Undecidability of propositional separation logic and its neighbours

James Brotherston *
Dept. of Computing
Imperial College London, UK

Max Kanovich
School of Electronic Engineering and Computer Science,
Queen Mary University of London, UK

Abstract

Separation logic has proven an adequate formalism for the analysis of programs that manipulate memory (in the form of pointers, heaps, stacks, etc.). In this paper, we consider the purely propositional fragment of separation logic, as well as a number of closely related substructural logical systems. We show that, surprisingly, all of these propositional logics are undecidable. In particular, we solve an open problem by establishing the undecidability of Boolean BI.

1 Introduction and Motivations

Separation logic has become well-established in the last decade as an effective formalism for reasoning about memory-manipulating programs [23]. Automated shape analysis tools based upon separation logic are now capable of verifying properties of large industrial programs [24, 5] and have been adapted to a variety of paradigms such as object-oriented programming [20, 9, 8] and concurrent programming [10, 14].

Separation logic is usually based upon a first-order extension of the propositional *bunched logic* Boolean BI (BBI) [15]. Bunched logics, originating in the *logic of bunched implications* BI [19], are substructural logics that combine a standard propositional logic with a multiplicative linear logic, and admit a Kripke-style truth interpretation in which "worlds" are understood as *resources* [15, 4]. In the case of separation logic, this interpretation takes place with respect to a model of *heaps* equipped with a partial operator for composing heaps whose domains are disjoint. Thus, in addition to the standard propositional connectives which are read in the usual way, the most important feature of separation logic is its multiplicative *separating conjunction* $*$, which denotes disjoint heap composition: $A_1 * A_2$ denotes the set of heaps which can be split into two disjoint heaps satisfying respectively A_1 and A_2. The multiplicative conjunction $*$ comes along with a unit I, which denotes the empty heap, and an adjoint implication $-\!\!*$, which denotes a property of heap extension: $A_1 -\!\!* A_2$ denotes those heaps which satisfy A_2 when extended with any heap satisfying A_1.

The original BI, which combines the aforementioned multiplicative connectives $*$, I and $-\!\!*$ with the standard intuitionistic connectives $\rightarrow$, $\wedge$, etc., was shown *decidable* by Galmiche et al [13]. In turn, BBI combines the same multiplicatives with the standard Boolean connectives. Since the Boolean component of BBI appears much simpler than the intuitionistic component of BI, it was expected for a long time that BBI might be decidable as well.

In this paper, we show that in fact BBI is *undecidable*, as are a number of related propositional systems that arise naturally in developing towards axiomatisations of separation logic. (Table 1 gives an extremely simple propositional system which is undecidable.) In addition, we show that, for any choice of *concrete* separation model employing a heap memory concept (see Example 1.1 below), validity in that model is undecidable as well. We also examine the relationship between our undecidability results and existing decidable fragments of separation logic.

Example 1.1. Examples of commonly-used separation models which employ a heap memory concept (cf. [6]):

(a) *Heap models* $(H, \circ, \{e\})$, where $H = L \rightharpoonup_{\text{fin}} RV$ is the set of *heaps*, which are finite partial functions from L to RV. L is supposed to be an infinite set. The unit e is the function with empty domain, and $h_1 \circ h_2$ is the union of h_1 and h_2 when their domains are disjoint (and undefined otherwise). By choosing suitable L and RV we may obtain a model of *hierarchical storage* [1], a model of *heaps of records* [15] and the *RAM model* [23].

(b) *Heap-with-permission models* [3] given by $(H, \circ, \{e\})$ with an underlying *permission algebra* $(P, \bullet, \mathbb{1})$, i.e. a set P equipped with a partial commutative, associative and cancellative operation $\bullet$, and a distinguished element $\mathbb{1}$ such that $\mathbb{1} \bullet \pi$ is undefined for all $\pi \in P$. (An example of a permission algebra is the interval $(0, 1]$, with $\bullet$ being addition but undefined when permissions add up to more than 1.) Then $H = L \rightharpoonup_{\text{fin}} (RV \times P)$ is the set of *heaps-with-permissions*, and $h_1 \circ h_2$ is again the union of disjoint h_1 and h_2. However,

*Research supported by an EPSRC postdoctoral fellowship.

some overlap is allowed: if $h_1(\ell) = \langle v, \pi_1 \rangle$, $h_2(\ell) = \langle v, \pi_2 \rangle$ and $\pi_1 \bullet \pi_2$ is defined then h_1 and h_2 are *compatible at* ℓ. When h_1 and h_2 are compatible at all common ℓ, then $(h_1 \circ h_2)(\ell) = \langle v, \pi_1 \bullet \pi_2 \rangle$, rather than being undefined.

(c) *Stack-and-heap models* [21] given by $(S \times H, \circ, E)$, where H is a set of *heaps* or *heaps-with-permissions* as above and $S = \mathtt{Var} \rightharpoonup_{\text{fin}} \mathtt{Val}$ is the set of *stacks*, partially mapping variables $\mathtt{Var}$ to values $\mathtt{Val}$. Here E consists of all pairs $\langle s, e \rangle$ in which e is the empty heap, and $\langle s_1, h_1 \rangle \circ \langle s_2, h_2 \rangle = \langle s_1, h_1 \circ h_2 \rangle$ iff $s_1 = s_2$ and $h_1 \circ h_2$ is defined in accordance with the previous items, and is undefined otherwise.

A plan of the remainder of the paper is as follows. In Section 2, we give the semantics of propositional separation logic and a number of the closely related proof systems for bunched logic. In Section 3, we present a proof-theoretic encoding of terminating computations of two-counter Minsky machine such that, whenever machine M terminates from configuration C, the corresponding sequent $\mathcal{F}_{M,C}$ is provable even in a minimal version of BBI (in which negation and falsum are disallowed). By soundness, the sequent $\mathcal{F}_{M,C}$ is then valid in all separation models, including any from Example 1.1 above. Then, in Section 4, we show that whenever $\mathcal{F}_{M,C}$ is valid in the most complex stack-and-heap model in Example 1.1 (where the heaps are heaps-with-permissions), machine M terminates from configuration C. Thus it follows that *any* property between provability of $\mathcal{F}_{M,C}$ in minimal BBI and validity of $\mathcal{F}_{M,C}$ in a heap-like model is undecidable. We state our undecidability results in Section 5, and examine some consequences concerning finite approximations in Section 6. We extend our undecidability result to a class of "dualising" separation models and its axiomatisation, given by Classical BI [4], in Section 7. Section 8 concludes.

2 Semantics and syntax of separation logic

Here we present the propositional language of separation logic, its interpretation in separation models and a number of related bunched logic proof systems.

We abstract from the concrete separation logic models found in the literature by the following definition (cf. [6, 15]):

Definition 2.1. A *separation model* is a cancellative partial commutative monoid $(H, \circ, E)$. That is, $\circ$ is a partial binary operation on H which is associative and commutative, where equalities of expressions $\alpha = \beta$ are understood to mean that either both α and β are undefined, or α and β are defined and equal. Cancellativity of $\circ$ means that if $z \circ x$ is defined and $z \circ x = z \circ y$ then $x = y$. We define $X \cdot Y =_{\text{def}} \{x \circ y \mid x \in X, y \in Y\}$. $E \subseteq H$ is a set of *units* such that $E \cdot \{h\} = \{h\}$ for all $h \in H$.

Comment 2.1. *In the above, for any $e_1, e_2 \in E$ we necessarily have $e_1 \circ e_1 = e_1$, and $e_1 \circ e_2$ is defined iff $e_1 = e_2$. In particular, if $\circ$ is total then E is forced to be a singleton $\{e\}$.*

Comment 2.2. *Practical reasons for our Definition 2.1:*

(1) *Most interesting applications of separation logic are based on "heap-like" separation models in which composition $\circ$ is not total (cf. Example 1.1).*

(2) *Although these models typically employ a single unit element e, we can consider more complex* stacks-and-heaps *models (see Ex. 1.1(c)) by generalising to a set of units E.*

Definition 2.2. A separation model $(H, \circ, E)$ is said to have *indivisible units* if $h_1 \circ h_2 \in E$ implies $h_1 \in E$ and $h_2 \in E$ for all $h_1, h_2 \in H$. (We remark that all of the separation models $(H, \circ, E)$ in Example 1.1 have indivisible units.)

Definition 2.3. *Formulas* are given by the following grammar (where p ranges over propositional variables):

$$A \quad ::= \quad p \mid \top \mid \bot \mid A \wedge A \mid A \vee A \mid A \to A \mid \neg A \mid \mathrm{I} \mid A * A \mid A \mathrel{-\!\!*} A$$

For the sake of readability, we often write a formula of the form $(A \to B)$ as the 'sequent' $A \vdash B$.

