	0.28	0.06	1 per process
Peterson	2/10/3/14	0.24	2	1 per process
Lamport Bakery (bounded)	2/22/4/32	52	19	2 per process
Lamport Fast	2/26/4/38	6.5	2	2 per process
CLH Queue Lock	2/48/4/60	26		0
Sense Reversing Barrier	2/16/2/24	1.1		0
Burns	2/9/2/11	0.07	0.02	1 per process
Dijkstra	2/14/3/24	9.5	0.35	1 per process
Tournament Barriers	2/8/2/8	1.2		0
Lamport Bakery (unbounded)	2/18/4/20	✖	154	2 per process
Linux Ticket Lock (unbounded)	2/8/2/8	✖	2	0

- Method SB: Sound and complete for boolean programs
- Method PB: Sound but incomplete for integer programs

Memory accesses in parallel programs are reordered by hardware.

- Makes lock free synchronization difficult
- Memory barriers often necessary for correctness
- Different reorderings possible on different architectures

Automatic Verification & Automatic Fence Insertion

Source Code Model (Simplified Dekker mutex lock)

P_0	P_1

x := 1
if(y == 0){
crit_sec$_0$()
}
	y := 1
	if(x == 0){
	crit_sec$_1$()
	}

⇒

Error!

P_0: y == 0
P_1: y := 1
P_1: x == 0
P_0: x := 1
(reordering)
P_0: crit_sec$_0$()
P_1: crit_sec$_1$()

⇒

Correction

P_0	P_1

x := 1
fence
if(y == 0){
crit_sec$_0$()
}
	y := 1
	fence
	if(x == 0){
	crit_sec$_1$()
	}

Prototype at https://github.com/memorax/memorax

Contact

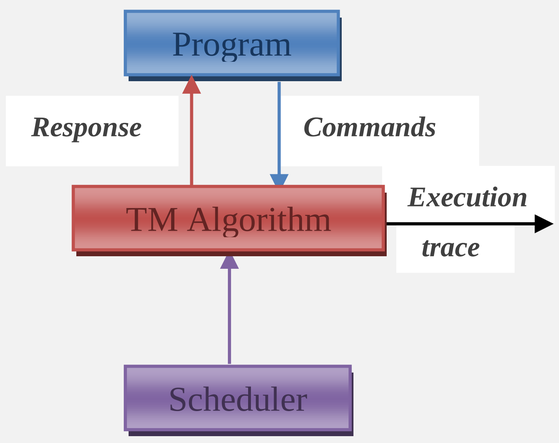
- Parosh Aziz Abdulla
- Mohamed Faouzi Atig
- Carl Leonardsson

2 Caches, Coherence and Accelerations of Transactional Memories

Motivation: Memory cashes in a multicore machine are distributed over many processors / many cores: Need for cash coherence protocols.

Challenge: Automatic verification of correctness (strict serializability) and liveness properties with an arbitrary number of caches or transactions of arbitrary length.

Case of study: **TMESI** *protocol.*



Transactional Memory (TM) system

Our approach is based on:

- Symbolic representations techniques.
- Language inclusion and simulation, Abstractions,
- Regular model checking

Long Term Goal

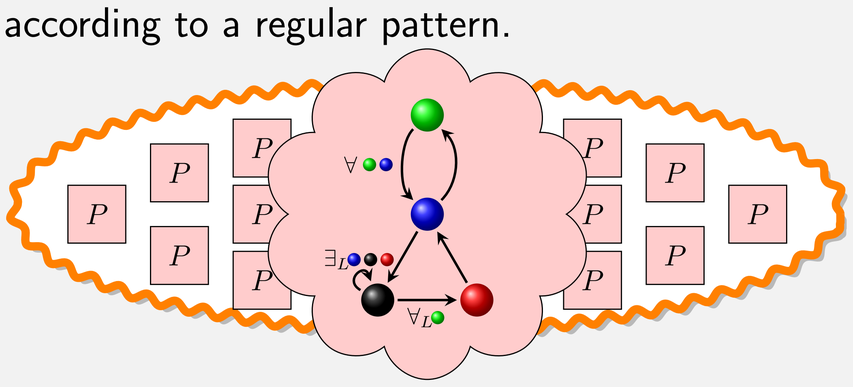
A platform that allows, with minimal human interaction, the verification and the validation of new cache protocols and hardware accelerated transactional memories.

