Undecidability of Boolean bunched logic

James Brotherston

Programming Principles, Logic and Verification Group
Dept. of Computer Science
University College London, UK
J.Brotherston@ucl.ac.uk

Logic Summer School, ANU, 10 December 2015

Introduction

Previously:

- we showed that the Hilbert system for BBI is sound and complete w.r.t. validity in an associated class of Kripke models;

Introduction

Previously:

- we showed that the Hilbert system for BBI is sound and complete w.r.t. validity in an associated class of Kripke models;

- we reformulated the Hilbert system as an analytic, cut-eliminating display calculus.

Introduction

Previously:

- we showed that the Hilbert system for BBI is sound and complete w.r.t. validity in an associated class of Kripke models;

- we reformulated the Hilbert system as an analytic, cut-eliminating display calculus.

You might think that BBI is therefore decidable: given a formula A, just conduct an exhaustive search for $\vdash A$ in the display calculus.

Introduction

Previously:

- we showed that the Hilbert system for BBI is sound and complete w.r.t. validity in an associated class of Kripke models;
- we reformulated the Hilbert system as an analytic, cut-eliminating display calculus.

You might think that BBI is therefore decidable: given a formula A, just conduct an exhaustive search for $\vdash A$ in the display calculus.

But, actually, it isn't. That's today's subject.

BBI, *proof-theoretically*

Recall:

Provability in BBI is given by extending a Hilbert system for propositional classical logic by

$$A * B \vdash B * A \qquad A * (B * C) \vdash (A * B) * C$$

$$A \vdash A * \mathrm{I} \qquad A * \mathrm{I} \vdash A$$

$$\frac{A_1 \vdash B_1 \quad A_2 \vdash B_2}{A_1 * A_2 \vdash B_1 * B_2} \qquad \frac{A * B \vdash C}{A \vdash B \mathrel{-\!\!*} C} \qquad \frac{A \vdash B \mathrel{-\!\!*} C}{A * B \vdash C}$$

BBI, *semantically (1)*

Recall:

A BBI-model is given by $\langle W, \circ, E \rangle$, where

- W is a set (of "worlds"),
- $\circ$ is a binary function $W \times W \to \mathcal{P}(W)$; we extend $\circ$ to $\mathcal{P}(W) \times \mathcal{P}(W) \to \mathcal{P}(W)$ by

$$W_1 \circ W_2 =_{\text{def}} \bigcup_{w_1 \in W_1, w_2 \in W_2} w_1 \circ w_2$$

- $\circ$ is commutative and associative;
- the set of units $E \subseteq W$ satisfies $w \circ E = \{w\}$ for all $w \in W$.

A valuation for BBI-model $M = \langle W, \circ, E \rangle$ is a function ρ from propositional variables to $\mathcal{P}(W)$.

BBI, *semantically (2)*

Given M, ρ, and $w \in W$, we define the forcing relation $w \models_\rho A$ by induction on formula A:

$$\begin{array}{rcl} w \models_\rho P & \Leftrightarrow & w \in \rho(P) \\ w \models_\rho A \to B & \Leftrightarrow & w \models_\rho A \text{ implies } w \models_\rho B \\ \vdots & & \\ w \models_\rho \mathrm{I} & \Leftrightarrow & w \in E \\ w \models_\rho A * B & \Leftrightarrow & w \in w_1 \circ w_2 \text{ and } w_1 \models_\rho A \text{ and } w_2 \models_\rho B \\ w \models_\rho A \mathrel{-\!\!*} B & \Leftrightarrow & \forall w', w'' \in W.\ \text{if } w'' \in w \circ w' \text{ and } w' \models_\rho A \\ & & \text{then } w'' \models_\rho B \end{array}$$

A is valid in M iff $w \models_\rho A$ for all ρ and $w \in W$.

Undecidability strategy

- There is basically one universal strategy for showing things undecidable: by reduction from some problem already known undecidable!

