Theory of Computing Systems manuscript No.
(will be inserted by the editor)

Separation Logic with One Quantified Variable

Stéphane Demri · Didier Galmiche · Dominique Larchey-Wendling · Daniel Méry

Received: date / Accepted: date

Abstract We investigate first-order separation logic with one record field restricted to a unique quantified variable (1SL1). Undecidability is known when the number of quantified variables is unbounded and the satisfiability problem is PSPACE-complete for the propositional fragment. We show that the satisfiability problem for 1SL1 is PSPACE-complete and we characterize its expressive power by showing that every formula is equivalent to a Boolean combination of atomic properties. This contributes to our understanding of fragments of first-order separation logic that can specify properties about the memory heap

Work partially supported by the ANR grant DynRes (project No. ANR-11-BS02-011) and by the EU Seventh Framework Programme under grant agreement No. PIOF-GA-2011-301166 (DATAVERIF). Revised and complete version of (Demri et al 2014).

S. Demri
LSV, ENS Cachan, CNRS, Université Paris-Saclay, 94235 Cachan, France
Tel.: +33 (0) 1 47 40 75 46
Fax.: +33 (0) 1 47 40 75 21
E-mail: demri@lsv.fr

D. Galmiche
LORIA & Université de Lorraine, BP 239, 54506 Vandœuvre-lès-Nancy, France
Tel.: +33 (0) 3 83 59 20 15
Fax.: +33 (0) 3 83 41 30 79
E-mail: didier.galmiche@loria.fr

D. Larchey-Wendling
LORIA & CNRS, BP 239, 54506 Vandœuvre-lès-Nancy, France
Tel.: +33 (0) 3 54 95 85 14
Fax.: +33 (0) 3 83 41 30 79
E-mail: dominique.larchey-wendling@loria.fr

D. Méry
LORIA & Université de Lorraine, BP 239, 54506 Vandœuvre-lès-Nancy, France
Tel.: +33 (0) 3 54 95 85 14
Fax.: +33 (0) 3 83 41 30 79
E-mail: daniel.mery@loria.fr

of programs with singly-linked lists. All the fragments we consider contain the magic wand operator and first-order quantification over a single variable.

Keywords Separation logic · Quantifier elimination · Model Checking · Satisfiability · Computational Complexity

1 Introduction

1.1 Separation Logic for Verifying Programs with Pointers

Separation logic (Reynolds 2002) is a well-known logic for analysing programs with pointers stemming from BI logic (Ishtiaq and O'Hearn 2001). Such programs have specific errors to be detected and separation logic is used as an assertion language for Hoare-like proof systems (Reynolds 2002) that are dedicated to verify programs manipulating heaps. Any procedure mechanizing the proof search requires subroutines that check the satisfiability or the validity of formulæ (more precisely entailment) from the assertion language. That is why, characterizing the computational complexity of separation logic and its fragments and designing optimal decision procedures remain essential tasks. Separation logic contains a structural separating connective and its adjoint (the separating implication, also known as the magic wand).

Concise and modular proofs can be derived using these connectives, since they can express properties such as non-aliasing (Reynolds 2002).

The main concern of the paper is to study a non-trivial fragment of first-order separation logic with one record field as far as expressive power, decidability and complexity are concerned. Herein, the models of separation logic are pairs made of a variable valuation (store) and a partial function with finite domain (heap), also known as memory states.

1.2 Decidability and Complexity Results for Separation Logic

The complexity of satisfiability and model-checking problems for separation logic fragments have been thoroughly studied (Calcagno et al 2001; Reynolds 2002; Cook et al 2011) (see also new decidability results in (Iosif et al 2013) or undecidability results in (Brotherston and Kanovich 2010; Larchey-Wendling and Galmiche 2010) in an alternative setting). Separation logic is equivalent to a Boolean propositional logic (Lozes 2004a,b) if first-order quantifiers are disabled. Separation logic without first-order quantifiers is decidable, but it becomes undecidable with first-order quantifiers (Calcagno et al 2001). For instance, model-checking and satisfiability for propositional separation logic are PSPACE-complete problems (Calcagno et al 2001). Decidable fragments with first-order quantifiers can be found in (Galmiche and Méry 2010; Brochenin et al 2012).

In order to study decidability or complexity issues for separation logic, two tracks have been observed in the literature. There is the verification approach

with decision procedures for fragments of practical use, see e.g. (Berdine et al 2005; Cook et al 2011; Haase et al 2013). Alternatively, fragments, extensions or variants of separation logic are considered from a logical viewpoint, see e.g. (Calcagno et al 2001; Brotherston and Kanovich 2010; Larchey-Wendling and Galmiche 2010) (see also Brotherston et al (2014); Antonopoulos et al (2014) for extensions with inductive predicates).