Definition 2.4. A *valuation* for a separation model $(H, \circ, E)$ is a function ρ that assigns to each propositional variable p a set $\rho(p) \subseteq H$. Given any $h \in H$ and formula A, we define the forcing relation $h \models_\rho A$ by induction on A:

$$\begin{array}{rcl}
h \models_\rho p & \Leftrightarrow & h \in \rho(p) \\
h \models_\rho \top & \Leftrightarrow & \text{always} \\
h \models_\rho \bot & \Leftrightarrow & \text{never} \\
h \models_\rho A_1 \wedge A_2 & \Leftrightarrow & h \models_\rho A_1 \text{ and } h \models_\rho A_2 \\
h \models_\rho A_1 \vee A_2 & \Leftrightarrow & h \models_\rho A_1 \text{ or } h \models_\rho A_2 \\
h \models_\rho A_1 \to A_2 & \Leftrightarrow & \text{if } h \models_\rho A_1 \text{ then } h \models_\rho A_2 \\
h \models_\rho \neg A & \Leftrightarrow & h \not\models_\rho A \\
h \models_\rho \mathrm{I} & \Leftrightarrow & h \in E \\
h \models_\rho A_1 * A_2 & \Leftrightarrow & \exists h_1, h_2.\ h = h_1 \circ h_2 \text{ and } h_1 \models_\rho A_1 \\
 & & \text{and } h_2 \models_\rho A_2 \\
h \models_\rho A_1 \mathrel{-\!\!*} A_2 & \Leftrightarrow & \forall h'.\ \text{if } h \circ h' \text{ defined and } h' \models_\rho A_1 \\
 & & \text{then } h \circ h' \models_\rho A_2
\end{array}$$

The intended meaning of any formula A under ρ is given by $[\![A]\!]_\rho =_{\text{def}} \{h \mid h \models_\rho A\}$. In particular, we have:

$$\begin{array}{rcl}
[\![\mathrm{I}]\!]_\rho & = & E \\
[\![A \wedge B]\!]_\rho & = & [\![A]\!]_\rho \cap [\![B]\!]_\rho \\
[\![A * B]\!]_\rho & = & [\![A]\!]_\rho \cdot [\![B]\!]_\rho \\
[\![A \to B]\!]_\rho & = & \text{largest } Z \subseteq H.\ [\![A]\!]_\rho \cap Z \subseteq [\![B]\!]_\rho \\
[\![A \mathrel{-\!\!*} B]\!]_\rho & = & \text{largest } Z \subseteq H.\ [\![A]\!]_\rho \cdot Z \subseteq [\![B]\!]_\rho
\end{array} \tag{1}$$

Definition 2.5. A formula A is *valid* in a separation model $(H, \circ, E)$ if for any valuation ρ, we have $[\![A]\!]_\rho = H$. A sequent $A \vdash B$ is *valid* if $[\![A]\!]_\rho \subseteq [\![B]\!]_\rho$ for any valuation ρ.

By *propositional separation logic* we mean the set of formulas / sequents valid in the class of all separation models.

Comment 2.3. *Separation models with total and non-total compositions behave differently. E.g., the sequent* $(p \wedge ((p \mathrel{-\!\!*} (p \wedge \mathrm{I}))) \vdash \mathrm{I}$ *is valid in any separation model in which* $\circ$ *is total. But, let* $(H, \circ, E)$ *be a separation model in which* $h \circ h$ *is undefined for some* $h \in H$ *(as is typical in heap models), and a valuation* ρ *with* $\rho(p) = \{h\}$*. Then* $h \models_\rho (p \wedge ((p \mathrel{-\!\!*} (p \wedge \mathrm{I})))$ *while* $h \not\models_\rho \mathrm{I}$*, so this sequent is invalid in* $(H, \circ, E)$*.*

Core proof systems for propositional separation logic are provided by various *bunched logics*, a class of substructural logics pioneered by O'Hearn and Pym [19].

Definition 2.6. We consider a chain of logics as follows:

• The *logic of bunched implications*, BI (cf. [19, 22, 13]) is given by the following axioms and inference rules:

(a) all instances of intuitionistically valid propositional formulas and intuitionistic inference rules;

(b) the axioms and inference rules for $*$, $-\!\!*$ and I given in section (a) of Table 1.

• *Boolean* BI, or BBI (cf. [15, 12]) is obtained from BI by expanding (a) above to also include all instances of *classically* valid propositional formulas.

• Since the presence of classical negation $\neg$ increases the general complexity of BBI, we shall also consider a positive fragment of BBI, which we call *minimal* BBI, in which the formula connectives are restricted to $\wedge$, $\rightarrow$, I, $*$ and $-\!\!*$. Minimal BBI is given by the rules of Table 1.

• As we shall see, the *restricted* $*$*-contraction* principle $(\mathrm{I} \wedge A) \vdash (A * A)$ holds in (minimal) BBI, whereas the analogous *restricted* $*$*-weakening* principle $(\mathrm{I} \wedge (A * B)) \vdash A$ does not. Thus we consider the system BBI$+$eW obtained by enriching BBI with the latter principle.

• Having considered restricted $*$-weakening, it is also natural to consider BBI$+$W, obtained by enriching BBI with the *unrestricted* $*$*-weakening* principle $(A * B) \vdash A$.

Proposition 2.1 (Soundness)**.** *If* A *is provable in* BBI *then* A *is valid in all separation models. Furthermore, if* A *is provable in* BBI$+$eW *then* A *is valid in all separation models with indivisible units.*

Completeness of BBI with respect to the class of partial commutative monoids is still open. However, BBI is complete for the class of *relational* commutative monoids [12].

Corollary 2.1. BI $\subset$ BBI $\subset$ BBI$+$eW $\subset$ BBI$+$W*, where* $\subset$ *is interpreted as strict inclusion between the set of sequents provable in each logic.*

Proof. The nonstrict inclusions hold easily. The strict inclusions hold because because BI is conservative over intuitionistic logic [22], and (un)restricted $*$-weakening is not valid in all separation models (with indivisible units) and hence not BBI$(+$eW$)$-provable by Proposition 2.1. □

Comment 2.4. *Both ends of the chain of logics in Corollary 2.1 are in fact decidable.* BI *was shown decidable in* [13]*, and* BBI$+$W *is decidable because, as we shall see, it collapses into ordinary classical logic (see Prop. 2.3).*

Because of the observation in Comment 2.4 it is not obvious, *a priori*, whether or not the intermediate systems BBI and BBI$+$eW ought to be decidable.

Proposition 2.2. *The following forms of the deduction theorem hold for minimal* BBI*:*

(a) $(A \wedge B) \vdash C$ *is provable iff* $A \vdash (B \rightarrow C)$ *is provable;*

(b) $(A * B) \vdash C$ *is provable iff* $A \vdash (B \mathrel{-\!\!*} C)$ *is provable;*

(c) $B \vdash C$ *is provable iff* $\mathrm{I} \vdash (B \mathrel{-\!\!*} C)$ *is provable.*

Lemma 2.1. *The following are derivable in minimal* BBI*:*

$$\frac{A \vdash C \qquad B \vdash C}{A \vee B \vdash C} \qquad\qquad \frac{A \vdash C \qquad A \rightarrow B \vdash C}{\vdash C}$$

where $A \vee B$ *is an* abbreviation *for* $(B \rightarrow A) \rightarrow A$*.*

Proof. With the help of Peirce's law. □

One of the important features of separation logic is that the $*$*-contraction principle*, $A \vdash (A * A)$, is not generally valid, and hence not provable in BBI by Prop. 2.1. Surprisingly, however, BBI enjoys the *restricted* $*$*-contraction* which holds only at the multiplicative unit I. (From now on, we write $A \equiv B$ to mean that both $A \vdash B$ and $B \vdash A$ are provable.)

Lemma 2.2. *The following is provable in minimal* BBI*:*

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

Proof. We can easily derive $(\mathrm{I} \wedge A) * (\mathrm{I} \wedge A) \vdash (A * A)$, whence by parts (b) and (a) of Proposition 2.2 we obtain:

$$A \vdash \mathrm{I} \rightarrow ((\mathrm{I} \wedge A) \mathrel{-\!\!*} (A * A)) \tag{2}$$

Now using weakening for $\wedge$ and the equivalence $(A * \mathrm{I}) \equiv A$ we can derive each of the following:

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

Thus we can derive $(\mathrm{I} \wedge (A \rightarrow (A * A))) * (\mathrm{I} \wedge A) \vdash (A * A)$ by modus ponens, whence by Prop. 2.2 (b) and (a) we obtain:

$$A \rightarrow (A * A) \vdash \mathrm{I} \rightarrow ((\mathrm{I} \wedge A) \mathrel{-\!\!*} (A * A)) \tag{3}$$

By combining (2) and (3) the second derived rule of Lemma 2.1 yields $\vdash \mathrm{I} \rightarrow ((\mathrm{I} \wedge A) \mathrel{-\!\!*} (A * A))$, which is equal to $\mathrm{I} \vdash (\mathrm{I} \wedge A) \mathrel{-\!\!*} (A * A)$. From this we obtain $(\mathrm{I} \wedge A) \vdash (A * A)$ by Prop. 2.2(c). □

Restricted $*$-contraction with restricted $*$-weakening induces a collapse of $\wedge$ and $*$ at the multiplicative unit I:

$(A * B) \vdash (B * A)$ $\quad (A * \mathrm{I}) \vdash A$

$(A*(B * C)) \vdash ((A * B) * C)$ $\quad A \vdash (A * \mathrm{I})$

$(A * (A \mathrel{-\!\!*} B)) \vdash B$

$$\frac{A \vdash B}{(A * C) \vdash (B * C)} \qquad \frac{(A * B) \vdash C}{A \vdash (B \mathrel{-\!\!*} C)}$$

(a) Axioms and rules for $*$, $\mathrel{-\!\!*}$ and I.

