Undecidability of propositional separation logic and its neighbours

James Brotherston

Computer Science Seminar
Institute of Cybernetics, Tallinn University of Technology
17 Nov 2011
Outline

1. An overview of propositional separation logic
Outline

1. An overview of propositional separation logic
2. Undecidability of separation logic
Outline

1. An overview of propositional separation logic
2. Undecidability of separation logic
3. Decidable fragments: finite vs. infinite valuations
Outline

1. An overview of propositional separation logic
2. Undecidability of separation logic
3. Decidable fragments: finite vs. infinite valuations
4. Additional results
Outline

1. An overview of propositional separation logic
2. Undecidability of separation logic
3. Decidable fragments: finite vs. infinite valuations
4. Additional results

This is joint work with Prof. Max Kanovich, Queen Mary University of London. This talk is based on the paper of the same name (in Proc. LICS’10).
Part I

Propositional separation logic
Separation models

Separation logic is well established as a formalism for expressing and reasoning about properties of memory.
Separation logic is well established as a formalism for expressing and reasoning about properties of memory.

Definition
A separation model is a cancellative partial commutative monoid \( \langle H, \circ, E \rangle \).
Separation logic is well established as a formalism for expressing and reasoning about properties of memory.

**Definition**

A separation model is a cancellative partial commutative monoid \( \langle H, \circ, E \rangle \). We define:

\[
X \cdot Y = \text{def} \{ x \circ y \mid x \in X, y \in Y \}
\]

whence \( E \subseteq H \) is a set of units such that \( X \cdot E = X \).
Separation models

Separation logic is well established as a formalism for expressing and reasoning about properties of memory.

Definition
A separation model is a cancellative partial commutative monoid \( \langle H, \circ, E \rangle \). We define:

\[ X \cdot Y = \text{def} \{ x \circ y \mid x \in X, y \in Y \} \]

whence \( E \subseteq H \) is a set of units such that \( X \cdot E = X \).

Definition
\( \langle H, \circ, E \rangle \) has indivisible units if \( h_1 \circ h_2 \in E \) implies \( h_1, h_2 \in E \). (NB. All models of practical interest have indivisible units!)
Practical examples of separation models (I)

- Heap models $\langle H, \circ, \{e\} \rangle$, where $H = L \rightarrow_{\text{fin}} RV$ is the set of heaps ($L$ is infinite). $e$ is the function with empty domain, and:

$$h_1 \circ h_2 = \begin{cases} h_1 \cup h_2 & \text{if } \text{dom}(h_1), \text{dom}(h_2) \text{ disjoint} \\ \text{undefined} & \text{otherwise} \end{cases}$$
Practical examples of separation models (I)

• Heap models $\langle H, \circ, \{e\} \rangle$, where $H = L \rightsquigarrow_{\text{fin}} RV$ is the set of heaps ($L$ is infinite). $e$ is the function with empty domain, and:

$$h_1 \circ h_2 = \begin{cases} h_1 \cup h_2 & \text{if \text{dom}(h_1), \text{dom}(h_2) disjoint} \\ \text{undefined} & \text{otherwise} \end{cases}$$

• A basic example of the above: the RAM-domain model $\langle D, \circ, \{e_0\} \rangle$ where $D$ is the class of finite subsets of $\mathbb{N}$, the operation $\circ$ is the union of disjoint sets, and the unit $e_0$ is $\emptyset$. 
Practical examples of separation models (II)

- Heap-with-permissions models \( \langle H, \circ, E \rangle \), where 
  \[ H = L \rightarrow_{\text{fin}} (RV \times P) \]
  is a set of heaps with permissions. 
  \( h_1 \circ h_2 \) is defined as before, except that for heaps with the 
  same value at overlapping locations, we add the permissions.
Practical examples of separation models (II)

- **Heap-with-permissions models** \( \langle H, \circ, E \rangle \), where
  \( H = L \rightarrow_{\text{fin}} (RV \times P) \) is a set of *heaps with permissions*. 
  \( h_1 \circ h_2 \) is defined as before, except that for heaps with the same value at overlapping locations, we add the permissions.

- **Stack-and-heap models** \( \langle S \times H, \circ, E \rangle \), where \( H \) is a set of *heaps* or *heaps-with-permissions*, \( S = \text{Var} \rightarrow_{\text{fin}} \text{Val} \) is a set of *stacks*, and \( \langle s_1, h_1 \rangle \circ \langle s_2, h_2 \rangle \) is defined when \( s_1 = s_2 \) and \( h_1 \circ h_2 \) is defined (as above).
Semantics (I)

Formulas extend standard propositional connectives with the “multiplicatives” $I$, $*$ and $\neg *$. 
Formulas extend standard propositional connectives with the “multiplicatives” $\text I$, $\ast$ and $\neg\ast$.