1.3 Our Contributions

In this paper, we study first-order separation logic with one quantified variable, with an unbounded number of program variables and with one record field (herein called 1SL1).

1. We introduce *test formulæ* that state simple properties about the memory states and we show that every formula in 1SL1 is equivalent to a Boolean combination of test formulæ. Moreover, we provide a simple algorithm to compute from a formula in 1SL1, an equivalent Boolean combination of test formulæ.
   For instance, separating connectives can be eliminated in a controlled way as well as first-order quantification over the single variable. In that way, we show a quantifier elimination property similar to the one for Presburger arithmetic (in that case, the test formulæ are linear and periodicity constraints). This result extends previous ones on propositional separation logic (Lozes 2004a,b; Brochenin et al 2009) and as far as we know, this is the first time that this approach is extended to a first-order version of separation logic with the magic wand operator.
2. Even though Boolean combinations of test formulæ and 1SL1 have identical expressive power, the complexity of model-checking Boolean combinations of test formulæ and the complexity of model-checking 1SL1 formulæ are different. Indeed we show that the satisfiability problem for Boolean combinations of test formulæ is NP-complete whereas we establish that model-checking and satisfiability problems for 1SL1 are PSPACE-complete. NP-completeness is shown thanks to a saturation algorithm for the theory of memory states with test formulæ, paving the way to use SMT solvers to decide 1SL1; see e.g. the use of such solvers in (Pérez and Rybalchenko 2013; Piskac et al 2013, 2014; Bansal et al 2015).
   The difference between these two complexities here is due to the conciseness of 1SL1. PSPACE-completeness is still a relatively low complexity but this result can be extended with more than one record field (but still with one quantified variable). This is the best we can hope for with one quantified variable and with the magic wand, that is notoriously known to easily increase computational complexity. Indeed, 1SL with two quantified variables and no program variables (1SL2) has been recently shown undecidable in (Demri and Deters 2014). Of course, other extensions of 1SL1 could be considered, for instance to add a bit of arithmetical constraints, but herein

we focus on 1SL1 that is theoretically nicely designed, even though it is still unclear how much 1SL1 is useful for formal verification.
By way of comparison with first-order predicate logic, propositional calculus is NP-complete and FO1 (first-order logic with one variable) is NP-complete too. We knew that propositional separation logic is PSPACE-complete and herein we also establish that 1SL1 satisfiability has the same worst-case complexity.

Structure of the paper. Section 2 is mainly dedicated to preliminary definitions about separation logic, basic properties that can be expressed in 1SL1 and several preliminary definitions and results about partitions on memory states. In Section 3, test formulæ are introduced as well as several relations between locations based on such test formulae. The second part of this section contains a series of technical lemmas that are useful to establish the correctness of the abstraction based on test formulæ, which is shown in Section 4. Decidability of 1SL1 satisfiability problem as well as admissibility of quantifiers are established in Section 4 too. Section 4 concludes by showing that the satisfiability and model-checking problems can be solved in polynomial space. In Section 5, we prove a result stated earlier in the paper, namely the satisfiability status of conjunctions of literals made of test formulae can be decided in polynomial time. Strictly speaking, this result is not used to establish complexity results about 1SL1 (and that is why, its proof has been postponed a bit) but it paves the way to decide 1SL1 with SMT solvers, this is at least our hope. Whereas Section 6 contains concluding remarks, a technical appendix concludes the paper and it contains the proofs that are not present in the main body of the paper.

2 Separation Logic 1SL and its Heap Memory Model

2.1 First-Order Separation Logic with One Selector 1SL

Let $\mathtt{PVAR}$ be a countably infinite set of *program variables* and $\mathtt{FVAR}$ be a countably infinite set of *quantified variables*. We write $\mathtt{x}, \mathtt{y}, \ldots, \mathtt{x}_1, \mathtt{x}_2, \ldots$ to denote program variables and $\mathtt{u}, \mathtt{u}_1, \mathtt{u}_2, \ldots$ to denote quantified variables. A *memory state* (also called a *model*) is a pair (s, h) such that

- s is a variable valuation of the form $s : \mathtt{PVAR} \to \mathbb{N}$ (the *store*),
- h is a partial function $h : \mathbb{N} \to \mathbb{N}$ with finite domain (the *heap*) and we write $\mathrm{dom}(h)$ to denote its *domain* and $\mathrm{ran}(h)$ to denote its *range*.