$A \vdash (B \to A)$ $\quad A \vdash (B \to (A \wedge B))$

$(A \to (B \to C)) \vdash ((A \to B) \to (A \to C))$ $\quad (A \wedge B) \vdash A$

$((A \to B) \to A) \vdash A$ (*Peirce's law*) $\quad (A \wedge B) \vdash B$

$$\frac{A \qquad A \vdash B}{B} \qquad \frac{(A \wedge B) \vdash C}{A \vdash (B \to C)}$$

(b) Axioms and rules for $\to$ and $\wedge$.

Table 1. Minimal Boolean BI, which employs only $\wedge$, $\to$, $*$, $\mathrel{-\!\!*}$ and I.

Corollary 2.2. *The following hold in* BBI$+$eW*:*

$$(\mathrm{I} \wedge (A * B)) \equiv (\mathrm{I} \wedge A \wedge B) \equiv ((\mathrm{I} \wedge A) * (\mathrm{I} \wedge B))$$

Proof. By restricted $*$-weakening and Lemma 2.2. □

Proposition 2.3. BBI$+$W *is ordinary classical logic.*

Proof. Easily, $A \equiv (A \wedge (A * \mathrm{I})) \equiv (A \wedge \mathrm{I})$ in BBI$+$W. Thus by Corollary 2.2, we have $(A * B) \equiv (A \wedge B)$. □

3 From computations to minimal BBI proofs

In this section we encode terminating computations of two-counter Minsky machines in minimal BBI.

Definition 3.1. A non-deterministic, two-counter *Minsky machine* M with counters c_1, c_2 is given by a finite set of *instructions*, which are each of one of the following labelled forms:

$$\begin{array}{ll} \text{“increment } c_k \text{ by 1”} & L_i\colon c_k{+}{+}; \mathbf{goto}\ L_j; \\ \text{“decrement } c_k \text{ by 1”} & L_i\colon c_k{-}{-}; \mathbf{goto}\ L_j; \\ \text{“zero-test } c_k\text{”} & L_i\colon \mathbf{if}\ c_k{=}0\ \mathbf{goto}\ L_j; \\ \text{“goto”} & L_i\colon \mathbf{goto}\ L_j; \end{array} \tag{4}$$

where $k \in \{1, 2\}$, $i \geq 1$ and $j \geq 0$. The labels L_0 and L_1 are reserved for the *final* and *initial* states of M, respectively. For technical reasons, we also add the special labels L_{-1} and L_{-2} which come equipped with the following four instructions:

$$\begin{array}{ll} L_{-1}\colon c_2{-}{-}; \mathbf{goto}\ L_{-1}; & L_{-1}\colon \mathbf{goto}\ L_0; \\ L_{-2}\colon c_1{-}{-}; \mathbf{goto}\ L_{-2}; & L_{-2}\colon \mathbf{goto}\ L_0; \end{array} \tag{5}$$

A *configuration* of M is given by $\langle L, n_1, n_2\rangle$, where the label L is the current state of M, and n_1 and n_2 are the current values of counters c_1 and c_2, respectively. We write $\rightsquigarrow_M$ for one instruction step of M, and write $\langle L, n_1, n_2\rangle \rightsquigarrow_M^* \langle L', n_1', n_2'\rangle$ if M can go from the configuration $\langle L, n_1, n_2\rangle$ to the configuration $\langle L', n_1', n_2'\rangle$ in a finite number of steps. We say that M *terminates from* $\langle L, n_1, n_2\rangle$, written $\langle L, n_1, n_2\rangle \Downarrow_M$, if $\langle L, n_1, n_2\rangle \rightsquigarrow_M^* \langle L_0, 0, 0\rangle$.

The specific role of L_{-1} and L_{-2} is explained by:

Lemma 3.1. $\langle L_{-k}, n_1, n_2\rangle \Downarrow_M$ *if and only if* $n_k = 0$.

Proof. The only instructions applicable to the configuration $\langle L_{-k}, n_1, n_2\rangle$ are those from the group (5). □

Definition 3.2. In our encoding we use the following abbreviation. We fix a propositional variable b, and henceforth define a “relative negation” by: $\mathrel{-}A =_{\text{def}} (A \mathrel{-\!\!*} b)$.

Lemma 3.2. *The following are derivable in minimal* BBI*:*

(a) $A \vdash \mathrel{-}\mathrel{-}A$ *and* $\mathrel{-}\mathrel{-}\mathrel{-}A \vdash \mathrel{-}A$

(b) $A * (\mathrel{-}B \mathrel{-\!\!*} \mathrel{-}A) \vdash \mathrel{-}\mathrel{-}B$

(c) $\dfrac{A * B \vdash C}{A * \mathrel{-}\mathrel{-}B \vdash \mathrel{-}\mathrel{-}C}$

(d) $\dfrac{A * B \vdash D \qquad A * C \vdash D}{A * \mathrel{-}\mathrel{-}(B \vee C) \vdash \mathrel{-}\mathrel{-}D}$

Definition 3.3 (Machine encoding)**.** We encode each instruction γ from (4) by the following formula $\kappa(I)$:

$$\begin{array}{rl} \kappa(L_i\colon c_k{+}{+}; \mathbf{goto}\ L_j;) & =_{\text{def}} (\mathrel{-}(l_j * p_k) \mathrel{-\!\!*} \mathrel{-}l_i) \\ \kappa(L_i\colon c_k{-}{-}; \mathbf{goto}\ L_j;) & =_{\text{def}} (\mathrel{-}l_j \mathrel{-\!\!*} \mathrel{-}(l_i * p_k)) \\ \kappa(L_i\colon \mathbf{if}\ c_k{=}0\ \mathbf{goto}\ L_j;) & =_{\text{def}} (\mathrel{-}(l_j \vee l_{-k}) \mathrel{-\!\!*} \mathrel{-}l_i) \\ \kappa(L_{-i}\colon \mathbf{goto}\ L_j;) & =_{\text{def}} (\mathrel{-}l_j \mathrel{-\!\!*} \mathrel{-}l_{-i}) \end{array}$$

where p_1, p_2, l_{-2}, l_{-1}, l_0, l_1, l_2,... are distinct propositional variables (p_1 and p_2 are used to represent the counters c_1 and c_2, respectively). Then for any Minsky machine M given by instructions γ_1,γ_2,...,γ_t define its encoding formula $\kappa(M)$ by: $\kappa(M) =_{\text{def}} (\mathrm{I} \wedge \bigwedge_{i=1}^{t} \kappa(\gamma_i))$.

Lemma 3.3. *For each instruction γ of a machine M, the sequent $\kappa(M) \vdash (\kappa(M) * \kappa(\gamma))$ is derivable in minimal* BBI*.*

Proof. Follows from Lemma 2.2. □

Theorem 3.1. *Suppose that $\langle L_i, n_1, n_2\rangle \Downarrow_M$ for some M. Then the following sequent is derivable in minimal* BBI*:*

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

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

Proof. Since $A * (\mathrm{I} \wedge \mathrel{-}l_0) \vdash b$ is easily derivable from $A \vdash \mathrel{-}\mathrel{-}l_0$ it suffices to prove the stronger sequent:

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

We proceed by induction on the length m of the computation of $\langle L, n_1, n_2\rangle \rightsquigarrow_M^* \langle L_0, 0, 0\rangle$. In the base case $m = 0$ we have $n_1 = n_2 = 0$, and thus we require to derive:

$$\kappa(M) * l_0 * \mathrm{I} * \mathrm{I} \vdash -\!\!-l_0$$

Using weakening for $\wedge$ and $\mathrm{I} * A \equiv A$, this reduces to $l_0 \vdash -\!\!-l_0$, which is provable by part (a) of Lemma 3.2.

Next, we assume that the result holds for all computations of length $m-1$, and show that it holds for any computation of length m. We then proceed by case distinction on the instruction γ which yields the first step of the computation. We show the cases for an increment instruction and for a zero-test instruction. The decrement and goto instructions are treated similarly to increment instructions.

