

# Undecidability of propositional separation logic

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

Separation logic has proven an effective formalism for the analysis of memory-manipulating programs. In this paper, we show that the purely propositional fragment of separation logic is undecidable.

The main novelty of our approach to the problem is that we focus on validity in specific, concrete memory models of practical interest (as opposed to validity in general classes of models). We prove the more subtle result that for *any* choice of concrete heap-like model of separation logic, validity in that model remains undecidable.

Besides its intrinsic technical interest, this result also provides new insights into the nature of decidable fragments of separation logic. In particular, we show that in order to obtain such decidable fragments, either the formula language must be severely restricted or valuations of propositional variables must be constrained to be finite.

In addition, we show that a number of propositional systems, based on bunched logic, that approximate separation logic are undecidable as well. In particular, this resolves the open problems of decidability for Boolean BI and Classical BI.

Moreover, we provide one of the simplest undecidable propositional systems currently known in the literature, called “Minimal Boolean BI”, by combining the (purely positive) implication-conjunction fragment of Boolean logic with the laws of multiplicative  $*$ -conjunction, its unit and its adjoint implication, originally provided by multiplicative intuitionistic linear logic. Each of these two components is individually decidable: the implication-conjunction fragment of Boolean logic is co-NP-complete, and multiplicative intuitionistic linear logic is NP-complete.

All of our undecidability results are obtained by means of a direct encoding of Minsky machines.

Categories and Subject Descriptors: F.3.1 [Logics and Meanings of Programs]: Specifying and Verifying and Reasoning about Programs—*Logics of programs; Assertions*; F.4.1 [Mathematical Logic and Formal Languages]: Mathematical Logic—*Model theory; Proof theory*

General Terms: Theory, Verification

Additional Key Words and Phrases: separation logic, undecidability, bunched logic

## ACM Reference Format:

Brotherston, J., and Kanovich, M. 2012. Undecidability of propositional separation logic and its neighbours. J. ACM V, N, Article A (January YYYY), 39 pages.  
DOI = 10.1145/0000000.0000000 <http://doi.acm.org/10.1145/0000000.0000000>

## 1. INTRODUCTION, MOTIVATIONS, SUMMARY

*Separation logic* has become well-established in the last decade as an effective formalism for reasoning about programs that manipulate memory (in the form of heaps, stacks, etc.) [Reynolds 2002; Ishtiaq and O’Hearn 2001; Calcagno et al. 2007]. Automated shape analysis tools based upon separation logic are capable of verifying properties of large industrial programs [Yang et al. 2008; Calcagno et al. 2011], and have been adapted to a variety of paradigms such as object-oriented programming [Parkinson and Bier-

---

Brotherston and Kanovich are supported by EPSRC.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).

© YYYY ACM 0004-5411/YYYY/01-ARTA \$10.00

DOI 10.1145/0000000.0000000 <http://doi.acm.org/10.1145/0000000.0000000>

man 2008; Distefano and Parkinson 2008; Gardner et al. 2012] and concurrent programming [Dodds et al. 2009; Gotsman et al. 2009].

Separation logic is usually based on a mathematical model of *heap partitioning*. In addition to the standard ‘additive’ Boolean connectives, which are read in the usual way, separation logic features certain ‘multiplicative’ connectives which are interpreted as operations in this model. The most important of these is the so-called *separating conjunction*  $*$ , which generally denotes a partial operator for composing heaps whose domains are disjoint:  $A_1 * A_2$  denotes the set of heaps which can be split into two disjoint heaps satisfying respectively  $A_1$  and  $A_2$ . The separating conjunction  $*$  comes along with its *unit*  $I$ , which denotes the empty heap, and its *adjoint implication*  $A_1 \multimap A_2$ , denoting those heaps whose extension with any heap satisfying  $A_1$ , satisfies  $A_2$ . Depending on the application, first-order quantifiers and inductively defined predicates may also be included in the logic.

Our main contribution in this paper is that, separation logic is *undecidable even at the purely propositional level and irrespective of the choice of underlying memory model*.

The novelty of our approach to the problem is that we focus on the logic for *concrete* memory models of practical interest, and prove that, whichever concrete heap-like model we choose, the validity of propositional separation logic formulas in that model is undecidable. The models we consider are taken from the literature on separation logic and its applications; the most common memory models used in practice are listed in Example 1.2 below.

As an immediate corollary of this strict result, we get undecidability of validity in various classes of such “separation models”, and undecidability of provability in several closely related propositional systems, including Boolean BI [Ishtiaq and O’Hearn 2001] and Classical BI [Brotherston and Calcagno 2009].

*Remark 1.1.* Validity in a fixed model of practical interest is a much more subtle problem than validity in general classes of models.

As usual, to demonstrate that a validity problem is undecidable, we aim to exhibit a reduction from the halting problem for a suitable class of machines. In our case, these are non-deterministic, 2-counter Minsky machines. Thus, given a formal system, we can construct a provable formula  $\mathcal{F}_{M,C}$  in its language which is intended to represent the termination of the Minsky machine  $M$  from the configuration  $C$ . By soundness of the system, provability of  $\mathcal{F}_{M,C}$  implies its validity in some class of models.

In order to prove the faithfulness of our encoding, we have to show the converse direction that if  $\mathcal{F}_{M,C}$  is valid in this class of models then  $M$  terminates from  $C$ . Now, traditionally, to show that a formula  $\mathcal{F}$  has a property  $Q$  given that  $\mathcal{F}$  is valid in a *class* of models, one constructs *some* model  $\mathcal{M}$  in the class such that validity of  $\mathcal{F}$  in this specially designed model  $\mathcal{M}$  implies  $Q$ .

However, when we consider the problem of validity in a concrete model  $\mathcal{M}_0$  *specified in advance*, we have no such freedom. Instead, we have to show  $Q$  given only that  $\mathcal{F}$  is valid in this fixed concrete model  $\mathcal{M}_0$ . The difficulty of this task is that there is no *ad hoc* connection between the given model  $\mathcal{M}_0$  and the “artificially” designed model  $\mathcal{M}$  one would normally seek to create.  $\square$

Since existing decidable fragments of separation logic are based on concrete models, an additional advantage of our approach is that our undecidability results for these models illuminate the restrictions on these fragments. In particular, it transpires that undecidability is closely connected to the trade-off between finite and infinite valuations of the atomic propositions (cf. Section 8).

*Example 1.2.* Here we list examples of commonly-used models of separation logic which employ a heap memory concept:

- (a) *Heap models* [Ahmed et al. 2003; Ishtiaq and O’Hearn 2001; Reynolds 2002], given by

$$(H, \circ, \{e\})$$

where  $H = L \multimap_{\text{fin}} RV$  is the set of *heaps*, i.e. finite partial functions from an infinite  $L$  to  $RV$ . The unit  $e$  is the function with the 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).

- (b) We will find it useful to consider a special case of the heap models (a) above: a very simplistic *RAM-domain model*

$$(\mathcal{D}, \circ, \{e_0\})$$

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  $\emptyset$ .

- (c) *Heap-with-permission models* [Bornat et al. 2005], given by

$$(H, \circ, \{e\})$$

with an underlying *permission algebra*

$$(P, \bullet, \mathbb{1})$$

which is a set  $P$  equipped with a partial commutative and associative operation  $\bullet$ , and a distinguished element  $\mathbb{1}$  (the “write permission”) such that  $\mathbb{1} \bullet \pi$  is undefined for all  $\pi \in P$ .

Then  $H = L \multimap_{\text{fin}} (RV \times P)$  is the set of *heaps-with-permissions*, and  $h_1 \circ h_2$  is again the union of  $h_1$  and  $h_2$  with disjoint domains. However, 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  $\ell$* . When  $h_1$  and  $h_2$  are compatible at all  $\ell$  from the intersection of their domains, then  $(h_1 \circ h_2)(\ell) = \langle v, \pi_1 \bullet \pi_2 \rangle$ , rather than being undefined.

- (d) *Stack-and-heap models*, as used e.g. to model Java memory [Parkinson et al. 2006], given by

$$(S \times H, \circ, E)$$

where  $H$  is a set of *heaps* or *heaps-with-permissions* as above and  $S = \mathbf{Var} \multimap_{\text{fin}} \mathbf{Val}$  is the set of *stacks*, partially mapping  $\mathbf{Var}$  to  $\mathbf{Val}$ . Here  $E$  consists of all pairs  $\langle s, e \rangle$  in which  $e$  is the empty heap, and

$$\langle s_1, h_1 \rangle \circ \langle s_2, h_2 \rangle = \langle s_1, h_1 \circ h_2 \rangle$$

if and only if  $s_1 = s_2$  and  $h_1 \circ h_2$  is defined in accordance with the previous items, and is undefined otherwise.

- (e) Models related to *Petri-net markings*, given by

$$(H, \circ, \{e_0\})$$

where for some  $K \subseteq \mathbb{N}$ ,  $H = L \multimap_{\text{fin}} K$  is the set of *markings*.

By taking  $K = \mathbb{N}$  we model states in traditional Petri nets, with  $\circ$  being the *total* operation of multiset union, i.e.,  $(h_1 \circ h_2)(\ell) = h_1(\ell) + h_2(\ell)$ .

By taking  $K = \{0, 1, \dots, k\}$  we model states in Petri nets with capacity  $k$ , where  $\circ$  is multiset union as above, but with  $(h_1 \circ h_2)(\ell)$  undefined when  $h_1(\ell) + h_2(\ell) > k$ .

- (f) Models related to *finite multisets*, (e.g., integer partitions), given by

$$(H, \circ, \{e_0\})$$

where  $H = L \multimap_{\text{fin}} \mathbb{N}$  is a set of *finite multisets*, with  $\circ$  being the *total* operation of multiset union as in the previous item. These models can be thought of as a certain kind of memory model, where every location  $\ell$  has an associated size or “number of blocks”  $h(\ell)$ .  $\square$

Seen as a proof system, separation logic invokes a first-order extension of the propositional *bunched logic* Boolean BI [Ishtiaq and O’Hearn 2001]. Bunched logics, originating in the “logic of bunched implications” BI [O’Hearn and Pym 1999], can be conceived of as substructural logics that combine a standard propositional logic with various “multiplicative” connectives originally provided by linear logic (cf. [Pym 2002; Brotherston 2011]). Their practical significance for computer science stems from their Kripke-style truth interpretation in which the “worlds” of the Kripke models are understood as *resources* [Ishtiaq and O’Hearn 2001; Pym et al. 2004; Brotherston and Calcagno 2010].

In proving our main undecidability results on separation logic, we also establish the undecidability of provability in several propositional systems based on bunched logic. In particular, these include the bunched logics Boolean BI [Ishtiaq and O’Hearn 2001] and Classical BI [Brotherston and Calcagno 2010]. (The decidability of Boolean BI, and to a lesser extent Classical BI, was widely considered an open problem for quite a long time.)

The remainder of this paper is structured as follows.

First, in Section 2, we give the formula language and semantics of propositional separation logic, then, in Section 3, present a number of propositional systems that arise naturally in developing towards an axiomatisation of separation logic.

An overview of our undecidability results and the proof methodology used to establish them is then given in Section 4. The proofs themselves occupy Sections 5–7.

In Section 5, we give a minimal version of Boolean BI, in which negation and falsum are disallowed; this “Minimal BBI” looks extremely simple, but is nonetheless undecidable.

In Section 6, we encode two-counter Minsky machines as sequents of Minimal BBI so that, whenever machine  $M$  terminates from configuration  $C$ , the corresponding sequent  $\mathcal{F}_{M,C}$  is provable in Minimal BBI. By soundness, the sequent  $\mathcal{F}_{M,C}$  is then valid in all separation models, including any from Example 1.2. Then, in Section 7, we show that whenever  $\mathcal{F}_{M,C}$  is valid in one of the *concrete* models in Example 1.2, the machine  $M$  must terminate from configuration  $C$ . Thus it follows that “*any property between provability in Minimal BBI and validity in a heap-like model is undecidable*” (cf. Figure 2 in Section 4).

In Section 8, we examine the limitations on decidable fragments of separation logic imposed by our undecidability results. Oddly enough, it happens that validity under all *finite valuations* for atomic propositions in a heap-like model does not imply general validity in that model. Consequently, at least in some models, decidable fragments of separation logic can be obtained by restricting valuations to be finite (cf. [Calcagno et al. 2001]).

In Section 9, we consider “dualising” separation models and related propositional systems based on Classical BI [Brotherston and Calcagno 2010]. Models of Classical BI differ from the models in Example 1.2 in that every element is required to have a unique dual; thus, for example, every Abelian group is a model of Classical BI (in which the dual of element is its group inverse), as are e.g. bit arrays (where the dual of an array is given by inverting each of its bits). Classical BI is a nonconservative extension of Boolean BI, and thus validity behaves quite differently. We show that our encoding of Minsky machines also yields undecidability of systems based on Classical BI, and undecidability of validity in associated classes of models.

This is an extended and revised version of a conference paper which appeared at LICS 2010 [Brotherston and Kanovich 2010]. Sections 5, 6 and 7 contain the details of our undecidability proofs, and are purely technical.

## 2. SYNTAX AND SEMANTICS OF PROPOSITIONAL SEPARATION LOGIC

Here we present the language of propositional separation logic and its interpretation in the usual class of ‘separation models’. Then we present a chain of proof systems, based on bunched logic, that partially axiomatise propositional separation logic.

The following definition of a *separation model* from [Calcagno et al. 2007] abstracts from the concrete heap-like models of separation logic employed in the literature (cf. Example 1.2):

*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 the equality of expressions  $\alpha = \beta$  means that either  $\alpha$  and  $\beta$  are both undefined, or  $\alpha$  and  $\beta$  are defined and equal. Cancellativity of  $\circ$  means that if  $h \circ h'$  is defined and  $h \circ h' = h \circ h''$  then  $h' = h''$ .

We extend  $\circ$  to a multiplication  $\cdot$  on subsets of  $H$  by

$$X \cdot Y =_{\text{def}} \{h \circ h' \mid h \in X, h' \in Y, \text{ and } h \circ h' \text{ is defined}\} \quad (1)$$

The set of *units*  $E$  is then a subset of  $H$  such that, for all  $h \in H$ ,

$$E \cdot \{h\} = \{h\}. \quad (2)$$

All the models from Example 1.2 are separation models. We allow a set of units  $E$  rather than a single unit  $e$  in Definition 2.1 in order to cover the whole spectrum of heap-like models (cf. the stack-and-heap models in Example 1.2(d)).

**PROPOSITION 2.2.** *For any  $h \in H$ , there is a unique  $e \in E$  such that  $h \circ e = h$ .*

**PROOF.** Assume  $h \circ e = h \circ e' = h$ . Then by cancellativity we get  $e = e'$ .  $\square$

**PROPOSITION 2.3.** *The set of units  $E$  in a separation model  $(H, \circ, E)$  forms a ‘unit matrix’. That is, for any  $e_i, e_j \in E$  we have:*

$$e_i \circ e_j = \begin{cases} e_i, & \text{if } e_i = e_j, \\ \text{undefined,} & \text{otherwise.} \end{cases}$$

*In particular, if  $\circ$  is total then  $E$  is forced to be a singleton  $\{e\}$ .*

**PROOF.** Let  $e_i, e_j \in E$ . Using (2) and commutativity of  $\circ$ , we have  $\{e_i\} \cdot E = \{e_i\}$  whence  $e_i \circ e_j$  is either  $e_i$  or undefined. But, by the same token,  $E \cdot \{e_j\} = \{e_j\}$  whence  $e_i \circ e_j$  is either  $e_j$  or undefined. Thus if  $e_i \neq e_j$  then  $e_i \circ e_j$  must be undefined. To see that  $e_i \circ e_i = e_i$ , observe that we must have  $e_i \circ e = e_i$  for some  $e \in E$  because  $e_i \in \{e_i\} \cdot E$ .  $\square$

*Definition 2.4.* 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  $h_1, h_2 \in H$ . (In fact, in this case  $h_1 = h_2$  because of the observation in Proposition 2.3 that  $h_1 \circ h_2$  must be undefined when  $h_1 \neq h_2$ .)

*Remark 2.5.* The memory models of theoretical/practical importance employed in the literature (see Example 1.2) all have indivisible units in the sense of Definition 2.4. This natural property can be summarised by the following slogan on ‘conservation of matter’:

“The empty memory cannot be split into non-empty pieces”.  $\square$

(However, we can easily construct a model whose units are divisible; e.g., the model  $(\mathbb{Z}, +, \{0\})$ , where  $+$  is addition on the integers  $\mathbb{Z}$ , is one such.)

*Definition 2.6.* *Formulas* are built from an infinite set of atomic propositions  $p$  and constants  $\top$ ,  $\perp$ , and  $\text{I}$  by a unary operator  $\neg$  and binary connectives  $\wedge$ ,  $\vee$ ,  $\rightarrow$ ,  $*$ , and  $\neg*$ .

For the sake of readability, we often write a formula of the form  $A \rightarrow B$  as the ‘sequent’  $A \vdash B$ .

By convention, negation  $\neg$  has the greatest precedence, followed by  $\wedge$ ,  $\vee$ , and  $*$ , with the implications  $\rightarrow$  and  $\neg*$  having lowest precedence.

*Definition 2.7.* A *valuation* for a separation model  $(H, \circ, E)$  is a function  $\rho$  that assigns to each atomic proposition  $p$  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  $A$ :

$$\begin{aligned}
h \models_\rho p &\Leftrightarrow h \in \rho(p) \\
h \models_\rho \top &\Leftrightarrow \text{always} \\
h \models_\rho \perp &\Leftrightarrow \text{never} \\
h \models_\rho \neg A &\Leftrightarrow h \not\models_\rho A \\
h \models_\rho A_1 \wedge A_2 &\Leftrightarrow h \models_\rho A_1 \text{ and } h \models_\rho A_2 \\
h \models_\rho A_1 \vee A_2 &\Leftrightarrow h \models_\rho A_1 \text{ or } h \models_\rho A_2 \\
h \models_\rho A_1 \rightarrow A_2 &\Leftrightarrow \text{if } h \models_\rho A_1 \text{ then } h \models_\rho A_2 \\
h \models_\rho I &\Leftrightarrow h \in E \\
h \models_\rho A_1 * A_2 &\Leftrightarrow \exists h_1, h_2. h = h_1 \circ h_2 \text{ and } h_1 \models_\rho A_1 \text{ and } h_2 \models_\rho A_2 \\
h \models_\rho A_1 \multimap A_2 &\Leftrightarrow \forall h'. \text{ if } (h \circ h' \text{ defined and } h' \models_\rho A_1) \text{ then } h \circ h' \models_\rho A_2
\end{aligned}$$

The intended meaning of any formula  $A$  under  $\rho$  is given by  $\llbracket A \rrbracket_\rho =_{\text{def}} \{h \mid h \models_\rho A\}$ .

