A Novel Approach to Parameterized verification of Cache Coherence Protocols

Yongjian Li[1] Kaiqiang Duan[1] Yi Lv[1] Jun Pang[2] Yi Lv[1] Shaowei Cai[1]

State Key Laboratory of Computer Science, China

Computer Science and Communications, University of Luxembourg, Luxembourg

2016 年 9 月 27 日

Problem of Parameterized Verification

Consdier a protocol P, a property Inv

- $P(N) \models Inv$ for any N
- not just for a single protocol instance $P(c) \models Inv$
- Our opinion: parameterized verification is a theorem proving problem

State of Arts

- CMP: parameter abstraction and parameter abstraction
- Proposed, by McMillan, elaborated by Chou, Mannava, and Park (CMP) , and formalized by Krstic
- construction of an abstract instance which can simulate any protocol instance
- human provides auxiliary invariants (non-interference lemmas)

State of Arts

- invisible invariants,
- auxiliary invariants are computed from reachable state set in a finite protocol instance $P(c)$
- raw formula translated from BDD
- the reachable state set can't be enumerated, e.g., the FLASH protocol

Two central and difficult problems

- searching auxiliary invariants is not automatic
- soundness problem: the theoretical foundation is not mechanized, and there is no a formal proof

Our Motivation

- automatically searching auxiliary invariants
- Formally proving all the things: both the theoretical foundation and case studies
- A formal proof script as a formal verification product

An Overview of Our Approach

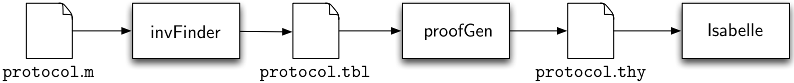

Figure: The workflow of paraVerifier.

Some Explanations

- paraVerifier=invFinder + proofGen
- protoocl.fl: a small reference instance of the protocol
- invFinder searches auxiliary invariants automatically
- protocol.tbl: stores the set of ground invariants and a causal relation table
- proofGen: create an Isabelle proof script protocol.thy which models and verifies the protocol
- run Isabelle script to automatically proof-check protocol.thy

Theoretical Foundation-Protocol

A protocol is formalized as a pair $(ini, rules)$, where

- ini is an initialization formula; and
- $rules$ is a set of transition rules. Each rule $r \in rules$ is defined as $g \triangleright S$, where g is a predicate, and S is a parallel assignment to distinct variables v_i with expressions e_i, where S is a parallel assignment $S = \{x_i := e_i | i > 0\}$.

We write pre $r = g$, and act $r = S$ if $r = g \triangleright S$.

Theoretical Foundation-Causal Relation

Definition

We define the following relations

1. $\text{invHoldForRule}_1\ s\ f\ r \equiv s \models \text{pre}\ r \longrightarrow s \models \text{preCond}\ f\ (\text{act}\ r)$, where $\text{preCond}\ S\ f = f[x_i := e_i]$, which substitutes each occurrence of x_i by e_i;
2. $\text{invHoldForRule}_2\ s\ f\ r \equiv s \models f \longleftrightarrow s \models \text{preCond}\ f\ (\text{act}\ r)$;
3. $\text{invHoldForRule}_3\ s\ f\ r\ F \equiv \exists f' \in F$ s.t. $s \models (f' \wedge (\text{pre}\ r)) \longrightarrow s \models \text{preCond}\ f\ (\text{act}\ r)$;
4. $\text{invHoldForRule}\ s\ f\ r\ F$ represents a disjunction of invHoldForRule_1, invHoldForRule_2 and invHoldForRule_3.

Theoretical Foundation - Consistency Relation)

Definition

A consistency relation, i.e., consistent *invs ini rules*, that holds between a protocol $(ini, rules)$ and a set of invariants $invs = \{inv_1, \ldots, inv_n\}$, is defined as:

- For any invariant $inv \in invs$ and state s, if *ini* is evaluated as true at state s (i.e., formEval $ini\ s = true$), then *inv* is also evaluated as true at the state s.
- For any $inv \in invs$, and $r \in rules$, and any state s, invHoldForRule *inv r invs*.

Theoretical Foundation - Consistent Lemma

Lemma

For a protocol $(ini, rules)$, *we use* reachableSet *ini rules to denote the set of reachable states of the protocol. Given a set of invariants invs, we have* $[|\text{consistent } invs\ ini\ rules; s \in \text{reachableSet } ini\ rules|] \Longrightarrow \forall inv \in invs.\text{formEval } inv\ s$