A valuation for a separation model $\langle H, \circ, E \rangle$ is a function $\rho$ from propositional variables to $\mathcal P(H)$. 
Semantics (I)

Formulas extend standard propositional connectives with the “multiplicatives” $I$, $*$ and $\neg\neg$.

A valuation for a separation model $\langle H, \circ, E \rangle$ is a function $\rho$ from propositional variables to $\mathcal{P}(H)$.

Given $h \in H$ and formula $A$ we define the relation $h \models_{\rho} A$ by induction on $A$: 
Semantics (I)

Formulas extend standard propositional connectives with the “multiplicatives” $I$, $*$ and $\neg\neg$.

A valuation for a separation model $\langle H, \circ, E \rangle$ is a function $\rho$ from propositional variables to $\mathcal{P}(H)$.

Given $h \in H$ and formula $A$ we define the relation $h \models_\rho A$ by induction on $A$:

- $h \models_\rho P \iff h \in \rho(P)$
- $h \models_\rho F_1 \land F_2 \iff h \models_\rho F_1$ and $r \models_\rho F_2$
- $h \models_\rho I \iff h = e$
- $h \models_\rho F_1 \ast F_2 \iff h = h_1 \circ h_2$ and $h_1 \models_\rho F_1$ and $h_2 \models_\rho F_2$
- $h \models_\rho F_1 \rightarrow F_2 \iff \forall h'. h \circ h'$ defined and $h' \models_\rho F_1$ implies $h \circ h' \models_\rho F_2$
Semantics (I)

Formulas extend standard propositional connectives with the “multiplicatives” $I$, $*$ and $\pdownarrow$.

A valuation for a separation model $\langle H, \circ, E \rangle$ is a function $\rho$ from propositional variables to $\mathcal{P}(H)$.

Given $h \in H$ and formula $A$ we define the relation $h \models_\rho A$ by induction on $A$:

\[
\begin{align*}
    h \models_\rho P & \iff h \in \rho(P) \\
    h \models_\rho F_1 \land F_2 & \iff h \models_\rho F_1 \text{ and } r \models_\rho F_2 \\
    \quad \vdots & \\
    h \models_\rho I & \iff h = e \\
    h \models_\rho F_1 \ast F_2 & \iff h = h_1 \circ h_2 \text{ and } h_1 \models_\rho F_1 \text{ and } h_2 \models_\rho F_2 \\
    h \models_\rho F_1 \pdownarrow F_2 & \iff \forall h'. h \circ h' \text{ defined and } h' \models_\rho F_1 \text{ implies } h \circ h' \models_\rho F_2
\end{align*}
\]

We define $\llbracket A \rrbracket_\rho = \text{def } \{h \mid h \models_\rho A\}$.
Semantics (I)

Formulas extend standard propositional connectives with the “multiplicatives” \( I, \ast \) and \( \ast \).

A valuation for a separation model \( \langle H, \circ, E \rangle \) is a function \( \rho \) from propositional variables to \( \mathcal{P}(H) \).

Given \( h \in H \) and formula \( A \) we define the relation \( h \models_{\rho} A \) by induction on \( A \):

\[
\begin{align*}
  h \models_{\rho} P & \iff h \in \rho(P) \\
  h \models_{\rho} F_1 \land F_2 & \iff h \models_{\rho} F_1 \text{ and } r \models_{\rho} F_2 \\
  & \vdots \\
  h \models_{\rho} I & \iff h = e \\
  h \models_{\rho} F_1 \ast F_2 & \iff h = h_1 \circ h_2 \text{ and } h_1 \models_{\rho} F_1 \text{ and } h_2 \models_{\rho} F_2 \\
  h \models_{\rho} F_1 \ast F_2 & \iff \forall h'. h \circ h' \text{ defined and } h' \models_{\rho} F_1 \text{ implies } h \circ h' \models_{\rho} F_2
\end{align*}
\]

We define \( [A]_{\rho} = \text{def} \{ h \mid h \models_{\rho} A \} \).