LEMMA 2.8. *Given a separation model  $(H, \circ, E)$  and a valuation  $\rho$  for the model, we have the following identities:*

$$\llbracket p \rrbracket_\rho = \rho(p) \quad (3)$$

$$\llbracket \perp \rrbracket_\rho = \emptyset \quad (4)$$

$$\llbracket I \rrbracket_\rho = E \quad (5)$$

$$\llbracket A \wedge B \rrbracket_\rho = \llbracket A \rrbracket_\rho \cap \llbracket B \rrbracket_\rho \quad (6)$$

$$\llbracket A * B \rrbracket_\rho = \llbracket A \rrbracket_\rho \cdot \llbracket B \rrbracket_\rho \quad (7)$$

$$\llbracket A \rightarrow B \rrbracket_\rho = \text{largest } Z \subseteq H. \llbracket A \rrbracket_\rho \cap Z \subseteq \llbracket B \rrbracket_\rho \quad (8)$$

$$\llbracket A \multimap B \rrbracket_\rho = \text{largest } Z \subseteq H. \llbracket A \rrbracket_\rho \cdot Z \subseteq \llbracket B \rrbracket_\rho \quad (9)$$

PROOF. We just show the identities (7) and (9). In the case of (7), we have:

$$\begin{aligned}
\llbracket A * B \rrbracket_\rho &= \{h \mid \exists h_1, h_2. h = h_1 \circ h_2 \text{ and } h_1 \models_\rho A \text{ and } h_2 \models_\rho B\} \\
&= \{h_1 \circ h_2 \mid h_1 \models_\rho A \text{ and } h_2 \models_\rho B \text{ and } h_1 \circ h_2 \text{ defined}\} \\
&= \{h_1 \circ h_2 \mid h_1 \in \llbracket A \rrbracket_\rho \text{ and } h_2 \in \llbracket B \rrbracket_\rho \text{ and } h_1 \circ h_2 \text{ defined}\} \\
&= \llbracket A \rrbracket_\rho \cdot \llbracket B \rrbracket_\rho
\end{aligned}$$

To show (9), we first show that  $\llbracket A \multimap B \rrbracket_\rho$  satisfies the required inclusion, that is

$$\llbracket A \rrbracket_\rho \cdot \llbracket A \multimap B \rrbracket_\rho \subseteq \llbracket B \rrbracket_\rho$$

To see this, let  $h \in \llbracket A \rrbracket_\rho \cdot \llbracket A \multimap B \rrbracket_\rho$ . Then there are  $h_1, h_2 \in H$  with  $h = h_1 \circ h_2$  and  $h_1 \models_\rho A$  and  $h_2 \models_\rho A \multimap B$ . In particular, this means that for any  $h'$  such that  $h' \circ h_2$  is defined and  $h' \models_\rho A$ , we have  $h' \circ h_2 \models_\rho B$ . Since  $h_1 \circ h_2$  is defined, we have  $h_1 \circ h_2 \models_\rho B$ , i.e.  $h \models_\rho B$  as required.

To see that  $\llbracket A \multimap B \rrbracket_\rho$  is the *largest* set satisfying the required inclusion, we let  $Z \subseteq H$  satisfy  $\llbracket A \rrbracket_\rho \cdot Z \subseteq \llbracket B \rrbracket_\rho$ , and show that  $Z \subseteq \llbracket A \multimap B \rrbracket_\rho$ . This means showing that if  $h \in Z$  then  $h \models_\rho A \multimap B$ . Let  $h' \in H$  be such that  $h \circ h'$  is defined and  $h' \models_\rho A$ , i.e.  $h' \in \llbracket A \rrbracket_\rho$ . Since  $\llbracket A \rrbracket_\rho \cdot Z \subseteq \llbracket B \rrbracket_\rho$ , we have  $h \circ h' \in \llbracket B \rrbracket_\rho$ , i.e.  $h \circ h' \models_\rho B$  as required.  $\square$

*Definition 2.9.* A formula  $A$  is *valid* in a separation model  $(H, \circ, E)$  if for any valuation  $\rho$ , we have  $\llbracket A \rrbracket_\rho = H$ .

In particular, a sequent  $A \vdash B$  is valid in  $(H, \circ, E)$  if  $\llbracket A \rrbracket_\rho \subseteq \llbracket B \rrbracket_\rho$  for any valuation  $\rho$ .

In the next theorem we establish the exact correlation between seemingly different concepts - the *indivisibility of units* in the memory models and the *restricted \*-weakening* in formal systems, given by axioms of the form

$$I \wedge (A * B) \vdash A$$

THEOREM 2.10.

- (a) For any formulas  $A$  and  $B$ , the restricted \*-weakening principle of the form

$$I \wedge (A * B) \vdash A$$

is valid in all separation models  $(H, \circ, E)$  with indivisible units.

- (b) In addition to that, we show that the above correlation is exact, namely, the basic instance of the restricted \*-weakening principle (here  $p$  and  $q$  are atomic propositions)

$$I \wedge (p * q) \vdash p$$

is valid in a separation model  $(H, \circ, E)$  if and only if  $(H, \circ, E)$  has indivisible units.  $\square$

PROOF.

- (a) For the first part of the proof, let  $(H, \circ, E)$  be a separation model with indivisible units, let  $h \in H$  and suppose that  $h \models_{\rho} I \wedge (A * B)$ . Thus  $h \in E$  and there exist  $h_1, h_2 \in H$  with  $h = h_1 \circ h_2$  and  $h_1 \models_{\rho} A$  and  $h_2 \models_{\rho} B$ . As  $(H, \circ, E)$  has indivisible units, we have  $h_1 \in E$  and  $h_2 \in E$ , whence by Proposition 2.3 we have  $h_1 = h_2 = h$ . Thus  $h \models_{\rho} A$  and, as  $h$  was arbitrarily chosen,  $(I \wedge (A * B)) \vdash A$  is valid in this model as required.

- (b) For the second part, let  $(H', \circ', E)$  be a separation model whose set of units  $E$  is divisible, i.e. such that  $h' \circ' h'' = e \in E$  for some  $h' \notin E$ .

Let  $\rho$  be a valuation with  $\rho(p) = \{h'\}$  and  $\rho(q) = \{h''\}$ . Then we easily have that  $e \models_{\rho} I \wedge (p * q)$  but  $e \not\models_{\rho} p$ , so  $I \wedge (p * q) \vdash p$  is not valid in this model as required.  $\square$

The following proposition shows that separation models with total and non-total compositions  $\circ$  behave quite differently with respect to validity.

PROPOSITION 2.11. A sequent of the form

$$I \wedge (p * q \multimap \perp) \vdash (p \multimap \perp) \vee (q \multimap \perp) \quad (10)$$

is valid in a separation model  $(H, \circ, E)$  if and only if the operation  $\circ$  is total.

PROOF. In fact, the sequent (10) represents the following ‘natural’ law:

$$\llbracket p \rrbracket_{\rho} \cdot \llbracket q \rrbracket_{\rho} = \emptyset \text{ implies } (\llbracket p \rrbracket_{\rho} = \emptyset \text{ or } \llbracket q \rrbracket_{\rho} = \emptyset)$$

( $\Leftarrow$ ) Let  $(H, \circ, E)$  be a separation model with  $\circ$  total, and suppose  $h \models_{\rho} I \wedge (p * q \multimap \perp)$ . Then  $h \in E$  and  $h \models_{\rho} (p * q \multimap \perp)$ . Using Lemma 2.8, we have  $\llbracket p * q \rrbracket_{\rho} \subseteq \llbracket \perp \rrbracket_{\rho} = \emptyset$ , and thus  $\llbracket p \rrbracket_{\rho} \cdot \llbracket q \rrbracket_{\rho} = \llbracket p * q \rrbracket_{\rho} = \emptyset$ .

Since  $\circ$  is total,  $(\llbracket p \rrbracket_{\rho} \cdot \llbracket q \rrbracket_{\rho} = \emptyset)$  implies that either  $(\llbracket p \rrbracket_{\rho} = \emptyset)$  or  $(\llbracket q \rrbracket_{\rho} = \emptyset)$ . Consequently, either  $h \models_{\rho} (p \multimap \perp)$  or  $h \models_{\rho} (q \multimap \perp)$ , i.e.  $h \models_{\rho} (p \multimap \perp) \vee (q \multimap \perp)$  as required.

( $\Rightarrow$ ) Let  $(H, \circ, E)$  be a separation model in which  $\circ$  is non-total, that is for some  $h'$  and  $h''$ , their product  $h' \circ h''$  is undefined. Let  $\rho$  be a valuation with  $\rho(p) = \{h'\}$  and  $\rho(q) = \{h''\}$ . Then  $\llbracket p \rrbracket_{\rho} \cdot \llbracket q \rrbracket_{\rho} = \emptyset$  by construction. Thus, using Lemma 2.8, we have

$$\begin{aligned} E \cdot \llbracket p * q \rrbracket_{\rho} &= \llbracket p * q \rrbracket_{\rho} \\ &= \llbracket p \rrbracket_{\rho} \cdot \llbracket q \rrbracket_{\rho} \\ &= \emptyset \\ &= \llbracket \perp \rrbracket_{\rho} \end{aligned}$$



Thus in particular  $E \cdot \llbracket p * q \rrbracket_\rho \subseteq \llbracket \perp \rrbracket_\rho$  and, by Lemma 2.8, we have  $E \subseteq \llbracket p * q \multimap \perp \rrbracket_\rho$ , which means that  $\llbracket I \wedge (p * q \multimap \perp) \rrbracket_\rho \neq \emptyset$ . On the other hand, by construction both  $\llbracket p \multimap \perp \rrbracket_\rho = \emptyset$  and  $\llbracket q \multimap \perp \rrbracket_\rho = \emptyset$ , i.e.  $\llbracket (p \multimap \perp) \vee (q \multimap \perp) \rrbracket_\rho = \emptyset$ . We conclude that our sequent is not valid in this model as required.  $\square$

### 3. FORMAL SYSTEMS FOR SEPARATION LOGIC

Core proof systems for propositional separation logic are provided by *bunched logic*, a class of substructural logics pioneered by O’Hearn and Pym. Specifically, the propositional proof theory of separation logic is usually considered to be given by the bunched logic Boolean BI, although other choices are also possible. Every separation model can be seen as a model of Boolean BI [Ishtiaq and O’Hearn 2001; Galmiche and Larchey-Wendling 2006].

$(\alpha_1) \quad A * B \vdash B * A$	$(\alpha_4) \quad A * I \vdash A$
$(\alpha_2) \quad A * (B * C) \vdash (A * B) * C$	$(\alpha_5) \quad A \vdash A * I$
$(\alpha_3) \quad A * (A \multimap B) \vdash B$	
$\frac{A \vdash B}{A * C \vdash B * C} (\alpha_6)$	$\frac{(A * B) \vdash C}{A \vdash (B \multimap C)} (\alpha_7)$

Fig. 1. Axioms and rules for  $*$ ,  $\multimap$  and  $I$ .

*Definition 3.1.* We consider the following chain of logical systems

$$\text{BI} \subseteq \text{BBI} \subseteq \text{BBI} + \text{eW} \subseteq \text{BBI} + \text{W}$$

each defined as follows:

- BI, a.k.a. the *logic of bunched implications* (cf. [O’Hearn and Pym 1999; Pym 2002; Galmiche et al. 2005]) is given by:

(A) all instances of intuitionistically valid formulas and inference rules, and

(B) the axioms and inference rules for  $*$ ,  $\multimap$  and  $I$  given in Figure 1, originally provided by multiplicative intuitionistic linear logic, MILL [Girard and Lafont 1987; Benton et al. 1993].

$\square$

- *Boolean BI*, a.k.a. BBI (cf. [Ishtiaq and O’Hearn 2001; Galmiche and Larchey-Wendling 2006]) is obtained from BI by expanding (A) above to include all instances of *classically* valid propositional formulas and inference rules.

- As we shall see (Lemma 5.3), the *restricted  $*$ -contraction*

$$I \wedge A \vdash A * A$$

holds in BBI, whereas the analogous *restricted  $*$ -weakening*

$$I \wedge (A * B) \vdash A$$



does not. However, the restricted  $*$ -weakening exactly expresses the indivisibility of units in a separation model (see Theorem 2.10). Thus we introduce the system BBI+eW by enriching BBI with all new axioms of the form  $I \wedge (A * B) \vdash A$ .

- Having considered restricted  $*$ -weakening, it is also natural to consider BBI+W, obtained by enriching BBI with the *unrestricted  $*$ -weakening* - that is, axioms of the form

$$A * B \vdash A$$

(or, equivalently,  $B \vdash I$ ). □

**PROPOSITION 3.2 (SOUNDNESS OF BBI).** *If  $A$  is provable in BBI then  $A$  is valid in all separation models.*

**PROOF.** We just need to show that validity in any separation model is preserved by the axioms and rules of classical propositional logic and by the axioms and rules given in Figure 1, which is an easy exercise. □

In many cases the connection between provability in the systems in Definition 3.1 and validity in the classes of corresponding separation models is not exact. E.g., BBI is not complete even for validity in the class of all partial commutative monoids [Larchey-Wendling and Galmiche 2010]. No complete natural axiomatisation is currently known for validity with respect to the class of all separation models (or an interesting subclass thereof).

**COROLLARY 3.3 (SOUNDNESS OF BBI+eW).** *If  $A$  is provable in BBI+eW then  $A$  is valid in all separation models with indivisible units.*

**PROOF.** Immediate from Proposition 3.2 and Theorem 2.10. □

**COROLLARY 3.4.** *Interpreting  $\subset$  as strict inclusion between the sets of sequents provable in each system, we have:*

$$\text{BI} \subset \text{BBI} \subset \text{BBI+eW} \subset \text{BBI+W}$$

**PROOF.** First, note that the non-strict inclusions  $\text{BI} \subseteq \text{BBI} \subseteq \text{BBI+eW} \subseteq \text{BBI+W}$  hold easily by the construction of the systems in Definition 3.1. Thus we just need to show  $\text{BBI+W} \not\subseteq \text{BBI+eW} \not\subseteq \text{BBI} \not\subseteq \text{BI}$ .

First,  $\text{BBI} \not\subseteq \text{BI}$  holds because BI is conservative over intuitionistic logic [Pym 2002] and thus, e.g., the classical tautology  $(p \rightarrow \perp) \rightarrow \perp \vdash p$  is invalid in BI, whereas it is valid in BBI.

$\text{BBI+eW} \not\subseteq \text{BBI}$  holds because, by Theorem 2.10, the instance  $(I \wedge (p * q)) \vdash p$  of BBI+eW's restricted  $*$ -weakening axiom is not valid in all separation models, and hence not provable in BBI by Proposition 3.2.

Finally,  $\text{BBI+W} \not\subseteq \text{BBI+eW}$  holds because the instance  $p \vdash I$  of BBI+W's unrestricted  $*$ -weakening axiom is not valid in all separation models with indivisible units, and hence not provable in BBI+eW by Corollary 3.3. To see this, let  $(H, \circ, E)$  be a separation model with indivisible units, and let  $\rho(p) = \{h\}$  for some  $h \notin E$ . Then, easily,  $h \models_\rho p$  but  $h \not\models_\rho I$ . □

One of the important features of separation logic is that the  $*$ -contraction,  $A \vdash A * A$ , is not generally valid, and hence not provable in BBI by Proposition 3.2. Surprisingly, however, BBI does enjoy the *restricted  $*$ -contraction* (see Lemma 5.3 in Section 5)

$$I \wedge A \vdash A * A$$

which holds only at the level of the multiplicative unit  $I$ .

*Remark 3.5.* We note that the following instance of restricted  $*$ -contraction

$$I \wedge p \vdash p * p$$

is *not* valid in ordinary intuitionistic BI. This can most easily be seen from the elementary Kripke semantics of BI in [Galmiche et al. 2005], whereby models are ordered partial commutative monoids  $(M, \circ, e, \sqsubseteq)$ , and we have the following clauses for satisfaction of formulas by  $m \in M$  and valuation  $\rho$ :

$$\begin{aligned} m \models_{\rho} I &\Leftrightarrow e \sqsubseteq m \\ m \models_{\rho} A \wedge B &\Leftrightarrow m \models_{\rho} A \text{ and } m \models_{\rho} B \\ m \models_{\rho} A * B &\Leftrightarrow \exists n, n' \in M. n \circ n' \leq m \text{ and } n \models_{\rho} A \text{ and } n \models_{\rho} B \end{aligned}$$

Now consider the 2-element BI-model

$$(\{e, b\}, \circ, e, \sqsubseteq)$$

where  $\circ$  is given by  $e \circ e = b \circ b = e$  and  $e \circ b = b \circ e = b$ , and  $\sqsubseteq$  is given by  $e \sqsubseteq b$ . Let  $\rho$  be a valuation with  $\rho(p) = \{b\}$ . Then, easily,  $b \models_{\rho} I \wedge p$  but  $b \not\models_{\rho} p * p$ , i.e.  $I \wedge p \vdash p * p$  is not valid in this model.<sup>1</sup>  $\square$

In the case of BBI+eW, its restricted \*-weakening with the restricted \*-contraction of Lemma 5.3 induces a collapse of  $\wedge$  and  $*$  at the level of the unit  $I$ , as follows:

COROLLARY 3.6. *The following equivalences hold in BBI+eW:*

$$I \wedge (A * B) \equiv I \wedge A \wedge B \equiv (I \wedge A) * (I \wedge B)$$

where the equivalence  $F \equiv G$  means that both  $F \vdash G$  and  $G \vdash F$  are provable.

PROOF. First,  $I \wedge (A * B) \vdash I \wedge A \wedge B$  is derivable using the restricted \*-weakening. Second, using the restricted \*-contraction and the usual contraction for  $\wedge$  we can derive:

$$I \wedge A \wedge B \vdash (I \wedge A \wedge B) * (I \wedge A \wedge B)$$

whence  $I \wedge A \wedge B \vdash (I \wedge A) * (I \wedge B)$  follows from weakening for  $\wedge$ . Finally, we can derive  $(I \wedge A) * (I \wedge B) \vdash I \wedge (A * B)$  using the equivalence  $I * C \equiv C$  (given by axioms  $(\alpha_4)$  and  $(\alpha_5)$ ) and weakening for  $\wedge$ . The required equivalences then follow by transitivity of  $\vdash$ .  $\square$

PROPOSITION 3.7. *BBI+W collapses into ordinary classical logic.*

PROOF. Trivially,  $I \wedge A \vdash A$  is provable in BBI+W. Using the equivalence  $I * A \equiv A$  and contraction for  $\wedge$  we also have  $A \vdash (I * A) \wedge A$  provable. Using the unrestricted \*-weakening of BBI+W we obtain  $A \vdash I \wedge A$ , and thus  $I \wedge A \equiv A$  holds in BBI+W. Using this equivalence together with Corollary 3.6, we have:

$$A * B \equiv I \wedge (A * B) \equiv I \wedge A \wedge B \equiv A \wedge B$$

which also guarantees that  $A \multimap B \equiv A \rightarrow B$ . Finally, using the equivalence  $I \wedge A \equiv A$  we have  $I \equiv I \wedge \top \equiv \top$ . Thus all the multiplicative connectives of BBI+W collapse into their classical additive equivalents.  $\square$

*Remark 3.8.* Both ends of the chain of logics  $BI \subset BBI \subset BBI+eW \subset BBI+W$  given by Corollary 3.4 are in fact *decidable*. BI was shown decidable in [Galmiche et al. 2005], and BBI+W is decidable because, by Proposition 3.7, it collapses into ordinary classical logic. Thus, technically speaking, it is relatively surprising that — as we shall see in the next section — the intermediate systems BBI and BBI+eW are both in fact *undecidable*.  $\square$

<sup>1</sup>The key difference compared to validity in separation models is that, in the satisfaction relation for BBI in Definition 2.7,  $b \models_{\rho} I \wedge p$  forces  $e = b$ , while in the satisfaction relation for BI above it only forces  $e \sqsubseteq b$ .

#### 4. UNDECIDABILITY OF PROPOSITIONAL SEPARATION LOGIC

In this section, we give a high-level outline of our undecidability results and their proof, the details of which occupy most of the remainder of the paper.