Case $\gamma = (L_i\colon k{+}{+}; \mathbf{goto}\ L_j;)$**.** We show the subcase $k = 1$; the subcase $k = 2$ is similar. By the case assumption we have $\langle L_i, n_1, n_2\rangle \rightsquigarrow_M \langle L_j, n_1 + 1, n_2\rangle$, and we are required to show that the following is derivable:

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

This derivation is produced roughly by the following chain of backward reasoning. First, since γ is an instruction of M and $\kappa(\gamma) = ((-\!\!*(l_j * p_1) \mathrel{-\!\!*} -\!\!*l_i)$, we can apply Lemma 3.3 to generate the obligation:

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

We can use part (b) of Lemma 3.2 to reduce this to:

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

Using part (a) of Lemma 3.2 this reduces again to:

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

which further reduces by part (c) of Lemma 3.2 to:

$$\kappa(M) * l_j * p_1^{n_1+1} * p_2^{n_2} \vdash -\!\!-l_0$$

which is provable by the induction hypothesis.

Case $\gamma = (L_i\colon \mathbf{if}\ c_k{=}0\ \mathbf{goto}\ L_j;)$**.** We show the subcase $k = 1$; the subcase $k = 2$ is similar. By the case assumption we have $\langle L_i, 0, r_j\rangle \rightsquigarrow_M \langle L_j, 0, n_2\rangle$, and require to derive:

$$\kappa(M) * l_i * \mathrm{I} * p_2^{n_2} \vdash -\!\!-l_0$$

By the equivalence $A * \mathrm{I} \equiv A$ and using Lemma 3.3 to duplicate $\kappa(\gamma)$ as in the previous case, it suffices to show:

$$\kappa(M) * (-\!\!*(l_j \vee l_{-1}) \mathrel{-\!\!*} -\!\!*l_i) * l_i * p_2^{n_2} \vdash -\!\!-l_0$$

By employing part (b) of Lemma 3.2 we can reduce this to:

$$\kappa(M) * -\!\!-(l_j \vee l_{-1}) * p_2^{n_2} \vdash -\!\!-l_0$$

which further reduces by part (d) of the same lemma to the pair of proof obligations:

$$\kappa(M) * l_j * p_2^{n_2} \vdash -\!\!-l_0$$
$$\kappa(M) * l_{-1} * p_2^{n_2} \vdash -\!\!-l_0$$

The first of these is immediate by induction hypothesis and $A * \mathrm{I} \equiv A$. For the second, we note that since L_{-1} is by definition labelled only by decrement and goto instructions, it follows by induction on n_2 and the proofs of the decrement and goto instruction cases that the present theorem already holds when $L_i = L_{-1}$. Thus, because $\langle L_{-1}, 0, n_2\rangle \Downarrow_M$ by Lemma 3.1, we have that the second obligation above is also provable. This completes the case, and the proof. □

4 From validity to terminating computations

In this section, our goal is to show that for each of the concrete models in Example 1.1, we have $\langle L_i, n_1, n_2\rangle \Downarrow_M$ whenever the following sequent is valid:

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

For the sake of perspicuity we establish this property first for the *RAM-domain model*, which can be seen as the simplest heap model from Example 1.1(a), obtained by taking $L = \mathbb{N}$ and RV a singleton set. Then, we extend our approach to the most complex stack-and-heap models from Example 1.1(c), of which all the other models can be seen as special instances.

Definition 4.1. The *RAM-domain model* is $(\mathcal{D}, \circ, \{e_0\})$ where $\mathcal{D}$ is the class of finite subsets of $\mathbb{N}$, and $d_1 \circ d_2$ is the union of the disjoint sets d_1 and d_2 (with $d_1 \circ d_2$ undefined if d_1 and d_2 are not disjoint). The unit e_0 is $\emptyset$.

Definition 4.2. We introduce the following valuation ρ_0 for the RAM-domain model $(\mathcal{D}, \circ, \{e_0\})$:

$$\rho_0(p_1) = \{\ \{2\}, \{4\}, \{8\}, \ldots, \{2^m\}, \ldots\}$$
$$\rho_0(p_2) = \{\ \{3\}, \{9\}, \{27\}, \ldots, \{3^m\}, \ldots\}$$
$$\rho_0(l_i) = \{\ \{\delta_i\}, \{\delta_i^2,\}\, \{\delta_i^3\}, \ldots, \{\delta_i^m\}, \ldots\}$$

where δ_i is taken as a fresh prime number for each of the propositional variables l_{-2}, l_{-1}, l_0, l_1, l_2,..., and:

$$\rho_0(b) = \bigcup_{\langle L_i, n_1, n_2\rangle \Downarrow_M} [\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0}$$

Lemma 4.1. *Definition 4.2 guarantees that for any n, a finite set d belongs to $[\![p_k^n]\!]_{\rho_0}$ if and only if d consists of exactly n distinct powers of the corresponding prime. Thus any element of $[\![p_k^n]\!]_{\rho_0}$ uniquely determines the number n.*

Proof. By induction on n. E.g., by definition, $[\![p_1 * p_1]\!]_{\rho_0}$ consists of two-element sets of the form $\{2^{m_1}, 2^{m_2}\}$. □

Comment 4.1. *Our choice of $\rho_0(p_1)$ and $\rho_0(p_2)$ to have* infinitely many disjoint elements *is dictated by peculiarities of composition in the heap model. For any* finite *choice of $\rho_0(p_k)$, we must have $[\![p_k^n]\!]_{\rho_0} = [\![p_k^m]\!]_{\rho_0}$ for sufficiently large n and m, which obstructs us in uniquely representing the contents n of counter c_k by the formula p_k^n. (We discuss decidability consequences in Section 6.)*

Lemma 4.2. *$e_0 \models_{\rho_0} \kappa(M)$ for any machine M.*

Proof. Writing $M = \{\gamma_1, \ldots, \gamma_t\}$, we have by Defn. 2.4:

$$\begin{array}{rcl} e_0 \models_{\rho_0} \kappa(M) & \Leftrightarrow & e_0 \models_{\rho_0} \mathrm{I} \wedge \bigwedge_{i=1}^{t} \kappa(\gamma_i) \\ & \Leftrightarrow & e_0 \in \{e_0\} \text{ and } \forall 1 \leq i \leq t.\ e_0 \models_{\rho_0} \kappa(\gamma_i) \end{array}$$

Thus it suffices to show that $e_0 \models_{\rho_0} \kappa(\gamma)$ for any instruction γ. We present the cases for an increment instruction and for a zero-test instruction; the other cases are similar.

Case $\gamma = (L_i\colon c_k{+}{+}; \mathbf{goto}\ L_j;)$. We show the subcase $k = 1$; the subcase $k = 2$ is similar. We have $\kappa(\gamma) = (\mathbin{-\!\!*}(l_j * p_1) \mathbin{-\!\!*} \mathbin{-\!\!*} l_i)$. To show $e_0 \models_{\rho_0} \kappa(I)$, we must show that $x \models_{\rho_0} \mathbin{-\!\!*}(l_j * p_1)$ implies $x \models_{\rho_0} \mathbin{-\!\!*} l_i$ for any $x \in \mathcal{D}$. First note that we have for all $x \in \mathcal{D}$:

$$\begin{array}{l} \quad x \models_{\rho_0} \mathbin{-\!\!*}(l_j * p_1) \\ \Leftrightarrow \forall x'.\ x \circ x' \text{ defined and } x' \models_{\rho_0} l_j * p_1 \text{ implies } x \circ x' \models_{\rho_0} b \\ \Leftrightarrow \forall x'.\ x \circ x' \text{ defined and } (x' = y \circ z \text{ and } y \models_{\rho_0} l_j \text{ and} \\ \quad z \models_{\rho_0} p_1) \text{ implies } x \circ x' \models_{\rho_0} b \\ \Leftrightarrow \forall y, z.\ x \circ y \circ z \text{ defined and } y \in \rho_0(l_j) \text{ and } z \in \rho_0(p_1) \\ \quad \text{implies } x \circ y \circ z \models_{\rho_0} b \\ \Leftrightarrow x \in [\![p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0} \text{ and } \langle L_j, n_1 + 1, n_2\rangle{\Downarrow_M} \end{array}$$

The last equivalence follows because, since the elements of $\mathcal{D}$ are finite sets whereas $\rho_0(l_j)$ and $\rho_0(p_1)$ contain *infinitely many disjoint sets*, $x \circ y \circ z$ must be defined for some $y \in \rho_0(l_j), z \in \rho_0(p_1)$, in which case $x \circ y \circ z \models_{\rho_0} b$ must hold. By a similar argument, we also have for all $x \in \mathcal{D}$:

$$x \models_{\rho_0} \mathbin{-\!\!*} l_i \ \Leftrightarrow\ x \in [\![p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0} \text{ and } \langle L_i, n_1, n_2\rangle{\Downarrow_M}$$

Since $\langle L_i, n_1, n_2\rangle \rightsquigarrow_M \langle L_j, n_1 + 1, n_2\rangle$ by applying the increment instruction γ, we have that $\langle L_j, n_1 + 1, n_2\rangle{\Downarrow_M}$ implies $\langle L_i, n_1, n_2\rangle{\Downarrow_M}$, so that $x \models_{\rho_0} \mathbin{-\!\!*}(l_j * p_1)$ implies $x \models_{\rho_0} \mathbin{-\!\!*} l_i$ as required.