Undecidability strategy

- There is basically one universal strategy for showing things undecidable: by reduction from some problem already known undecidable!
- That is, we show that if we could decide validity of BBI-formulas, then we could decide some other undecidable problem.

Undecidability strategy

- There is basically one universal strategy for showing things undecidable: by reduction from some problem already known undecidable!
- That is, we show that if we could decide validity of BBI-formulas, then we could decide some other undecidable problem.
- Classic undecidable problem: the halting problem, as famously considered by Turing.

Undecidability strategy

- There is basically one universal strategy for showing things undecidable: by reduction from some problem already known undecidable!
- That is, we show that if we could decide validity of BBI-formulas, then we could decide some other undecidable problem.
- Classic undecidable problem: the halting problem, as famously considered by Turing.
- Turing machines are not very convenient for our purposes (why not?), so we shall instead consider the halting problem for two counter Minsky machines.

Minsky machines

A Minsky machine M with counters c_1, c_2 is given by a finite set of labelled instructions of the following types, where $k \in \{1, 2\}$:

Minsky machines

A Minsky machine M with counters c_1, c_2 is given by a finite set of labelled instructions of the following types, where $k \in \{1, 2\}$:

L_i: c_k++; **goto** L_j;	"increment c_k (and jump)"
L_i: c_k−−; **goto** L_j;	"decrement c_k (and jump)"
L_i: **if** $c_k = 0$ **goto** L_j;	"zero-test c_k (and jump)"
L_i: **goto** L_j;	"jump"

Minsky machines

A Minsky machine M with counters c_1, c_2 is given by a finite set of labelled instructions of the following types, where $k \in \{1, 2\}$:

L_i: c_k++; **goto** L_j;	"increment c_k (and jump)"
L_i: c_k−−; **goto** L_j;	"decrement c_k (and jump)"
L_i: **if** $c_k = 0$ **goto** L_j;	"zero-test c_k (and jump)"
L_i: **goto** L_j;	"jump"

Configurations of M have the form $\langle L_i, n_1, n_2\rangle$. We write $\langle L_i, n_1, n_2\rangle{\Downarrow_M}$ if $\langle L_i, n_1, n_2\rangle \rightsquigarrow_M^* \langle L_0, 0, 0\rangle$.

Minsky machines

A Minsky machine M with counters c_1, c_2 is given by a finite set of labelled instructions of the following types, where $k \in \{1,2\}$:

L_i: c_k++; **goto** L_j;	"increment c_k (and jump)"
L_i: c_k−−; **goto** L_j;	"decrement c_k (and jump)"
L_i: **if** $c_k = 0$ **goto** L_j;	"zero-test c_k (and jump)"
L_i: **goto** L_j;	"jump"

Configurations of M have the form $\langle L_i, n_1, n_2\rangle$. We write $\langle L_i, n_1, n_2\rangle{\Downarrow_M}$ if $\langle L_i, n_1, n_2\rangle \leadsto_M^* \langle L_0, 0, 0\rangle$.
We introduce special labels L_{-1}, L_{-2} with instructions:

L_{-1}: c_2−−; **goto** L_{-1};	L_{-1}: **goto** L_0;
L_{-2}: c_1−−; **goto** L_{-2};	L_{-2}: **goto** L_0;

whence $\langle L_{-k}, n_1, n_2\rangle{\Downarrow_M}$ iff $n_k = 0$.

Outline proof of undecidability

Theorem

It is undecidable whether a given Minsky machine terminates from a given configuration.

Outline proof of undecidability

Theorem

It is undecidable whether a given Minsky machine terminates from a given configuration.

Idea: given a machine M and configuration C, we encode M, C as a formula $\mathcal{F}_{M,C}$ of BBI such that

$$M \text{ terminates from } C \Leftrightarrow \mathcal{F}_{M,C} \text{ is valid .}$$

Outline proof of undecidability

Theorem

It is undecidable whether a given Minsky machine terminates from a given configuration.