Figure 2 shows the overall development of our proof. As is typical, we show that a property  $Q$  is undecidable by showing that a problem already known to be undecidable reduces to the problem of deciding  $Q$ . In our case, the undecidable problem to be encoded is the termination of a two-counter, non-deterministic Minsky machine  $M$  from an arbitrary configuration  $C$ , shown as the top-centre node in Figure 2.

The proof strategy then goes as follows.

We encode  $M$  and  $C$  as a formula  $\mathcal{F}_{M,C}$  such that termination of  $M$  from  $C$  implies that  $\mathcal{F}_{M,C}$  is provable in BBI (Theorem 6.8). In fact, we can obtain this result even for a *minimal* version of BBI whose Boolean connectives are restricted to  $\wedge$  and  $\rightarrow$ : in particular, negation  $\neg$  and falsum  $\perp$  are disallowed. We present Minimal BBI in Section 5, and the encoding of Minsky machines in Section 6.

By construction of the proof systems in Definition 3.1, provability of  $\mathcal{F}_{M,C}$  in Minimal BBI implies its provability in BBI and BBI+eW. By our soundness results (Proposition 3.2 and Corollary 3.3) this implies validity of  $\mathcal{F}_{M,C}$  in the class of all separation models, and in the class of all separation models with indivisible units, respectively. Specifically,  $\mathcal{F}_{M,C}$  must be valid in *any particular* model chosen from those in Example 1.2.

Finally, we can complete the circle of implications — and thus the reduction from the halting problem for Minsky machines — by establishing that validity of  $\mathcal{F}_{M,C}$  in *any particular* such model implies that the machine  $M$  terminates from configuration  $C$ . This is established by Theorem 7.14.

As a consequence, *every property of formulas between provability in Minimal BBI and validity in a model from Example 1.2 is undecidable.*

The right-hand side of Figure 2 essentially repeats this proof structure, but for *Classical* BI (CBI) and its models, which obey stronger properties than BBI and its separation models. Classical BI, its models and the corresponding undecidability results are presented in Section 9.

**COROLLARY 4.1.** *The following properties of formulas are undecidable, even when restricted to the language  $(\wedge, \rightarrow, \text{I}, *, -*)$  of Minimal BBI:*

- (a) *provability in Minimal BBI;*
- (b) *provability in BBI;*
- (c) *provability in BBI+eW;*
- (d) *validity in the class of all separation models;*
- (e) *validity in the class of all separation models with indivisible units;*
- (f) *validity in the class of all total separation models;*
- (g) *validity in the class of all total separation models with indivisible units;*
- (h) *validity in one of the concrete models in Example 1.2, for arbitrarily chosen locations  $L$ , values  $RV$ , stacks  $S$ , and permission algebra  $P$  (note  $L$  must be infinite).*

**PROOF.** The termination problem for Minsky machines, which is undecidable [Minsky 1967], reduces to each of the problems above by the diagram in Figure 2.  $\square$

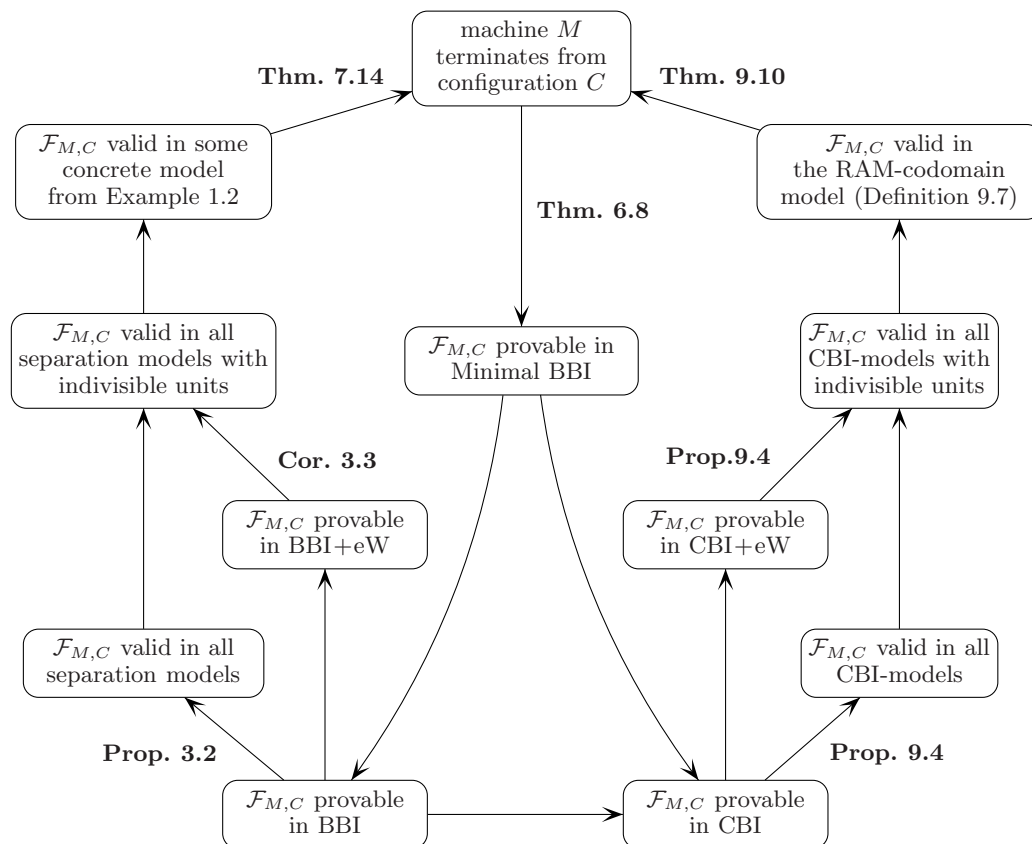


Fig. 2. Diagrammatic proof of undecidability. The arrows are implications, and  $\mathcal{F}_{M,C}$  is a sequent built from machine  $M$  and configuration  $C$ . The problems at each node are all undecidable.

**COROLLARY 4.2.** *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.  $\square$

## 5. MINIMAL BOOLEAN BI

One might be tempted to think that the undecidability of BBI, in particular, is mainly an artifact of Boolean negation, which is its most visible point of difference to standard (intuitionistic) BI. In fact, this is not the case. In this section we introduce a minimal positive fragment of BBI, called *Minimal* BBI, in which the formula connectives are restricted to the minimal set of  $\wedge$ ,  $\rightarrow$ ,  $I$ ,  $*$  and  $-*$ .

Minimal BBI is given by the minimal axioms and rules for  $*$ ,  $-*$  and  $I$  (originally provided by linear logic) in Figure 1 together with those for  $\rightarrow$  and  $\wedge$  in Figure 3.

*Remark 5.1.* As foreshadowed in the previous section, Minimal BBI, even though it looks extremely simple, is still *undecidable*, notwithstanding the fact that both its components are decidable: the implication-conjunction fragment of Boolean logic is co-NP-complete, and multiplicative intuitionistic linear logic is NP-complete [Kanovich 1992].

Oddly enough, removing *Peirce's law* from Figure 3 results in (a restricted version of) standard intuitionistic BI, which is decidable. This is quite counterintuitive, since by removing Peirce's law we replace the simple Boolean logic component in Figure 3 with the implication-conjunction fragment of intuitionistic logic, the complexity of which is higher (it is PSPACE-complete).  $\square$

$(\alpha_1) \quad A * B \vdash B * A$	$(\alpha_4) \quad A * I \vdash A$
$(\alpha_2) \quad A * (B * C) \vdash (A * B) * C$	$(\alpha_5) \quad A \vdash A * I$
$(\alpha_3) \quad A * (A \multimap B) \vdash B$	
$\frac{A \vdash B}{A * C \vdash B * C} (\alpha_6)$	$\frac{(A * B) \vdash C}{A \vdash (B \multimap C)} (\alpha_7)$

(a) Axioms and rules for  $*$ ,  $\multimap$  and  $I$ .

$(\beta_1) \quad A \vdash (B \rightarrow A)$	$(\beta_4) \quad A \vdash (B \rightarrow (A \wedge B))$
$(\beta_2) \quad (A \rightarrow (B \rightarrow C)) \vdash ((A \rightarrow B) \rightarrow (A \rightarrow C))$	$(\beta_5) \quad A \wedge B \vdash A$
$(\beta_3) \quad ((A \rightarrow B) \rightarrow A) \vdash A \quad (\text{Peirce's law})$	$(\beta_6) \quad A \wedge B \vdash B$
$\frac{A \quad A \vdash B}{B} (\text{MP})$	$\frac{A \wedge B \vdash C}{A \vdash (B \rightarrow C)} (\beta_7)$

(b) Axioms and rules for  $\rightarrow$  and  $\wedge$ .Fig. 3. A minimal set of axioms and rules for  $\rightarrow$ ,  $\wedge$ ,  $*$ ,  $\multimap$  and  $I$ , which axiomatises Minimal Boolean BI.

Note that in Figure 3 we must include the axioms and rules for  $\wedge$ , since  $\wedge$  itself is not definable exclusively in terms of  $\rightarrow$ . Also, neither  $\neg$  nor  $\perp$  is expressible by  $\wedge$  and  $\rightarrow$  alone. As a consequence, it is necessary to establish that various expected principles of classical logic, present in full BBI by construction, still hold in Minimal BBI.

**PROPOSITION 5.2.** *The following proof rules are derivable in Minimal BBI, where we write  $A \vee B$  as an abbreviation for  $((B \rightarrow A) \rightarrow A)$ :*

$\frac{}{A \vdash A} (\text{Id})$	$\frac{A \vdash B \quad B \vdash C}{A \vdash C} (\text{Tr})$	$\frac{A \vdash B \quad A \vdash C}{A \vdash B \wedge C} (\wedge)$
$\frac{A \vdash (B \rightarrow C)}{A \wedge B \vdash C} (\rightarrow)$	$\frac{A \vdash (B \multimap C)}{A * B \vdash C} (\multimap)$	$\frac{B \vdash C}{I \vdash (B \multimap C)} (I)$
$\frac{A \vdash B}{B \rightarrow C \vdash A \rightarrow C} (\text{MT})$	$\frac{A \vdash C \quad B \vdash C}{A \vee B \vdash C} (\vee)$	$\frac{A \vdash C \quad (A \rightarrow B) \vdash C}{C} (\text{EM})$

Moreover, the rules  $(\rightarrow)$ ,  $(\multimap)$  and (I) are all reversible, i.e. their premise and conclusion are interchangeable.

PROOF. We show how to derive each proof rule directly. In the following, recall that the sequent  $A \vdash B$  is merely another notation for the formula  $A \rightarrow B$ .

(Id). By instantiating the axioms  $(\beta_1)$  and  $(\beta_2)$  we obtain, respectively,

$$A \vdash ((A \rightarrow A) \rightarrow A)$$

$$(A \rightarrow ((A \rightarrow A) \rightarrow A)) \vdash ((A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A))$$

Thus by (MP) we obtain  $(A \rightarrow (A \rightarrow A)) \vdash (A \rightarrow A)$ . Thus, since  $A \vdash (A \rightarrow A)$  is an instance of  $(\beta_1)$ , we obtain  $A \vdash A$  by (MP).

(Tr). We have  $B \vdash C$  provable by assumption and  $(B \rightarrow C) \vdash (A \rightarrow (B \rightarrow C))$  an instance of  $(\beta_1)$ . Thus by (MP) we obtain  $A \rightarrow (B \rightarrow C)$ . Now by instantiating the axiom  $(\beta_2)$  we obtain:

$$(A \rightarrow (B \rightarrow C)) \vdash ((A \rightarrow B) \rightarrow (A \rightarrow C))$$

whence by (MP) we obtain  $(A \rightarrow B) \vdash (A \rightarrow C)$ . Then, since  $A \vdash B$  is provable by assumption we obtain  $A \vdash C$  by (MP) again as required.

( $\wedge$ ). We have  $A \vdash B$  by assumption and  $B \vdash (C \rightarrow B \wedge C)$  an instance of  $(\beta_4)$ . Thus by (Tr) we obtain  $A \vdash (C \rightarrow B \wedge C)$ . Now by instantiating  $(\beta_2)$  we have:

$$(A \rightarrow (C \rightarrow B \wedge C)) \vdash ((A \rightarrow C) \rightarrow (A \rightarrow B \wedge C))$$

Hence by (MP) we obtain  $(A \rightarrow C) \vdash (A \rightarrow B \wedge C)$ . Since we have  $A \vdash C$  by assumption we obtain  $A \vdash B \wedge C$  by (MP) as required.

( $\rightarrow$ ). The bottom-to-top direction is immediate by  $(\beta_7)$ . For the top-to-bottom direction, note that we have  $A \wedge B \vdash A$  an instance of  $(\beta_5)$  and  $A \vdash (B \rightarrow C)$  provable by assumption. Thus by (Tr) we obtain  $A \wedge B \vdash (B \rightarrow C)$ . Now by instantiating  $(\beta_2)$  we have:

$$(A \wedge B \rightarrow (B \rightarrow C)) \vdash ((A \wedge B \rightarrow B) \rightarrow (A \wedge B \rightarrow C))$$

Hence by (MP) we obtain  $(A \wedge B \rightarrow B) \vdash (A \wedge B \rightarrow C)$ . Since  $A \wedge B \vdash B$  is an instance of  $(\beta_6)$ , we can apply (MP) again to obtain  $A \wedge B \vdash C$  as required.

( $\multimap$ ). The bottom-to-top direction is immediate by  $(\alpha_7)$ . For the top-to-bottom direction, since  $A \vdash (B \multimap C)$  is provable by assumption we have  $A * B \vdash (B \multimap C) * B$  by the rule  $(\alpha_6)$ . Since  $(B \multimap C) * B \vdash B * (B \multimap C)$  is an instance of  $(\alpha_1)$  we then obtain  $A * B \vdash B * (B \multimap C)$  by (Tr). Then since  $B * (B \multimap C) \vdash C$  is an instance of  $(\alpha_3)$  we can apply (Tr) again to obtain  $A * B \vdash C$  as required.

(I). For the top-to-bottom direction, we have  $B * I \vdash B$  an instance of  $(\alpha_4)$  and  $I * B \vdash B * I$  an instance of  $(\alpha_1)$  whence  $I * B \vdash B$  is provable by (Tr). Since  $B \vdash C$  is provable by assumption, we obtain  $I * B \vdash C$  by (Tr), whence  $I \vdash (B \multimap C)$  is provable using the rule  $(\multimap)$  above. The reverse direction is similar.

(MT). (This rule is named for being “modus tollens-like”.) First,  $(B \rightarrow C) \vdash (B \rightarrow C)$  is derivable using (Id). Thus by the rule  $(\rightarrow)$  together with (Tr) and commutativity of  $\wedge$  (easily derivable from the rule  $(\wedge)$ ), we obtain  $B \vdash ((B \rightarrow C) \rightarrow C)$ . Since we have  $A \vdash B$  by assumption, we obtain  $A \vdash ((B \rightarrow C) \rightarrow C)$  by (Tr). Thus by applying

the deduction theorem again, together with commutativity of  $\wedge$  and (Tr), we have  $(B \rightarrow C) \vdash (A \rightarrow C)$  as required.

( $\vee$ ). Since we have  $B \vdash C$  by assumption, we can obtain by applying the rule (MT) twice

$$((B \rightarrow A) \rightarrow A) \vdash ((C \rightarrow A) \rightarrow A)$$

By applying the rule ( $\rightarrow$ ) we then obtain  $(B \rightarrow A) \rightarrow A \wedge (C \rightarrow A) \vdash A$ . Since  $A \vdash C$  is provable by assumption, we obtain  $(B \rightarrow A) \rightarrow A \wedge (C \rightarrow A) \vdash C$  by applying (Tr). By applying ( $\rightarrow$ ) again, we have

$$((B \rightarrow A) \rightarrow A) \vdash ((C \rightarrow A) \rightarrow C)$$

Now we have  $((C \rightarrow A) \rightarrow C) \vdash C$  an instance of Peirce's law ( $\beta_3$ ). Thus we obtain  $((B \rightarrow A) \rightarrow A) \vdash C$  by (Tr), which is equal to  $A \vee B \vdash C$ .

(EM). (This rule is named for being “excluded middle-like”.) We have by assumption  $A \vdash C$  and  $(A \rightarrow B) \vdash C$ , so we obtain  $A \vee (A \rightarrow B) \vdash C$  by applying the rule ( $\vee$ ). By definition,  $A \vee (A \rightarrow B)$  is equal to  $((A \rightarrow B) \rightarrow A) \vdash A$ , which is an instance of Peirce's law ( $\alpha_3$ ). Thus we obtain  $C$  by (MP).

□

We remark that the rules ( $\wedge$ ) and ( $\vee$ ) given by Proposition 5.2 immediately imply the associativity and commutativity of  $\wedge$  and  $\vee$ , whereas associativity and commutativity of  $*$  is directly expressed by axioms ( $\alpha_1$ ) and ( $\alpha_2$ ) of Minimal BBI. From now on, when we work in Minimal BBI, we implicitly treat  $\wedge$ ,  $\vee$  and  $*$  as being commutative and associative.

The following lemma is of utmost importance for our undecidability results.

LEMMA 5.3. *The restricted  $*$ -contraction, given by a sequent of the form,*

$$I \wedge A \vdash A * A \tag{11}$$

*is provable in Minimal BBI.*

PROOF. We make free use of the derived rules of Minimal BBI given by Proposition 5.2. According to the reversible version of the rule (I), the sequent (11) follows from

$$I \vdash (I \wedge A) \multimap (A * A)$$

which is notation for the formula

$$I \rightarrow ((I \wedge A) \multimap (A * A)) \tag{12}$$

Using the “excluded middle” rule (EM) in Proposition 5.2, we can derive the formula (12) from the following two sequents:

$$A \vdash I \rightarrow ((I \wedge A) \multimap (A * A)) \tag{13}$$

and

$$A \rightarrow (A * A) \vdash I \rightarrow ((I \wedge A) \multimap (A * A)) \tag{14}$$

We proceed by demonstrating how to derive each of (13) and (14).

- We derive the sequent (13) as follows. First, using the axioms ( $\beta_5$ ) and ( $\beta_6$ ) for  $\wedge$ -weakening, the rule ( $\alpha_6$ ) for  $*$  and the transitivity rule (Tr) we can easily derive

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

Using the derived rule ( $\multimap$ ), we obtain:

$$(I \wedge A) \vdash ((I \wedge A) \multimap (A * A))$$



and, by means of the derived rule ( $\rightarrow$ ), we conclude with the desired sequent (13).

- We derive the sequent (14) in the following way. First, using  $\wedge$ -weakening and the axiom  $(B * I) \vdash B$ , we can derive each of the following

$$(I \wedge (A \rightarrow (A * A))) * (I \wedge A) \vdash I * (I \wedge A)$$

and

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

whence by transitivity (Tr) we obtain:

$$(I \wedge (A \rightarrow (A * A))) * (I \wedge A) \vdash A \quad (15)$$

Next, using the same principles, we can derive

$$(I \wedge (A \rightarrow (A * A))) * (I \wedge A) \vdash (I \wedge (A \rightarrow (A * A))) * I,$$

and

$$(I \wedge (A \rightarrow (A * A))) * I \vdash (A \rightarrow (A * A))$$

resulting by transitivity (Tr) in

$$(I \wedge (A \rightarrow (A * A))) * (I \wedge A) \vdash (A \rightarrow (A * A)) \quad (16)$$

Using the axiom  $(\beta_2)$  from Figure 3 and the derived rules  $(\wedge)$ ,  $(\rightarrow)$  and (Tr) it is straightforward to derive from (15) and (16):

$$(I \wedge (A \rightarrow (A * A))) * (I \wedge A) \vdash A * A$$

Hence by applying the derived rules  $(-*)$  we obtain

$$(I \wedge (A \rightarrow (A * A))) \vdash (I \wedge A) \multimap (A * A)$$

and, by means of the derived rules  $(\rightarrow)$ , we conclude with the desired sequent (14).  $\square$

The proof principles of Proposition 5.2 and Lemma 5.3 are the main ones that are needed for correctness of our encoding of Minsky machines in the next section.