Two heaps h_1 and h_2 are said to be *disjoint*, noted $h_1 \perp h_2$, if their domains are disjoint; when this holds, we write $h_1 \uplus h_2$ to denote the heap corresponding to the disjoint union of the graphs of h_1 and h_2, hence $\mathrm{dom}(h_1 \uplus h_2) = \mathrm{dom}(h_1) \uplus \mathrm{dom}(h_2)$. When the domains of h_1 and h_2 are not disjoint, the composition $h_1 \uplus h_2$ is not defined even if h_1 and h_2 have the same values on $\mathrm{dom}(h_1) \cap \mathrm{dom}(h_2)$.

The *empty heap*, denoted $\square$, is the only heap with an empty domain. It is a neutral element for $\uplus$: we have $h \uplus \square = \square \uplus h = h$ for any heap h. The heap h_1 is a *subheap* of h_2, noted $h_1 \sqsubseteq h_2$, if $\mathrm{dom}(h_1) \subseteq \mathrm{dom}(h_2)$ and $h_1(l) = h_2(l)$ for any $l \in \mathrm{dom}(h_1)$. Alternatively, we say that the heap h_2 is an *extension* of the heap h_1. Obviously, $h_1 \sqsubseteq h_1 \uplus h_2$ whenever the composition is defined. Moreover, any extension of the heap h_1 has the form $h_1 \uplus h_2$ for some heap h_2. A *heap is atomic* if its domain is a singleton set. We write $[l_1 \mapsto l_2]$ for the unique atomic heap h such that $\mathrm{dom}(h) = \{l_1\}$ and $h(l_1) = l_2$.

Formulæ in 1SL are built from *expressions* (composed of either program or quantified variables) and *atomic formulæ* (either *equality* tests or *points-to*). Formulæ in 1SL are closed under Boolean connectives, first-order quantification (as in first-order classical logic) but also under *separating conjunction* $*$, its *unit* $\mathtt{emp}$, and the *separating implication* $-\!\!*$ usually called the *magic wand*.

Definition 2.1 The *1SL formulæ* are defined by the following grammar:

$$\begin{array}{rcl} \mathtt{e} & ::= & \mathtt{x} \mid \mathtt{u} \quad \text{where } \mathtt{x} \in \mathtt{PVAR},\ \mathtt{u} \in \mathtt{FVAR} \\ \pi & ::= & \mathtt{e} = \mathtt{e}' \mid \mathtt{e} \hookrightarrow \mathtt{e}' \\ \mathcal{A} & ::= & \pi \mid \bot \mid \mathtt{emp} \mid \mathcal{A} \wedge \mathcal{B} \mid \neg\mathcal{A} \mid \mathcal{A} * \mathcal{B} \mid \mathcal{A} -\!\!* \mathcal{B} \mid \exists \mathtt{u}\, \mathcal{A} \quad \text{where } \mathtt{u} \in \mathtt{FVAR}. \end{array}$$

We make use of standard notations for the derived connectives of classical logic. The *size* of a formula $\mathcal{A}$, written $|\mathcal{A}|$, is defined as the number of (logical) nodes of its syntactic tree, or equivalently by the following recursive equations: $|\pi| = |\bot| = |\mathtt{emp}| \stackrel{def}{=} 1$, $|\mathcal{A} \wedge \mathcal{B}| = |\mathcal{A} * \mathcal{B}| = |\mathcal{A} -\!\!* \mathcal{B}| \stackrel{def}{=} 1 + |\mathcal{A}| + |\mathcal{B}|$, and $|\neg\mathcal{A}| = |\exists \mathtt{u}\, \mathcal{A}| \stackrel{def}{=} 1 + |\mathcal{A}|$. An *assignment* is a map $\mathfrak{f} : \mathtt{FVAR} \to \mathbb{N}$. The satisfaction relation $\models$ is parameterized by assignments (the obvious clauses for Boolean connectives are omitted):

- $(s,h) \models_\mathfrak{f} \mathtt{e} = \mathtt{e}'$ iff $[\![\mathtt{e}]\!] = [\![\mathtt{e}']\!]$ where $[\![\mathtt{x}]\!] \stackrel{def}{=} s(\mathtt{x})$ and $[\![\mathtt{u}]\!] \stackrel{def}{=} \mathfrak{f}(\mathtt{u})$.
- $(s,h) \models_\mathfrak{f} \mathtt{e} \hookrightarrow \mathtt{e}'$ iff $[\![\mathtt{e}]\!] \in \mathrm{dom}(h)$ and $h([\![\mathtt{e}]\!]) = [\![\mathtt{e}']\!]$.
- $(s,h) \models_\mathfrak{f} \mathtt{emp}$ iff $h = \square$.
- $(s,h) \models_\mathfrak{f} \mathcal{A}_1 * \mathcal{A}_2$ iff $(s,h_1) \models_\mathfrak{f} \mathcal{A}_1$ and $(s,h_2) \models_\mathfrak{f} \mathcal{A}_2$ hold for some heaps h_1 and h_2 such that $h = h_1 \uplus h_2$.
- $(s,h) \models_\mathfrak{f} \mathcal{A}_1 -\!\!* \mathcal{A}_2$ iff for any heap h', if $h \perp h'$ and $(s,h') \models_\mathfrak{f} \mathcal{A}_1$ then $(s, h \uplus h') \models_\mathfrak{f} \mathcal{A}_2$.
- $(s,h) \models_\mathfrak{f} \exists \mathtt{u}\, \mathcal{A}$ iff there is $l \in \mathbb{N}$ such that $(s,h) \models_{\mathfrak{f}[\mathtt{u} \mapsto l]} \mathcal{A}$ where $\mathfrak{f}[\mathtt{u} \mapsto l]$ is the assignment equal to $\mathfrak{f}$ except that $\mathtt{u}$ takes the value l.

