J Autom Reasoning (2013) 51:241–280
DOI 10.1007/s10817-012-9258-1

Specification and Verification of Concurrent Programs Through Refinements

Sandip Ray · Rob Sumners

Received: 2 October 2011 / Accepted: 31 July 2012 / Published online: 30 August 2012

Abstract We present a framework for the specification and verification of reactive concurrent programs using general-purpose mechanical theorem proving. We define specifications for concurrent programs by formalizing a notion of refinements analogous to stuttering trace containment. The formalization supports the definition of intuitive specifications of the intended behavior of a program. We present a collection of proof rules that can be effectively orchestrated by a theorem prover to reason about complex programs using refinements. The proof rules systematically reduce the correctness proof for a concurrent program to the definition and proof of an invariant. We include automated support for discharging this invariant proof with a predicate abstraction tool that leverages the existing theorems proven about the components of the concurrent programs. The framework is integrated with the ACL2 theorem prover and we demonstrate its use in the verification of several concurrent programs in ACL2.

Keywords Abstraction · Fairness · Reactive systems · Theorem proving · Trace containment

1 Introduction

Reactive concurrent programs consist of a number of interacting components (e.g., processes, threads, etc.) that perform ongoing, non-terminating computations while

This work was done when the corresponding author was at the University of Texas at Austin.

S. Ray (✉)
Strategic CAD Labs, Intel Corporation, Hillsboro, OR 97124, USA
e-mail: sandip.ray@intel.com

R. Sumners
Advanced Micro Devices, Inc., Austin, TX 78741, USA
e-mail: robert.sumners@amd.com