A “sequent” \( A \vdash B \) is valid in \( \langle H, \circ, E \rangle \) if \( [A]_{\rho} \subseteq [B]_{\rho} \) for all \( \rho \).
Semantics (II)

In any separation model \( \langle H, \circ, E \rangle \) we have:

\[
\begin{align*}
[I]_{\rho} &= E \\
[A \ast B]_{\rho} &= [A]_{\rho} \cdot [B]_{\rho} \\
[A \rightarrow B]_{\rho} &= \text{largest } Z \subseteq H. \ Z \cdot [A]_{\rho} \subseteq [B]_{\rho}
\end{align*}
\]
Semantics (II)

In any separation model $⟨H, \circ, E⟩$ we have:

\[
\begin{align*}
[I]_\rho &= E \\
[A \ast B]_\rho &= [A]_\rho \cdot [B]_\rho \\
[A \ast \ast B]_\rho &= \text{largest } Z \subseteq H. \ Z \cdot [A]_\rho \subseteq [B]_\rho
\end{align*}
\]

In particular this implies restricted $\ast$-contraction:

\[
[I \wedge A]_\rho = [I \wedge A]_\rho \cdot [I \wedge A]_\rho = [(I \wedge A) \ast (I \wedge A)]_\rho
\]
Semantics (II)

In any separation model \( \langle H, \circ, E \rangle \) we have:

\[
[I]_\rho = E \\
[A \ast B]_\rho = [A]_\rho \cdot [B]_\rho \\
[A \rightarrow B]_\rho = \text{largest } Z \subseteq H. Z \cdot [A]_\rho \subseteq [B]_\rho
\]

In particular this implies restricted \(*\)-contraction:

\[
[I \land A]_\rho = [I \land A]_\rho \cdot [I \land A]_\rho = [(I \land A) \ast (I \land A)]_\rho
\]

which doesn’t hold in linear logic because, e.g.:

\[
[A \ast B]_\rho = \text{Cl}([A]_\rho \cdot [B]_\rho)
\]

where Cl is a closure operator. This is less precise, and rules out finite valuations since, e.g., Cl(\(\emptyset\)) is infinite.
Possible axiomatisations of separation logic

- \textbf{BI}, obtained by extending intuitionistic logic with the standard \textbf{MILL} axioms and rules for I, $\ast$ and $\neg\ast$;
Possible axiomatisations of separation logic

- **BI**, obtained by extending *intuitionistic* logic with the standard **MILL** axioms and rules for I, *, and ¬*;
- **BBI**, obtained by extending *classical* logic with the standard **MILL** axioms and rules for I, *, and ¬*;
Possible axiomatisations of separation logic

- **BI**, obtained by extending *intuitionistic* logic with the standard **MILL** axioms and rules for I, *, and \( \neg * \);
- **BBI**, obtained by extending *classical* logic with the standard **MILL** axioms and rules for I, *, and \( \neg * \);
- a *minimal BBI* with additives restricted to \( \land \) and \( \rightarrow \), i.e. no negation and no falsum (see next slide);
Possible axiomatisations of separation logic

- **BI**, obtained by extending intuitionistic logic with the standard MILL axioms and rules for I, * and —*;
- **BBI**, obtained by extending classical logic with the standard MILL axioms and rules for I, * and —*;
- a *minimal BBI* with additives restricted to ∧ and →, i.e. no negation and no falsum (see next slide);
- **BBI+eW** where eW is the restricted *-*weakening: 
  \[ I \land (A \ast B) \vdash I \land A, \]  
  which holds in all models with indivisible units. Because of restricted *-*contraction we have 
  \[ I \land (A \ast B) \equiv I \land A \land B; \]
Possible axiomatisations of separation logic

- **BI**, obtained by extending intuitionistic logic with the standard MILL axioms and rules for I, * and -*;
- **BBI**, obtained by extending classical logic with the standard MILL axioms and rules for I, * and -*;
- a minimal BBI with additives restricted to ∧ and →, i.e. no negation and no falsum (see next slide);
- **BBI+eW** where eW is the restricted *-weakening: I ∧ (A * B) ⊢ I ∧ A, which holds in all models with indivisible units. Because of restricted *-contraction we have I ∧ (A * B) ≡ I ∧ A ∧ B;
- **BBI+W** where W is the full *-weakening: A * B ⊢ A. This system collapses into classical logic!
**Minimal BBI**

\[(A \ast B) \vdash (B \ast A)\]
\[(A \ast (B \ast C)) \vdash ((A \ast B) \ast C)\]
\[(A \ast (A \rightarrow B)) \vdash B\]

\[
\begin{align*}
A & \vdash B \\
\hline
(A \ast C) & \vdash (B \ast C)
\end{align*}
\]

\[
\begin{align*}
(A \ast B) & \vdash C \\
\hline
A & \vdash (B \ast C)
\end{align*}
\]