Whereas '$\exists \mathtt{u}$' is clearly a first-order quantifier, the connectives $*$ and $-\!\!*$ are known to be second-order quantifiers. In the paper, we show how to eliminate these three connectives when only one quantified variable is used. The logic 1SL is not minimal for its expressive power; e.g. $\mathtt{emp}$ is logically equivalent to $\forall \mathtt{u}\ \neg((\mathtt{u} \hookrightarrow \mathtt{u}) -\!\!* \bot)$.

Proposition 2.2 *Let $s, s' : \mathtt{PVAR} \to \mathbb{N}$ be two stores, $h : \mathbb{N} \rightharpoonup \mathbb{N}$ be a heap, $\mathfrak{f}, \mathfrak{f}' : \mathtt{FVAR} \to \mathbb{N}$ be two assignments and let $\mathcal{A}$ be a 1SL formula. If $s(\mathtt{x}) = s'(\mathtt{x})$ holds for every program variable $\mathtt{x}$ that occurs in $\mathcal{A}$ and $\mathfrak{f}(\mathtt{u}) = \mathfrak{f}'(\mathtt{u})$ holds for any quantified variable $\mathtt{u}$ that occurs freely in $\mathcal{A}$ then the equivalence $(s,h) \models_\mathfrak{f} \mathcal{A}$ iff $(s',h) \models_{\mathfrak{f}'} \mathcal{A}$ holds.*

The proof is by an easy verification and it is left to the reader.

As a consequence of Proposition 2.2, we might abusively use the notation $(s,h) \models_{\mathfrak{f}} \mathcal{A}$ when the "store" $s : \{\mathtt{x}_1, \ldots, \mathtt{x}_q\} \to \mathbb{N}$ is only defined on a superset of the program variables that occur in $\mathcal{A}$, because the interpretation of the program variables that do not occur in $\mathcal{A}$ does not matter.

Proposition 2.3 *Let $\varphi : \mathbb{N} \to \mathbb{N}$ be permutation on locations, i.e. a one-to-one map. For any 1SL formula $\mathcal{A}$, any memory state (s,h) and any assignment $\mathfrak{f} : \mathtt{FVAR} \to \mathbb{N}$, we have $(s,h) \models_{\mathfrak{f}} \mathcal{A}$ iff $(\varphi \circ s, \varphi \circ h \circ \varphi^{-1}) \models_{\varphi \circ \mathfrak{f}} \mathcal{A}$.*

In the above statement, it is worth noting that the domain of $\varphi \circ h \circ \varphi^{-1}$ is $\varphi(\mathrm{dom}(h))$. The proof of Proposition 2.3 is left to the reader.

Since models for a formula $\mathcal{A}$ are closed under permutations on locations, without any loss of generality, we can assume that every variable $\mathtt{x}$ in $\mathcal{A}$ (being either a program variable or a freely occurring quantified variable) is interpreted by a value less than m, where $\mathcal{A}$ contains at most m program or free variables (see Corollary 2.4 below).

Corollary 2.4 *Let $\mathcal{A}$ be an 1SL formula, $\mathcal{V} \subseteq \mathtt{PVAR}$ and $\mathcal{F} \subseteq \mathtt{FVAR}$ be finite subsets such that $\mathcal{V}$ contains the program variables that occur in $\mathcal{A}$ and $\mathcal{F}$ contains the quantified variables that occur freely in $\mathcal{A}$. If there exist a memory state (s,h) and an assignment $\mathfrak{f} : \mathtt{FVAR} \to \mathbb{N}$ such that $(s,h) \models_{\mathfrak{f}} \mathcal{A}$, then there exist a memory state (s',h') and an assignment $\mathfrak{f}' : \mathtt{FVAR} \to \mathbb{N}$ such that $(s',h') \models_{\mathfrak{f}'} \mathcal{A}$ and $s'(\mathcal{V}) \cup \mathfrak{f}'(\mathcal{F}) \subseteq \{0,1,\ldots,m-1\}$ with $m = \mathrm{card}(\mathcal{V}) + \mathrm{card}(\mathcal{F})$.*