## 6. FROM COMPUTATIONS TO MINIMAL BBI PROOFS

In this section we give our encoding of terminating computations of (non-deterministic) two-counter Minsky machines as provable sequents of Minimal BBI. That is to say, given a Minsky machine  $M$  and initial configuration  $C$  our aim is to give a sequent  $\mathcal{F}_{M,C}$  such that  $M$  terminates from  $C$  just in case  $\mathcal{F}_{M,C}$  is provable in Minimal BBI (as per the overarching proof strategy in Figure 2 of Section 4).

Here, we give the construction of  $\mathcal{F}_{M,C}$  and show that it is provable in Minimal BBI whenever  $M$  terminates on  $C$ . The converse direction will be established in Section 7.

*Definition 6.1.* A non-deterministic, two-counter *Minsky machine*  $M$  [Minsky 1967] with non-negative counters  $c_1, c_2$  is given by a finite set of labelled *instructions* of the form:

$$\begin{array}{ll} \text{“increment } c_k \text{ by 1”} & L_i: c_k ++; \mathbf{goto} L_j; \\ \text{“decrement } c_k \text{ by 1”} & L_i: c_k --; \mathbf{goto} L_j; \\ \text{“zero-test on } c_k \text{”} & L_i: \mathbf{if } c_k = 0 \mathbf{ goto } L_j; \\ \text{“goto”} & L_i: \mathbf{goto} L_j; \end{array} \quad (17)$$

where  $k \in \{1, 2\}$ ,  $i \geq 1$  and  $j \geq 0$ .

$$\begin{array}{c}
\frac{(L_i: c_1++; \mathbf{goto} L_j;) \in M}{\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1 + 1, n_2 \rangle} \qquad \frac{(L_i: c_2++; \mathbf{goto} L_j;) \in M}{\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 + 1 \rangle} \\
\\
\frac{(L_i: c_1--; \mathbf{goto} L_j;) \in M}{\langle L_i, n_1 + 1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle} \qquad \frac{(L_i: c_2--; \mathbf{goto} L_j;) \in M}{\langle L_i, n_1, n_2 + 1 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle} \\
\\
\frac{(L_i: \mathbf{if} c_1 = 0 \mathbf{goto} L_j;) \in M}{\langle L_i, 0, n_2 \rangle \rightsquigarrow_M \langle L_j, 0, n_2 \rangle} \qquad \frac{(L_i: \mathbf{if} c_2 = 0 \mathbf{goto} L_j;) \in M}{\langle L_i, n_1, 0 \rangle \rightsquigarrow_M \langle L_j, n_1, 0 \rangle} \\
\\
\frac{(L_i: \mathbf{goto} L_j;) \in M}{\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle}
\end{array}$$

Fig. 4. One-step semantics of two-counter Minsky machines.

Non-determinism of  $M$  means that each “state label”  $L_i$  may have multiple associated instructions. The labels  $L_0$  and  $L_1$  are reserved for the *final* and *initial* states of  $M$ , respectively. In particular,  $L_1$  may not label any instructions.

To cope more easily with zero-test instructions, we also add the special labels  $L_{-1}$  and  $L_{-2}$  which come equipped with the following four instructions (and which may not label any other instructions):

$$\begin{array}{ll}
L_{-1}: c_2--; \mathbf{goto} L_{-1}; & L_{-1}: \mathbf{goto} L_0; \\
L_{-2}: c_1--; \mathbf{goto} L_{-2}; & L_{-2}: \mathbf{goto} L_0;
\end{array} \tag{18}$$

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 the one-step relation between configurations of  $M$ ; the one-step semantics of Minsky machines is given by Figure 4. We let  $\rightsquigarrow_M^*$  be the reflexive-transitive closure of  $\rightsquigarrow_M$ , so that  $\langle L, n_1, n_2 \rangle \rightsquigarrow_M^* \langle L', n'_1, n'_2 \rangle$  if  $M$  can go from  $\langle L, n_1, n_2 \rangle$  to  $\langle L', n'_1, n'_2 \rangle$  in a finite number of steps. If  $\langle L, n_1, n_2 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$ , we say that  $M$  *terminates from*  $\langle L, n_1, n_2 \rangle$ , written  $\langle L, n_1, n_2 \rangle \Downarrow_M$ .

The specific role of the special labels  $L_{-1}$  and  $L_{-2}$  is explained by the following lemma:

LEMMA 6.2.  $\langle L_{-k}, n_1, n_2 \rangle \Downarrow_M$  if and only if  $n_k = 0$ , where  $k \in \{1, 2\}$ .

PROOF. We show the case  $k = 1$ ; the other case is similar. We must show that  $\langle L_{-1}, n_1, n_2 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$  iff  $n_1 = 0$ . We proceed by induction on  $n_2$ .

If  $n_2 = 0$  then we are immediately done, since  $\langle L_{-1}, 0, 0 \rangle \rightsquigarrow_M \langle L_0, 0, 0 \rangle$  by the relevant goto instruction in the group (18) but, by construction, no other instruction in  $M$  is applicable to the configuration  $\langle L_{-1}, n_1, 0 \rangle$ .

If  $n_2 = m + 1$  then, by the induction hypothesis, it suffices to prove that

$$\langle L_{-1}, n_1, m + 1 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle \text{ iff } \langle L_{-1}, n_1, m \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$$

The “if” direction is straightforward, since  $\langle L_{-1}, n_1, m + 1 \rangle \rightsquigarrow_M \langle L_{-1}, n_1, m \rangle$  via the relevant decrement instruction from the group (18).

For the “only if” direction, suppose  $\langle L_{-1}, n_1, m + 1 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$ , from which we must have  $\langle L_{-1}, n_1, m + 1 \rangle \rightsquigarrow_M C \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$  for some configuration  $C$ . The only instructions applicable to the configuration  $\langle L_{-1}, n_1, m + 1 \rangle$  are the relevant decrement instruction and goto instruction from (18). If the decrement instruction is applied, we have  $C = \langle L_{-1}, n_1, m \rangle$  and are done. If instead the goto instruction is applied, we have  $C = \langle L_0, n_1, m + 1 \rangle$ . But, by construction, no instructions may be labelled by the “final

state"  $L_0$  whereas  $\langle L_0, n_1, m+1 \rangle$  fails to be a final state. This contradicts the assumption that  $\langle L_{-1}, n_1, m+1 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$ .  $\square$

### 6.1. Machine encoding

*Definition 6.3.* In our encoding we use the following abbreviation. We fix an atomic proposition  $b$ , and henceforth define a “relative negation” by:  $\neg A =_{\text{def}} (A \multimap b)$ .

The following lemma derives useful properties of  $\neg A$  which show that it behaves much like a multiplicative version of intuitionistic negation.

LEMMA 6.4. *The following sequents and proof rules are derivable in Minimal BBI:*

- (a)  $A \vdash \neg\neg A$
- (b)  $\neg\neg\neg A \vdash \neg A$
- (c)  $A * (\neg B \multimap \neg A) \vdash \neg\neg B$
- (d)  $\frac{A * B \vdash C}{A * \neg\neg B \vdash \neg\neg C}$
- (e)  $\frac{A * B \vdash D \quad A * C \vdash D}{A * \neg\neg (B \vee C) \vdash \neg\neg D}$   $\square$

PROOF. We derive each sequent or proof rule directly in Minimal BBI, making use of the derived rules given by Proposition 5.2.

- (a) According to the Minimal BBI axiom  $(\alpha_3)$ , we have  $A * (A \multimap b) \vdash b$  provable. Thus by applying the derived rule  $(\multimap)$  we obtain  $A \vdash ((A \multimap b) \multimap b)$ , i.e.  $A \vdash \neg\neg A$  as required.
- (b) According to the Minimal BBI axiom  $(\alpha_3)$ , we have  $\neg\neg A * (\neg\neg A \multimap b) \vdash b$  provable. Also,  $A \vdash \neg\neg A$  is provable by part (a) above. Thus by the derived rule  $(\text{Tr})$  we obtain  $A * (\neg\neg A \multimap b) \vdash b$ . By applying  $(\multimap)$  again we obtain  $(\neg\neg A \multimap b) \vdash A \multimap b$ , i.e.  $\neg\neg\neg A \vdash \neg A$  as required.
- (c) We have  $\neg B * (\neg B \multimap \neg A) \vdash \neg A$  an instance of the Minimal BBI axiom  $(\alpha_3)$ , which can then be rewritten as  $\neg B * (\neg B \multimap \neg A) \vdash A \multimap b$ . By applying  $(\multimap)$  we obtain  $A * (\neg B \multimap \neg A) \vdash \neg B \multimap b$ , i.e.  $A * (\neg B \multimap \neg A) \vdash \neg\neg B$  as required.
- (d) We have  $A * B \vdash C$  by assumption and  $C \vdash \neg\neg C$  by part (a) above, whence by  $(\text{Tr})$  we obtain  $A * B \vdash \neg\neg C$ . By applying  $(\multimap)$ , we obtain  $A * \neg C \vdash \neg B$ . Since  $\neg B \vdash \neg\neg\neg B$  is derivable by part (a), we obtain  $A * \neg C \vdash \neg\neg\neg B$  by  $(\text{Tr})$ . By applying  $(\multimap)$  again, this can be transformed to the required  $A * \neg\neg B \vdash \neg\neg C$ .
- (e) By applying  $(\multimap)$  to the premises  $A * B \vdash D$  and  $A * C \vdash D$  we have  $B \vdash (A \multimap D)$  and  $C \vdash (A \multimap D)$ . Thus by applying the derived rule  $(\vee)$  we obtain  $B \vee C \vdash (A \multimap D)$ . Applying  $(\multimap)$ , we get  $A * (B \vee C) \vdash D$ . Finally, we apply the derived rule given by part (d) above to obtain  $A * \neg\neg (B \vee C) \vdash \neg\neg D$  as required.  $\square$

*Definition 6.5 (Machine encoding).* For any instruction  $\gamma$  from Definition 6.1, we define the formula  $\kappa(\gamma)$  as follows:

$$\begin{aligned}\kappa(L_i; c_k ++; \mathbf{goto} L_j;) &=_{\text{def}} (\neg(l_j * p_k) \multimap \neg l_i) \\ \kappa(L_i; c_k --; \mathbf{goto} L_j;) &=_{\text{def}} (\neg l_j \multimap \neg(l_i * p_k)) \\ \kappa(L_i; \mathbf{if} c_k = 0 \mathbf{goto} L_j;) &=_{\text{def}} (\neg(l_j \vee l_{-k}) \multimap \neg l_i) \\ \kappa(L_i; \mathbf{goto} L_j;) &=_{\text{def}} (\neg l_j \multimap \neg l_i)\end{aligned}$$

where  $p_1, p_2, l_{-2}, l_{-1}, l_0, l_1, l_2, \dots$  are distinct atomic propositions ( $p_1$  and  $p_2$  are used to represent the counters  $c_1$  and  $c_2$ , respectively).

Then for a Minsky machine  $M$  given by instructions  $\{\gamma_1, \gamma_2, \dots, \gamma_t\}$  define its encoding formula  $\kappa(M)$  by:

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

In the process of computation each instruction of a Minsky machine can be reused an unlimited number of times. In order for us to simulate Minsky computations within Minimal BBI proofs, it is essential that the formulas  $\kappa(\gamma)$  representing reusable instructions can be duplicated as needed, which follows from the restricted  $*$ -contraction given by Lemma 5.3.

LEMMA 6.6. *For each instruction  $\gamma$  of a machine  $M$ , the sequent*

$$\kappa(M) \vdash \kappa(M) * \kappa(\gamma)$$

*is derivable in Minimal BBI.*

PROOF. Writing  $M = \{\gamma_1, \gamma_2, \dots, \gamma_t\}$ , we have  $\kappa(M) = (I \wedge \bigwedge_{i=1}^t \kappa(\gamma_i))$  and  $\gamma = \gamma_j$  for some  $1 \leq j \leq t$ . First, using the derived rules (Id) and  $(\wedge)$ , and the axiom  $(\beta_5)$  for  $\wedge$ , we can derive

$$\kappa(M) \vdash I \wedge \kappa(M)$$

Next, we have as an instance of restricted  $*$ -contraction (Lemma 5.3)

$$I \wedge \kappa(M) \vdash \kappa(M) * \kappa(M)$$

We have  $\kappa(M) \vdash \kappa(\gamma_j)$  derivable in Minimal BBI by using weakening for  $\wedge$  (given by the axioms  $(\beta_5)$  and  $(\beta_6)$ ) and the derived transitivity rule (Tr) of Proposition 5.2. Thus by applying the rule  $(\alpha_6)$  we obtain

$$\kappa(M) * \kappa(M) \vdash \kappa(M) * \kappa(\gamma_j)$$

Thus using the transitivity rule (Tr) we obtain  $\kappa(M) \vdash \kappa(M) * \kappa(\gamma_j)$  as required.  $\square$

We encode a configuration  $C = \langle L_i, n_1, n_2 \rangle$  as a  $*$ -conjunction of the propositional variable  $l_i$  with  $n_1$  copies of  $p_1$  and  $n_2$  copies of  $p_2$ :

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

The fact that  $\langle L_0, 0, 0 \rangle$  is a terminating configuration is encoded by the following formula:

$$I \wedge \neg l_0$$

Putting everything together, termination of  $M$  from  $\langle L_i, n_1, n_2 \rangle$  will be encoded by the following sequent  $\mathcal{F}_{M,C}$ :

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

### 6.2. Intuition for our encoding

*Remark 6.7.* A direct adaptation of the encoding of Minsky machines developed for full linear logic in [Kanovich 1995] does not work properly for separation logic due to the differences between linear logic and separation logic (discussed further in Section 10). Roughly speaking, in the encoding of [Kanovich 1995], each main step in the derivation of the “encoding sequent” corresponds to a single forward step  $C \leadsto_M C'$  in the terminating computation of machine  $M$ . In contrast, in our encoding, each main step in the derivation corresponds to a ‘backward move’ from the class of terminating computations starting from  $C'$  to to the class of longer terminating computations starting from  $C$ .

In accordance with our proof strategy in Section 4, our encoding is intended to achieve two complementary goals.

- (1) On the one hand, we have to show provability (in Minimal BBI) of  $\mathcal{F}_{M,C}$  whenever  $M$  terminates from  $C$ . We construct the proof of  $\mathcal{F}_{M,C}$  by induction on the length  $m$  of the terminating computation, using as sub-proofs the proofs given by induction for terminating computations of shorter length.
- (2) On the other hand, for a fixed memory model from Example 1.2, we have to find a valuation  $\rho_0$  such that  $M$  terminates from  $C$  of the form  $\langle L_i, n_1, n_2 \rangle$  whenever  $\mathcal{F}_{M,C}$  is valid under this valuation  $\rho_0$ , i.e. whenever

$$\llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \rrbracket_{\rho_0} \subseteq \rho_0(b)$$

To obtain the required termination of  $M$  from  $C$  from this inclusion, it stands to reason to define  $\rho_0(b)$  so as to represent *all* terminating configurations, and then to show that  $e_0 \in \llbracket \kappa(M) \rrbracket_{\rho_0}$  and  $e_0 \in \llbracket I \wedge \neg l_0 \rrbracket_{\rho_0}$ , resulting in the inclusion:

$$\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \subseteq \rho_0(b)$$

Providing that each element of  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  uniquely determines the configuration  $\langle L_i, n_1, n_2 \rangle$ , we can then deduce that  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .

(However, an additional twist to the problem is that we need a rather complicated valuation  $\rho_0$  in the memory models of Example 1.2, because of the way the *partial* composition is defined in these models. We look at this point in more detail in Remark 7.5.)

□

### 6.3. Formal details

**THEOREM 6.8.** *Let  $M$  be a Minsky machine, and for a configuration  $C$  of the form  $\langle L_i, n_1, n_2 \rangle$ , suppose that  $M$  terminates from  $C$ . Then the following sequent  $\mathcal{F}_{M,C}$  is derivable in Minimal BBI:*

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \vdash b \quad (19)$$

where  $p_k^n$  denotes the formula  $\underbrace{p_k * p_k * \dots * p_k}_{n \text{ times}}$ , with  $p_k^0 = I$ .

**PROOF.** We make free use of the derived rules of Minimal BBI given by Proposition 5.2. First we show that it suffices to derive the sequent:

$$\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} \vdash \neg \neg l_0 \quad (20)$$

To see this, suppose we have  $A \vdash \neg \neg l_0$ , whence by applying the derived rule  $(\neg *)$  we obtain  $\neg l_0 \vdash (A \neg * b)$ . Since  $I \wedge \neg l_0 \vdash \neg l_0$  is an instance of the axiom  $(\beta_6)$  we obtain  $I \wedge \neg l_0 \vdash (A \neg * b)$  by applying the derived rule  $(\text{Tr})$ . By applying  $(\neg *)$  again we obtain  $A * (I \wedge \neg l_0) \vdash b$  as required.

We show that (20) is derivable in Minimal BBI by induction on the length  $m$  of the computation of  $\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$ . In the base case  $m = 0$  we have  $n_1 = n_2 = 0$ , and must derive:

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

This is easily derivable using the equivalence  $I * A \equiv A$ , weakening for  $\wedge$  and part (a) of Lemma 6.4.

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  $\gamma$  which yields the first step of the computation. We show the cases for a goto instruction, and for increment, decrement and zero-test instructions with the counter variable  $k = 1$ ; the cases for  $k = 2$  are similar.

**Case  $\gamma = (L_i: \text{goto } L_j;)$ .**

By the case assumption we have  $\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle$ , and we are required to show that the following sequent is derivable:

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

By the induction hypothesis, we can derive

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

By applying part (d) of Lemma 6.4, we obtain

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

Since  $\text{-----} l_0 \vdash \text{---} l_0$  is derivable by part (b) of Lemma 6.4, we obtain by (Tr)

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

Now since  $(\neg l_j \multimap \neg l_i) * l_i \vdash \neg l_j$  is provable by part (c) of Lemma 6.4, we obtain by (Tr) and the derived rule  $(\multimap)$

$$\kappa(M) * (\neg l_j \multimap \neg l_i) * l_i * p_1^{n_1} * p_2^{n_2} \vdash \text{---} l_0$$

Since  $\gamma \in M$  and  $\kappa(\gamma) = (\neg l_j \multimap \neg l_i)$ , we have  $\kappa(M) \vdash \kappa(M) * (\neg l_j \multimap \neg l_i)$  provable by Lemma 6.6. Thus we derive by (Tr) and  $(\multimap)$ :

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

which completes the case.