(a) Axioms and rules for \(*\), \(-\ast\) and I.
**Minimal BBI**

\[(A \ast B) \vdash (B \ast A)\]  \hspace{2cm} \[(A \ast I) \vdash A\]

\[(A \ast (B \ast C)) \vdash ((A \ast B) \ast C)\]  \hspace{2cm} \[A \vdash (A \ast I)\]

\[(A \ast (A \rightarrow B)) \vdash B\]

\[
\frac{A \vdash B}{(A \ast C) \vdash (B \ast C)} \frac{(A \ast B) \vdash C}{A \vdash (B \rightarrow C)}
\]

(a) Axioms and rules for \(\ast\), \(\rightarrow\) and \(I\).

\[A \vdash (B \rightarrow A)\]  \hspace{2cm} \[A \vdash (B \rightarrow (A \land B))\]

\[\left(A \rightarrow (B \rightarrow C)\right) \vdash \left((A \rightarrow B) \rightarrow (A \rightarrow C)\right)\]  \hspace{2cm} \[(A \land B) \vdash A\]

\[\left((A \rightarrow B) \rightarrow A\right) \vdash A \hspace{2cm} (Peirce’s law)\]  \hspace{2cm} \[(A \land B) \vdash B\]

\[
\frac{A \hspace{1cm} A \vdash B}{B} \hspace{2cm} \frac{(A \land B) \vdash C}{A \vdash (B \rightarrow C)}
\]

(b) Axioms and rules for \(\rightarrow\) and \(\land\).
Part II

Undecidability
Outline proof of undecidability

\[ M \text{ terminates from } C \]
Outline proof of undecidability

$M$ terminates from $C$ (Thm 1)

$\mathcal{F}_{M,C}$ provable in minimal BBI
Outline proof of undecidability

\[ M \text{ terminates from } C \] (Thm 1)

\[ \mathcal{F}_{M,C} \text{ provable in minimal BBI} \]

\[ \mathcal{F}_{M,C} \text{ provable in BBI} \]

\[ \mathcal{F}_{M,C} \text{ provable in BBI+eW} \]
Outline proof of undecidability

$M$ terminates from $C$ (Thm 1)

$\mathcal{F}_{M,C}$ provable in minimal BBI

$\mathcal{F}_{M,C}$ valid in any separation model with indivisible units

$\mathcal{F}_{M,C}$ valid in any separation model

$\mathcal{F}_{M,C}$ provable in BBI

$\mathcal{F}_{M,C}$ provable in BBI$+eW$
Outline proof of undecidability

\[ M \text{ terminates from } C \]  
(Thm 1)

\[ \mathcal{F}_{M,C} \text{ valid in some concrete heap model} \]

\[ \mathcal{F}_{M,C} \text{ valid in any separation model with indivisible units} \]

\[ \mathcal{F}_{M,C} \text{ valid in any separation model} \]

\[ \mathcal{F}_{M,C} \text{ provable in minimal BBI} \]

\[ \mathcal{F}_{M,C} \text{ provable in BBI} \]

\[ \mathcal{F}_{M,C} \text{ provable in BBI+eW} \]
Outline proof of undecidability

- $M$ terminates from $C$ (Thm 1)
- $\mathcal{F}_{M,C}$ provable in minimal BBI
  - $\mathcal{F}_{M,C}$ valid in any separation model with indivisible units
  - $\mathcal{F}_{M,C}$ valid in any separation model
  - $\mathcal{F}_{M,C}$ provable in BBI

- $\mathcal{F}_{M,C}$ provable in BBI+eW
Outline proof of undecidability

$M$ terminates from $C$

(Thm 2)

$\mathcal{F}_{M,C}$ valid in some concrete heap model

$\mathcal{F}_{M,C}$ valid in any separation model with indivisible units

$\mathcal{F}_{M,C}$ valid in any separation model

(Thm 1)

$\mathcal{F}_{M,C}$ provable in minimal BBI

$\mathcal{F}_{M,C}$ provable in BBI

$\mathcal{F}_{M,C}$ provable in BBI+eW

All problems above are undecidable. Undecidability of BBI also established by Larchey-Wendling and Galmiche 2010.
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 $k \in \{1, 2\}$:

- $L_i: c_k++; \text{goto } L_j;$ “increment $c_k$ (and jump)"
- $L_i: c_k--; \text{goto } L_j;$ “decrement $c_k$ (and jump)"
- $L_i: \text{if } c_k=0 \text{ goto } L_j;$ “zero-test $c_k$ (and jump)"
- $L_i: \text{goto } L_j;$ “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 $k \in \{1, 2\}$:

- $L_i: c_k++; \text{goto } L_j$; "increment $c_k$ (and jump)"
- $L_i: c_k--; \text{goto } L_j$; "decrement $c_k$ (and jump)"
- $L_i: \text{if } c_k=0 \text{ goto } L_j$; "zero-test $c_k$ (and jump)"
- $L_i: \text{goto } L_j$; "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 $k \in \{1, 2\}$:

- $L_i: c_k++; \text{ goto } L_j$; “increment $c_k$ (and jump)”
- $L_i: c_k--; \text{ goto } L_j$; “decrement $c_k$ (and jump)”
- $L_i: \text{ if } c_k = 0 \text{ goto } L_j$; “zero-test $c_k$ (and jump)”
- $L_i: \text{ goto } L_j$; “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 $\langle L_i, n_1, n_2 \rangle \rightsquigarrow^*_M \langle L_0, 0, 0 \rangle$.

We introduce special labels $L_{-1}, L_{-2}$ with instructions:

- $L_{-1}: c_2--; \text{ goto } L_{-1}$; $L_{-1}: \text{ goto } L_0$;
- $L_{-2}: c_1--; \text{ goto } L_{-2}$; $L_{-2}: \text{ goto } L_0$;

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

13/27
For each label $L_i$ we have a propositional variable $l_i$. We also pick two propositional variables $p_1, p_2$ to represent counters $c_1, c_2$. 
For each label $L_i$ we have a propositional variable $l_i$. We also pick two propositional variables $p_1, p_2$ to represent counters $c_1, c_2$. 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 $p_k * p_k * \cdots * p_k$, with $p_k^0 = I$. 


Encoding configurations in minimal BBI

For each label $L_i$ we have a propositional variable $l_i$. We also pick two propositional variables $p_1, p_2$ to represent counters $c_1, c_2$. A configuration $\langle L_i, n_1, n_2 \rangle$ will be represented as:

$$l_i * p_k^{n_1} * p_k^{n_2}$$

where $p_k^n$ denotes the formula $p_k * p_k * \cdots * p_k$, with $p_k^0 = I$. Also pick propositional variable $b$ and write

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

$b$ will be interpreted as “all terminating configurations”. $\rightarrow*$ corresponds to replacement of parts of configurations.
Encoding machines in minimal BBI

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

- $L_i$: $c_k \leftarrow c_k + 1$; $\text{goto } L_j$; $\Rightarrow (-(l_j \star p_k) \star -l_i)$
- $L_i$: $c_k \leftarrow c_k - 1$; $\text{goto } L_j$; $\Rightarrow (-l_j \star -(l_i \star p_k))$
- $L_i$: if $c_k = 0$ $\text{goto } L_j$; $\Rightarrow (-(l_j \lor l_{-k}) \star -l_i)$
- $L_i$: $\text{goto } L_j$; $\Rightarrow (-l_j \star -l_i)$
Encoding machines in minimal BBI

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

\[
L_i: c_k \texttt{++}; \texttt{goto } L_j; \quad \Rightarrow \quad -(l_j \ast p_k) \ast -l_i
\]

\[
L_i: c_k \texttt{--}; \texttt{goto } L_j; \quad \Rightarrow \quad -(l_j \ast -(l_i \ast p_k))
\]

\[
L_i: \texttt{if } c_k = 0 \texttt{ goto } L_j; \quad \Rightarrow \quad -(l_j \lor l_{-k}) \ast -l_i
\]

\[
L_i: \texttt{goto } L_j; \quad \Rightarrow \quad -(l_j \ast -l_i)
\]

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

\[
\kappa(M) = I \land \bigwedge_{i=1}^{t} \kappa(\gamma_i)
\]

We’ll use restricted $\ast$-contraction to duplicate instructions as needed!
First main theorem

Theorem

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

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

Theorem

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

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

Proof relies heavily on “quasi-negation” properties of $-$ (e.g. $-A \equiv ---A$) and the restricted $*$-contraction:

$$I \land A \vdash (I \land A) * (I \land A)$$

which is derivable in minimal BBI.
Second main theorem

Theorem

\[ \langle L_i, n_1, n_2 \rangle \downarrow_M \text{ whenever the following sequent is valid in some } \textbf{concrete heap-like model used in practice (recall examples)}: \]

\[ \kappa(M) \ast l_i \ast p_1^{n_1} \ast p_2^{n_2} \ast (I \land \lnot l_0) \vdash b \]
Second main theorem