Proof The set $s(\mathcal{V}) \cup \mathfrak{f}(\mathcal{F})$ is a finite subset of $\mathbb{N}$. Its cardinal is less than or equal to $m = \mathrm{card}(\mathcal{V}) + \mathrm{card}(\mathcal{F})$. Hence, there exists a permutation on locations $\varphi : \mathbb{N} \to \mathbb{N}$ such that $\varphi(s(\mathcal{V}) \cup \mathfrak{f}(\mathcal{F})) \subseteq \{0,1,\ldots,m-1\}$. We apply Proposition 2.3 and get $s' = \varphi \circ s$, $h' = \varphi \circ h \circ \varphi^{-1}$ and $\mathfrak{f}' = \varphi \circ \mathfrak{f}$. □

We write 1SL0 to denote the propositional fragment of 1SL where no variable from $\mathtt{FVAR}$ occurs. Similarly, we write 1SL1 to denote the fragment of 1SL restricted to a single quantified variable, say $\mathtt{u}$. In that case, the satisfaction relation can be denoted by $\models_l$ where l is understood as the value $l = \mathfrak{f}(\mathtt{u})$ of the quantified variable $\mathtt{u}$ under the assignment $\mathfrak{f}$.

Let $\mathfrak{L}$ be a logic among 1SL, 1SL1 and 1SL0. As usual, the *satisfiability problem* for $\mathfrak{L}$ takes as input a formula $\mathcal{A}$ in $\mathfrak{L}$ and asks whether there is a memory state (s,h) and an assignment $\mathfrak{f}$ such that $(s,h) \models_{\mathfrak{f}} \mathcal{A}$. The *model-checking problem* for $\mathfrak{L}$ takes as input a formula $\mathcal{A}$ in $\mathfrak{L}$, a memory state (s,h) and an assignment $\mathfrak{f}$ for free variables from $\mathcal{A}$ and asks whether $(s,h) \models_{\mathfrak{f}} \mathcal{A}$.

When checking the satisfiability status of a formula $\mathcal{A}$ in 1SL1, we assume that its program variables are contained in $\{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$ for some $q \geqslant 1$ and the quantified variable is $\mathtt{u}$. So, $\mathtt{PVAR}$ is unbounded but as usual, when dealing with a specific formula, the set of program variables is finite.

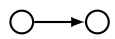
Theorem 2.5 *(Calcagno et al 2001; Brochenin et al 2012; Demri and Deters 2014) The satisfiability and model-checking problems for 1SL0 are* PSPACE*-complete. The satisfiability problem for 1SL is undecidable, even when restricted to two quantified variables.*

2.2 A First Glimpse of Properties that can be Stated in 1SL1

The logic 1SL1 allows to express different types of properties on memory states. The examples below indeed illustrate the expressivity of 1SL1 and in the paper we characterize precisely what can be expressed in 1SL1:

- the variable $\mathtt{e}$ is allocated in the heap:

$$\mathtt{alloc(e)} \stackrel{def}{=} (\mathtt{e} \hookrightarrow \mathtt{e}) \mathrel{-\!\!*} \bot$$

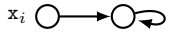


  where $\mathtt{e}$ is either $\mathtt{x}_i$ or $\mathtt{u}$. Intuitively: it is not possible to add a loop at $\mathtt{e}$;[1]
- the program variables $\mathtt{x}_i$ and $\mathtt{x}_j$ have the same value (without explicit equality predicate and with only one quantified variable):

$$\forall \mathtt{u}\ (\mathtt{u} \hookrightarrow \mathtt{x}_i) \mathrel{-\!\!*} (\mathtt{u} \hookrightarrow \mathtt{x}_j)$$

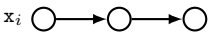
- the variable $\mathtt{x}_i$ points to a location that is a loop:

$$\mathtt{toloop}(\mathtt{x}_i) \stackrel{def}{=} \exists \mathtt{u}\, (\mathtt{x}_i \hookrightarrow \mathtt{u} \wedge \mathtt{u} \hookrightarrow \mathtt{u})$$



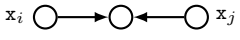
- the variable $\mathtt{x}_i$ points to a location that is allocated:

$$\mathtt{toalloc}(\mathtt{x}_i) \stackrel{def}{=} \exists \mathtt{u}\, \big(\mathtt{x}_i \hookrightarrow \mathtt{u} \wedge \mathtt{alloc}(\mathtt{u})\big)$$