**Case  $\gamma = (L_i: c_1++; \text{goto } L_j;)$ .**

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 sequent is derivable:

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

By the induction hypothesis, we can derive

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

By applying part (d) of Lemma 6.4, we obtain

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

Since  $\text{-----} l_0 \vdash \text{---} l_0$  is derivable by part (b) of Lemma 6.4, we obtain by (Tr)

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

Now since  $(\neg(l_j * p_1) \multimap \neg l_i) * l_i \vdash \neg\neg(l_j * p_1)$  is provable by part (c) of Lemma 6.4, we obtain by (Tr) and  $(\multimap)$

$$\kappa(M) * (\neg(l_j * p_1) \multimap \neg l_i) * l_i * p_1^{n_1} * p_2^{n_2} \vdash \neg\neg l_0$$

As  $\gamma \in M$  and  $\kappa(\gamma) = (\neg(l_j * p_1) \multimap \neg l_i)$ , we have  $\kappa(M) \vdash \kappa(M) * (\neg(l_j * p_1) \multimap \neg l_i)$  provable by Lemma 6.6. Thus we derive by (Tr) and  $(\multimap)$

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

which completes the case.

**Case**  $\gamma = (L_i; c_1 \multimap; \text{goto } L_j;)$ .

By the case assumption we have  $\langle L_i, n_1 + 1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle$ , and we are required to show that the following sequent is derivable:

$$\kappa(M) * l_i * p_1^{n_1+1} * p_2^{n_2} \vdash \neg\neg l_0$$

By the induction hypothesis, we can derive

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

By applying part (d) of Lemma 6.4, we obtain

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

Since  $\neg\neg\neg\neg l_0 \vdash \neg\neg l_0$  is derivable by part (b) of Lemma 6.4, we obtain by (Tr)

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

Now since  $(\neg l_j \multimap \neg(l_i * p_1)) * l_i * p_1 \vdash \neg\neg l_j$  is provable by part (c) of Lemma 6.4, we obtain by (Tr) and  $(\multimap)$

$$\kappa(M) * (\neg l_j \multimap \neg(l_i * p_1)) * l_i * p_1^{n_1+1} * p_2^{n_2} \vdash \neg\neg l_0$$

Since  $\gamma \in M$  and  $\kappa(\gamma) = (\neg l_j \multimap \neg(l_i * p_1))$ , we have  $\kappa(M) \vdash \kappa(M) * (\neg l_j \multimap \neg(l_i * p_1))$  provable by Lemma 6.6. Thus we derive by (Tr) and  $(\multimap)$

$$\kappa(M) * l_i * p_1^{n_1+1} * p_2^{n_2} \vdash \neg\neg l_0$$

which completes the case.

**Case**  $\gamma = (L_i; \text{if } c_1 = 0 \text{ goto } L_j;)$ .

By the case assumption we have  $\langle L_i, 0, n_2 \rangle \rightsquigarrow_M \langle L_j, 0, n_2 \rangle$ , and must derive the following sequent:

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

First, by the induction hypothesis we can derive

$$\kappa(M) * l_j * I * p_2^{n_2} \vdash \neg\neg l_0 \tag{21}$$

Next, note that by Lemma 6.2 we have  $\langle L_{-1}, 0, n_2 \rangle \Downarrow_M$ . Since the computation of  $\langle L_{-1}, 0, n_2 \rangle \rightsquigarrow_M^* \langle L_0, 0, 0 \rangle$  involves only decrement and goto instructions (see group (18) in Definition 6.1) by construction, the cases already considered above are sufficient to establish the present theorem for the configuration  $\langle L_{-1}, 0, n_2 \rangle$ . Thus we can also derive

$$\kappa(M) * l_{-1} * I * p_2^{n_2} \vdash \neg\neg l_0 \tag{22}$$

Thus by applying part (e) of Lemma 6.4 we obtain from (21) and (22)

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



Since  $\text{---}l_0 \vdash \text{---}l_0$  is derivable by part (b) of Lemma 6.4, we obtain by (Tr)

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

Then as  $(\text{---}(l_j \vee l_{-1}) * \text{---}l_i) * l_i \vdash \text{---}(l_j \vee l_{-1})$  is provable by part (c) of Lemma 6.4, we obtain by (Tr) and  $(-*)$

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

Since  $\gamma \in M$  and  $\kappa(\gamma) = (\text{---}(l_j \vee l_{-1}) * \text{---}l_i)$ , we have  $\kappa(M) \vdash \kappa(M) * (\text{---}(l_j \vee l_{-1}) * \text{---}l_i)$  provable by Lemma 6.6. Thus we derive by (Tr) and  $(-*)$

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

which completes the case, and the proof.  $\square$

## 7. FROM VALIDITY TO TERMINATING COMPUTATIONS

In this section, our goal is to show that for each of the concrete memory models given in Example 1.2, we have  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$  whenever the following sequent  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  from Theorem 6.8 is valid:

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

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 1.2(a), obtained by taking the set of locations  $L$  to be  $\mathbb{N}$  and the set of values  $RV$  to be a singleton set.

Then we can extend our approach to any of the memory models from Example 1.2. In particular, we extend our approach to the most general stack-and-heap models from Example 1.2(c), of which the other models can be seen as special instances, obtained by choosing appropriate  $L$ ,  $RV$ , set of stacks  $S$  and permission algebra  $P$ .

*Definition 7.1.* The *RAM-domain model* (cf. Example 1.2(b)) 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  $\emptyset$ .

### 7.1. Valuation for the RAM-domain model

*Definition 7.2.* Given a machine  $M$ , we introduce the following valuation  $\rho_0$  for the RAM-domain model  $(\mathcal{D}, \circ, \{e_0\})$ :

$$\begin{aligned} \rho_0(p_1) &= \{ \{2\}, \{4\}, \{8\}, \dots, \{2^m\}, \dots \} \\ \rho_0(p_2) &= \{ \{3\}, \{9\}, \{27\}, \dots, \{3^m\}, \dots \} \\ \rho_0(l_i) &= \{ \{\delta_i\}, \{\delta_i^2\}, \{\delta_i^3\}, \dots, \{\delta_i^m\}, \dots \} \end{aligned}$$

where each  $\delta_i$  is taken as a fresh prime number for each of the atomic propositions  $l_{-2}, l_{-1}, l_0, l_1, l_2, \dots$ , and:

$$\rho_0(b) = \bigcup_{\langle L_i, n_1, n_2 \rangle \Downarrow_M} \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$$

where  $b$  is the distinguished propositional variable introduced by Definition 6.3. (The reason behind our choice for  $\rho_0(b)$  is given in Remark 6.7)

**LEMMA 7.3.** *For any  $d \in \mathcal{D}$ ,  $n \in \mathbb{N}$  and  $k \in \{1, 2\}$ , Definition 7.2 guarantees that  $d \in \llbracket p_k^n \rrbracket_{\rho_0}$  if and only if  $d$  consists of exactly  $n$  distinct powers of the corresponding prime  $k + 1$  (that is, 2 or 3).*

*Thus  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  is not empty, and each of the elements of  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  uniquely determines the configuration  $\langle L_i, n_1, n_2 \rangle$ .*

PROOF. The first part is by induction on  $n$ . In the case  $n = 0$  we have  $\llbracket p_k^0 \rrbracket_{\rho_0} = \llbracket \mathbf{I} \rrbracket_{\rho_0} = \{e_0\}$ , where  $e_0$  is  $\emptyset$ . Thus  $d \in \llbracket p_k^0 \rrbracket_{\rho_0}$  iff  $d = \emptyset$  iff  $d$  consists of zero powers of the prime  $k + 1$ .

In the case  $n > 0$  we have, using Lemma 2.8:

$$\begin{aligned} \llbracket p_k^n \rrbracket_{\rho_0} &= \llbracket p_k * p_k^{n-1} \rrbracket_{\rho_0} \\ &= \llbracket p_k \rrbracket_{\rho_0} \cdot \llbracket p_k^{n-1} \rrbracket_{\rho_0} \\ &= \{d' \circ d'' \mid d' \in \rho_0(p_k), d'' \in \llbracket p_k^{n-1} \rrbracket_{\rho_0}\} \end{aligned}$$

Thus  $d \in \llbracket p_k^n \rrbracket_{\rho_0}$  iff it is the union of disjoint sets  $d'$  and  $d''$  with  $d' \in \rho_0(p_k)$  and  $d'' \in \llbracket p_k^{n-1} \rrbracket_{\rho_0}$ . By induction hypothesis,  $d'' \in \llbracket p_k^{n-1} \rrbracket_{\rho_0}$  iff  $d''$  consists of exactly  $n - 1$  distinct powers of  $k + 1$ , and by construction it holds that  $d' \in \rho_0(p_k)$  iff  $d' = \{(k + 1)^m\}$  for some  $m$ . Since  $d'$  and  $d''$  are disjoint,  $d$  consists of exactly  $n$  powers of  $k + 1$  as required.

For the second part of the lemma, we have, using Lemma 2.8:

$$\begin{aligned} \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} &= \llbracket l_i \rrbracket_{\rho_0} \cdot \llbracket p_1^{n_1} \rrbracket_{\rho_0} \cdot \llbracket p_2^{n_2} \rrbracket_{\rho_0} \\ &= \{d_1 \circ d_2 \circ d_3 \mid d_1 \in \rho_0(l_i), d_2 \in \llbracket p_1^{n_1} \rrbracket_{\rho_0}, d_3 \in \llbracket p_2^{n_2} \rrbracket_{\rho_0}\} \end{aligned}$$

Thus if  $d \in \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  then it is the union of disjoint sets  $d_1$ ,  $d_2$  and  $d_3$  with  $d_1 \in \rho_0(l_i)$ ,  $d_2 \in \llbracket p_1^{n_1} \rrbracket_{\rho_0}$  and  $d_3 \in \llbracket p_2^{n_2} \rrbracket_{\rho_0}$ . By the first part of the lemma,  $d_2$  and  $d_3$  consist of  $n_1$  distinct powers of 2 and  $n_2$  distinct powers of 3, respectively, whereas by construction  $d_1 = \{\delta_i^n\}$  for some  $n$ , where  $\delta_i$  is the unique prime corresponding to  $l_i$ .

Note that by the uniqueness of prime decompositions,  $d_1$ ,  $d_2$  and  $d_3$  are all non-overlapping. Thus  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  is not empty, and each of its elements uniquely determines  $L_i$ ,  $n_1$  and  $n_2$ , and hence the configuration  $\langle L_i, n_1, n_2 \rangle$ .  $\square$

*Remark 7.4.* Our choice of  $\rho_0(p_1)$  and  $\rho_0(p_2)$  to have *infinitely many disjoint elements* is dictated by peculiarities of composition  $\circ$  in the heap model. Moreover, for any *finite* choice of  $\rho_0(p_k)$ , we must have

$$\llbracket p_k^n \rrbracket_{\rho_0} = \llbracket p_k^m \rrbracket_{\rho_0}$$

for all sufficiently large  $n$  and  $m$ , which obstructs us in uniquely representing the contents  $n$  of the Minsky machine counter  $c_k$  by the formula  $p_k^n$ . We discuss the impact upon decidability of a restriction to finite valuations in Section 8.  $\square$

## 7.2. Intuition for our encoding (cont.)

*Remark 7.5.* As discussed in Remark 6.7, to guarantee the faithfulness of our encoding it suffices to check that the encoding guarantees that  $e_0 \in \llbracket \kappa(M) \rrbracket_{\rho_0}$ , that is, for each of the machine instruction  $\gamma$  we can provide  $e_0 \in \llbracket \kappa(\gamma) \rrbracket_{\rho_0}$ .

For any instruction  $\gamma$ , the formula  $\kappa(\gamma)$  is of the form  $(\neg A \multimap \neg B)$  where  $A$  is of the form  $l_j$ ,  $(l_j \vee l_{-k})$  or  $(l_j * p_k)$ , and  $B$  is of the form  $l_i$  or  $(l_i * p_k)$ . Thus, recalling that  $\neg A =_{\text{def}} (A \multimap b)$ , and  $\neg B =_{\text{def}} (B \multimap b)$ , the statement  $e_0 \in \llbracket \kappa(\gamma) \rrbracket_{\rho_0}$  is equivalent to the inclusion

$$\llbracket A \multimap b \rrbracket_{\rho_0} \subseteq \llbracket B \multimap b \rrbracket_{\rho_0}$$

Since, by Lemma 2.8, the statement  $d \in \llbracket A \multimap b \rrbracket_{\rho_0}$  means that  $\llbracket A \rrbracket_{\rho_0} \cdot \{d\} \subseteq \rho_0(b)$ , this inclusion can be rewritten as the rule that for all  $d$ ,

$$\frac{\llbracket A \rrbracket_{\rho_0} \cdot \{d\} \subseteq \rho_0(b)}{\llbracket B \rrbracket_{\rho_0} \cdot \{d\} \subseteq \rho_0(b)}$$

For instance, in the case of the increment instruction “ $L_i: c_1++; \text{goto } L_j;$ ”, we must show that the following rule is correct for any  $d$ :

$$\frac{\llbracket l_j * p_1 \rrbracket_{\rho_0} \cdot \{d\} \subseteq \rho_0(b)}{\llbracket l_i \rrbracket_{\rho_0} \cdot \{d\} \subseteq \rho_0(b)}$$

Our choice of the encoding (see details below) ensures that if  $\llbracket l_j * p_1 \rrbracket_{\rho_0} \cdot \{d\} \subseteq \rho_0(b)$ , then  $d$  must belong to a set of the form  $\llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , so that this rule becomes

$$\frac{\llbracket l_j * p_1^{n_1+1} * p_2^{n_2} \rrbracket_{\rho_0} \subseteq \rho_0(b)}{\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \subseteq \rho_0(b)}$$

or, paraphrasing,

$$\frac{\text{“}M \text{ terminates from the configuration } \langle L_j, n_1 + 1, n_2 \rangle\text{”}}{\text{“}M \text{ terminates from the configuration } \langle L_i, n_1, n_2 \rangle\text{”}}$$

which is a correct statement according to the increment instruction at hand.  $\square$

### 7.3. Formal details

Now we prove the basic lemma.

LEMMA 7.6.  $e_0 \models_{\rho_0} \kappa(M)$  for any machine  $M$ .

PROOF. Writing  $M = \{\gamma_1, \dots, \gamma_t\}$ , we have by Definition 2.7:

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

Thus it suffices to show that  $e_0 \models_{\rho_0} \kappa(\gamma)$  for any instruction  $\gamma$ . Recalling that  $\neg A =_{\text{def}} (A \multimap b)$ , we shall make use of the fact that, for any  $d \in \mathcal{D}$  and any formula  $A$ ,

$$d \models_{\rho_0} \neg A \Leftrightarrow \forall d'. (d, d' \text{ disjoint and } d' \models_{\rho_0} A) \text{ implies } d \circ d' \in \rho_0(b) \quad (23)$$

We present the cases for a goto instruction, and for increment, decrement and zero-test instructions with counter variable  $k = 1$ ; the cases where  $k = 2$  are similar.

**Case**  $\gamma = (L_i: \text{goto } L_j;)$ . We have  $\kappa(\gamma) = (\neg l_j \multimap \neg l_i)$ , so must show  $e_0 \models_{\rho_0} \neg l_j \multimap \neg l_i$ . Using the fact that  $e_0 \circ d = d$ , this amounts to showing, for any  $d \in \mathcal{D}$ ,

$$d \models_{\rho_0} \neg l_j \text{ implies } d \models_{\rho_0} \neg l_i$$

To show this implication, suppose  $d \models_{\rho_0} \neg l_j$ . Since  $d \in \mathcal{D}$  is a finite set whereas  $\rho_0(l_j)$  contains infinitely many disjoint sets by construction, there must be some (in fact infinitely many)  $d_j \in \rho_0(l_j)$  such that  $d$  and  $d_j$  are disjoint. Thus, using the equivalence (23) and Lemma 2.8, we have:

$$\begin{aligned} d \circ d_j &\in \rho_0(b) \\ &= \bigcup_{\langle L_i, n_1, n_2 \rangle \downarrow_M} \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \\ &= \bigcup_{\langle L_i, n_1, n_2 \rangle \downarrow_M} \rho_0(l_i) \cdot \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \end{aligned}$$

By construction of  $\rho_0$  and using the fact that  $d_j \in \rho_0(l_j)$ , we must have  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  for some  $n_1, n_2$ . Moreover, by Lemma 7.3,  $d \circ d_j$  uniquely determines  $\langle L_j, n_1, n_2 \rangle$ , so we must have  $\langle L_j, n_1, n_2 \rangle \downarrow_M$ . Since  $\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle$  by applying the goto instruction  $\gamma$ , we then have  $\langle L_i, n_1, n_2 \rangle \downarrow_M$ .

Now to see that  $d \models_{\rho_0} \neg l_i$ , let  $d' \in \mathcal{D}$  be disjoint from  $d$  with  $d' \in \rho_0(l_i)$ . By (23) it suffices to show that  $d \circ d' \in \rho_0(b)$ . Since  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , we have  $d \circ d' \in \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$

where  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ , and thus  $d \circ d' \in \rho_0(b)$  as required.

**Case  $\gamma = (L_i: c_1++; \text{goto } L_j;)$ .** We have  $\kappa(\gamma) = (\neg(l_j * p_1) \multimap \neg l_i)$ . As in the previous case, to show  $e_0 \models_{\rho_0} \kappa(\gamma)$ , we must show that, for any  $d \in \mathcal{D}$ ,

$$d \models_{\rho_0} \neg(l_j * p_1) \text{ implies } d \models_{\rho_0} \neg l_i$$

Suppose  $d \models_{\rho_0} \neg(l_j * p_1)$ . Since  $\rho_0(l_j)$  and  $\rho_0(p_1)$  each contain infinitely many disjoint sets, we can find  $d_j \in \rho_0(l_j)$  and  $d_1 \in \rho_0(p_1)$  such that  $d, d_j$  and  $d_1$  are disjoint. Thus  $d_j \circ d_1 \models_{\rho_0} l_j * p_1$ . Using the equivalence (23) and Lemma 2.8, we have:

$$\begin{aligned} d \circ d_j \circ d_1 &\in \rho_0(b) \\ &= \bigcup_{\langle L_i, n_1, n_2 \rangle \Downarrow_M} \rho_0(l_i) \cdot \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \end{aligned}$$

By construction of  $\rho_0$  and using the fact that  $d_j \in \rho_0(l_j)$ , we must have  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  for some  $n_1, n_2$ . Moreover, by Lemma 7.3, and using the fact that  $d_1 \in \rho_0(p_1)$ , the set  $d \circ d_j \circ d_1$  uniquely determines  $\langle L_j, n_1 + 1, n_2 \rangle$ , whence  $\langle L_j, n_1 + 1, n_2 \rangle \Downarrow_M$ . Since  $\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1 + 1, n_2 \rangle$  via the increment instruction  $\gamma$ , we have  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .

Now to see that  $d \models_{\rho_0} \neg l_i$ , let  $d' \in \mathcal{D}$  be disjoint from  $d$  with  $d' \in \rho_0(l_i)$ . By (23) it suffices to show that  $d \circ d' \in \rho_0(b)$ . Since  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , we have  $d \circ d' \in \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  where  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ , and thus  $d \circ d' \in \rho_0(b)$  as required.