Theorem
\(\langle L_i, n_1, n_2\rangle \downarrow_M\) whenever the following sequent is valid in some concrete heap-like model used in practice (recall examples):

\[ \kappa(M) * l_i * p_{n_1} * p_{n_2} * (I \land \neg l_0) \vdash b \]

Proof outline. Consider for simplicity the RAM-domain model \(\langle D, \circ, \{e_0\}\rangle\) based on subsets of \(\mathbb{N}\). We have for any \(\rho\):

\[ \llbracket \kappa(M) * l_i * p_{n_1} * p_{n_2} * (I \land \neg l_0) \rrbracket \rho \subseteq \llbracket b \rrbracket \rho \]
Second main theorem

Theorem

\[ \langle L_i, n_1, n_2 \rangle \downarrow_M \text{ whenever the following sequent is valid in some concrete heap-like model used in practice (recall examples)}: \]

\[ \kappa(M) \ast l_i \ast p_1^{n_1} \ast p_2^{n_2} \ast (I \land \neg l_0) \vdash b \]

Proof outline. Consider for simplicity the RAM-domain model \( \langle \mathcal{D}, \circ, \{e_0\} \rangle \) based on subsets of \( \mathbb{N} \). We have for any \( \rho \):

\[ \left[ \kappa(M) \ast l_i \ast p_1^{n_1} \ast p_2^{n_2} \ast (I \land \neg l_0) \right] \rho \subseteq \llbracket b \rrbracket \rho \]

We want to pick \( \rho \) with \( e_0 \in \left[ \kappa(M) \right] \rho \) and \( e_0 \in \left[ I \land \neg l_0 \right] \rho \) to get:

\[ \left[ l_i \ast p_1^{n_1} \ast p_2^{n_2} \right] \rho \subseteq \llbracket b \rrbracket \rho \]

and infer \( \langle L_i, n_1, n_2 \rangle \downarrow_M \).
$e_0 \in \llbracket \kappa(M) \rrbracket_\rho$: The edge of disaster

To check $e_0 \in \llbracket \kappa(M) \rrbracket_\rho$ we check $e_0 \in \llbracket \kappa(\gamma) \rrbracket_\rho$ for each instruction $\gamma$. 
$e_0 \in [\kappa(M)]_\rho$: The edge of disaster

To check $e_0 \in [\kappa(M)]_\rho$ we check $e_0 \in [\kappa(\gamma)]_\rho$ for each instruction $\gamma$.

Why do we encode, e.g., $L_i: c_k++; \textbf{goto} L_j$; as

$$-(l_j \ast p_k) \ast -l_i)$$

and not $l_i \ast (l_j \ast p_k)$?
$e_0 \in \lceil \kappa(M) \rceil_\rho$: The edge of disaster

To check $e_0 \in \lceil \kappa(M) \rceil_\rho$ we check $e_0 \in \lceil \kappa(\gamma) \rceil_\rho$ for each instruction $\gamma$.

Why do we encode, e.g., $L_i: c_k++; \textbf{goto } L_j;$ as

$$(-(l_j * p_k) \multimap -l_i)$$

and not $l_i \multimap (l_j * p_k)$?

Let’s try to check: $e_0 \in \lceil l_i \multimap (l_j * p_k) \rceil_\rho$, i.e. $\lceil l_i \rceil_\rho \subseteq \lceil l_j * p_k \rceil_\rho$. 
$e_0 \in \llbracket \kappa(M) \rrbracket_\rho$: The edge of disaster

To check $e_0 \in \llbracket \kappa(M) \rrbracket_\rho$ we check $e_0 \in \llbracket \kappa(\gamma) \rrbracket_\rho$ for each instruction $\gamma$.

Why do we encode, e.g., $L_i: c_k++; \textbf{goto } L_j$; as

$$(−(l_j * p_k) → −l_i)$$

and not $l_i → (l_j * p_k)$?

Let’s try to check: $e_0 \in \llbracket l_i → (l_j * p_k) \rrbracket_\rho$, i.e. $\llbracket l_i \rrbracket_\rho \subseteq \llbracket l_j * p_k \rrbracket_\rho$.

But suppose $L_i = L_j$. In separation models this means:

$$\llbracket l_i \rrbracket_\rho \subseteq \llbracket l_i \rrbracket_\rho \cdot \llbracket p_k \rrbracket_\rho \subseteq \llbracket l_i \rrbracket_\rho \cdot \llbracket p_k \rrbracket_\rho \cdot \llbracket p_k \rrbracket_\rho \subseteq \cdots$$

i.e., any heap can be split into arbitrarily many pieces!