Idea: given a machine M and configuration C, we encode M, C as a formula $\mathcal{F}_{M,C}$ of BBI such that

$$M \text{ terminates from } C \Leftrightarrow \mathcal{F}_{M,C} \text{ is valid} .$$

Then, if we could decide validity of formulas in BBI, we could decide the halting problem for Minsky machines, contradiction!

Encoding configurations (1)

First, for each label L_i we have a propositional variable l_i.

Encoding configurations (1)

First, for each label L_i we have a propositional variable l_i.

We also pick two propositional variables p_1, p_2 to represent the counters c_1, c_2.

Encoding configurations (1)

First, for each label L_i we have a propositional variable l_i.

We also pick two propositional variables p_1, p_2 to represent the counters c_1, c_2.

Then, a configuration $\langle L_i, n_1, n_2 \rangle$ will be represented as:

$$l_i * p_1^{n_1} * p_2^{n_2}$$

where p_k^n denotes the formula $\overbrace{p_k * p_k * \cdots * p_k}^{n \text{ times}}$, with $p_k^0 = \mathrm{I}$.

Encoding configurations (2)

Now we pick a new propositional variable b and write

$$-A =_{\text{def}} A \mathrel{-\!\!*} b$$

Encoding configurations (2)

Now we pick a new propositional variable b and write

$$-A =_{\mathrm{def}} A \mathrel{-\!\!*} b$$

b will be interpreted as "all terminating configurations of the machine".

Encoding configurations (2)

Now we pick a new propositional variable b and write

$$-A =_{\text{def}} A \mathrel{-\!\!*} b$$

b will be interpreted as "all terminating configurations of the machine".

So $-A$ should be read as "whenever I add A to my current state, I get a terminating configuration".

Restricted $*$-contraction

Contraction does not hold for $*$:

$$A \nvdash A * A$$

Restricted $*$-contraction

Contraction does not hold for $*$:

$$A \nvdash A * A$$

However, a restricted form of contraction does hold:

$$\mathrm{I} \wedge A \vdash (\mathrm{I} \wedge A) * (\mathrm{I} \wedge A)$$

Easy to see semantically, but quite hard to derive!

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

$$L_i\colon c_k{+}{+};\, \textbf{goto } L_j; \qquad \Rightarrow \quad (-(l_j * p_k) \mathrel{-\!\!*} -l_i)$$

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

$$
\begin{array}{lcl}
L_i\colon c_k{+}{+};\ \textbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}(l_j * p_k) \mathrel{-\!\!*} \boldsymbol{-}l_i) \\
L_i\colon c_k{-}{-};\ \textbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}l_j \mathrel{-\!\!*} \boldsymbol{-}(l_i * p_k))
\end{array}
$$

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

$$
\begin{array}{lcl}
L_i\colon c_k{+}{+};\, \textbf{goto } L_j; & \Rightarrow & (-(l_j * p_k) \mathrel{-\!\!*} -l_i) \\
L_i\colon c_k{-}{-};\, \textbf{goto } L_j; & \Rightarrow & (-l_j \mathrel{-\!\!*} -(l_i * p_k)) \\
L_i\colon \textbf{if } c_k = 0 \textbf{ goto } L_j; & \Rightarrow & (-(l_j \vee l_{-k}) \mathrel{-\!\!*} -l_i)
\end{array}
$$

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

$$
\begin{array}{lcl}
L_i\colon c_k{+}{+};\,\mathbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}(l_j * p_k) \mathrel{-\!\!*} \boldsymbol{-} l_i) \\
L_i\colon c_k{-}{-};\,\mathbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-} l_j \mathrel{-\!\!*} \boldsymbol{-}(l_i * p_k)) \\
L_i\colon \mathbf{if}\ c_k = 0\ \mathbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}(l_j \vee l_{-k}) \mathrel{-\!\!*} \boldsymbol{-} l_i) \\
L_i\colon \mathbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-} l_j \mathrel{-\!\!*} \boldsymbol{-} l_i)
\end{array}
$$