**Case  $\gamma = (L_i: c_1--; \text{goto } L_j;)$ .** We have  $\kappa(\gamma) = (\neg l_j \multimap \neg(l_i * p_1))$ . To show  $e_0 \models_{\rho_0} \kappa(\gamma)$ , we must show that, for any  $d \in \mathcal{D}$ ,

$$d \models_{\rho_0} \neg l_j \text{ implies } d \models_{\rho_0} \neg(l_i * p_1)$$

Now suppose  $d \models_{\rho_0} \neg l_j$ . As in the previous cases, we can find  $d_j \in \rho_0(l_j)$  such that  $d_j$  and  $d$  are disjoint. Thus, using the equivalence (23), we have:

$$\begin{aligned} d \circ d_j &\in \rho_0(b) \\ &= \bigcup_{\langle L_i, n_1, n_2 \rangle \Downarrow_M} \rho_0(l_i) \cdot \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \end{aligned}$$

By the same argument as in previous cases, this implies that there exist  $n_1$  and  $n_2$  such that  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , and  $\langle L_j, n_1, n_2 \rangle \Downarrow_M$ . Since  $\langle L_i, n_1 + 1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle$  via the decrement instruction  $\gamma$ , we have  $\langle L_i, n_1 + 1, n_2 \rangle \Downarrow_M$ .

Now let  $d' \in \mathcal{D}$  be disjoint from  $d$  with  $d' \models_{\rho_0} l_i * p_1$ . By (23) it suffices to show that  $d \circ d' \in \rho_0(b)$ . First, we have  $d' = d_i \circ d_1$  where  $d_i \in \rho_0(l_i)$  and  $d_1 \in \rho_0(p_1)$ . Since  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , we have  $d \circ d_i \circ d_1 \in \llbracket l_i * p_1^{n_1+1} * p_2^{n_2} \rrbracket_{\rho_0}$  where  $\langle L_i, n_1 + 1, n_2 \rangle \Downarrow_M$ . Thus  $d \circ d_i \circ d_1 = d \circ d' \in \rho_0(b)$  as required.

**Case  $\gamma = (L_i: \text{if } c_1=0 \text{ goto } L_j;)$ .** In this case, we have  $\kappa(\gamma) = (\neg(l_j \vee l_{-1}) \multimap \neg l_i)$ . To show  $e_0 \models_{\rho_0} \kappa(\gamma)$ , we must show that, for any  $d \in \mathcal{D}$ ,

$$d \models_{\rho_0} \neg(l_j \vee l_{-1}) \text{ implies } d \models_{\rho_0} \neg l_i$$

Now suppose that  $d \models_{\rho_0} \neg(l_j \vee l_{-1})$ . As in previous cases, we can find  $d_j \in \rho_0(l_j)$  and  $d_{-1} \in \rho_0(l_{-1})$  such that  $d, d_j$  and  $d_{-1}$  are all disjoint. Note that both  $d_j \models_{\rho_0} l_j \vee l_{-1}$  and  $d_{-1} \models_{\rho_0} l_j \vee l_{-1}$ . Thus, using the equivalence (23), we have:

$$\begin{aligned} d \circ d_j, d \circ d_{-1} &\in \rho_0(b) \\ &= \bigcup_{\langle L_i, n_1, n_2 \rangle \Downarrow_M} \rho_0(l_i) \cdot \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \end{aligned}$$

By the same argument as in previous cases, there are  $n_1$  and  $n_2$  such that  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , and we have both  $\langle L_j, n_1, n_2 \rangle \Downarrow_M$  and  $\langle L_{-1}, n_1, n_2 \rangle \Downarrow_M$ . However, by Lemma 6.2,

$\langle L_{-1}, n_1, n_2 \rangle \Downarrow_M$  implies  $n_1 = 0$ . Thus  $\langle L_i, n_1, n_2 \rangle \rightsquigarrow_M \langle L_j, n_1, n_2 \rangle \Downarrow_M$  by applying the zero-test instruction  $\gamma$ , whence  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ . The argument that  $d \models_{\rho_0} \neg l_i$  is then exactly as in the goto and increment instruction cases. This completes the case, and the proof.  $\square$

LEMMA 7.7. *For any machine  $M$  we have  $e_0 \models_{\rho_0} (I \wedge \neg l_0)$ .*

PROOF. We trivially have  $e_0 \models_{\rho_0} I$ . To show  $e_0 \models_{\rho_0} \neg l_0$ , taking into account that  $e_0 \circ x = x$ , we must show that  $x \models_{\rho_0} l_0$  implies  $x \models_{\rho_0} b$ , i.e. that  $\rho_0(l_0) \subseteq \rho_0(b)$ . This is immediate by construction of  $\rho_0(b)$ , since  $\langle L_0, 0, 0 \rangle \Downarrow_M$  by definition and  $\rho_0(l_0) = \llbracket l_0 * I * I \rrbracket_{\rho_0} = \llbracket l_0 * p_1^0 * p_2^0 \rrbracket_{\rho_0}$ .  $\square$

THEOREM 7.8. *For any machine  $M$ , if the sequent*

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

*is valid in the RAM-domain model  $(\mathcal{D}, \circ, \{e_0\})$ , then  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .*

PROOF. By the definition of validity and using Lemma 2.8 we have:

$$\begin{aligned} \llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \rrbracket_{\rho_0} &\subseteq \llbracket b \rrbracket_{\rho_0} \\ \text{i.e. } \llbracket \kappa(M) \rrbracket_{\rho_0} \cdot \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \cdot \llbracket I \wedge \neg l_0 \rrbracket_{\rho_0} &\subseteq \rho_0(b) \end{aligned}$$

Since  $e_0 \in \llbracket \kappa(M) \rrbracket_{\rho_0}$  by Lemma 7.6 and  $e_0 \in \llbracket I \wedge \neg l_0 \rrbracket_{\rho_0}$  by Lemma 7.7 we have in particular:

$$\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0} \subseteq \rho_0(b)$$

By Lemma 7.3, each of the elements of the set  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  uniquely determines the configuration  $\langle L_i, n_1, n_2 \rangle$ , whence our construction of  $\rho_0(b)$  yields  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .  $\square$

#### 7.4. The general case of stack-and-heap models

Having established as Theorem 7.8 that validity of a sequent of the form  $\mathcal{F}_{M,C}$  in the basic RAM-domain model implies termination of machine  $M$  from configuration  $C$ , we now extend this result to the most sophisticated *stack-and-heap* models from Example 1.2(c), in which the heaps are heaps-with-permissions. All the models from Example 1.2(a)–(c) can be seen as special instances of such models by taking appropriate  $L$ ,  $RV$ ,  $S$  and  $P$ . We write  $\circ'$  for the composition in a stack-and-heap model and  $\cdot'$  for its extension to sets to disambiguate from the corresponding operations  $\circ$  and  $\cdot$  of the RAM-domain model.

*Definition 7.9.* Let  $(S \times H, \circ', E)$  be a stack-and-heap model from Example 1.2(c), where  $S$  is a set of stacks,  $H = \mathbb{N} \rightarrow_{\text{fin}} (RV \times P)$  is a set of heaps-with-permissions and  $(P, \bullet, \mathbb{1})$  is a permission algebra. (Recall that  $\mathbb{1} \bullet \pi$  is undefined for all  $\pi \in P$ .) We abuse notation slightly by writing the unit  $e_0$  of the RAM-domain model for the heap with empty domain. Thus  $E = \{\langle s, e_0 \rangle \mid s \in S\}$ .

Based on our valuation  $\rho_0$  for the RAM-domain model and a given machine  $M$  in Definition 7.2, we introduce a valuation  $\rho_1$  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:

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

Then for any atomic proposition  $p$  we define its valuation by:

$$\rho_1(p) = \bigcup_{d \in \rho_0(p)} [d]$$

LEMMA 7.10. *For any stack-and-heap model  $(S \times H, \circ', E)$  and any atomic propositions  $p$  and  $q$ , we have the following identity:*

$$\llbracket p * q \rrbracket_{\rho_1} = \bigcup_{d \in \llbracket p * q \rrbracket_{\rho_0}} [d]$$

PROOF. First, we show that for any  $d_1 \in \rho_0(p)$ ,  $d_2 \in \rho_0(q)$ ,

$$[d_1 \circ d_2] = [d_1] \cdot' [d_2] \quad (24)$$

For disjoint  $d_1$  and  $d_2$  this is given by construction, since

$$\begin{aligned} [d_1] \cdot' [d_2] &= \{\langle s_0, h_1 \rangle \circ' \langle s_0, h_2 \rangle \mid \langle s_0, h_1 \rangle \in [d_1], \langle s_0, h_2 \rangle \in [d_2]\} \\ &= \{\langle s_0, h_1 \rangle \circ' \langle s_0, h_2 \rangle \mid \text{domain}(h_1) = [d_1], \text{domain}(h_2) = [d_2], \\ &\quad \forall \ell \in d_1. h_1(\ell) = \langle \_, \mathbb{1} \rangle, \forall \ell \in d_2. h_2(\ell) = \langle \_, \mathbb{1} \rangle\} \\ &= \{\langle s_0, h \rangle \mid \text{domain}(h) = d_1 \circ d_2, \forall \ell \in d_1 \circ d_2. h(\ell) = \langle \_, \mathbb{1} \rangle\} \\ &= [d_1 \circ d_2] \end{aligned}$$

For overlapping  $d_1$  and  $d_2$ , we have  $d_1 \circ d_2$  undefined and thus  $[d_1 \circ d_2] = \emptyset$ . We show that  $[d_1] \cdot' [d_2] = \emptyset$ . Assume for contradiction that  $\langle s_0, h_1 \rangle \circ' \langle s_0, h_2 \rangle$  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  $[d_2]$ , and since  $d_1, d_2$  are overlapping, this implies  $h_1(\ell) = h_2(\ell) = \langle \_, \mathbb{1} \rangle$  for some  $\ell \in d_1 \cap d_2$ . 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 (because in heap-with-permissions models (cf. Example 1.2(b)) we add the permissions when composing compatible heaps), which is a contradiction. Thus  $[d_1] \cdot' [d_2]$  is empty when  $d_1 \circ d_2$  is undefined.

Now, using Lemma 2.8, we have as required:

$$\begin{aligned} \llbracket p * q \rrbracket_{\rho_1} &= \llbracket p \rrbracket_{\rho_1} \cdot' \llbracket q \rrbracket_{\rho_1} \\ &= \{\langle s_0, h_1 \rangle \circ' \langle s_0, h_2 \rangle \mid \langle s_0, h_1 \rangle \in \rho_1(p), \langle s_0, h_2 \rangle \in \rho_1(q)\} \\ &= \{\langle s_0, h_1 \rangle \circ' \langle s_0, h_2 \rangle \mid \langle s_0, h_1 \rangle \in \bigcup_{d \in \rho_0(p)} [d], \langle s_0, h_2 \rangle \in \bigcup_{d \in \rho_0(q)} [d]\} \\ &= \bigcup_{d_1 \in \rho_0(p), d_2 \in \rho_0(q)} \{\langle s_0, h_1 \rangle \circ' \langle s_0, h_2 \rangle \mid \langle s_0, h_1 \rangle \in [d_1], \langle s_0, h_2 \rangle \in [d_2]\} \\ &= \bigcup_{d_1 \in \rho_0(p), d_2 \in \rho_0(q)} [d_1] \cdot' [d_2] \\ &= \bigcup_{d_1 \in \rho_0(p), d_2 \in \rho_0(q)} [d_1 \circ d_2] \quad (\text{by (24)}) \\ &= \bigcup_{d \in \{d_1 \circ d_2 \mid d_1 \in \rho_0(p), d_2 \in \rho_0(q)\}} [d] \\ &= \bigcup_{d \in \llbracket p \rrbracket_{\rho_0} \cdot' \llbracket q \rrbracket_{\rho_0}} [d] \\ &= \bigcup_{d \in \llbracket p * q \rrbracket_{\rho_0}} [d] \end{aligned}$$

□

LEMMA 7.11. *For any stack-and-heap model  $(S \times H, \circ', E)$  and any formula  $A$  of the form  $l_i$ ,  $l_i * p_k$ , or  $l_i \vee l_j$ , we have the equivalence*

$$\llbracket A \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\} \subseteq \llbracket b \rrbracket_{\rho_1} \Leftrightarrow \langle s_0, h \rangle \in [d] \text{ and } \llbracket A \rrbracket_{\rho_0} \cdot \{d\} \subseteq \llbracket b \rrbracket_{\rho_0}$$

where  $d = \text{domain}(h)$ .

PROOF. First, note that by distinguishing cases on  $A$ , and using Lemma 7.10 in the case  $A = l_i * p_k$ , we easily have that

$$\llbracket A \rrbracket_{\rho_1} = \bigcup_{d \in \llbracket A \rrbracket_{\rho_0}} [d] \quad (25)$$

We show each of the required implications separately.

( $\Rightarrow$ ) Suppose that  $\llbracket A \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\} \subseteq \llbracket b \rrbracket_{\rho_1}$ . That is, we have:

$$\{\langle s', h' \rangle \circ' \langle s_0, h \rangle \mid \langle s', h' \rangle \models_{\rho_1} A\} \subseteq \bigcup_{d \in \rho_0(b)} [d] \quad (26)$$

Now let  $d = \text{domain}(h)$ , and let  $x \in \llbracket A \rrbracket_{\rho_0} \cdot \{d\}$ . Thus  $x = d \circ d'$ , where  $d$  and  $d'$  are disjoint and  $d' \models_{\rho_0} A$ . We require to show that  $d \circ d' \in \llbracket b \rrbracket_{\rho_0}$  and  $\langle s_0, h \rangle \in [d]$ .

Let  $\langle s_0, h' \rangle \in [d']$ . Since  $d' \in \llbracket A \rrbracket_{\rho_0}$ , we have by (25)  $\langle s_0, h' \rangle \in \llbracket A \rrbracket_{\rho_1}$ , i.e.  $\langle s_0, h' \rangle \models_{\rho_1} A$ . Since  $d$  and  $d'$  are disjoint,  $\langle s_0, h \rangle \circ' \langle s_0, h' \rangle$  is defined and, by (26),

$$\langle s_0, h \rangle \circ' \langle s_0, h' \rangle \in \bigcup_{d \in \rho_0(b)} [d]$$

That is,  $\langle s_0, h \rangle \circ' \langle s_0, h' \rangle \in [d'']$  for some  $d'' \in \rho_0(b)$ . Since the domain of  $\langle s_0, h \rangle \circ' \langle s_0, h' \rangle$  is  $d \circ d'$  by construction, we must have  $d'' = d \circ d'$ . Thus  $d \circ d' \in \rho_0(b) = \llbracket b \rrbracket_{\rho_0}$  as required. Furthermore,  $\langle s_0, h \rangle \in [d]$  because  $d$  and  $d'$  are disjoint, and thus  $h$  and  $h'$  do not overlap, but  $\langle s_0, h \rangle \circ' \langle s_0, h' \rangle \in [d \circ d']$ , which means that  $h(\ell) = \langle \_, \mathbb{1} \rangle$  for all  $\ell \in d$ . This completes the “only if” direction.

( $\Leftarrow$ ) Let  $d = \text{domain}(h)$ , let  $\langle s_0, h \rangle \in [d]$  and suppose  $\llbracket A \rrbracket_{\rho_0} \cdot \{d\} \subseteq \llbracket b \rrbracket_{\rho_0}$ . The latter inclusion means that:

$$\{d' \circ d \mid d' \models_{\rho_0} A\} \subseteq \rho_0(b) \quad (27)$$

Now suppose that  $x \in \llbracket A \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\}$ . Thus  $x = \langle s', h' \rangle \circ' \langle s_0, h \rangle$  (which implies  $s' = s_0$ ), where  $\langle s', h' \rangle \models_{\rho_1} A$ . We require to show that  $\langle s_0, h' \rangle \circ' \langle s_0, h \rangle \in \llbracket b \rrbracket_{\rho_1}$ .

Since  $\langle s_0, h' \rangle \in \llbracket A \rrbracket_{\rho_1}$  by the above, we have by (25) some  $d'$  such that  $d' \in \llbracket A \rrbracket_{\rho_0}$ , i.e.  $d' \models_{\rho_0} A$ , and  $\langle s_0, h' \rangle \in [d']$  (which implies  $\text{domain}(h') = d'$ ). Because  $\langle s_0, h \rangle \in [d]$  and  $\langle s', h' \rangle \in [d']$ , we have  $h(\ell) = \langle \_, \mathbb{1} \rangle$  for all  $\ell \in d$  and  $h'(\ell') = \langle \_, \mathbb{1} \rangle$  for all  $\ell' \in d'$ . As  $\langle s_0, h' \rangle \circ' \langle s_0, h \rangle$  is defined by assumption, it follows that  $h$  and  $h'$  cannot overlap, and so  $d$  and  $d'$  are disjoint. Thus, by (27),  $d' \circ d \in \rho_0(b)$ . We easily have  $\langle s_0, h' \rangle \circ' \langle s_0, h \rangle \in [d' \circ d]$ , and so

$$\langle s_0, h' \rangle \circ' \langle s_0, h \rangle \in \bigcup_{d \in \rho_0(b)} [d] = \rho_1(b) = \llbracket b \rrbracket_{\rho_1}$$

as required. This completes the “if” direction, and the proof.  $\square$

**LEMMA 7.12.**  $\langle s_0, e_0 \rangle \models_{\rho_1} \kappa(M)$  for any machine  $M$  and any stack-and-heap model  $(S \times H, \circ', E)$ .

**PROOF.** As in Lemma 7.6, we show that  $\langle s_0, e_0 \rangle \models_{\rho_1} \kappa(\gamma)$  for any instruction  $\gamma$  in Definition 6.1. For any such  $\gamma$ , the formula  $\kappa(\gamma)$  is of the form  $(\neg A \multimap \neg B)$  where  $A$  and  $B$  are of the form  $l_i$ ,  $l_i \vee l_j$  or  $l_i * p_k$ . Since  $\langle s_0, e_0 \rangle$  is a unit of the model and  $\langle s_0, e_0 \rangle \circ' \langle s, h \rangle$  is undefined for  $s \neq s_0$ , we have

$$\begin{aligned} & \langle s_0, e_0 \rangle \models_{\rho_1} \neg A \multimap \neg B \\ \Leftrightarrow & \forall \langle s, h \rangle. \langle s_0, e_0 \rangle \circ' \langle s, h \rangle \text{ defined and } \langle s, h \rangle \models_{\rho_1} \neg A \text{ implies } \langle s_0, e_0 \rangle \circ' \langle s, h \rangle \models_{\rho_1} \neg B \\ \Leftrightarrow & \forall \langle s', h' \rangle. s = s_0 \text{ and } \langle s, h \rangle \models_{\rho_1} \neg A \text{ implies } \langle s, h \rangle \models_{\rho_1} \neg B \\ \Leftrightarrow & \forall h. \langle s_0, h \rangle \models_{\rho_1} \neg A \text{ implies } \langle s_0, h \rangle \models_{\rho_1} \neg B \end{aligned}$$

Recalling that  $\neg A = A \multimap b$ , and using Lemma 2.8, this amounts to showing, for any  $h \in H$ :

$$\llbracket A \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\} \subseteq \llbracket b \rrbracket_{\rho_1} \Rightarrow \llbracket B \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\} \subseteq \llbracket b \rrbracket_{\rho_1} \quad (28)$$

Assume  $\llbracket A \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\} \subseteq \llbracket b \rrbracket_{\rho_1}$ , and let  $d = \text{domain}(h)$ . By Lemma 7.11 we have  $\langle s_0, h \rangle \in [d]$ , and  $\llbracket A \rrbracket_{\rho_0} \cdot \{d\} \subseteq \llbracket b \rrbracket_{\rho_0}$ . By Lemma 7.6 we have  $e_0 \models_{\rho_0} \kappa(\gamma)$  and thus (by the same reasoning as above)  $\llbracket \neg A \rrbracket_{\rho_0} \subseteq \llbracket \neg B \rrbracket_{\rho_0}$ , which means that  $\llbracket A \rrbracket_{\rho_0} \cdot \{d\} \subseteq \llbracket b \rrbracket_{\rho_0}$  implies  $\llbracket B \rrbracket_{\rho_0} \cdot \{d\} \subseteq \llbracket b \rrbracket_{\rho_0}$ . Thus we have  $\llbracket B \rrbracket_{\rho_0} \cdot \{d\} \subseteq \llbracket b \rrbracket_{\rho_0}$ , whence Lemma 7.11 yields  $\llbracket B \rrbracket_{\rho_1} \cdot' \{\langle s_0, h \rangle\} \subseteq \llbracket b \rrbracket_{\rho_1}$  as required.  $\square$

**LEMMA 7.13.**  $\langle s_0, e_0 \rangle \models_{\rho_1} \mathbf{I} \wedge \neg l_0$  for any machine  $M$  and any stack-and-heap model  $(S \times H, \circ', E)$ .