Case $\gamma = (L_i\colon \mathbf{if}\ c_k{=}0\ \mathbf{goto}\ L_j;)$. We show only the subcase $k = 1$. We have $\kappa(\gamma_) = (\mathbin{-\!\!*}(l_j \vee l_{-1}) \mathbin{-\!\!*} \mathbin{-\!\!*} l_i)$. To show $e_0 \models_{\rho_0} \kappa(I)$, we must show that $x \models_{\rho_0} \mathbin{-\!\!*}(l_j \vee l_{-1})$ implies $x \models_{\rho_0} \mathbin{-\!\!*} l_i$ for any $x \in \mathcal{D}$. As in the previous case, we have for all $x \in \mathcal{D}$:

$$x \models \mathbin{-\!\!*} l_i \ \Leftrightarrow\ x \in [\![p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0} \text{ and } \langle L_i, n_1, n_2\rangle{\Downarrow_M}$$

We also have, for all $x \in \mathcal{D}$:

$$\begin{array}{l} \quad x \models \mathbin{-\!\!*}(l_j \vee l_{-1}) \\ \Leftrightarrow \forall x'.\ x \circ x' \text{ defined and } (x' \models_{\rho_0} l_j \text{ or } x' \models_{\rho_0} l_{-1}) \\ \quad \text{implies } x \circ x' \models_{\rho_0} b \\ \Leftrightarrow \forall x'.\ x \circ x' \text{ defined and } x' \in \rho_0(l_j) \cup \rho_0(l_{-1}) \text{ implies} \\ \quad x \circ x' \models_{\rho_0} b \\ \Leftrightarrow x \in [\![p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0} \text{ and } \langle L_j, n_1, n_2\rangle{\Downarrow_M} \text{ and } \langle L_{-1}, n_1, n_2\rangle{\Downarrow_M} \\ \Leftrightarrow x \in [\![p_2^{n_2}]\!]_{\rho_0} \text{ and } \langle L_j, 0, n_2\rangle{\Downarrow_M} \ \text{(by Lemma 3.1)} \end{array}$$

The penultimate equivalence above requires reasoning similar to that employed in the previous case: $x \circ x'$ must be defined for some $x' \in \rho_0(l_j)$ and for some $x' \in \rho_0(l_{-1})$.

Since $\langle L_i, 0, n_2\rangle \rightsquigarrow_M \langle L_j, 0, n_2\rangle$ by the operational semantics of the zero-test instruction γ, we have: $\langle L_j, 0, n_2\rangle{\Downarrow_M}$ implies $\langle L_i, 0, n_2\rangle{\Downarrow_M}$, i.e. $x \models_{\rho_0} \mathbin{-\!\!*}(l_j \vee l_{-1})$ implies $x \models_{\rho_0} \mathbin{-\!\!*} l_i$ as required. □

Lemma 4.3. *$e_0 \models_{\rho_0} \mathrm{I} \wedge \mathbin{-\!\!*} l_0$.*

Proof. We trivially have $e_0 \models_{\rho_0} \mathrm{I}$. Since $\langle L_0, 0, 0\rangle{\Downarrow_M}$, we have $\rho_0(l_0) \subseteq \rho_0(b)$ by construction of ρ_0, which straightforwardly entails $e_0 \models_{\rho_0} l_0 \mathbin{-\!\!*} b$. □

Theorem 4.1. *If $(\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge \mathbin{-\!\!*} l_0)) \vdash b$ is valid in $(\mathcal{D}, \circ, \{e_0\})$ then $\langle L_i, n_1, n_2\rangle{\Downarrow_M}$.*

Proof. By the definition of validity and using the equations (1) we have:

$$\begin{array}{rl} & [\![\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge \mathbin{-\!\!*} l_0)]\!]_{\rho_0} \subseteq \rho_0(b) \\ \text{i.e.} & [\![\kappa(M)]\!]_{\rho_0} \cdot [\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0} \cdot [\![\mathrm{I} \wedge \mathbin{-\!\!*} l_0]\!]_{\rho_0} \subseteq \rho_0(b) \end{array}$$

Since $e_0 \in [\![\kappa(M)]\!]_{\rho_0}$ by Lemma 4.2 and $e_0 \in [\![\mathrm{I} \wedge \mathbin{-\!\!*} l_0]\!]_{\rho_0}$ by Lemma 4.3 we have in particular:

$$[\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0} \subseteq \rho_0(b)$$

By Lemma 4.1 and (1), the set $[\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_0}$ uniquely determines the numbers n_1 and n_2, whence our construction of $\rho_0(b)$ yields $\langle L_i, n_1, n_2\rangle{\Downarrow_M}$. □

Having established our Theorem 4.1 for the basic RAM-domain model, we now extend it to the most sophisticated *stack-and-heap* models from Example 1.1(c), in which the heaps are heaps-with-permissions. All the models from Example 1.1 can be seen as special instances of such models.

Definition 4.3. Let $(S \times H, \circ, E)$ be a stack-and-heap model from Example 1.1(c), where S is a set of stacks and $H = \mathbb{N} \rightharpoonup_{\text{fin}} (RV \times P)$ is a set of heaps-with-permissions and underlying permission algebra $(P, \bullet, \mathbb{1})$. (Recall that $\mathbb{1} \bullet \pi$ is undefined for all $\pi \in P$.)

Based on our valuation ρ_0 for the RAM-domain model in Definition 4.2, we introduce a valuation ρ_1 for $(S \times H, \circ, E)$

as follows. First, we fix an arbitrary stack $s_0 \in S$, and for each finite set $d \subseteq \mathbb{N}$ we define the set $[d] \subseteq S \times H$ by:

$$[d] =_{\text{def}} \{\langle s_0, h\rangle \mid \mathtt{domain}(h) = d \text{ and } \forall \ell \in d.\ h(\ell) = \langle \_, \mathbb{1}\rangle\}$$

Then for any propositional variable p we define its valuation by: $\rho_1(p) = \bigcup_{d \in \rho_0(p)} [d]$.

Lemma 4.4. *For any propositional variable p and q:*

$$[\![p * q]\!]_{\rho_1} = [\![p]\!]_{\rho_1} \cdot [\![q]\!]_{\rho_1} = \bigcup_{d \in [\![p*q]\!]_{\rho_0}} [d].$$

Proof. It suffices to show that $[d_1 \circ d_2] = [d_1] \cdot [d_2]$. For disjoint d_1 and d_2 this is given by construction. For overlapping d_1 and d_2, assume that $\ell \in d_1 \cap d_2$, and $\langle s_0, h_1\rangle \circ \langle s_0, h_2\rangle$ is defined for some $\langle s_0, h_1\rangle \in [d_1]$ and $\langle s_0, h_2\rangle \in [d_2]$. By construction of $[d_1]$ and $[d_2]$, this implies $h_1(\ell) = h_2(\ell) = \langle \_, \mathbb{1}\rangle$. But then since $\langle s_0, h_1\rangle \circ \langle s_0, h_2\rangle$ is defined, we must have $\mathbb{1} \bullet \mathbb{1}$ defined, which is a contradiction. Thus $[d_1] \cdot [d_2]$ is empty when $d_1 \circ d_2$ is undefined. □

Lemma 4.5. *For any formula A of the form l, $(l * p)$, or $(l \vee l')$, the set $[\![A]\!]_{\rho_1} \cdot \{\langle s, h\rangle\}$ is not empty, and we have $[\![A]\!]_{\rho_1} \cdot \{\langle s, h\rangle\} \subseteq [\![b]\!]_{\rho_1}$ if and only if $[\![A]\!]_{\rho_0} \cdot \{d\} \subseteq [\![b]\!]_{\rho_0}$ with $\langle s, h\rangle \in [d]$, where $d = \mathtt{domain}(h)$.*

Lemma 4.6. $\langle s_0, e_0\rangle \models_{\rho_1} \kappa(M)$ *for any machine M.*

Proof. As in Lemma 4.2, we show that $\langle s_0, e_0\rangle \models_{\rho_1} \kappa(\gamma)$ for any γ in the group (4). Recalling that $-\!\!*A = (A \mathrel{-\!\!*} b)$, each $\kappa(\gamma)$ is of the form $((A \mathrel{-\!\!*} b) \mathrel{-\!\!*} (B \mathrel{-\!\!*} b))$, so it suffices to prove $[\![A \mathrel{-\!\!*} b]\!]_{\rho_1} \subseteq [\![B \mathrel{-\!\!*} b]\!]_{\rho_1}$. Using the equations (1), this amounts to showing, for any $\langle s, h\rangle$:

$$[\![A]\!]_{\rho_1} \cdot \{\langle s, h\rangle\} \subseteq [\![b]\!]_{\rho_1} \;\Rightarrow\; [\![B]\!]_{\rho_1} \cdot \{\langle s, h\rangle\} \subseteq [\![b]\!]_{\rho_1} \quad (6)$$

