Thread-Modular Verification by Context Inference

Ranjit Jhala*
Computer Science Department
Jacobs School of Engineering
University of California, San Diego, USA
jhala@cs.ucsd.edu

ABSTRACT

Multithreaded programs are notoriously difficult to verify: the interleaving of concurrent threads causes an exponential explosion of the control state, and if threads can be dynamically created, the number of control states is unbounded. A classical thread-modular approach to verification is to consider the system as composed of a "main" thread and a context which is an abstraction of all the other "environment" threads of the system, and to then verify that (a) that this composed system is safe ("assume"), and, (b) that the context is indeed a sound abstraction ("guarantee"). Once the appropriate context has been divined, the above checks can be discharged by existing methods. Previously, such a context had to be provided manually, and if the given context is imprecise, then either check may fail leaving us with no information about whether the system is safe or not.

We show how to automate the thread-modular approach by: (a) finding a model for the context that is simultaneously (i) abstract enough to permit efficient checking and (ii) precise enough to preclude false positives as well as yield real error traces when the checks fail, and (b) showing how to infer such a context automatically. We give a novel way to construct stateful contexts, by representing individual environment threads as abstract finite state machines, and tracking arbitrarily many threads by counting the number of threads at each abstract state. We infer stateful contexts by iteratively weakening the abstract reachability analysis used for sequential programs, until an appropriate context is obtained.

We have implemented this algorithm in our C model checker BLAST, and used it to look for race conditions on networked embedded systems applications written in NesC, which use non-trivial synchronization idioms, that cause previous, imprecise analyses to race false alarms. We were able to find potential races in some cases and prove the absence of races in others.

*Joint work with Thomas A. Henzinger(EPFL, Switzerland) and Rupak Majumdar (UCLA, USA).

Biography

Ranjit Jhala is an Assistant Professor of Computer Science at the University of California, San Diego. Previously, he received a Ph.D. in Computer Science from the University of California, Berkeley, and before that, a B. Tech in Computer Science and Engineering from the Indian Institute of Technology, Delhi. He is interested in Programming Languages and Software Engineering, more specifically, in techniques for building reliable computer systems. The majority of his work has been on the BLAST software verification system which draws from, combines and contributes to techniques for Automated Deduction, Program Analysis and Model Checking.