Study of stepwise simulation between ASMs

Patrick Cegielski Julien Cervelle

JAF 37 • Florence • 2018-05-29

LACL • Université Paris-Est Créteil

ASMs

$A = \langle D, S, V, P \rangle$

- D is the domain
- S are the static symbols
- $\langle D, S \rangle$ forms an algebra
- V are the dynamic symbols
- P is the program

A subset $I \subset V$ are the input symbols

$\perp \in D$ is a special value

P is a loop of instructions $\varphi \implies a$

- φ is a quantifier-free formula of $\mathcal{L}(S \cup V \cup \{\mathtt{undef}\})$
- a is an assignment $\mathsf{s}(\bar{t}) := v$
    - s is a dynamic symbols of arity $|\bar{t}|$
    - v and $\bar{t}$ are terms of $\mathcal{L}(S \cup V \cup \{\mathtt{undef}\})$

Instructions in the loop are executed in parallel and must be non contradictory

ASM halts on fixpoint

Trace

The **state** of an ASM is the values stored in its *dynamic* symbols.

The **initial state** for an ASM on input x is such that

- The input symbols are filled with x
- The other dynamic symbols are filled with $\perp$

The **trace** of ASM A on input x: $t_0, t_1, \ldots, t_n, \ldots$

- t_0 = the state of A initialized with x
- t_{i+1} = the state after one step of A from state t_i
- If the run halts, the last element is halting (fix point)
- Otherwise the trace is infinite

Simulation

ASM were introduced as a universal algorithm model

Any sequential algorithm is simulated by an ASM

- $1 - 1$ **simulation**: one step of the ASM = one step of the algorithm
- $n - 1$ **simulation**: exactly n steps of the ASM = one step of the algorithm

Often, the ASM is padded with "skip" instructions to reach n steps

n **weak** simulation: **at most** n steps of the ASM = one step of the algorithm

Question: *could one use weak simulation?*

Example: classic simulation

$\texttt{x:=}\bot \Longrightarrow \texttt{x:=1}$

$\texttt{x:=1} \Longrightarrow \texttt{x:=2}$

$\texttt{x:=2} \Longrightarrow \texttt{x:=5}$

$\texttt{x:=5} \Longrightarrow \texttt{x:=1}$

$\texttt{s}\neq 9 \wedge \texttt{s}\neq\bot \Longrightarrow \texttt{s:=s+1}$

$\texttt{s}=9 \vee \texttt{s}=\bot \Longrightarrow \texttt{s:=1}$

$\texttt{s=2} \Longrightarrow \texttt{x:=1}$

$5\leqslant \texttt{s} \leqslant 8 \Longrightarrow \texttt{x:=x+1}$

x
$\bot$
1
2
5
1
$\vdots$

s	x	s	x
$\bot$	$\bot$	7	3
1	$\bot$	8	4
2	$\bot$	9	5
3	1	1	5
4	1	2	5
5	1	3	1
6	2	$\vdots$	

Example: weak simulation

`x:=⊥` $\Longrightarrow$ `x:=1`

`x:=1` $\Longrightarrow$ `x:=2`

`x:=2` $\Longrightarrow$ `x:=5`

`x:=5` $\Longrightarrow$ `x:=1`

`x≠5∧x≠⊥` $\Longrightarrow$ `x:=x+1`

`x=5∨x=⊥` $\Longrightarrow$ `x:=1`

$$\begin{array}{c} \times \\ \perp \\ 1 \\ 2 \\ 5 \\ 1 \\ \vdots \end{array} \qquad \begin{array}{c} \times \\ \perp \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 1 \\ \vdots \end{array}$$

Motivation

- Intuitively, with weak simulation, simulated machine can compute more than the simulating machine
- Weak simulation need an oracle to tell when a simulating step is reached
- The oracle can be described by the set of index to remove from the simulating trace to get only the simulating steps
- Question: *can we find a case where these oracle are "non-computable" while the ASM use only computable elements?*

Formalization

Let A and B be two ASMs where B n-weak-simulates A

A and B must have the same domain, the dynamic symbols of B greater or equal the one of A, and the same input set

Let $t_A(x)$ and $t_B(x)$ be the **traces** of A and B on input x

We call **witness** for this weak-simulation a set $W_x \subseteq \mathbb{N}$ such that:

1. $t_A(x) = V_A^{*,\omega} \cap t_B(x) \restriction_{W_x^{\complement}}$ (simulation)
2. no interval grater than $n - 1$ is included in W_x (n-weak)

Definition

An ASM is *arithmetic* if:

- it's domain is $\mathbb{N}$
- all members of the algebra are computable

Proposition

Let A and B be two arithmetic ASMs. If A is n-weakly simulated by B then $\mathcal{W} = \{W_x \mid x \text{ halting input for } A\}$ is computable.