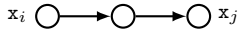
- the variables $\mathtt{x}_i$ and $\mathtt{x}_j$ point to a shared location:

$$\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \stackrel{def}{=} \exists \mathtt{u}\, (\mathtt{x}_i \hookrightarrow \mathtt{u} \wedge \mathtt{x}_j \hookrightarrow \mathtt{u})$$



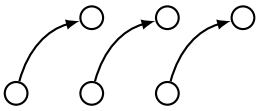
- there is a location between $\mathtt{x}_i$ and $\mathtt{x}_j$:

$$\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \stackrel{def}{=} \exists \mathtt{u}\ (\mathtt{x}_i \hookrightarrow \mathtt{u} \wedge \mathtt{u} \hookrightarrow \mathtt{x}_j)$$



- the domain of the heap has at least k elements:

$$\#\,\mathtt{dom} \geqslant k \stackrel{def}{=} \neg\mathtt{emp} * \cdots * \neg\mathtt{emp}$$



  where $\neg\mathtt{emp}$ occurs k times (in the example, $k = 3$ on the right-hand side of the equality symbol);
- the memory heap has exactly one memory cell at address $\mathtt{x}_i$ (expressed in 1SL0):

$$\mathtt{atomic}(\mathtt{x}_i) \stackrel{def}{=} \mathtt{alloc}(\mathtt{x}_i) \wedge \neg(\#\,\mathtt{dom} \geqslant 2)$$

- the location interpreted by $\mathtt{x}_i$ has no predecessor:

$$\#\,\mathtt{pred}(\mathtt{x}_i) = 0 \stackrel{def}{=} \neg(\exists \mathtt{u}\ \mathtt{u} \hookrightarrow \mathtt{x}_i)$$

[1] It is of course possible and perhaps more intuitive to define $\mathtt{alloc(e)}$ using a quantifier by $\exists \mathtt{u}\, (\mathtt{e} \hookrightarrow \mathtt{u})$, but that definition would not be correct for $\mathtt{alloc(u)}$ because we are limited to the use of one quantified variable.

– the location interpreted by $\mathtt{x}_i$ has exactly one predecessor:

$$\#\,\mathtt{pred}(\mathtt{x}_j) = 1 \stackrel{def}{=} (\exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{x}_i) \wedge \neg(\exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{x}_i * \exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{x}_i)$$

– the location interpreted by $\mathtt{x}_i$ has exactly $k > 0$ predecessors:

$$\#\,\mathtt{pred}(\mathtt{x}_j) = k \stackrel{def}{=} (\#\,\mathtt{pred}(\mathtt{x}_j) = 1) * \cdots * (\#\,\mathtt{pred}(\mathtt{x}_j) = 1)$$

where $\#\,\mathtt{pred}(\mathtt{x}_j) = 1$ occurs k times;

– the heap contains at least three self-loops:

$$\#\,\mathtt{loop} \geqslant 3 \stackrel{def}{=} (\exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{u}) * (\exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{u}) * (\exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{u}).$$

We also illustrate briefly the expressive power of 1SL2, i.e. with two quantified variables. But be aware that 1SL2 is strictly more expressive than 1SL1; see for instance Corollary 4.12. Moreover, 1SL2 is proved non recursively enumerable in (Demri and Deters 2014) whereas in this paper we show that 1SL1 is PSPACE-complete. In 1SL2, it is possible to express the existence of paths/lists within memory states:

– the heap is composed of exactly a path of strictly positive length from $\mathtt{x}_i$ to $\mathtt{x}_j$ together with an arbitrary number of cycles, written $\mathtt{ls}'(\mathtt{x}_i, \mathtt{x}_j)$:

$$\begin{array}{l} \#\,\mathtt{pred}(\mathtt{x}_i) = 0 \wedge \mathtt{alloc}(\mathtt{x}_i) \\ \wedge\ \#\,\mathtt{pred}(\mathtt{x}_j) = 1 \wedge \neg\mathtt{alloc}(\mathtt{x}_j) \\ \wedge\ \forall \mathtt{u}_1\ (\#\,\mathtt{pred}(\mathtt{u}_1) = 0 \wedge \mathtt{alloc}(\mathtt{u}_1)) \Rightarrow \mathtt{u}_1 = \mathtt{x}_i \\ \wedge\ \forall \mathtt{u}_1\ (\#\,\mathtt{pred}(\mathtt{u}_1) \neq 0 \wedge \mathtt{u}_1 \neq \mathtt{x}_j) \Rightarrow (\#\,\mathtt{pred}(\mathtt{u}_1) = 1 \wedge \mathtt{alloc}(\mathtt{u}_1)) \end{array}$$

– there is a path from $\mathtt{x}_i$ to $\mathtt{x}_j$ can be expressed by

$$\mathtt{ls}(\mathtt{x}_i, \mathtt{x}_j) \stackrel{def}{=} (\mathtt{x}_i = \mathtt{x}_j) \vee \big(\mathtt{ls}'(\mathtt{x}_i, \mathtt{x}_j) * \top\big)$$

to be found originally in (Brochenin et al 2012, Lemma 2.4); a similar property was established for graph logics in (Dawar et al 2007).

