HSF(C): A Software Verifier based on Horn Clauses

Corneliu Popeea
Technical University Munich

Joint work with Sergey Grebenshchikov,
Ashutosh Gupta,
Nuno P. Lopes and
Andrey Rybalchenko

Developing verifiers today

Program Model

transition system, program with procedures, multi-threaded program, functional program, ...

+

Proof Rule

Invariance, summarization, rely/guarantee, transition invariance, refinement typing, ...

+

Complex verification effort

=

Verification Tool

Developing verifiers tomorrow

Verification Tool = Synthesizer (Program Model, Proof Rule)

Proof rules

Init(V) -> Inv(V)
Inv(V) ∧ Step(V, V') -> Inv(V')
Inv(V) ∧ Error(V) -> ⊥

Transition system is safe

Inv(V) ∧ Step(V, V') -> TransInv(V, V')
TransInv(V, V') ∧ Step(V', V'') ->
TransInv(V, V'')
dwf(TransInv(V, V'))

Transition system terminates

Init(V) ∧ V'=V -> Summ(V,V')
Summ(V,V') ∧ Step(V', V'') -> Summ(V,V'')
Summ(V,V') ∧ Call(V', V'') ∧ V'''=V'' -> Summ(V'',V''')
Summ(V,V') ∧ Call(V', V'') ∧ Summ(V'', V''') ∧
Return(V''', V'''') ∧ Local(V', V'''') -> Summ(V,V'''')
Summ(V,V') ∧ Error(V') -> ⊥

Procedural program is safe

Init(V) → Inv_i(V)
Inv_i(V) ∧ $Step_i$(V, V') → Inv_i(V')
(∨ Inv_i(V) ∧ $Step_i$(V,V')) → Env_j(V,V')
Inv_i(V) ∧ Env_i(V,V') → Inv_i(V')
Inv_1(V) ∧ .. ∧ Inv_N(V) ∧ Error(V) → ⊥

Multi-threaded program is safe

HSF(C)

- C frontend based on CIL [Necula-et-al, CC 2002]
  - translates input program to Horn clauses
- Summarization proof rule [Reps, Horwitz, Sagiv - POPL 1995]
- HSF algorithm
  [Grebenshchikov, Lopes, Popeea, Rybalchenko - PLDI 2012]

Competition candidate	CEGAR	Predicate Abstraction	Bounded Model Checking	Shape Analysis	ART-based Analysis	Lazy Abstraction	Interpolation	Concurrency Support
BLAST	✓	✓			✓	✓	✓	
CPA-ABE	✓	✓			✓	✓	✓	
CPA-MEMO	✓	✓			✓	✓	✓	
ESBMC			✓					✓
FSHELL			✓					
LLBMC			✓					
PREDATOR				✓				
QARMC-HSF(C)	✓	✓			✓		✓	✓
SATABS	✓	✓						✓
WOLVERINE	✓				✓	✓	✓	

HSF(C) results

ControlFlowInteger category:
- 96 benchmarks

Place	Tool	Points (144 max)
1st	CPAChecker-ABE	141
2nd	CPAChecker-Memo	140
3rd	**HSF(C)**	**140**
4th	ESBMC	102

94 correct results in 80 minutes
2 time/outs

HSF and related work

- Software verification tools
  - Slam, Blast, Terminator, CPAchecker, DSolve
- Verifiers - target for automated synthesis
  - XSB: generates model checkers for CCS programs
  - Getafix: generates model checkers for boolean programs

HSF: generates model checkers for C and OCaml programs competitive with mature software verification tools

Synthesizing software verifiers from proof rules [Grebenshchikov, Lopes, Popeea, Rybalchenko - PLDI 2012]

Questions?