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

$$
\begin{array}{lcl}
L_i\colon c_k{+}{+};\,\mathbf{goto}\ L_j; & \Rightarrow & (-(l_j * p_k) \mathrel{-\!\!*} -l_i) \\
L_i\colon c_k{-}{-};\,\mathbf{goto}\ L_j; & \Rightarrow & (-l_j \mathrel{-\!\!*} -(l_i * p_k)) \\
L_i\colon \mathbf{if}\ c_k = 0\ \mathbf{goto}\ L_j; & \Rightarrow & (-(l_j \vee l_{-k}) \mathrel{-\!\!*} -l_i) \\
L_i\colon \mathbf{goto}\ L_j; & \Rightarrow & (-l_j \mathrel{-\!\!*} -l_i)
\end{array}
$$

We code a whole machine $M = \{\gamma_1, \ldots, \gamma_t\}$ as:

$$\kappa(M) = \mathrm{I} \wedge \bigwedge_{i=1}^{t} \kappa(\gamma_i)$$

Encoding machines in BBI

We code each instruction γ of a machine M as a formula $\kappa(\gamma)$ of BBI:

$$
\begin{array}{lcl}
L_i\colon c_k{+}{+};\,\textbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}(l_j * p_k) \mathrel{-\!\!*} \boldsymbol{-}l_i) \\
L_i\colon c_k{-}{-};\,\textbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}l_j \mathrel{-\!\!*} \boldsymbol{-}(l_i * p_k)) \\
L_i\colon \textbf{if}\ c_k{=}0\ \textbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}(l_j \vee l_{-k}) \mathrel{-\!\!*} \boldsymbol{-}l_i) \\
L_i\colon \textbf{goto}\ L_j; & \Rightarrow & (\boldsymbol{-}l_j \mathrel{-\!\!*} \boldsymbol{-}l_i)
\end{array}
$$

We code a whole machine $M = \{\gamma_1, \ldots, \gamma_t\}$ as:

$$
\kappa(M) = \mathrm{I} \wedge \bigwedge_{i=1}^{t} \kappa(\gamma_i)
$$

Finally, we code termination from $\langle L_0, 0, 0\rangle$ as $(I \wedge \boldsymbol{-}l_0)$.

Master encoding

Putting everything together, the formula $\mathcal{F}_{M,C}$ encoding termination of M from C will be

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge - l_0) \vdash b$$

Master encoding

Putting everything together, the formula $\mathcal{F}_{M,C}$ encoding termination of M from C will be

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge - l_0) \vdash b$$

Plan of proof:

M terminates from C

Master encoding

Putting everything together, the formula $\mathcal{F}_{M,C}$ encoding termination of M from C will be

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge - l_0) \vdash b$$

Plan of proof:

$\quad\quad M$ terminates from C

$\Rightarrow \quad \mathcal{F}_{M,C}$ provable $\quad\quad$ (Theorem 1)

Master encoding

Putting everything together, the formula $\mathcal{F}_{M,C}$ encoding termination of M from C will be

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge -l_0) \vdash b$$

Plan of proof:

$\quad M$ terminates from C

$\Rightarrow \quad \mathcal{F}_{M,C}$ provable (Theorem 1)

$\Rightarrow \quad \mathcal{F}_{M,C}$ valid in all models (soundness)

Master encoding

Putting everything together, the formula $\mathcal{F}_{M,C}$ encoding termination of M from C will be

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \boldsymbol{-} l_0) \vdash b$$

Plan of proof:

M terminates from C

$\Rightarrow$ $\mathcal{F}_{M,C}$ provable (Theorem 1)

$\Rightarrow$ $\mathcal{F}_{M,C}$ valid in all models (soundness)

$\Rightarrow$ $\mathcal{F}_{M,C}$ valid in a specially chosen model and valuation