Contact

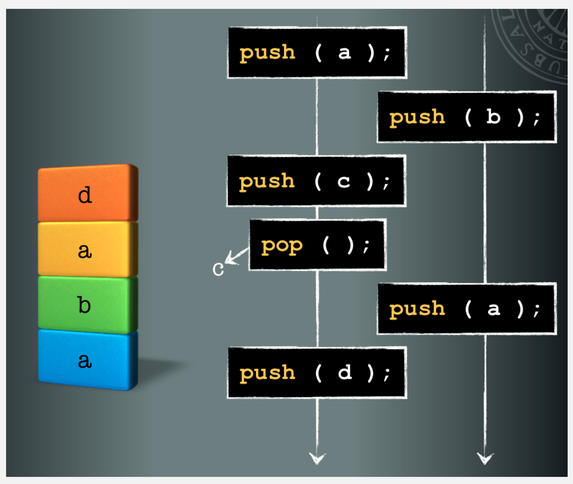
☐ Parosh Aziz Abdulla ☐ Lukáš Holík ☐ Yunyun Zhu.

3 Verification of highly concurrent algorithms / parameterized systems

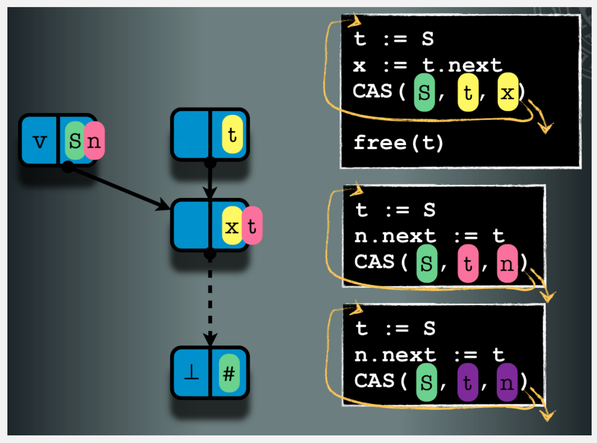
Objective: Verifying safety properties for *parameterized systems* that consist of arbitrary numbers of components (processes) organized according to a regular pattern.



Expected Results: Proving safety of an infinite family (one for each possible size of the system) by only inspecting small instances of the system.



Target: Highly concurrent data structures, such as Treiber's lock-free stack or Michael&Scott's lock-free queue



Expected Results: Proving stack or queue properties such as orderedness, lossyness or duplication, *in the absence of garbage collection*, by using the method for parametrized systems (on the left).

Contact

- Parosh Aziz Abdulla
- Frédéric Haziza
- Lukáš Holík

4 Verification of Multi-Push Down automata

Idea: Analysis tools for sequential programs are more mature than those for concurrent program. *Can we leverage that power?*

Yes! By translating concurrent programs to sequential ones.

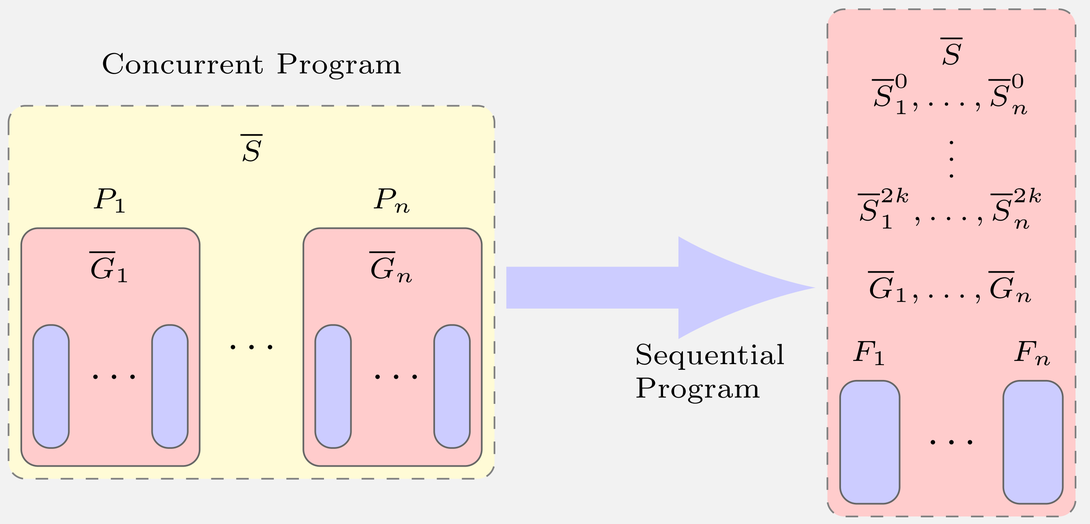
Sequential program construction:

- Each process P_i has *stack bound* k_i and a *context budget* b_i.
- Construct scheduler which simulates interleaving executions.
- A process P_i is only allowed b_i preemptions as long as its call stack is above k_i.

Prototype at http://user.it.uu.se/ jarst116/fmcad2012/

Results:

- Detection of **deep** bugs (big # interleavings needed).
- **Complete** correctness proof of certain programs.



Contact

- Parosh Aziz Abdulla
- Mohamed Faouzi Atig
- Jari Stenman
- Othmane Rezine