2.3 Decomposition and Graphical Representation

We fix a finite set of q distinct program variables $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\} \subseteq \mathtt{PVAR}$; to each memory state (s, h), we associate several subsets of locations that together define two partitions of $\mathrm{dom}(h)$. *Beware that these subsets and partitions depend on the choice of q and $\mathcal{V}$*:

– one partition takes care of self-loops and predecessors of interpretations of program variables;
– the other one takes care of locations which are "close" to the interpretations of program variables; see below.

This allows us to decompose the heap domains in such a way that we can easily identify the properties that can be indeed expressed by the formulæ of 1SL1 that contain only the program variables of $\mathcal{V}$.

We introduce a first partition of the heap domain by distinguishing the selfloops and the predecessors of variable interpretations on the one hand, and the remaining locations in the domain on the other hand:

$$\mathrm{pred}(s,h,i) \stackrel{\text{def}}{=} \big\{l \mid h(l) = s(\mathtt{x}_i)\big\} \quad \mathrm{pred}(s,h) \stackrel{\text{def}}{=} \mathrm{pred}(s,h,1) \cup \cdots \cup \mathrm{pred}(s,h,q)$$
$$\mathrm{loop}(s,h) \stackrel{\text{def}}{=} \big\{l \mid h(l) = l\big\} \qquad \mathrm{rem}(s,h) \stackrel{\text{def}}{=} \mathrm{dom}(h)\backslash\big(\mathrm{pred}(s,h) \cup \mathrm{loop}(s,h)\big)$$

where $\mathrm{pred}(s,h,i)$ is defined for any $i \in [1,q]$. From the definition of $\mathrm{rem}(s,h)$, we derive the obvious identity $\mathrm{dom}(h) = \mathrm{rem}(s,h) \uplus \big(\mathrm{pred}(s,h) \cup \mathrm{loop}(s,h)\big)$. However, the sets $\mathrm{pred}(s,h)$ and $\mathrm{loop}(s,h)$ are not necessarily disjoint. As a consequence of h being a partial function, the sets $\mathrm{pred}(s,h,i)$ and $\mathrm{pred}(s,h,j)$ intersect only if $s(\mathtt{x}_i) = s(\mathtt{x}_j)$, in which case the identity $\mathrm{pred}(s,h,i) = \mathrm{pred}(s,h,j)$ holds.

We introduce a second partition of $\mathrm{dom}(h)$ by distinguishing the locations related to a memory cell involving a program variable interpretation on the one hand, and the remaining locations in the domain on the other hand. So, the sets below depend also implicitly on $\mathcal{V}$:

$$\mathrm{ref}(s,h) \stackrel{\text{def}}{=} \mathrm{dom}(h) \cap s(\mathcal{V}) \qquad \heartsuit(s,h) \stackrel{\text{def}}{=} \mathrm{ref}(s,h) \cup \mathrm{acc}(s,h)$$
$$\mathrm{acc}(s,h) \stackrel{\text{def}}{=} \mathrm{dom}(h) \cap h\big(s(\mathcal{V})\big) \qquad \overline{\heartsuit}(s,h) \stackrel{\text{def}}{=} \mathrm{dom}(h)\backslash\heartsuit(s,h).$$

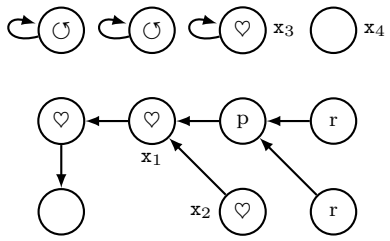
The *core* of the memory state (s,h), written $\heartsuit(s,h)$, contains the locations l in $\mathrm{dom}(h)$ such that either l is the interpretation of a program variable or it is an image by h of a program variable (that is also in the domain). We have $\mathrm{ref}(s,h) \subseteq s(\mathcal{V})$, whence $\mathrm{card}(\mathrm{ref}(s,h)) \leqslant q$. Since h is a partial function, from $\mathrm{acc}(s,h) \subseteq h(s(\mathcal{V}))$ we deduce $\mathrm{card}(\mathrm{acc}(s,h)) \leqslant q$. As a consequence, we have $\mathrm{card}(\heartsuit(s,h)) \leqslant 2q$. Hence, the core of a memory heap is always a "small part" of the domain, small meaning of bounded size when q is fixed.