Master encoding

Putting everything together, the formula $\mathcal{F}_{M,C}$ encoding termination of M from C will be

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge -\!\!\!* \, l_0) \vdash b$$

Plan of proof:

M terminates from C

$\Rightarrow$ $\mathcal{F}_{M,C}$ provable (Theorem 1)

$\Rightarrow$ $\mathcal{F}_{M,C}$ valid in all models (soundness)

$\Rightarrow$ $\mathcal{F}_{M,C}$ valid in a specially chosen model and valuation

$\Rightarrow$ M terminates from C (Theorem 2)

First theorem

Theorem

Suppose $\langle L_i, n_1, n_2 \rangle {\Downarrow_M}$. *Then the following is derivable in* BBI:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge -l_0) \vdash b$$

First theorem

Theorem

Suppose $\langle L_i, n_1, n_2 \rangle \Downarrow_M$. Then the following is derivable in BBI*:*

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \boldsymbol{-} l_0) \vdash b$$

We actually derive the stronger

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} \vdash \boldsymbol{-}\boldsymbol{-} l_0$$

First theorem

Theorem

Suppose $\langle L_i, n_1, n_2 \rangle {\Downarrow_M}$. *Then the following is derivable in* BBI:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \boldsymbol{-}\, l_0) \vdash b$$

We actually derive the stronger

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} \vdash \boldsymbol{-}\boldsymbol{-}\, l_0$$

Proof is by induction on the length of the computation $\langle L_i, n_1, n_2 \rangle {\Downarrow_M}$. Restricted $*$-contraction is used to duplicate instructions from $\kappa(M)$ as needed.

Choosing a model

Given that $\mathcal{F}_{M,C}$ is provable, it is valid by soundness.

Choosing a model

Given that $\mathcal{F}_{M,C}$ is provable, it is valid by soundness.

It's enough to show that M terminates from C given only that $\mathcal{F}_{M,C}$ is valid in some model of our choice, under some valuation of our choice.

Choosing a model

Given that $\mathcal{F}_{M,C}$ is provable, it is valid by soundness.

It's enough to show that M terminates from C given only that $\mathcal{F}_{M,C}$ is valid in some model of our choice, under some valuation of our choice.

We use the RAM-domain model $\langle \mathcal{D}, \circ, \{e_0\} \rangle$, where:

- $\mathcal{D}$ is the set of all finite subsets of $\mathbb{N}$;

Choosing a model

Given that $\mathcal{F}_{M,C}$ is provable, it is valid by soundness.

It's enough to show that M terminates from C given only that $\mathcal{F}_{M,C}$ is valid in some model of our choice, under some valuation of our choice.

We use the RAM-domain model $\langle \mathcal{D}, \circ, \{e_0\} \rangle$, where:

- $\mathcal{D}$ is the set of all finite subsets of $\mathbb{N}$;
- $\circ$ is union of disjoint sets, undefined otherwise;

Choosing a model

Given that $\mathcal{F}_{M,C}$ is provable, it is valid by soundness.

It's enough to show that M terminates from C given only that $\mathcal{F}_{M,C}$ is valid in some model of our choice, under some valuation of our choice.

We use the RAM-domain model $\langle \mathcal{D}, \circ, \{e_0\} \rangle$, where:

- $\mathcal{D}$ is the set of all finite subsets of $\mathbb{N}$;
- $\circ$ is union of disjoint sets, undefined otherwise;
- e_0 is the empty set.