Proof: The Turing machine recover the input and simulates A and B in parallel on input x and once finished, it outputs W_x.

Upper bound : Undecidable

Proposition

There exists some arithmetic ASMs A and some non-arithmetic B such that $W_\emptyset$ is not recursive.

`n=⊥` $\Longrightarrow$ `m:=0∧n:=1`

`n≠⊥` $\Longrightarrow$ `n:=n+1`

n	$\perp$	1	2	3	4	5	6
m	$\perp$	0	0	0	0	0	0

`n=⊥` $\Longrightarrow$ `m:=0∧n:=1`

`n is even ∧ f(n/2)=1` $\Longrightarrow$ `m:=1`

`m=1` $\Longrightarrow$ `m:=0∧n:=n+1`

`n is even ∧ f(n/2)=0` $\Longrightarrow$ `n:=n+1`

`n is odd` $\Longrightarrow$ `n:=n+1`

n	$\perp$	1	2	2	3	4	5	6	6
m	$\perp$	0	0	1	0	0	0	0	1

where f is the characteristic funtion of some non-computable set (in the example, $f(1) = f(3) = 1$ and $f(2) = 0$).

Main result

Theorem

There exists some arithmetic ASMs A and B such that $\mathcal{W} = \{W_x \mid x \text{ input for } A\}$ contains only finite sets and is non-computable.

Proof: Both perform the following steps but when c reached 0, some ASM K is executed and in parallel on the same input.

```
s=⊥ ⟹ c:=input+1∧s:=0
s=1∧c>0 ⟹ c:=c-1
c=0∧¬KHasHalted ⟹ s:=s+1
```

```
s=⊥ ⟹ c:=input+1∧s:=1
s=1∧c>1 ⟹ c:=c-1
s=1∧c=1∧m=⊥ ⟹ m:=1
s=1∧c=1∧m=1 ⟹ m:=⊥∧c:=0
c=0∧¬KHasHalted ⟹ s:=s+1
c=0∧KHasHalted ⟹ m:=1
```

Where K is some ASM which has non computable halting set

Proof

When `input`$= 6$, K does not halt:

$$\begin{array}{ccccccccccccccc}
c & \perp & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
m & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \cdots \\
d & \perp & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 5 & \cdots \\
K & & & & & & & & & & K_0 & K_1 & K_2 & K_3 & K_4 & \cdots
\end{array}$$

$$\begin{array}{cccccccccccccccc}
c & \perp & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
m & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & 1 & \perp & \perp & \perp & \perp & \perp & \perp & \cdots \\
d & \perp & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 5 & \cdots \\
K & & & & & & & & & & & K_0 & K_1 & K_2 & K_3 & K_4 & \cdots
\end{array}$$

When K does not halt on input x, W_x is the singleton $\{x + 2\}$

Proof

When `input`$=5$, K halts after 3 step:

$$\begin{array}{c} c \\ m \\ d \\ K \end{array} \begin{bmatrix} \perp & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp & \perp \\ \perp & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 \\ & & & & & & & & K_0 & K_1 & K_2 & K_3 \end{bmatrix}$$

$$\begin{array}{c} c \\ m \\ d \\ K \end{array} \begin{bmatrix} \perp & 6 & 5 & 4 & 3 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \perp & \perp & \perp & \perp & \perp & \perp & \perp & 1 & \perp & \perp & \perp & \perp & \perp & 1 \\ \perp & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 4 \\ & & & & & & & & & K_0 & K_1 & K_2 & K_3 & K_3 \end{bmatrix}$$

When K halts on input x, W_x is a pair.

The enumeration of $\mathcal{W}$ allows to enumerate the halting set of K and its complement and thus decide K.

Conclusion

We confirm that simulation cannot be replaced by weak-simulation

Also, instead of padding with “skip”, one can add a special dymamic boolean symbol “sim” set to `true` only for simulating steps