Assume $[\![A]\!]_{\rho_1} \cdot \{\langle s, h\rangle\} \subseteq [\![b]\!]_{\rho_1}$. Let $d = \mathtt{domain}(h)$. By Lemma 4.5 we have $\langle s, h\rangle \in [d]$, and $[\![A]\!]_{\rho_0} \cdot \{d\} \subseteq [\![b]\!]_{\rho_0}$. Lemma 4.2 shows that $e_0 \models_{\rho_0} \kappa(\gamma)$ and thus $[\![A \mathrel{-\!\!*} b]\!]_{\rho_0} \subseteq [\![B \mathrel{-\!\!*} b]\!]_{\rho_0}$, so that $[\![B]\!]_{\rho_0} \cdot \{d\} \subseteq [\![b]\!]_{\rho_0}$, whence Lemma 4.5 yields $[\![B]\!]_{\rho_1} \cdot \{\langle s, h\rangle\} \subseteq [\![b]\!]_{\rho_1}$. □

Lemma 4.7. $\langle s_0, e_0\rangle \models_{\rho_1} (l_0 \mathrel{-\!\!*} b)$.

Proof. Similar to Lemma 4.3. □

Theorem 4.2. *If a sequent $\mathcal{F}_{M,l_i,n_1,n_2}$ of the form*

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

is valid in some concrete model from Example 1.1 then $\langle L_i, n_1, n_2\rangle \Downarrow_M$.

Proof. Without loss of generality, we may assume that $\mathcal{F}_{M,l_i,n_1,n_2}$ is valid in a stack-and-heap model given in Definition 4.3. Thus we have by definition of validity:

$$[\![\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge (l_0 \mathrel{-\!\!*} b))]\!]_{\rho_1} \subseteq [\![b]\!]_{\rho_1}$$

Taking into account Lemmas 4.6 and 4.7, we get:

$$[\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_1} \subseteq \rho_1(b).$$

According to Lemmas 4.4 and 4.1, the set $[\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_1}$ uniquely determines the numbers n_1 and n_2, so that our construction of $\rho_1(b)$ yields $\langle L_i, n_1, n_2\rangle \Downarrow_M$. □

5 Undecidability of separation logic

Now, based upon Figure 1, we may state the following:

Corollary 5.1. *The following problems are undecidable:*

- *provability in minimal* BBI*;*
- *provability in* BBI*;*
- *provability in* BBI+eW*;*
- *validity in the class of all separation models;*
- *validity in the class of all separation models with indivisible units;*
- *validity in any of the concrete separation models in Ex. 1.1.*

Proof. The termination of a Minsky machine M from configuration $C = \langle L_1, n, 0\rangle$ is undecidable [18], and reduces to each of the problems above by the diagram in Figure 1. □

Corollary 5.2. *Neither minimal* BBI *nor* BBI *nor* BBI+eW *has the finite model property.*

Proof. A recursive enumeration of proofs and finite countermodels for any of the logics above would yield a decision procedure for provability, which is impossible. □

6 Finite approximations

Our undecidability results for propositional separation logic seem to be at odds with the decidability of the quantifier-free fragment of a certain separation theory over an *infinite* heap model, due to Calcagno et al.[7]. The crucial difference is that their decidability result is restricted to *finite valuations* ρ such that $\rho(p)$ is finite for every atomic proposition p. Namely, in [7] each p represents one *cell*, i.e. a heap whose domain is a singleton. The reason why their decidability is highly non-trivial is that their language contains $\mathrel{-\!\!*}$ and the underlying separation model employs a non-total $\circ$, so that, e.g., whenever $[\![A]\!]_\rho$ is *finite*, $[\![A \mathrel{-\!\!*} B]\!]_\rho$ becomes *infinite*. In this section, we investigate this phenomenon.

Theorem 6.1. *Let $(H, \circ, \{e_0\})$ be a heap model. Then there is an algorithm that, for any finite valuation ρ, and any sequent $\mathcal{F}_{M,l_1,n_1,n_2}$ of the form:*

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

decides whether this sequent is valid under the valuation ρ.

Proof. In principle, this can be deduced from [7]. Our direct construction in Lemma 6.1 show subtleties of the problem.

Lemma 6.1. *There is an algorithm that, for any finite valuation ρ, decides whether $e_0 \models_\rho \kappa(M)$.*

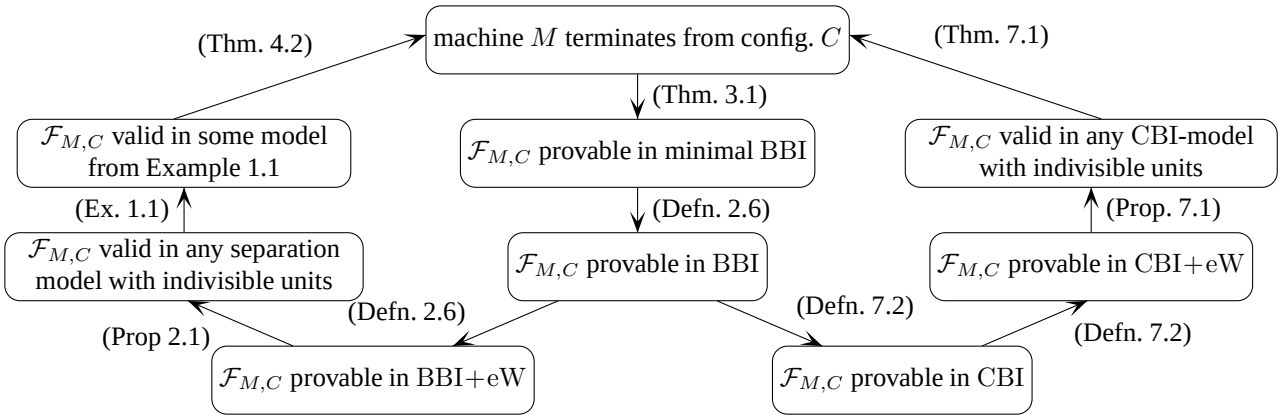

Figure 1. Diagrammatic proof of undecidability. The arrows are implications, and $\mathcal{F}_{M,C}$ is a formula / sequent built from machine M and configuration C. The problems at each node are all undecidable.

Proof. As in Lemmas 4.2 and 4.6 (cf. (6)), we have to check $e_0 \models_\rho \kappa(\gamma)$ for any γ of the form $((A \mathrel{-\!\!*} b) \mathrel{-\!\!*} (B \mathrel{-\!\!*} b))$, whence it suffices to show how to check the sentence:

$$\forall z((\llbracket A \rrbracket_\rho \cdot \{z\} \subseteq \llbracket b \rrbracket_\rho) \;\Rightarrow\; (\llbracket B \rrbracket_\rho \cdot \{z\} \subseteq \llbracket b \rrbracket_\rho))$$

where $\llbracket A \rrbracket_\rho$, $\llbracket B \rrbracket_\rho$ and $\llbracket b \rrbracket_\rho$ are finite. We consider two cases depending on the domain of z:

(1) In the case that $(\llbracket A \rrbracket_\rho \cdot \{z\} \neq \emptyset)$, we can construct a *finite* list of all z such that $(\llbracket A \rrbracket_\rho \cdot \{z\} \subseteq \llbracket b \rrbracket_\rho)$. Then we may check for these z whether $(\llbracket B \rrbracket_\rho \cdot \{z\} \subseteq \llbracket b \rrbracket_\rho)$.

(2) If $(\llbracket A \rrbracket_\rho \cdot \{z\} = \emptyset)$, then trivially $(\llbracket A \rrbracket_\rho \cdot \{z\} \subseteq \llbracket b \rrbracket_\rho)$, so it suffices to show how to check a sentence of the form:

$$\forall z((\llbracket A \rrbracket_\rho \cdot \{z\} = \emptyset) \;\Rightarrow\; (\llbracket B \rrbracket_\rho \cdot \{z\} \subseteq \llbracket b \rrbracket_\rho)) \qquad (7)$$

Let $\llbracket A \rrbracket_\rho = \{f_1, f_2, .., f_m\}$ and $\alpha_i = \mathtt{domain}(f_i)$ for all i, and $\llbracket B \rrbracket_\rho = \{g_1, .., g_t\}$ and $\beta_j = \mathtt{domain}(g_j)$ for all j.
For each choice of ℓ_1, ℓ_2,.., ℓ_m from α_1, α_2,.., α_m, respectively, we write $d_{\ell_1,\ell_2,..,\ell_m}$ for the set $\{\ell_1, \ell_2, .., \ell_m\}$. In the following we rely on the fact that, although $\llbracket A \rrbracket_\rho \cdot \{z\} = \emptyset$ for *infinitely many* z, the domain of each of these z must be a superset of some $d_{\ell_1,..,\ell_m}$.