Second main theorem

Theorem

$\langle L_i, n_1, n_2 \rangle \Downarrow_M$ *whenever the following sequent is valid:*

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \vdash b$$

Second main theorem

Theorem

$\langle L_i, n_1, n_2 \rangle \Downarrow_M$ *whenever the following sequent is valid:*

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \vdash b$$

Proof outline. In our RAM-domain model $\langle \mathcal{D}, \circ, \{e_0\} \rangle$, we have for any ρ:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \models_\rho b$$

Second main theorem

Theorem
$\langle L_i, n_1, n_2\rangle\Downarrow_M$ *whenever the following sequent is valid:*

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge -l_0) \vdash b$$

Proof outline. In our RAM-domain model $\langle \mathcal{D}, \circ, \{e_0\}\rangle$, we have for any ρ:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge -l_0) \models_\rho b$$

We want to pick ρ with $e_0 \models_\rho \kappa(M)$ and $e_0 \models_\rho I \wedge -l_0$ to get:

$$l_i * p_1^{n_1} * p_2^{n_2} \models_\rho b$$

and infer $\langle L_i, n_1, n_2\rangle\Downarrow_M$.

$[\![p_k^n]\!]_\rho$: *The (second) edge of disaster*

We intend that $l_i * p_1^{n_1} * p_2^{n_2}$ should encode configuration $\langle L_i, n_1, n_2 \rangle$. Thus $d \models_\rho p_k^{n_k}$ should determine the number n_k.

$[\![p_k^n]\!]_\rho$: *The (second) edge of disaster*

We intend that $l_i * p_1^{n_1} * p_2^{n_2}$ should encode configuration $\langle L_i, n_1, n_2 \rangle$. Thus $d \models_\rho p_k^{n_k}$ should determine the number n_k.

But composition is disjoint so that, e.g., if we take $\rho(p_k) = \{h\}$ for a nonempty heap h, then $\rho(p_k^2) = \rho(p_k * p_k)$ is empty!

$[\![p_k^n]\!]_\rho$: *The (second) edge of disaster*

We intend that $l_i * p_1^{n_1} * p_2^{n_2}$ should encode configuration $\langle L_i, n_1, n_2 \rangle$. Thus $d \models_\rho p_k^{n_k}$ should determine the number n_k.

But composition is disjoint so that, e.g., if we take $\rho(p_k) = \{h\}$ for a nonempty heap h, then $\rho(p_k^2) = \rho(p_k * p_k)$ is empty!

In general, whenever $\rho(p_k)$ is finite we must have:

$$[\![p_k^n]\!]_\rho = [\![p_k^m]\!]_\rho$$

for sufficiently large n and m. So we need an infinite valuation.

Choosing a valuation

We choose a valuation ρ for $\langle \mathcal{D}, \circ, \{e_0\} \rangle$ as follows:

$$\begin{array}{rcl} \rho(p_1) & = & \{\{2^m\} \mid m \in \mathbb{N}\} \\ \rho(p_2) & = & \{\{3^m\} \mid m \in \mathbb{N}\} \end{array}$$

Choosing a valuation

We choose a valuation ρ for $\langle \mathcal{D}, \circ, \{e_0\} \rangle$ as follows:

$$\begin{array}{rcl} \rho(p_1) & = & \{\{2^m\} \mid m \in \mathbb{N}\} \\ \rho(p_2) & = & \{\{3^m\} \mid m \in \mathbb{N}\} \\ \rho(l_i) & = & \{\{\delta_i^m\} \mid m \in \mathbb{N}\} \end{array}$$

where δ_i is a fresh prime number for each propositional variable $l_{-2}, l_{-1}, l_0, l_1, \ldots$

Choosing a valuation

We choose a valuation ρ for $\langle \mathcal{D}, \circ, \{e_0\} \rangle$ as follows:

$$\begin{array}{rcl} \rho(p_1) & = & \{\{2^m\} \mid m \in \mathbb{N}\} \\ \rho(p_2) & = & \{\{3^m\} \mid m \in \mathbb{N}\} \\ \rho(l_i) & = & \{\{\delta_i^m\} \mid m \in \mathbb{N}\} \end{array}$$

where δ_i is a fresh prime number for each propositional variable $l_{-2}, l_{-1}, l_0, l_1, \ldots$
Finally, we define:

$$\rho(b) = \bigcup_{\langle L_i, n_1, n_2 \rangle \Downarrow_M} \{d \mid d \models_\rho l_i * p_1^{n_1} * p_2^{n_2}\}$$

so $\rho(b)$ is the set of interpretations of all terminating configurations.