**PROOF.** We have  $\langle s_0, e_0 \rangle \models_{\rho_1} \mathbf{I}$  because  $\langle s_0, e_0 \rangle \in E$ . By Lemma 7.7 we have  $e_0 \models_{\rho_0} \neg l_0$  and thus  $\rho_0(l_0) \subseteq \rho_0(b)$ . Thus we have

$$\rho_1(l_0) = \bigcup_{d \in \rho_0(l_0)} [d] \subseteq \bigcup_{d \in \rho_0(b)} [d] = \rho_1(b)$$



which implies that  $\langle s_0, e_0 \rangle \models_{\rho_1} \neg l_0$ , as required.  $\square$

**THEOREM 7.14.** *For any machine  $M$  and configuration  $\langle L_i, n_1, n_2 \rangle$ , if a sequent  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  of the form*

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

*is valid in some concrete model listed in Example 1.2 then  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .*

**PROOF.** The case of Petri nets and multiset models with their *total*  $\circ$  can be covered by the original valuation  $\rho_0$  from Definition 7.2.

For other models, by taking appropriate  $L, RV$ , set of stacks  $S$  and permission algebra  $P$ , we may assume that  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  is valid in a stack-and-heap model  $(S \times H, \circ', E)$  as given in Definition 7.9. Thus we have by definition of validity, and using Lemma 2.8:

$$\begin{aligned} & \llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \rrbracket_{\rho_1} \subseteq \llbracket b \rrbracket_{\rho_1} \\ \text{i.e. } & \llbracket \kappa(M) \rrbracket_{\rho_1} \cdot' \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_1} \cdot' \llbracket I \wedge \neg l_0 \rrbracket_{\rho_1} \subseteq \rho_1(b) \end{aligned}$$

Taking into account that  $\langle s_0, e_0 \rangle \in \llbracket \kappa(M) \rrbracket_{\rho_1}$  and  $\langle s_0, e_0 \rangle \in \llbracket I \wedge \neg l_0 \rrbracket_{\rho_1}$  according to Lemmas 7.12 and 7.13 respectively, we get:

$$\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_1} \subseteq \rho_1(b)$$

Using Lemma 7.10 and our definition of  $\rho_1(b)$ , we obtain:

$$\bigcup_{d \in \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}} [d] \subseteq \bigcup_{d \in \rho_0(b)} [d]$$

According to Lemma 7.3, each element from the set  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  uniquely determines the configuration  $\langle L_i, n_1, n_2 \rangle$ , so that our construction of  $\rho_0(b)$  yields  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .  $\square$

## 8. FINITE APPROXIMATIONS IN INFINITE MODELS

At first sight, 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. [Calcagno et al. 2001].

However, the crucial difference between our setting and theirs is that their decidability result is restricted to *finite* valuations  $\rho$  such that  $\rho(p)$  is finite for every atomic proposition  $p$ . More precisely, in [Calcagno et al. 2001] each  $p$  represents one *cell*, i.e. a heap whose domain is a singleton. Their decidability result is nevertheless highly non-trivial because their language contains  $\neg*$  and the underlying separation model employs a non-total  $\circ$ , so that, e.g., whenever  $\llbracket A \rrbracket_\rho$  is *finite*,  $\llbracket A \neg* B \rrbracket_\rho$  becomes *infinite*. Below we investigate this phenomenon.

**LEMMA 8.1.** *Let  $(H, \circ, \{e_0\})$  be a memory model from Example 1.2. There is an algorithm that, for any finite valuation  $\rho$ , decides whether  $e_0 \models_\rho \kappa(M)$  holds or not.*

**PROOF.** For the sake of perspicuity, we consider here a heap model from Example 1.2(a). In principle, the result can be deduced from [Calcagno et al. 2001]. The following direct construction illustrates subtleties of the problem, caused by non-totality of the composition  $\circ$ .

As in Lemmas 7.6 and 7.12, we have to check whether  $e_0 \models_\rho \kappa(\gamma)$  for any machine instruction  $\gamma$  taken from Definition 6.1. For any such  $\gamma$ , the formula  $\kappa(\gamma)$  is of the form  $(\neg A \neg* \neg B)$  where  $A$  and  $B$  are each of the form  $l_i$ ,  $(l_i \vee l_j)$  or  $(l_i * p_k)$ . Note that, as  $\rho$  is a finite valuation,  $\llbracket A \rrbracket_\rho$  and  $\llbracket B \rrbracket_\rho$  are also finite.

Checking whether  $e_0 \models_\rho \kappa(\gamma)$  thus means checking whether the following sentence is true or not (cf. (7.5)):

$$\text{for all } h \in H, \llbracket A \rrbracket_\rho \cdot \{h\} \subseteq \rho(b) \text{ implies } \llbracket B \rrbracket_\rho \cdot \{h\} \subseteq \rho(b) \quad (29)$$

By negating this statement, it suffices to check whether or not there is an  $h \in H$  satisfying the following sentence:

$$\llbracket A \rrbracket_\rho \cdot \{h\} \subseteq \rho(b) \quad \text{and} \quad \llbracket B \rrbracket_\rho \cdot \{h\} \not\subseteq \rho(b) \quad (30)$$

The tricky point, caused by the fact that  $\circ$  is not total, is that the equation (30) may have an infinite number of solutions  $h$ . Nevertheless, we can construct, in advance, a finite but representative list of ‘solution candidates’ so that any  $h$ , a solution to (30), can be transformed into  $\tilde{h}$  which is still a solution to (30) but belongs to this finite list of ‘candidates’.

Let  $\llbracket A \rrbracket_\rho = \{f_1, f_2, \dots, f_m\}$  and  $\llbracket B \rrbracket_\rho = \{g_1, \dots, g_t\}$ . Given a solution  $h$  to (30), we consider two cases depending on whether or not  $\llbracket A \rrbracket_\rho \cdot \{h\} = \emptyset$ :

- **Case**  $\llbracket A \rrbracket_\rho \cdot \{h\} \neq \emptyset$ .

In this case, the product  $f_i \circ h$  must be defined for some  $i$ , and, by (30), we have  $f_i \circ h \in \rho(b)$ , which implies that  $h$  is a sub-heap of some heap from  $\rho(b)$ . Thus  $h$  is guaranteed to belong to the *finite set*  $\mathcal{H}_1$  of all sub-heaps from  $\rho(b)$ .

- **Case**  $\llbracket A \rrbracket_\rho \cdot \{h\} = \emptyset$ .

In this case, for each  $i$  such that  $1 \leq i \leq m$ , the product  $f_i \circ h$  is undefined and hence there is an  $\ell_i$  such that  $\ell_i \in \text{domain}(f_i) \cap \text{domain}(h)$ . Notice that, by (30), we have  $\llbracket B \rrbracket_\rho \cdot \{h\} \not\subseteq \rho(b)$ . Consequently, for some  $1 \leq j \leq t$ , we have

$$\text{domain}(g_j) \cap \{\ell_1, \ell_2, \dots, \ell_m\} = \emptyset \quad (31)$$

(Otherwise, for all  $j$  we have had that  $\text{domain}(g_j) \cap \{\ell_1, \ell_2, \dots, \ell_m\} \neq \emptyset$ , and thereby  $\text{domain}(g_j) \cap \text{domain}(h) \neq \emptyset$ , resulting in  $\llbracket B \rrbracket_\rho \cdot \{h\} = \emptyset$ , which contradicts to (30).)

Now we construct a new solution  $\tilde{h}$  in the following way. Let  $\tilde{a}$  be a fresh element of  $L$  such that  $\tilde{a}$  does not occur in  $\llbracket A \rrbracket_\rho$ ,  $\llbracket B \rrbracket_\rho$  or  $\rho(b)$ , and let  $\tilde{b}$  be a fixed element of  $RV$ . Then the domain of  $\tilde{h}$  is defined as the extended set  $\{\ell_1, \ell_2, \dots, \ell_m, \tilde{a}\}$ , and for each  $x$  from  $\text{domain}(\tilde{h})$  we set:

$$\tilde{h}(x) = \tilde{b}$$

It remains to show that  $\tilde{h}$  indeed satisfies the sentence (30).

Firstly, for all  $1 \leq i \leq m$ , we have  $\text{domain}(f_i) \cap \text{domain}(\tilde{h}) \neq \emptyset$  because  $\ell_i \in \text{domain}(\tilde{h})$  by construction. This implies that  $\llbracket A \rrbracket_\rho \cdot \{\tilde{h}\} = \emptyset$ .

Secondly, by (31),  $\text{domain}(g_j)$  and  $\text{domain}(\tilde{h})$  are disjoint for some  $1 \leq j \leq t$ , so that  $g_j \circ \tilde{h}$  is defined and, because of the choice of  $\tilde{a}$ , we have  $g_j \circ \tilde{h} \notin \rho(b)$ . Therefore, we can conclude that  $\tilde{h}$  is still a solution to (30).

Now we construct a *finite set*  $\mathcal{H}_2$  of ‘solution candidates’ as follows. For each choice of  $\ell_1, \ell_2, \dots, \ell_m$  from  $\text{domain}(f_1), \text{domain}(f_2), \dots, \text{domain}(f_m)$ , respectively,  $\mathcal{H}_2$  contains the heap  $h'$  defined as follows:

$$\text{domain}(h') = \{\ell_1, \ell_2, \dots, \ell_m, \tilde{a}\} \quad \text{and} \quad \forall x \in \text{domain}(h'). \quad h'(x) = \tilde{b}$$

□

Bringing the two cases together, we can conclude that if the equation (30) has a solution, then the equation (30) has a solution  $\tilde{h}$  belonging to the *finite set*  $\mathcal{H}_1 \cup \mathcal{H}_2$ .

In other words, the statement (29), in which the range of the quantified  $h$  is infinite, is equivalent to the decidable statement (32) below, in which the *finite* range  $\mathcal{H}_1 \cup \mathcal{H}_2$  of the

quantified  $h$  can be computed in advance:

$$\text{for all } h \in \mathcal{H}_1 \cup \mathcal{H}_2, \llbracket A \rrbracket_\rho \cdot \{h\} \subseteq \rho(b) \text{ implies } \llbracket B \rrbracket_\rho \cdot \{h\} \subseteq \rho(b) \quad (32)$$

The practical upshot of this transformation is that checking whether  $e_0 \models_\rho \kappa(\gamma)$  can be done in a finite number of steps.

**THEOREM 8.2.** *Let  $(H, \circ, \{e_0\})$  be a memory model from Example 1.2. Then there is an algorithm that, for any finite valuation  $\rho$ , and any sequent  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  of the form:*

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

*decides whether this sequent is valid under the valuation  $\rho$ .*

**PROOF.** Given an  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  we first use Lemma 8.1 to compute  $\llbracket \kappa(M) \rrbracket_\rho$  and  $\llbracket I \wedge \neg l_0 \rrbracket_\rho$  (note that each of these is either  $\{e_0\}$  or  $\emptyset$ ). If either of these sets is empty then trivially  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  is valid under the valuation  $\rho$ .

Otherwise,  $\llbracket \kappa(M) \rrbracket_\rho = \llbracket I \wedge \neg l_0 \rrbracket_\rho = \{e_0\}$ , so deciding the validity of  $\mathcal{F}_{M, \langle L_i, n_1, n_2 \rangle}$  under  $\rho$  means deciding the inclusion

$$\begin{aligned} & \llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \rrbracket_\rho \subseteq \rho(b) \\ \text{i.e. } & \llbracket \kappa(M) \rrbracket_\rho \cdot \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_\rho \cdot \llbracket (I \wedge \neg l_0) \rrbracket_\rho \subseteq \rho(b) \\ \text{i.e. } & \{e_0\} \cdot \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_\rho \cdot \{e_0\} \subseteq \rho(b) \\ \text{i.e. } & \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_\rho \subseteq \rho(b) \end{aligned}$$

which is straightforward since both  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_\rho$  and  $\rho(b)$  are finite.  $\square$

**COROLLARY 8.3.** *We can construct a sequent  $\mathcal{F}_{M, \langle L_1, n_0, 0 \rangle}$  of the form*

$$\kappa(M) * l_1 * p_1^{n_0} * (I \wedge \neg l_0) \vdash b$$

*so that, for each model  $(H, \circ, \{e_0\})$  from Example 1.2,  $\mathcal{F}_{M, \langle L_1, n_0, 0 \rangle}$  is not valid in this model (and hence the corresponding computation of  $M$  from  $\langle L_1, n_0, 0 \rangle$  does not terminate), but  $\mathcal{F}_{M, \langle L_1, n_0, 0 \rangle}$  is valid in this model under all finite valuations  $\rho$ .*

**PROOF.** Take  $M$  such that  $K_M = \{n \mid \langle L_1, n, 0 \rangle \Downarrow_M\}$  is undecidable (the existence of such machines is guaranteed by [Minsky 1967]).  $K_M$  is recursively enumerable. According to Theorem 6.8 and Proposition 3.2, we have:

$$\begin{aligned} K_M &= \{n \mid \mathcal{F}_{M, \langle L_1, n, 0 \rangle} \text{ provable in Minimal BBI}\} \\ &= \{n \mid \mathcal{F}_{M, \langle L_1, n, 0 \rangle} \text{ valid in all separation models}\} \\ &= \{n \mid \mathcal{F}_{M, \langle L_1, n, 0 \rangle} \text{ valid in } (H, \circ, \{e_0\})\} \end{aligned}$$

Now let  $(H, \circ, \{e_0\})$  be a model from Example 1.2, and define

$$W_M = \{n \mid \mathcal{F}_{M, \langle L_1, n, 0 \rangle} \text{ is not valid in } (H, \circ, \{e_0\}) \text{ under some finite valuation } \rho\}$$

By Theorem 8.2,  $W_M$  is also recursively enumerable.

However, by construction,  $K_M$  and  $W_M$  are disjoint. Moreover, since  $K_M$  is recursively enumerable but undecidable,  $W_M$  is not the whole complement of  $K_M$ . 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 7.14 implies that  $\mathcal{F}_{M, \langle L_1, n_0, 0 \rangle}$  is not valid in  $(H, \circ, \{e_0\})$ . However,  $n_0 \notin W_M$  implies that  $\mathcal{F}_{M, \langle L_1, n_0, 0 \rangle}$  is valid in  $(H, \circ, \{e_0\})$  under all finite valuations  $\rho$ .  $\square$

## 9. 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 the bunched logic Classical BI.

*Definition 9.1.* 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  $\cdot^{-1} : H \rightarrow H$  satisfies  $h \circ h^{-1} = e^{-1}$  for all  $h \in H$ .

The CBI-models we consider here form a subclass of the more general *relational* CBI-models given in [Brotherston and Calcagno 2010].

*Example 9.2.* Examples of CBI-models (cf. [Brotherston and Calcagno 2010]):

(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).  $(\Sigma, \circ, \varepsilon, \cdot^{-1})$  where  $\Sigma$  is any class of *languages* containing the empty language  $\varepsilon$  and closed under union  $\cup$  and complement  $\bar{\cdot}$ . 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.,  $\Sigma$  may be the class of regular languages, or the class of finite and co-finite sets.

(c). *Effect algebras* [Foulis and Bennett 1994], which arise in the foundations of quantum mechanics, are exactly CBI-models with indivisible units.

(d). *Permission algebras*  $(P, \bullet, 1)$  [Bornat et al. 2005] enriched with a ‘formal unit’  $e$  and ‘formal equalities’  $e \bullet h = h \bullet e = e$  can be shown to be exactly non-degenerate (i.e. having more than one element) CBI-models with indivisible units.

*Definition 9.3.* Following Definition 3.1, we introduce a second chain of logics as follows:

$$\text{BBI} \subseteq \text{CBI} \subseteq \text{CBI} + \text{eW} \subseteq \text{CBI} + \text{W}$$

- *Classical* BI, a.k.a. CBI [Brotherston and Calcagno 2010] is obtained from BBI by extending its language with a constant  $\tilde{I}$ , and adding the axiom  $\sim\sim A \vdash A$ , where  $\sim A$  is an abbreviation for  $(A \multimap \tilde{I})$ .
- CBI+eW is obtained by extending CBI with the restricted  $\ast$ -weakening  $(I \wedge (A \ast B)) \vdash A$ .
- CBI+W is obtained by extending CBI with the unrestricted  $\ast$ -weakening  $(A \ast B) \vdash A$ .  $\square$

Validity of CBI-formulas with respect to CBI-models  $(H, \circ, e, \cdot^{-1})$  is given by extending the satisfaction relation in Definition 2.9 with the clause:

$$h \models_{\rho} \tilde{I} \Leftrightarrow h \neq e^{-1}$$

**PROPOSITION 9.4.** *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.*

**PROOF.** Since CBI-models are special cases of separation models, and BBI is sound wrt. separation models (Proposition 3.2), it suffices to show for the first part that the CBI axiom  $\sim\sim A \vdash A$  is valid in an arbitrary CBI-model  $(H, \circ, e, \cdot^{-1})$ .

First, we observe that for any  $h \in H$ , the element  $h^{-1}$  is the *unique* element such that  $h \circ h^{-1} = e^{-1}$ . For, suppose that  $h \circ x = e^{-1}$ , then we have by associativity and commutativity of  $\circ$ :

$$e^{-1} \circ x = (h \circ h^{-1}) \circ x = (h \circ x) \circ h^{-1} = e^{-1} \circ h^{-1}$$

whence we obtain  $x = h^{-1}$  by cancellativity<sup>2</sup> of  $\circ$ . As an immediate corollary, we also have  $(h^{-1})^{-1} = h$ .

Now by the definition of validity above and recalling that  $\sim A =_{\text{def}} (A \multimap \tilde{I})$ , we have:

$$h \models_{\rho} \sim A \Leftrightarrow \forall h'. h \circ h' \text{ defined and } h' \models_{\rho} A \text{ implies } h \circ h' \neq e^{-1}$$

Since, by the above,  $h^{-1}$  is the unique element of  $H$  such that  $h \circ h^{-1} = e^{-1}$ , we obtain:

$$h \models_{\rho} \sim A \Leftrightarrow h^{-1} \not\models_{\rho} A$$

Thus, using the fact that  $(h^{-1})^{-1} = h$ , we have:

$$h \models_{\rho} \sim \sim A \Leftrightarrow h^{-1} \not\models_{\rho} \sim A \Leftrightarrow (h^{-1})^{-1} \models_{\rho} A \Leftrightarrow h \models_{\rho} A$$

and so  $\sim \sim A \vdash A$  (as well as its converse) is valid as required.

