Parametric Verification of Industrial Cache Protocols

MURALI TALUPUR SAVA KRSTIĆ JOHN O'LEARY MARK R. TUTTLE

Strategic CAD Labs, Intel Corporation
Contact: murali.talupur@intel.com

Distributed protocols like cache coherence protocols form the bedrock on which modern multi-processor systems are built. Such distributed protocols are typically designed *parametrically*, that is, independent of the precise number of processors involved. Given that distributed programs are hard to reason about for humans and that no amount of testing/simulation can cover all scenarios, it becomes necessary that we find methods to formally and parametrically verify the correctness of such systems.

In this talk we will relate our practical experience with parameterized verification of an on-die cache coherence protocol for a many-core microprocessor design in progress at Intel. The protocol contains complexity that is not present in standard examples such as the FLASH or German protocols. To give an idea: the standard academic version of the German protocol has 7 different messages [4]; the FLASH protocol has 16 different messages and only 2 or 3 methods have ever been successful in verifying it parametrically. The Intel protocol with 54 different types of messages is vastly more complex. Moreover, the number of caching agents in the systems we are interested in is large enough to make parameterized verification a must: we found that conventional (non-parameterized) model checking techniques ran out of gas when confronted with more than four or five agents.

The verification technique we used is based on a method first described by McMillan [3] and subsequently elaborated by Chou, Mannava and Park [1] and Krstić [2]. This method, which we call the CMP method, is based on circular compositional reasoning and uses model checkers as proof assistants. Briefly, a parameterized system containing a directory and N caching agents is abstracted to a system containing a directory, two caching agents, and a third, highly nondeterministic, process representing "all the other agents". The user then supplies a series of lemmas that refine the behavior of the third agent; these lemmas are used mutually to prove one another and also the final property of interest. Coming up with these lemmas is a time-consuming process requiring a deep understanding of the protocol. It took us about a month and 25-odd lemmas to prove the cache coherence of the protocol. As far as we are aware this is the first time a protocol of this size and complexity has been verified parametrically.

The next step of the project was to make the CMP method easier to use by automating as much of it as possible. The method has three stages: (i) creating the inital abstraction, (ii) running the model checker and coming up with lemmas after examining counterexample traces, and (iii) refining the abstract model in light of the new lemmas. While discovering the lemmas requires ingenuity, the other two parts can be automated. We have built a tool that creates an initial abstraction and refines the abstract model with user-supplied lemmas automatically, and our talk will also describe the principles behind this tool.

The most important limitation of the CMP method is that it does not deal with systems in which processes are ordered by their indices. Commonly occurring algorithms break symmetry between processes by ordering them either linearly (as in the bakery algorithm), or placing them on a ring, grid or other network topology. The third element of our talk will explain our ongoing work to extend the CMP method to handle such asymmetric systems as well. Considering asymmetric systems led us to discover what we believe are new circular compositional reasoning principles. Besides enabling the CMP method to handle asymmetric systems, we anticipate that these new principles will also allow to formulate the CMP method itself much more intuitively and succinctly.

References

[1] C.-T. Chou, P. K. Mannava, and S. Park. A simple method for parameterized verification of cache coherence protocols. In A. J. Hu and A. K. Martin, editors, *Proc. Conf. on Formal Methods in Computer-Aided Design (FMCAD04)*, volume 3312 of *Lecture Notes in Computer Science*, pages 382–398. Springer, 2004.

[2] S. Krstić. Parameterized system verification with guard strengthening and parameter abstraction. In *Fourth Int. Workshop on Automatic Verification of Finite State Systems*, 2005. To appear in *Electronic Notes in Theoretical Computer Science*.

[3] K. L. McMillan. Parameterized verification of the FLASH cache coherence protocol by compositional model checking. In *Correct Hardware Design and Verification Methods (CHARME'01)*, volume 2144 of *LNCS*, pages 179–195, 2001.

[4] A. Pnueli, J. Xu, and L. D. Zuck. Liveness with (0, 1, ∞)-counter abstraction. In *CAV*, pages 107–122, 2002.