SRC Effort 1031.001: Verification of Shared Memory Consistency Protocols: Overview of Research and Technology Transfer

Principal Investigators: Ganesh Gopalakrishnan and Konrad Slind

8th September 2003

Abstract

This document provides an update of our research in the area of Shared Memory Consistency Protocols. Our activities fall into three categories: (1) Modeling Shared memory consistency and the creation of a library of these models; (2) Improving the performance of a widely used tool for shared memory protocol verification - namely the Murphi tool - through symbolic-analysis (SAT-based) assisted Partial Order reduction, and (3) Improving the capabilities of the Murphi verification tool through parallel BFS model-checking as well as parallel heuristic combinations of BFS and Random-walk. We also provide an update on the students and industrial liaisons involved in these efforts.

1 Overview

Our research on Shared Memory Consistency Protocols is being conducted by Faculty PIs Ganesh Gopalakrishnan and Konrad Slind. Graduate students who have worked on this effort include

- Ritwik Bhattacharya (PhD, expected during Fall 2004); Advisor: Gopalakrishnan, with Slind on the committee
- Hemanthkumar Sivaraj (MS, expected during Fall 2003); Advisor: Gopalakrishnan
- Yue Yang (PhD, expected during Spring 2004); Advisors: Gopalakrishnan and Prof. Gary Lindstrom, with Slind on the committee
- Sudhindra Pandav (MS, expected during Fall 2004); Advisor: Slind, with Gopalakrishnan on the committee

In addition, our student ‘pipeline’ includes the following students contributing directly to our efforts:
• Robert Palmer (PhD, expected during Spring 2005); Advisor: Gopalakrishnan, with Slind on the committee

• Jacob Tripp (Undergraduate; expected to finish his BS degree by the end of 2003); Currently supported as an NSF REU undergraduate researcher; prospective SRC fellowship applicant.

The source of funding to support the students directly associated with the SRC effort has been the SRC contract. The source of support for the indirectly associated students has been an NSF grant shared by Gopalakrishnan and Slind. However, Yang and Pandav have been partly supported by our NSF awards as well.

Our industrial liaisons to date are:

• Ching-Tsun Chou, Intel Corporation, Santa Clara, CA
• Steven German, IBM, Yorktown Heights, NY
• Abdelillah Mokkedem, Intel Corporation, Shrewsbury, MA

2 Highlights

The highlights of our efforts are described along with key results. This deliverables report focusses on a subset of these items.

2.1 Efficient Verification of Shared Memory Consistency Protocols

Industrial shared memory consistency protocols are described and modeled using explicit state enumeration tools, the primary tool of choice being Murphi for many groups. BDDs are simply unable to handle the 3000+ bits of state present in these models [German, FMSD March’03]. While bounded model-checking methods are promising, currently there is not much hope for applying unbounded model-checking methods currently under active development [McMillan, CAV’03 Tutorials]. In the near term, therefore, one must

• make advances in explicit state enumeration model-checking of these protocols, and
• begin developing a combination of symbolic and explicit methods that can handle large state-spaces

Student Ritwik Bhattacharya, under the guidance of German (industrial liaison) and Gopalakrishnan, is developing a symbolic analysis method for Murphi descriptions, and a novel partial order reduction algorithm based on this analysis. We have finished implementing a version of Murphi containing our algorithm. Preliminary experiments are very encouraging, and will be submitted to one of the TACAS 2003 workshops. This work will be reported in detail in our December 2003 deliverable.
2.2 Parallel model-checking methods

The huge state-space of industrial shared memory consistency protocols makes it impossible to exhaustively model-check the protocols for system size-parameters larger than, say, a few cache lines and processing nodes. Parallel and distributed model-checking methods can help exploit the additive storage capabilities of many processors, thus helping finish model-checking in many cases. In this context, Sivaraj has developed two tools:

- A breadth-first search version of distributed Murphi that can run on clusters of computers running MPI. This is a porting of the original algorithm due to Stern and Dill (developed for the Berkeley Active networks platform) to the much more widely available MPI library.

- A parallel heuristic random-walk exploration tool that can also run on MPI clusters. This tool includes four novel combinations of breadth-first search and random-walk exploration that increases search effectiveness and dramatically reduces both memory consumption as well as the number of messages exchanged.

The first tool has been described in our CAV 2002 and FMCAD 2002 papers. The second tool was described in two submissions: (i) A paper published in the workshop on Parallel and Distributed Model Checking (post-CAV 2003), and (ii) A paper in TECHCON 2003. In addition, a paper will be submitted to one of the TACAS 2003 workshops.