(Not a problem in linear logic.)
\([p_k^n]_\rho\): The (second) edge of disaster

We intend that \([l_i * p_1^{n_1} * p_2^{n_2}]_\rho\) should encode configuration \(\langle L_i, n_1, n_2 \rangle\). Thus \([p_k^{n_k}]_\rho\) should determine the number \(n_k\).
\[ [p_k^n]_\rho: \text{The (second) edge of disaster} \]

We intend that \([l_i * p_1^{n_1} * p_2^{n_2}]_\rho\) should encode configuration \(\langle L_i, n_1, n_2 \rangle\). Thus \([p_k^{n_k}]_\rho\) should determine the number \(n_k\).

But composition of heaps 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!
\[ [p^n_k]_\rho: \text{The (second) edge of disaster} \]

We intend that \( [l_i \ast p^{n_1}_1 \ast p^{n_2}_2]_\rho \) should encode configuration \( \langle L_i, n_1, n_2 \rangle \). Thus \( [p^{n_k}_k]_\rho \) should determine the number \( n_k \).

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

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

\[ [p^n_k]_\rho = [p^m_k]_\rho \]

for sufficiently large \( n \) and \( m \), which obstructs us in uniquely representing the number \( n_k \) by the formula \( p^n_k \).

(We discuss decidability consequences shortly.)
Choosing a valuation

We choose a valuation \( \rho \) for \( \langle \mathcal{D}, \circ, \{e_0\} \rangle \) as follows:

\[
\begin{align*}
\rho(p_1) &= \{2^m \mid m \in \mathbb{N}\} \\
\rho(p_2) &= \{3^m \mid m \in \mathbb{N}\} \\
\rho(l_i) &= \{\delta_i^m \mid m \in \mathbb{N}\}
\end{align*}
\]

where \( \delta_i \) 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 \(\rho\) for \(\langle D, \circ, \{e_0\} \rangle\) as follows:

\[
\rho(p_1) = \{\{2^m\} \mid m \in \mathbb{N}\} \\
\rho(p_2) = \{\{3^m\} \mid m \in \mathbb{N}\} \\
\rho(l_i) = \{\{\delta^m_i\} \mid m \in \mathbb{N}\}
\]

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

Finally, we define:

\[
\rho(b) = \bigcup \langle L_i, n_1, n_2 \rangle \downarrow_M [l_i \ast p_1^{n_1} \ast p_2^{n_2}]_\rho
\]

so \(\rho(b)\) is the set of interpretations of all terminating configurations.
Proof of Theorem 2

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

$$\llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket b \rrbracket_\rho$$
Proof of Theorem 2

If $\kappa(M) \ast l_i \ast p^{n_1}_1 \ast p^{n_2}_2 \ast (I \land \neg l_0) \vdash b$ is valid in $\langle D, \circ, \{e_0\} \rangle$ then:

$$\llbracket \kappa(M) \ast l_i \ast p^{n_1}_1 \ast p^{n_2}_2 \ast (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket b \rrbracket_\rho$$

Since $e_0 \in \llbracket \kappa(M) \rrbracket_\rho$ we get:

$$\llbracket l_i \ast p^{n_1}_1 \ast p^{n_2}_2 \ast (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket \neg \neg l_0 \rrbracket_\rho$$
Proof of Theorem 2

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

$$\llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket b \rrbracket_\rho$$

Since $e_0 \in \llbracket \kappa(M) \rrbracket_\rho$ we get:

$$\llbracket l_i * p_1^{n_1} * p_2^{n_2} * (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket \neg \neg l_0 \rrbracket_\rho$$

Since $e_0 \in \llbracket I \land \neg l_0 \rrbracket_\rho$ (because $\langle L_0, 0, 0 \rangle \downarrow_M$), we get:

$$\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_\rho \subseteq \llbracket b \rrbracket_\rho$$
If \( \kappa(M) \ast l_i \ast p_1^{n_1} \ast p_2^{n_2} \ast (I \land \neg l_0) \vdash b \) is valid in \( \langle D, \circ, \{e_0\} \rangle \) then:
\[
\llbracket \kappa(M) \ast l_i \ast p_1^{n_1} \ast p_2^{n_2} \ast (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket b \rrbracket_\rho
\]
Since \( e_0 \in \llbracket \kappa(M) \rrbracket_\rho \) we get:
\[
\llbracket l_i \ast p_1^{n_1} \ast p_2^{n_2} \ast (I \land \neg l_0) \rrbracket_\rho \subseteq \llbracket \neg \neg l_0 \rrbracket_\rho
\]
Since \( e_0 \in \llbracket I \land \neg l_0 \rrbracket_\rho \) (because \( \langle L_0, 0, 0 \rangle \Downarrow_M \)), we get:
\[
\llbracket l_i \ast p_1^{n_1} \ast p_2^{n_2} \rrbracket_\rho \subseteq \llbracket b \rrbracket_\rho
\]
Since \( \llbracket l_i \ast p_1^{n_1} \ast p_2^{n_2} \rrbracket_\rho \) uniquely determines \( n_1 \) and \( n_2 \) we conclude \( \langle L_i, n_1, n_2 \rangle \Downarrow_M \) from definition of \( \rho(b) \).
Part III