Needed lemma

Lemma
For our chosen model and valuation ρ,

$$e_0 \models_\rho \mathrm{I} \wedge \neg l_0 \ .$$

This is easy.

Needed lemma

Lemma

For our chosen model and valuation ρ,

$$e_0 \models_\rho \mathrm{I} \wedge \neg l_0 \ .$$

This is easy.

Lemma

$$e_0 \models_\rho \kappa(M).$$

Needed lemma

Lemma

For our chosen model and valuation ρ,

$$e_0 \models_\rho \mathrm{I} \wedge - l_0 \ .$$

This is easy.

Lemma

$$e_0 \models_\rho \kappa(M).$$

We have to show $e_0 \models_\rho \kappa(\gamma)$ for each possible instruction γ.

Needed lemma

Lemma

For our chosen model and valuation ρ,

$$e_0 \models_\rho \mathrm{I} \wedge - l_0 \ .$$

This is easy.

Lemma

$$e_0 \models_\rho \kappa(M).$$

We have to show $e_0 \models_\rho \kappa(\gamma)$ for each possible instruction γ.

This involves wrangling with the semantics of $-\!\!*$ and with the details of our valuation.

Proof of Lemma 2

If $\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \vdash b$ is valid in $\langle \mathcal{D}, \circ, \{e_0\} \rangle$ then:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \models_\rho b$$

Proof of Lemma 2

If $\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \vdash b$ is valid in $\langle \mathcal{D}, \circ, \{e_0\} \rangle$ then:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \models_\rho b$$

Since $e_0 \models_\rho \kappa(M)$ we get:

$$l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \models_\rho b$$

Proof of Lemma 2

If $\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \vdash b$ is valid in $\langle \mathcal{D}, \circ, \{e_0\} \rangle$ then:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \models_\rho b$$

Since $e_0 \models_\rho \kappa(M)$ we get:

$$l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge - l_0) \models_\rho b$$

Since $e_0 \models_\rho \mathrm{I} \wedge - l_0$ (because $\langle L_0, 0, 0 \rangle \Downarrow_M$), we get:

$$l_i * p_1^{n_1} * p_2^{n_2} \models_\rho b$$

Proof of Lemma 2

If $\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge -l_0) \vdash b$ is valid in $\langle \mathcal{D}, \circ, \{e_0\}\rangle$ then:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge -l_0) \models_\rho b$$

Since $e_0 \models_\rho \kappa(M)$ we get:

$$l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge -l_0) \models_\rho b$$

Since $e_0 \models_\rho \mathrm{I} \wedge -l_0$ (because $\langle L_0, 0, 0\rangle \Downarrow_M$), we get:

$$l_i * p_1^{n_1} * p_2^{n_2} \models_\rho b$$

Since $d \models_\rho l_i * p_1^{n_1} * p_2^{n_2}$ uniquely determines n_1 and n_2 we conclude $\langle L_i, n_1, n_2\rangle \Downarrow_M$ from definition of $\rho(b)$.

Further reading

J. Brotherston and M. Kanovich.
Undecidability of propositional separation logic and its neighbours.
In *Journal of the ACM* 61(2). ACM, 2014.
Original version in *Proc. LICS-25*. IEEE, 2010.

D. Larchey-Wendling and D. Galmiche.
Nondeterministic phase semantics and the undecidability of Boolean BI.
In *ACM Trans. Comput. Logic* 14(1). ACM, 2013.
Original version in *Proc. LICS-25*. IEEE, 2010.

C. Calcagno, P. O'Hearn and H. Yang.
Computability and complexity results for a spatial assertion language for data structures.
In *Proceedings of FSTTCS*, 2001.