A preliminary assessment of these tools was made during PI Gopalakrishnan’s recent sabbatical at Intel Corporation, Santa Clara. The first tool helped pool the memory resources of multiple machines, and even exhibited modest speed-ups. The second tool was able to detect certain deep bugs in actual protocols under development. While heuristic combinations of breadth-first search and parallel random-walk are efficient in many ways, an error-trail generation facility is essential to have before a thorough assessment of the random-walk tool can be made. We are implementing an innovative state checkpointing method in our distributed random-walk tool to support error-trail generation; after the completion of this facility, a thorough assessment of our tools will be made, and the tools made available to our industrial partners. These tools were described in our December deliverables. Updated reports will be generated once we write papers for the TACAS workshop in October 2003.

2.3 Analyzing the Intel Itanium Memory Ordering Rules

Shared memory consistency model specifications are notoriously hard to understand for designers, despite the utmost care taken to present things lucidly. The complexity is inherent in the complex set of ordering rules - see “A Formal Specification of Intel Itanium Processor Family Memory Orderings.” During his sabbatical at Intel, Gopalakrishnan began discussions with Intel engineers who summarized the issues that they are facing on a day-to-day basis:
• An executable specification of the memory ordering rules will help engineers understand the complex ordering rules. Current specifications are not executable.

• Such a tool can also greatly help during post-silicon verification of MP systems. Currently, this verification is based on ad-hoc non-scalable methods.

• An executable specification should, however, be based on a modular as well as declarative set of rules, so that the specification is reliable. Many current specifications are monolithic, making it very difficult to adapt memory model specifications.

In our Charme 2003 paper (“Analyzing the Intel Itanium Memory Ordering Rules using Constraint Programming and SAT,” by Yue Yang, Ganesh Gopalakrishnan, Gary Lindstrom, and Konrad Slind), we demonstrate how the Intel Itanium shared memory consistency model can be captured using a declarative specification. This specification

• directly translated into the language supported by most theorem-provers to support reasoning,

• directly executed using constraint-programming systems such as GnuProlog, and

• can be used for SAT-instance generation when verifying litmus tests.

As this topic pertains to our August 31 deliverable, it is described in detail in the accompanying deliverables report.

2.4 A Library of Operational and Axiomatic Specifications for Memory Models

This is also part of our August deliverable, and hence described fully in our deliverables report. The highlights of our work are as follows:

• We contribute high-level specifications in two complementary styles - namely the operational and the axiomatic. The advantages of obtaining specifications in these two styles are well-known: to reiterate,

  – the former assists implementers (cache coherence protocol designers and compiler writers) design their implementations;

  – the latter assists those industrial teams which manage architectural specifications across implementations of specific processor families.

Unlike most prior efforts, we provide executable specifications that directly benefit engineers:
• Our operational specifications are written in a parameterizable style—meaning, designers can redefine a few “bypass tables” and change a few “ordering rules” and obtain an executable specification for another memory model. In fact, we have one operational specification written in the popular Murphi model-checking notation, with switches that define which memory model is being defined. In our experience, memory model specifications are notoriously tricky to develop. Making them executable inside a model-checker greatly reduces the likelihood of having overlooked corner cases.

• Our axiomatic specifications are written in Constraint Prolog through an orthogonal collection of rules. By enabling / disabling these rules, a designer can understand how the rules contribute to the overall ordering relation - something hitherto not achieved in shared memory model specifications that define the ordering relation monolithically.

• To demonstrate that our Prolog descriptions are virtually formal specifications in mathematical logic, we also have a specification of memory ordering rules available in the notation of the Higher Order Logic (HOL) theorem-prover. Such rigorous specifications will allow us to prove properties about memory models using a theorem-prover. Many industrial experts are trained in theorem-proving in microprocessor companies, and this specification will especially appeal to them.

3 Publications

To summarize, the following papers related to these efforts have been published or in the works:

• Paper on SAT-based partial-order reduction methods - in preparation for the TACAS workshop

• Paper on parallel model-checking using Random Walk - presented at the Parallel and Distributed Model Checking workshop and also TECHCON 2003 (copies have been uploaded to SRC). A paper will possibly be written also for TACAS.

• Paper on modeling the Itanium memory model using Logic Programming and SAT (our Charme 2003 paper - will be uploaded to the SRC site).

• A paper is in preparation for the International Parallel and Distributed Processing Symposium (IPDPS) with deadline on October 3, 2003.

• A paper entitled “A Generic Operational Memory Model Specification Framework for Multithreaded Program Verification” is in preparation for submission to the journal Concurrency: Practice and Experience.
4 Software

As noted above, we have developed specifications in Murphi, Constraint Prolog, as well as Higher Order Logic. These will also be uploaded to the SRC site.

5 Technology Transfer

Technology transfer to Intel and IBM will be expedited thanks to our active involvement with our industrial liaisons. We will also seek newer liaisons; for instance, a few industrial experts who manage industrial shared memory consistency model specifications have been invited to serve as our liaisons.

6 Manpower Training

Two PhD students and one MS student are being trained on this project, with another PhD and MS student gaining exposure.

7 Concluding Remarks

This report is an overview of our activities to date. For the deliverables report, please see our accompanying report.