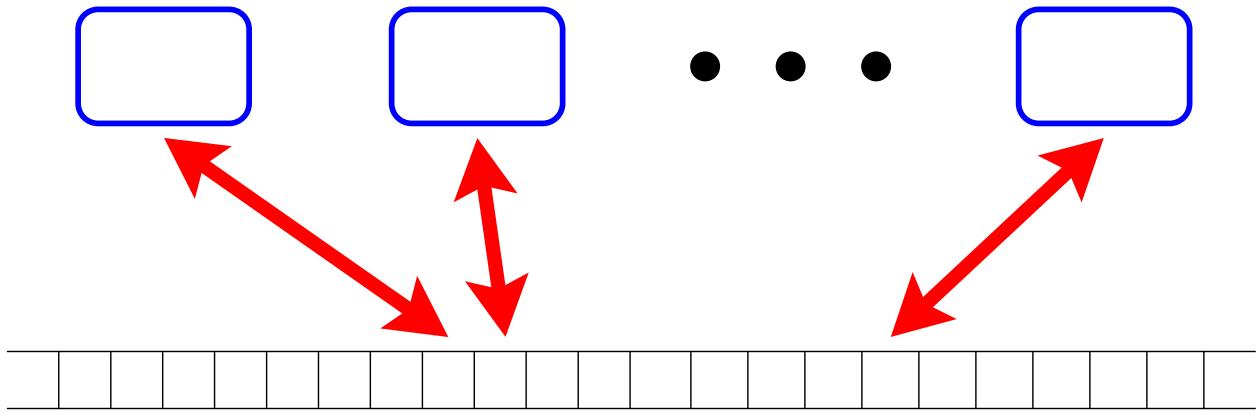
Verifying Multiple Compare and Swap

Joe Hurd
joe.hurd@cl.cam.ac.uk

University of Cambridge

Introduction

This work shows how a parallel architecture can be formalized in HOL, with many processors acting on a shared memory.



Relevant to possible future work of the ARM project (individual processors could be retargeted to ARM instruction set).

The Formalization Task

Create a logical model of a parallel architecture, supporting:

- independent execution of many processors communicating only through a shared memory; $\surd$
- arbitrary interleaving of instructions at the granularity of the primitive atomic operations; $\surd$
- plugging-in different memory models, with different rules for access reordering and barrier instructions; $\surd$
- automatic tools and techniques for specification and verification of concurrent programs. $\rightsquigarrow$current work

Parallel Architecture: Overview

- Identify a processor with its register file:

$$processor = register \rightarrow value$$

- The whole parallel architecture can then be modelled by a list of processors and a shared memory:

$$architecture = memory \times processor\ list$$

- The global state advances by any non-halted processor executing a primitive atomic instruction.

Parallel Architecture: Instruction Set

An *instruction* corresponds to a primitive atomic operation:

$$
\begin{aligned}
\textit{instruction} \;=\; & \text{NOP} \\
\mid\; & \text{UP of } \textit{value} \times \textit{register} \\
\mid\; & \text{RD of } \textit{register} \times \textit{register} \\
\mid\; & \text{WR of } \textit{register} \times \textit{register} \\
\mid\; & \text{CAS of } \textit{register} \times \textit{register} \times \textit{register} \times \textit{register} \\
\mid\; & \text{ALLOC of } \mathbb{N} \times \textit{register} \\
\mid\; & \text{MB}
\end{aligned}
$$

Some memory models (such as the Alpha) use memory barrier (MB) instructions to impose an ordering on accesses.

The CAS(n) Algorithm

- A case study to investigate verification in our model.
- CAS(n) stands for Multiple Compare And Swap:

  if the n memory addresses $a_1, \ldots, a_n$
  contain the expected values $x_1, \ldots, x_n$
  then replace them with the values $y_1, \ldots, y_n$

  (and this must all happen atomically!)

- A fast CAS(n) algorithm was recently developed by Harris, Fraser and Pratt in Cambridge.
- So far has led to:
  - instruction macros to simplify programming;
  - and a simulation engine to test arbitrary interleaving.

Conclusion

- We have laid out a formalization of a parallel architecture in HOL.
- We're working on case studies to develop techniques for specification and verification for a simple instantiation.
- There's scope for making the memory model more realistic (c.f. Gordon's model of the Alpha architecture),
- and also the processor instruction set (c.f. Fox's model of the ARM).
- The intention is that proofs completed for the simple instantiation lift up to more realistic models (we'll see).