Case 1. Assume $\beta_j \cap d_{\ell_1,..,\ell_m} \neq \emptyset$ for all β_j and $d_{\ell_1,..,\ell_m}$. Since, for each z in question, $\mathtt{domain}(z)$ is a superset of some $d_{\ell_1,..,\ell_m}$, we have $\llbracket B \rrbracket_\rho \cdot \{z\} = \emptyset$, and so (7) is true.

Case 2. Assume $\beta_j \cap d_{\ell_1,..,\ell_m} = \emptyset$ for some β_j and $d_{\ell_1,..,\ell_m}$. Let $\widetilde{n}$ be a number greater than every number occurring in β_j, $d_{\ell_1,..,\ell_m}$, and $\llbracket b \rrbracket_\rho$, and let $\widetilde{z}$ be a heap such that $\mathtt{domain}(\widetilde{z}) = \{\widetilde{n}\} \cup d_{\ell_1,..,\ell_m}$. Then $g_j \circ \widetilde{z}$ is defined but $g_j \circ \widetilde{z} \notin \llbracket b \rrbracket_\rho$, and (7) is false. □

We complete the proof for Theorem 6.1 as follows. Given an $\mathcal{F}_{M,l_i,n_1,n_2}$ we first use Lemma 6.1 to compute $\llbracket \kappa(M) \rrbracket_\rho$ and $\llbracket \mathrm{I} \wedge (l_0 \mathrel{-\!\!*} b) \rrbracket_\rho$ (both are subsets of $\{e_0\}$). If either of these sets is empty then trivially $\mathcal{F}_{M,l_i,n_1,n_2}$ is valid under the ρ. Otherwise, $\llbracket \kappa(M) \rrbracket_\rho = \llbracket \mathrm{I} \wedge (l_0 \mathrel{-\!\!*} b) \rrbracket_\rho = \llbracket \mathrm{I} \rrbracket_\rho$, and it only remains to check if the finite set $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_\rho$ is a subset of $\llbracket b \rrbracket_\rho$, which is straightforward. □

Corollary 6.1. *There is a sequent $\mathcal{F}_{M,l_1,n_0,0}$ of the form*

$$(\kappa(M) * l_1 * p_1^{n_0} * (\mathrm{I} \wedge (l_0 \mathrel{-\!\!*} b))) \vdash b$$

such that, for each separation model from Example 1.1, $\mathcal{F}_{M,l_1,n_0,0}$ is not valid in this model, but $\mathcal{F}_{M,l_1,n_0,0}$ is valid in this model under all finite valuations ρ.

Proof. Take M such that $K_M = \{n \mid \langle L_1, n, 0\rangle \Downarrow_M\}$ is undecidable. Let W_M be the set of all n such that $\mathcal{F}_{M,l_1,n,0}$ is not valid in the model under some finite valuation ρ. By Theorem 6.1, W_M is recursively enumerable. According to Theorem 3.1 and Proposition 2.1, K_M and W_M are disjoint. Therefore, we can find a number n_0 such that $n_0 \notin K_M \cup W_M$. Since $n_0 \notin K_M$, Theorem 4.2 implies that $\mathcal{F}_{M,l_1,n_0,0}$ is not valid in the model. However, $n_0 \notin W_M$ implies that $\mathcal{F}_{M,l_1,n_0,0}$ is valid under all finite valuations ρ. □

7 Extension to Classical BI

In this section, we extend our undecidability results to the class of "dualising separation models", whose proof-theoretical basis is given by Classical BI, or CBI [4].

Definition 7.1 (CBI-models)**.** A CBI-*model* is given by $(H, \circ, e, \cdot^{-1})$, where $\langle H, \circ, \{e\}\rangle$ is a separation model (with a single unit e) and $\cdot^{-1} : H \to H$ satisfies $h \circ h^{-1} = e \circ e^{-1} = e^{-1}$ for all $h \in H$.

The CBI-models we consider in this paper form a subclass of the more general *relational* CBI-models given in [4].

Example 7.1. Examples of CBI-models (cf. [4]):

(a) $([0,1], \circ, 0, \cdot^{-1})$, where $x_1 \circ x_2$ is $x_1 + x_2$ but undefined when $x_1 + x_2 > 1$. The inverse x^{-1} is $1 - x$.

(b) $(\Sigma, \circ, \varepsilon, \overline{\cdot})$ where Σ is any class of *languages* containing the empty language ε and closed under union $\cup$ and complement $\overline{\cdot}$. Here $d_1 \circ d_2$ is the union of disjoint languages d_1 and d_2 (in the overlapping case, $d_1 \circ d_2$ is undefined). E.g., Σ may be the class of regular languages, or Σ may be the class of finite and co-finite sets.

(c) *Effect algebras* [11], which arise in the foundations of quantum mechanics, are precisely CBI-models with indivisible units.

(d) *Permission algebras* $(P, \bullet, \mathbb{1})$ [3] enriched with a 'formal unit' e and 'formal equalities' $e \bullet h = h \bullet e = e$ are exactly non-degenerate CBI-models with indivisible units.

Definition 7.2. Following Definition 2.6, we introduce a second chain of logics as follows:

• CBI [4] is obtained from BBI by extending its language with a constant $\tilde{\mathrm{I}}$, and adding the axioms $A \vdash {\sim}{\sim}A$ and ${\sim}{\sim}A \vdash A$, where $\sim A$ is an abbreviation for $(A \mathrel{-\!\!*} \tilde{\mathrm{I}})$.

• CBI+eW is obtained by extending CBI with the restricted $*$-weakening $(\mathrm{I} \wedge (A * B)) \vdash A$;

• CBI+W is obtained by extending CBI with the unrestricted $*$-weakening $(A * B) \vdash A$.

Validity of CBI-formulas with respect to CBI-models $(H, \circ, e, \cdot^{-1})$ is defined as in Definition 2.5 once we extend the forcing relation $h \models_\rho A$ given in Definition 2.4 with the clause $h \models_\rho \tilde{\mathrm{I}} \Leftrightarrow h \neq e^{-1}$.

Proposition 7.1 (Soundness)**.** *If A is provable in* CBI *then A is valid in all* CBI*-models, and if A is provable in* CBI+eW *then A is valid in all* CBI*-models with indivisible units.*

Corollary 7.1. *Using Proposition 7.1 we have, similar to Corollary 2.1:* BBI $\subset$ CBI $\subset$ CBI+eW $\subset$ CBI+W *(the inclusion* BBI $\subset$ CBI *was established in* [4]*).*

Proposition 7.2. CBI+W *collapses into classical logic.*

Proof. As in Prop. 2.3, $A * B \equiv A \wedge B$ and $\mathrm{I} \equiv \top$. Furthermore, $\tilde{\mathrm{I}} \equiv {\sim}\mathrm{I} \equiv {\sim}\top \equiv \bot$, which forces ${\sim}A \equiv \neg A$. □

Since minimal BBI-provability implies CBI(+eW)-provability, to establish undecidability for CBI it suffices (see Fig. 1) to prove the analogue of Thm. 4.1 for a CBI-model.

Definition 7.3. We introduce the model $(\mathcal{D}^+, \circ, e_0, \cdot^{-1})$, where $\mathcal{D}^+$ is the class of finite and co-finite subsets of $\mathbb{N}$, $\circ$ is disjoint union, the unit e_0 is $\emptyset$ and $\cdot^{-1}$ is set complement.

Definition 7.4. We define a valuation ρ_C for $(\mathcal{D}^+, \circ, e_0, \cdot^{-1})$ by extending the valuation ρ_0 in Definition 4.2 as follows:

$$\begin{array}{rcl} \rho_C(x) & = & \rho_0(x) \quad \text{for all } x \in \{p_1, p_2\} \cup \{l_i \mid i \geq -2\} \\ \rho_C(b) & = & \rho_0(b) \cup \{d \in \mathcal{D}^+ \mid d \text{ is cofinite}\} \end{array}$$

Lemma 7.1. $e_0 \models_{\rho_C} \kappa(M)$ *for any machine M.*

Proof. As in Lemma 4.2, we must show $e_0 \models_{\rho_C} \kappa(\gamma)$ for any instruction γ. We only examine the increment instruction case, $\gamma = (L_i\colon c_k{+}{+}; \mathbf{goto}\ L_j;)$ for $k = 1$ here. As in the corresponding case of Lemma 4.2, we must show that $[\![-(l_j * p_1)]\!]_{\rho_C} \subseteq [\![-l_i]\!]_{\rho_C}$. Assuming that $x \models_{\rho_C} -(l_j * p_1)$ we have:

$$\forall y, z.\ ((x \circ y \circ z \text{ defined and } y \in \rho_C(l_j) \text{ and } z \in \rho_C(p_1)) \text{ implies } x \circ y \circ z \in \rho_C(b))$$

and need to show $x \models_{\rho_C} -l_i$, for which we have two cases. First, in the case that x is ***finite*** then the reasoning from the analogous case of Lemma 4.2 applies since ρ_C coincides with ρ_0 on all variables except b. That is, for some $y \in \rho_C(l_j)$ and $z \in \rho_C(p_1)$, $x \circ y \circ z$ is defined and thus $x \circ y \circ z \in \rho_C(b)$, whence $x \in [\![p_1^{n_1} * p_2^{n_2}]\!]_{\rho_C}$ and $\langle L_j, n_1 + 1, n_2\rangle \Downarrow_M$. By applying the instruction γ, we have $\langle L_i, n_1, n_2\rangle \Downarrow_M$, which implies that $x \circ x' \in \rho_C(b)$ whenever $x \circ x'$ is defined and $x' \in \rho_C(l_i)$, i.e. $x \models_{\rho_C} -l_i$ as required. In the case that x is *cofinite* then, by the same token, $x \circ x'$ is either undefined or cofinite for any $x' \in \rho_C(l_i)$, in which case $x \models_{\rho_C} -l_i$ as required because $\rho_C(b)$ contains all cofinite sets. □

Lemma 7.2. $e_0 \models_{\rho_C} \mathrm{I} \wedge -l_0$.