For soundness of CBI+eW, the fact that restricted \*-weakening holds in all CBI-models with indivisible units follows from the fact that it holds in all separation models with indivisible units, as established in Proposition 3.2.  $\square$

**COROLLARY 9.5.** *Interpreting  $\subset$  as strict inclusion between the set of sequents provable in each system using only the language of BBI, we have:*

$$\text{BBI} \subset \text{CBI} \subset \text{CBI+eW} \subset \text{CBI+W}$$

**PROOF.** The nonstrict versions of the inclusions hold easily by construction. The non-inclusion  $\text{CBI} \not\subseteq \text{BBI}$  was established in [Brotherston and Calcagno 2010]. By the same arguments as in Theorem 2.10 and Corollary 3.4, restricted \*-weakening is not valid in all CBI-models, and unrestricted \*-weakening is not valid in all CBI-models with indivisible units. Thus the non-inclusions  $\text{CBI+W} \not\subseteq \text{CBI+eW} \not\subseteq \text{CBI}$  hold by Proposition 9.4.  $\square$

**PROPOSITION 9.6.** *CBI+W is ordinary classical logic.*

**PROOF.** By the same argument as in Proposition 3.7, the presence of unrestricted \*-weakening forces the equivalences  $A * B \equiv A \wedge B$ ,  $A \multimap B \equiv A \rightarrow B$  and  $I \equiv \top$ .

It just remains to show that  $\tilde{I} \equiv \perp$ . To see this, first note that  $\perp \vdash \tilde{I}$  is trivially provable by the usual *ex falso quodlibet* of classical logic. For the reverse direction, first note that  $(\perp \multimap \tilde{I}) \vdash \top$  is trivially provable. As a consequence,  $(\top \multimap \tilde{I}) \vdash ((\perp \multimap \tilde{I}) \multimap \tilde{I})$  is also provable. By the CBI axiom, we have  $((\perp \multimap \tilde{I}) \multimap \tilde{I}) \vdash \perp$ , whence by transitivity of  $\vdash$  we obtain  $(\top \multimap \tilde{I}) \vdash \perp$ . By the equivalence  $I \equiv \top$ , we then have  $(I \multimap \tilde{I}) \vdash \perp$ . Since  $\tilde{I} \vdash (I \multimap \tilde{I})$  is easily derivable, we have  $\tilde{I} \vdash \perp$  provable by transitivity as required.

We observe that, as a consequence,  $\sim A \equiv A \rightarrow \perp \equiv \neg A$ .  $\square$

Since (Minimal) BBI-provability implies CBI-provability, to establish undecidability for CBI it suffices (see Figure 2) to prove the analogue of Theorem 7.8 for a CBI-model.

**Definition 9.7.** We introduce the *RAM-codomain 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 the union of disjoint sets (and undefined for non-disjoint sets), the unit  $e_0$  is  $\emptyset$  and  $\cdot^{-1}$  is set complement.

By extending the valuation  $\rho_0$  in Definition 7.2, we define a valuation  $\rho_C$  for  $(\mathcal{D}^+, \circ, e_0, \cdot^{-1})$  as follows:  $\rho_C$  coincides with  $\rho_0$  on all atomic propositions except  $b$ , and

$$\rho_C(b) = \rho_0(b) \cup \{d \in \mathcal{D}^+ \mid d \text{ is cofinite}\}$$

**LEMMA 9.8.**  $e_0 \models_{\rho_C} \kappa(M)$  for any machine  $M$ .

<sup>2</sup>This is the only point in our work for which cancellativity is required. The original definition of a CBI-model in [Brotherston and Calcagno 2010] requires as an axiom that the dual  $h^{-1}$  be unique for each  $h$ , whereas here we drop this requirement in favour of cancellativity, which is arguably more “natural”.

PROOF. As in Lemma 7.6, we must show  $e_0 \models_{\rho_C} \kappa(\gamma)$  for any instruction  $\gamma$ . Here we only examine the case of an increment instruction  $\gamma = (L_i: c_k++; \text{goto } L_j;)$  for  $k = 1$ . We note that we have, as in Lemma 7.6,

$$d \models_{\rho_C} \neg A \Leftrightarrow \forall d'. d, d' \text{ disjoint and } d' \models_{\rho_C} A \text{ implies } d \circ d' \in \rho_C(b) \quad (33)$$

We have  $\kappa(\gamma) = (\neg(l_j * p_1) \neg * \neg l_i)$ . To show  $e_0 \models_{\rho_C} \kappa(\gamma)$ , we must show for any  $d \in \mathcal{D}^+$ , as in the corresponding case of Lemma 7.6,

$$d \models_{\rho_C} \neg(l_j * p_1) \text{ implies } d \models_{\rho_C} \neg l_i$$

Assuming that  $d \models_{\rho_C} \neg(l_j * p_1)$ , there are two cases to consider.

First, if  $d$  is finite then we can find (as in the corresponding case of Lemma 7.6)  $d_j \in \rho_C(l_j)$  and  $d_1 \in \rho_C(p_1)$  such that  $d, d_j$  and  $d_1$  are disjoint, and thus in particular  $d_j \circ d_1 \models_{\rho_C} l_j * p_1$ . Using the equivalence (33), we have:

$$\begin{aligned} d \circ d_j \circ d_1 &\in \rho_C(b) \\ &= \rho_0(b) \cup \{d \in \mathcal{D}^+ \mid d \text{ is cofinite}\} \end{aligned}$$

Since  $d, d_j$  and  $d_1$  are finite, we must have  $d \circ d_j \circ d_1 \in \rho_0(b)$  and thus, by the same argument as in the corresponding case of Lemma 7.6,  $d \in \llbracket p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$  where  $\langle L_j, n_1 + 1, n_2 \rangle \Downarrow_M$  and thus  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$  by applying the instruction  $\gamma$ . Consequently,  $d \circ d' \in \rho_0(b) \subseteq \rho_C(b)$  for any  $d'$  such that  $d, d'$  are disjoint and  $d' \in \rho_C(l_i)$ . Thus by (33) we have  $d \models_{\rho_C} \neg l_i$  as required.

If, on the other hand,  $d$  is cofinite then, for any choice of  $d' \in \rho_C(l_i)$  with  $d, d'$  disjoint,  $d \circ d'$  must also be cofinite, in which case  $d \circ d' \in \rho_C(b)$  because  $\rho_C(b)$  contains *all* cofinite sets. Thus, again by (33), we have  $d \models_{\rho_C} \neg l_i$  as required.

The other cases follow from their analogues in Lemma 7.6 in a similar fashion.  $\square$

LEMMA 9.9.  $e_0 \models_{\rho_C} I \wedge \neg l_0$  for any machine  $M$ .

PROOF. Similar to Lemma 7.7.  $\square$

THEOREM 9.10. *If the sequent  $\kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \vdash b$  is valid in the model  $(\mathcal{D}^+, \circ, e_0, \cdot^{-1})$ , then  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .*

PROOF. By the definition of validity, and using Lemma 2.8, we have:

$$\begin{aligned} \llbracket \kappa(M) * l_i * p_1^{n_1} * p_2^{n_2} * (I \wedge \neg l_0) \rrbracket_{\rho_C} &\subseteq \rho_C(b) \\ \text{i.e. } \llbracket \kappa(M) \rrbracket_{\rho_C} \cdot \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_C} \cdot \llbracket I \wedge \neg l_0 \rrbracket_{\rho_C} &\subseteq \rho_C(b) \end{aligned}$$

By Lemmas 9.8 and 9.9 we have in particular:

$$\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_C} \subseteq \rho_C(b)$$

Since  $\rho_C$  coincides with  $\rho_0$  on all atomic propositions except  $b$ , we have  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_C} = \llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_0}$ , so that  $\langle L_i, n_1, n_2 \rangle$  is uniquely determined according to Lemma 7.3. In particular,  $\llbracket l_i * p_1^{n_1} * p_2^{n_2} \rrbracket_{\rho_C}$  is finite, so our construction of  $\rho_C(b)$  yields  $\langle L_i, n_1, n_2 \rangle \Downarrow_M$ .  $\square$

Again, based on Figure 2, we can assert the following:

COROLLARY 9.11. *The following properties of formulas are undecidable, even when restricted to the language  $(\wedge, \rightarrow, I, *, \neg)$  of Minimal BBI.<sup>3</sup>*

(a) *provability in CBI;*

<sup>3</sup>It seems that we are within the framework of Corollary 4.1, since the formulas are taken in the language  $(\wedge, \rightarrow, I, *, \neg)$ , which does not include the new CBI constant  $\tilde{I}$ . However, the problem remains non-trivial because of the fact that CBI is *not* a conservative extension of BBI.

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

PROOF. Similar to Corollary 4.1.  $\square$

COROLLARY 9.12. *Neither CBI nor CBI+eW has the finite model property.*

PROOF. Similar to Corollary 4.2.  $\square$

## 10. CONCLUDING REMARKS

Our main contribution is that separation logic, the logic of memory-manipulating programs, is undecidable even at the *purely propositional* level.

Moreover, we have established undecidability of validity in any particular memory model drawn from the literature on separation logic and its applications. Corollary 4.1(h) provides an infinite number of concrete undecidable models of practical and theoretical importance. That is, however we choose  $L$ ,  $RV$ ,  $S$  and  $P$  in Example 1.2 (with  $L$  infinite and  $RV$ ,  $S$ ,  $P$  possibly degenerate), we *always* get an undecidable model.

We also have established the undecidability of validity in various classes of separation models, and of provability in BBI and CBI and their siblings. In fact, to obtain a new exhibit for the ‘Undecidability Zoo’, we need add only *classical* conjunction and implication to the multiplicatives, without invoking any kind of negation and falsum (see Section 5).

From a technical point of view, we kill *all* our ‘undecidable birds’ with one stone – a direct encoding of Minsky machines, which single encoding suffices to cover all cases of interest (see Figure 2).

One of the above problems, namely whether or not BBI is decidable, was widely considered to be open for quite a long time. As we see, undecidability of BBI is an immediate corollary of our main results on separation logic. However, immediately prior to publication of our LICS paper, we discovered that undecidability of BBI can in fact be deduced from the undecidability of equational theories over certain algebras stated in [Kurucz et al. 1995]. An alternative proof of BBI undecidability has also been given independently in [Larchey-Wendling and Galmiche 2010].

However, our paper overlaps with [Kurucz et al. 1995] and [Larchey-Wendling and Galmiche 2010] *only* with respect to undecidability of BBI (which is obtained by very different techniques in all cases). In this paper, we give a direct proof not only of undecidability of BBI, but also of Minimal BBI, BBI+eW, CBI, CBI+eW, and of validity in various classes of separation models, as well as validity in any *particular* heap-like model of practical interest. There is no *ad hoc* connection between these problems and BBI.

### 10.1. New principles for separation logic

In proving our main undecidability results on separation logic, we have found a number of new principles which should be taken into account at the level of the multiplicative unit  $I$ .

- (i) Though *\*-contraction*,  $A \vdash A * A$ , is not generally valid in separation logic, in Lemma 5.3 we deduce the following *restricted \*-contraction*

$$I \wedge A \vdash A * A \tag{34}$$

- (ii) As for the systems BBI+eW and CBI+eW newly established in this paper, we remark that the memory models of theoretical/practical importance employed in the literature (cf. Example 1.2) all have indivisible units in the sense of Definition 2.4, which is



exactly axiomatized by the *restricted \*-weakening* principle of BBI+eW and CBI+eW (see Theorem 2.10):

$$I \wedge (A * B) \vdash A \quad (35)$$

This principle is in full accordance with the following law on ‘conservation of matter’:  
*“The empty memory cannot be split into non-empty pieces”*. □

## 10.2. Separation logic vs. linear logic

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

From the point of view of logical principles, there are clear differences between the two. E.g., distributivity of additive conjunction over disjunction:

$$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$$

holds even in BI but fails in linear logic. More specific to Boolean BI, the restricted \*-contraction:

$$I \wedge A \vdash A * A$$

holds even in Minimal BBI as shown by our Lemma 5.3, but this too fails in linear logic. Finally, while adding the unrestricted \*-weakening  $(A * B) \vdash A$  to linear logic gives us *affine logic*, adding it to BBI forces a collapse into classical logic (Proposition 3.7).

From a semantic perspective, the precise expression of properties of memory in separation logic is based on the fact that we have:

$$\llbracket A * B \rrbracket_\rho = \llbracket A \rrbracket_\rho \cdot \llbracket B \rrbracket_\rho$$

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* with respect to a certain closure operator  $Cl$ , which, in particular, violates the above exact equality. Indeed, the same is true of BI interpretations [O’Hearn and Pym 1999]. Not only is this less precise, it admits no possibility of finite valuations in these logics since, e.g., in linear logic even  $Cl(\emptyset)$  is always infinite. □

## 10.3. Undecidability vs. decidable fragments

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 [Berdine et al. 2004; Calcagno et al. 2001], 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 [Calcagno et al. 2001]) or restrict the formula language (as in [Berdine et al. 2004]). □

## ACKNOWLEDGMENTS

We are greatly indebted to Peter O’Hearn and Hongseok Yang for their fruitful discussions and suggestions. Thanks to Sam Staton for his observation on effect algebras [Foullis and Bennett 1994], and to Tadeusz Litak who has attracted our attention to [Kurucz et al. 1995].

## REFERENCES

- AHMED, A., JIA, L., AND WALKER, D. 2003. Reasoning about hierarchical storage. In *Proceedings of LICS-18*. IEEE Computer Society, 33–44.
- BENTON, N. P., BIERMAN, G. M., AND DE PAIVA AND MARTIN HYLAND, V. 1993. A term calculus for intuitionistic linear logic. In *Proceedings of TLCA*. Springer-Verlag, 75–90.
- BERDINE, J., CALCAGNO, C., AND O’HEARN, P. 2004. A decidable fragment of separation logic. In *Proceedings of FSTTCS*. LNCS Series, vol. 3328. Springer, 97–109.

- BORNAT, R., CALCAGNO, C., AND AND MATTHEW PARKINSON, P. O. 2005. Permission accounting in separation logic. In *Proceedings of POPL-32*. 59–70.
- BROTHERSTON, J. 2011. Bunched logics displayed. *Studia Logica: Special Issue on Recent Developments related to Residuated Lattices and Substructural Logics*. Awaiting publication.
- BROTHERSTON, J. AND CALCAGNO, C. 2009. Classical BI (A logic for reasoning about dualising resources). In *Proceedings of POPL-36*. ACM, 328–339.
- BROTHERSTON, J. AND CALCAGNO, C. 2010. Classical BI: Its semantics and proof theory. *Logical Methods in Computer Science* 6, 3.
- BROTHERSTON, J. AND KANOVICH, M. 2010. Undecidability of propositional separation logic and its neighbours. In *Proceedings of LICS-25*. IEEE Computer Society, 137–146.
- CALCAGNO, C., DISTEFANO, D., AND AND HONGSEOK YANG, P. O. 2011. Compositional shape analysis by means of bi-abduction. *Journal of the ACM* 58, 6.
- CALCAGNO, C., O’HEARN, P., AND YANG, H. 2007. Local action and abstract separation logic. In *Proceedings of LICS-22*. IEEE Computer Society, 366–378.
- CALCAGNO, C., YANG, H., AND O’HEARN, P. W. 2001. Computability and complexity results for a spatial assertion language for data structures. In *Proceedings of FSTTCS*. Springer-Verlag, 108–119.
- DISTEFANO, D. AND PARKINSON, M. 2008. jStar: Towards practical verification for Java. In *Proceedings of OOPSLA*. ACM, 213–226.
- DODDS, M., FENG, X., PARKINSON, M., AND VAFEIADIS, V. 2009. Deny-guarantee reasoning. In *Proceedings of 18th ESOP*. LNCS Series, vol. 5502. Springer-Verlag, 363–377.
- FOULIS, D. AND BENNETT, M. 1994. Effect algebras and unsharp quantum logics. *Foundations of Physics* 24, 1331–1352.
- GALMICHE, D. AND LARCHEY-WENDLING, D. 2006. Expressivity properties of Boolean BI through relational models. In *Proceedings of FSTTCS*. LNCS Series, vol. 4337. Springer, 357–368.
- GALMICHE, D., MÉRY, D., AND PYM, D. 2005. The semantics of BI and resource tableaux. *Mathematical Structures in Computer Science* 15, 1033–1088.
- GARDNER, P., MAFFEIS, S., AND SMITH, G. D. 2012. Towards a program logic for JavaScript. In *Proceedings of POPL*. 31–44.
- GIRARD, J.-Y. AND LAFONT, Y. 1987. Linear logic and lazy computation. In *Proceedings of TAPSOFT’87*. Springer-Verlag, 52–66.
- GOTSMAN, A., COOK, B., PARKINSON, M., AND KTOR VAFEIADIS, V. 2009. Proving that non-blocking algorithms don’t block. In *Proceedings of POPL-36*. ACM, 16–28.
- ISHTIAQ, S. AND O’HEARN, P. W. 2001. BI as an assertion language for mutable data structures. In *Proceedings of POPL-28*. ACM, 14–26.
- KANOVICH, M. 1992. Horn programming in linear logic is NP-complete. In *Proceedings of LICS-1992*. 200–210.
- KANOVICH, M. 1995. The direct simulation of Minsky machines in linear logic. In *Advances in Linear Logic*, J.-Y. Girard, Y. Lafont, and L. Regnier, Eds. London Mathematical Society Lecture Notes Series, vol. 222. Cambridge University Press, 123–145.
- KURUCZ, A., NEMETI, I., SAIN, I., AND SIMON, A. 1995. Decidable and undecidable modal logics with a binary modality. *Journal of Logic, Language and Information* 4, 191–206.
- LARCHEY-WENDLING, D. AND GALMICHE, D. 2010. The undecidability of Boolean BI through phase semantics. In *Proceedings of LICS-25*. IEEE Computer Society, 140–149.
- MINSKY, M. 1967. *Computation: finite and infinite machines*. Prentice-Hall, Inc.
- O’HEARN, P. W. AND PYM, D. J. 1999. The logic of bunched implications. *Bulletin of Symbolic Logic* 5, 2, 215–244.
- PARKINSON, M. AND BIERMAN, G. 2008. Separation logic, abstraction and inheritance. In *Proceedings of POPL-35*. ACM, 75–86.
- PARKINSON, M., BORNAT, R., AND O, C. C. 2006. Variables as resource in Hoare logics. In *Proceedings of LICS-21*. IEEE Computer Society, 137–146.
- PYM, D. 2002. *The Semantics and Proof Theory of the Logic of Bunched Implications*. Applied Logic Series. Kluwer. Errata and remarks (Pym 2008) maintained at <http://www.abdn.ac.uk/~csc335/BI-monograph-errata.pdf>.
- PYM, D., O’HEARN, P., AND YANG, H. 2004. Possible worlds and resources: The semantics of BI. *Theoretical Computer Science* 315, 1, 257–305.
- REYNOLDS, J. C. 2002. Separation logic: A logic for shared mutable data structures. In *Proceedings of LICS-17*. IEEE Computer Society, 55–74.

- YANG, H., LEE, O., BERDINE, J., CALCAGNO, C., OOK, B. C., DISTEFANO, D., AND O'HEARN, P. 2008. Scalable shape analysis for systems code. In *Proceedings of CAV*. LNCS Series, vol. 5123. Springer, 385–398.