Bounded Model-Checking Unbounded Object Creation

F.S.de Boer (CWI & LIACS)
Joint work with
Marcello Bonsangue and Jurriaan Rot (LIACS)
Dorel Lucano and Irina Mariuca (UAIC)

Pushdown Automata

$$(P, \Gamma, \delta)$$

where

- P is a finite set of *control locations*
- Γ is a finite *stack alphabet*
- $\Delta \subseteq (PX\Gamma)X(PX\Gamma^*)$ is a finite set of *transitions*:

$$(p, \gamma) \rightarrow (p', \gamma')$$

Model-checking LTL properties of Pushdown Automata:

A. Bouajjani, J. Esparza, O. Maler.
Reachability Analysis of Pushdown Automata: Application to Model-Checking.
Concur 1997.

Semantics of Object-Oriented Programs

Let $V = L \cup G$ be the set of *local* and *global* variables and D a set of Data.

Control locations $Stat \times (V \to D)$

Stack Alphabet $Stat \times (L \to D)$

Method call

$$\langle (m; S, H), \gamma \rangle \to \langle (B, I), (S, H \downarrow_L) \circ \gamma \rangle$$

Return

$$\langle (return, H), (S, H') \circ \gamma \rangle \to \langle (S, H \downarrow_G \cup H'), \gamma \rangle$$

Object Creation

$$\langle (x = new; S, H), \gamma \rangle \to (c = c + 1; x = c; S, H), \gamma \rangle$$

Unbounded Object Creation ("It's All Too Much")

Even this very simple program

$$m\{u = new; m\}$$

we cannot model-check!

Can we do better?

Yes, we can!

Incremental Approach ("Easy Does it")

1. Global variables only;
2. add local variables;
3. add fields.

For Global Variables Only

Heap:

$$G \rightarrow |G|$$

Object Creation:

$$\langle (x = new; S, H), \gamma \rangle \rightarrow (S, H[x = k]), \gamma \rangle$$

where $k \in 1..|G|$ is *unused* in H.

And Now ... Local Variables!

Object Creation

$$\langle (x = new; S, H), \gamma \rangle \rightarrow (S, H[x = k]), \gamma \rangle$$

where $k \in 1..|G| + |L|$ *not used in the visible heap* H.
Problem:

name **clashes**

Example: Let x be a global variable x in

$$\langle (x = new, H), (S, H') \circ \gamma \rangle \rightarrow \langle (return, H), (S, H') \circ \gamma \rangle$$

where

$$H(x) = H'(u)$$

for some local variable u.

Renaming ("What's In A Name"): First Attempt

$$\langle (return, H), (S, H') \circ \gamma \rangle \rightarrow \langle (S, \rho(H \downarrow_G) \cup H'), \gamma \rangle$$

where

$$\rho : N \rightarrow N$$

is a bijection ($\rho(H)(x) = \rho(H(x))$).

However

we cannot distinguish valid identifications between global variables of the visible heap H and the stacked local variables of H'!

Second Attempt: The Cutting Edge

The Twain **Shall** Meet:

$$H(G) \cap H(L)$$

Shadowing Cut-points

Let $C \subseteq L$ be a finite set of *cut-point variables* (not used in programs).

Method call

$$\langle (m; S, H), \gamma \rangle \rightarrow \langle (B, I), (S, H \downarrow_L) \circ \gamma \rangle$$

where

$$I(C) = H(G) \cap H(L)$$

Return

$$\langle (return, H), (S, H') \circ \gamma \rangle \rightarrow \langle (S, \rho(H \downarrow_G) \cup H'), \gamma \rangle$$

where $\rho(H(c)) = H(c)$, for $c \in C$.

Adding Fields ("Nothing To Get Hung About")

Requires:

- Generalization of cut-points.
- Bound on the size of the visible heap.
  Note: the resulting language is Turing Complete.

Standing On The Shoulders Of Giants

1. N. Rinetzky, J. Bauer, T. Reps, M. Sagiv, R. Wilhelm.
   A semantics for procedure local heaps and its abstractions.
   POPL 2005, ACM.
2. A. Bouajjani, S. Fratani, S. Qadeer. Context-Bounded Analysis of Multithreaded Programs with Dynamic Linked Structures.
   CAV 2007, LNCS 4590.

Our Contribution

- Explicit naming scheme;
- Formal proof of correctness;
- Implementation:
  - Executable semantics in K.
  - Maude model-checker of LTL properties over regular heap expressions, e.g.,

$$\Box(\mathit{first}; \mathit{next}^*; \mathit{last})$$