Proof. Similar to the proof of Lemma 4.3. □

Theorem 7.1. *If $(\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2}) * (\mathrm{I} \wedge -l_0) \vdash b$ is valid in the RAM-codomain model, then $\langle L_i, n_1, n_2\rangle \Downarrow_M$.*

Proof. By the definition of validity we have:

$$[\![\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (\mathrm{I} \wedge -l_0)]\!]_{\rho_C} \subseteq \rho_C(b)$$

By Lemma 7.1 and Lemma 7.2 we have in particular:

$$[\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_C} \subseteq \rho_C(b)$$

Since ρ_C coincides with ρ_0 on all propositional variables except b, the set $[\![l_i * p_1^{n_1} * p_2^{n_2}]\!]_{\rho_C}$ is finite, and uniquely determines n_1 and n_2 by Lemma 4.1 and equations (1). Our construction of $\rho_C(b)$ then yields $\langle L_i, n_1, n_2\rangle \Downarrow_M$. □

Again, based on Figure 1, we can assert the following:

Corollary 7.2. *The following problems are undecidable:*

- *provability in* CBI*;*
- *provability in* CBI+eW*;*
- *validity in the class of all* CBI*-models;*
- *validity in the class of* CBI*-models with indivisible units;*
- *validity in the concrete model $(\mathcal{D}^+, \circ, e_0, \cdot^{-1})$.*

Proof. Similar to Corollary 5.1. □

8 Concluding remarks

Our main contribution is that separation logic, the logic of memory-manipulating programs, is undecidable even at the propositional level. Oddly enough, it also turns out that while BI (which combines intuitionistic multiplicatives with intuitionistic additives) is decidable [13], its sibling BBI, (which combines intuitionistic multiplicatives with *Boolean* additives) is undecidable. In fact, to obtain an undecidable system, one need add only Boolean conjunction and implication to the multiplicatives.

Our undecidability results shed also new light on correlations between separation logic and linear logic.

From the point of view of logical principles, there are clear differences between separation logic and linear logic: e.g., distributivity of additive conjunction over disjunction, $A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C)$, holds even in BI but fails in linear logic. More specific to BBI, the restricted $*$-contraction $(I \wedge A) \vdash (A * A)$ holds in BBI as shown by our Lemma 2.2, but this too fails in linear logic. Finally, while adding the unrestricted $*$-weakening principle $(A * B) \vdash A$ to linear logic gives us the well-known *affine logic*, adding it to BBI forces a collapse into classical logic (Prop. 2.3).

From a semantical perspective, the *precise* expression of properties of memory in separation logic is based on the fact that $[\![A * B]\!]_\rho = [\![A]\!]_\rho \cdot [\![B]\!]_\rho$, i.e. the interpretation of $A * B$ is *exactly* the product of the interpretations of A and B. (This fact is also of crucial importance to its undecidability.) Linear logic interpretations deal only with sets that are *closed* w.r.t. a certain closure operator Cl, which, in particular, violates the above exact equality. Indeed, the same is true of BI interpretations [19]. Not only is this less precise, it admits no possibility of *finite valuations* in these logics since, e.g., in linear logic $Cl(\emptyset)$ is always infinite.

We also note that a direct adaptation of the encoding of Minsky machines developed for full linear logic in [16] does not work properly for BBI, so that we have had to develop a new encoding. Roughly speaking, in the linear logic encoding, each step in the derivation corresponds to a single forward step in the computation. In contrast, in our encoding, each step in the derivation corresponds to a 'move' from a class of terminating computations to a class of shorter terminating computations. An additional twist to the problem is that we require a much more complicated interpretation in the heap-like models of Example 1.1, because of the way *partial* composition is defined in these models.

Finally, our undecidability results for *concrete* heap-like models give new insights into the nature of decidable fragments of separation logic such as those given in [2, 7], as well as imposing boundaries on decidability. E.g., we can deduce that to obtain decidability in a heap-like model, one should either give up infinite valuations (as in [7]) or restrict the formula language (as in [2]).

References

[1] A. Ahmed, L. Jia, and D. Walker. Reasoning about hierarchical storage. In *Proceedings of LICS-18*, pp. 33–44, 2003.

[2] J. Berdine, C. Calcagno, and P. O'Hearn. A decidable fragment of separation logic. In *Proceedings of FSTTCS*, 2004.

[3] R. Bornat, C. Calcagno, P. O'Hearn, and M. Parkinson. Permission accounting in separation logic. In *Proceedings of POPL-32*, pp. 59–70, 2005.

[4] J. Brotherston and C. Calcagno. Classical BI (A logic for reasoning about dualising resource). In *Proceedings of POPL-36*, pp. 328–339, 2009.

[5] C. Calcagno, D. Distefano, P. O'Hearn, and H. Yang. Compositional shape analysis by means of bi-abduction. In *Proceedings of POPL-36*, pp. 289–300, 2009.

[6] C. Calcagno, P. O'Hearn, and H. Yang. Local action and separation logic. In *Proceedings of LICS-22*, pp. 366–378, 2007.

[7] C. Calcagno, H. Yang, and P. O'Hearn. Computability and complexity results for a spatial assertion language for data structures. In *Proceedings of FSTTCS*, pp. 108–119, 2001.

[8] W.-N. Chin, C. David, H. H. Nguyen, and S. Qin. Enhancing modular OO verification with separation logic. In *Proceedings of POPL-35*, 2008.

[9] D. Distefano and M. Parkinson. jStar: Towards practical verification for Java. In *Proc. OOPSLA*, pp. 213–226, 2008.

[10] M. Dodds, X. Feng, M. Parkinson, and V. Vafeiadis. Deny-guarantee reasoning. In *Proc. ESOP*, pp. 363–377, 2009.

[11] D. Foulis and M. Bennett. Effect algebras and unsharp quantum logics. *Foundations of Physics*, 24:1331–1352, 1994.

[12] D. Galmiche and D. Larchey-Wendling. Expressivity properties of Boolean BI through relational models. In *Proceedings of FSTTCS*, 2006.

[13] D. Galmiche, D. Mery, and D. Pym. The semantics of BI and resource tableaux. *Mathematical Structures in Computer Science*, 15:1033–1088, 2005.

[14] A. Gotsman, B. Cook, M. Parkinson, and V. Vafeiadis. Proving that non-blocking algorithms don't block. In *Proceedings of POPL-36*, pp. 16–28, 2009.

[15] S. Ishtiaq and P. W. O'Hearn. BI as an assertion language for mutable data structures. In *Proceedings of POPL-28*, 2001.

[16] M. Kanovich. The direct simulation of Minsky machines in linear logic. In *Advances in Linear Logic*, pp. 123–145, 1995.

[17] P. Lincoln, J. C. Mitchell, A. Scedrov, and N. Shankar. Decision problems for propositional linear logic. *Annals of Pure and Applied Logic*, 56(1–3):239–311, 1992.

[18] M. Minsky. *Computation: finite and infinite machines*, 1967.

[19] P. O'Hearn and D. J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, June 1999.

[20] M. Parkinson and G. Bierman. Separation logic, abstraction and inheritance. In *Proceedings of POPL-35*, 2008.

[21] M. Parkinson, R. Bornat, and C. Calcagno. Variables as resource in Hoare logics. In *Proc. LICS*, pp. 137–146, 2006.

[22] D. Pym. *The Semantics and Proof Theory of the Logic of Bunched Implications*. Applied Logic Series, Kluwer, 2002.

[23] J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of 17th LICS*, 2002.

[24] H. Yang, O. Lee, J. Berdine, C. Calcagno, B. Cook, D. Distefano, and P. O'Hearn. Scalable shape analysis for systems code. In *Proceedings of CAV*, 2008.