Decidability: finite vs. infinite valuations
Finite valuations

The quantifier-free fragment of a certain separation theory over an infinite heap model is **decidable** (Calcagno et al., 2001). WTF?
Finite valuations

The quantifier-free fragment of a certain separation theory over an infinite heap model is \textit{decidable} (Calcagno et al., 2001). WTF?

There, valuations are constrained to be \textit{finite}, whereas our valuation $\rho$ is necessarily \textit{infinite}.
**Finite valuations**

The quantifier-free fragment of a certain separation theory over an infinite heap model is **decidable** (Calcagno et al., 2001).

WTF?

There, valuations are constrained to be finite, whereas our valuation \( \rho \) is necessarily **infinite**.

**Theorem**

There is a sequent of the form \( \kappa(M) \ast l_i \ast p_{1}^{n_1} \ast (I \land -l_0) \vdash b \) such that, for any choice of heap-like model \( \langle H, o, E \rangle \), the sequent is **invalid** in the model, but **valid** under all finite valuations \( \rho \).
Finite valuations

The quantifier-free fragment of a certain separation theory over an infinite heap model is decidable (Calcagno et al., 2001). WTF?

There, valuations are constrained to be finite, whereas our valuation $\rho$ is necessarily infinite.

**Theorem**

There is a sequent of the form $\kappa(M) \ast l_i \ast p_1^{n_1} \ast (I \land \neg l_0) \vdash b$ such that, for any choice of heap-like model $\langle H, \circ, E \rangle$, the sequent is invalid in the model, but valid under all finite valuations $\rho$.

So to obtain decidable fragments of separation logic, one should either give up infinite valuations (Calcagno et al., 2001), or restrict the formula language (Berdine et al., 2004).
Part IV

Additional results
A CBI-model is a separation model $\langle H, \circ, E \rangle$ enriched with a total involution $\cdot ^{-1}$ such that for all $h \in H. \; h \circ h^{-1} = e^{-1}$. (Cf. effect algebras in quantum mechanics.)
A **CBI**-model is a separation model $\langle H, \circ, E \rangle$ enriched with a total involution $\cdot^{-1}$ such that for all $h \in H$, $h \circ h^{-1} = e^{-1}$. (Cf. **effect algebras** in quantum mechanics.)

E.g., can take $\langle \mathcal{D}, \circ, \{e_0\}, \cdot^{-1} \rangle$ where $\mathcal{D}$ is now the class of **finite and cofinite** subsets of $\mathbb{N}$, $\circ$ is union of disjoint sets, $e_0 = \emptyset$ and $\cdot^{-1}$ is set complement.
A CBI-model is a separation model \( \langle H, \circ, E \rangle \) enriched with a total involution \( \cdot^{-1} \) such that for all \( h \in H \). \( h \circ h^{-1} = e^{-1} \). (Cf. effect algebras in quantum mechanics.)

E.g., can take \( \langle D, \circ, \{e_0\}, \cdot^{-1} \rangle \) where \( D \) is now the class of finite and cofinite subsets of \( \mathbb{N} \), \( \circ \) is union of disjoint sets, \( e_0 = \emptyset \) and \( \cdot^{-1} \) is set complement.

CBI extends BBI with a multiplicative negation \( \sim \) defined by:

\[
h \models_\rho \sim A \iff h^{-1} \not\models_\rho A
\]
Undecidability of CBI and related problems

Proof of Thm 2 now uses a slightly modified valuation $\rho$. All problems above are again undecidable.
Some references

J. Berdine, C. Calcagno and P. O’Hearn.
A decidable fragment of separation logic.

J. Brotherston and C. Calcagno.
Classical BI (a logic for reasoning about dualising resources).

Computability and complexity results for a spatial assertion language for data structures.

D. Larchey-Wendling and D. Galmiche.
Undecidability of Boolean BI through phase semantics.

J.C. Reynolds.
Separation logic: a logic for shared mutable data structures.