In the sequel, we need to consider locations that belong to the intersection of those sets from different partitions. Here are their formal definitions:

$$\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \stackrel{\text{def}}{=} \mathrm{pred}(s,h,i)\backslash\heartsuit(s,h) \qquad \mathrm{pred}_{\heartsuit}(s,h,i) \stackrel{\text{def}}{=} \mathrm{pred}(s,h,i) \cap \heartsuit(s,h)$$
$$\mathrm{pred}_{\overline{\heartsuit}}(s,h) \stackrel{\text{def}}{=} \mathrm{pred}(s,h)\backslash\heartsuit(s,h) \qquad \mathrm{pred}_{\heartsuit}(s,h) \stackrel{\text{def}}{=} \mathrm{pred}(s,h) \cap \heartsuit(s,h)$$
$$\mathrm{loop}_{\overline{\heartsuit}}(s,h) \stackrel{\text{def}}{=} \mathrm{loop}(s,h)\backslash\heartsuit(s,h) \qquad \mathrm{loop}_{\heartsuit}(s,h) \stackrel{\text{def}}{=} \mathrm{loop}(s,h) \cap \heartsuit(s,h)$$
$$\mathrm{rem}_{\overline{\heartsuit}}(s,h) \stackrel{\text{def}}{=} \mathrm{rem}(s,h)\backslash\heartsuit(s,h) \qquad \mathrm{rem}_{\heartsuit}(s,h) \stackrel{\text{def}}{=} \mathrm{rem}(s,h) \cap \heartsuit(s,h).$$

For instance, $\mathrm{pred}_{\overline{\heartsuit}}(s,h)$ contains the set of locations l in $\mathrm{dom}(h)$ that are predecessors of a variable interpretation but no program variable $\mathtt{x}_i$ in $\{\mathtt{x}_1,\ldots,\mathtt{x}_q\}$ satisfies $s(\mathtt{x}_i) = l$ or $h(s(\mathtt{x}_i)) = l$ (which means $l \notin \heartsuit(s,h)$).

We insist on the fact that the definitions of subsets like $\mathrm{ref}(s,h)$, $\mathrm{acc}(s,h)$ and $\heartsuit(s,h)$ and hence $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$, $\mathrm{pred}_{\heartsuit}(s,h,i)$, $\mathrm{loop}_{\overline{\heartsuit}}(s,h)$, ... depend on a particular choice of q and $\mathcal{V} = \{\mathtt{x}_1,\ldots,\mathtt{x}_q\}$ even though *our notation does not reflect that dependency*. We made this choice for the sake of readability.



This graph represents an example of memory state (s,h) with the variables $\mathtt{x}_1,\ldots,\mathtt{x}_4$. Nodes labelled by '♡' belong to $\heartsuit(s,h)$; those labelled by '↺' belong to $\mathrm{loop}_{\overline{\heartsuit}}(s,h)$; those labelled by 'p' belong to $\mathrm{pred}_{\overline{\heartsuit}}(s,h)$ and those labelled by 'r' belong to $\mathrm{rem}_{\overline{\heartsuit}}(s,h)$.

Fig. 2.1 The decomposition of a memory state

However, in most of developments of this paper, and unless stated otherwise, we assume a fixed choice for q and $\mathcal{V}$.

A memory state (s,h) restricted to the finite set of program variables $\mathtt{x}_1,\ldots,\mathtt{x}_q$ can be represented by a *finite graph* encoding the (graph) representation of the heap h. Moreover, each variable $\mathtt{x}_i$ labels the location $s(\mathtt{x}_i)$, which may add a few more nodes in the case $s(\mathtt{x}_i)$ does not belong to the domain and codomain of h. The graph of Figure 2.1 illustrates the previous definitions of subsets on a simple memory state.

The introduction of the above sets provides a canonical way to decompose the heap domains, which will be helpful in the sequel.

Lemma 2.6 (Canonical decomposition) *For all stores s and all heaps h, the following identity holds:*

$$\mathrm{dom}(h) = \heartsuit(s,h) \uplus \mathrm{pred}_{\overline{\heartsuit}}(s,h) \uplus \mathrm{loop}_{\overline{\heartsuit}}(s,h) \uplus \mathrm{rem}_{\overline{\heartsuit}}(s,h).$$

The proof is by straightforward verification using the fact that $\mathrm{pred}(s,h) \cap \mathrm{loop}(s,h) \subseteq \heartsuit(s,h)$.

Proposition 2.7 $\{\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \mid i \in [1,q]\}$ *is a partition of* $\mathrm{pred}_{\overline{\heartsuit}}(s,h)$.

The (easy) proof is left to the reader.

Remember that both $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) = \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ and $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) = \varnothing$ are possible. Below, we present properties about the canonical decomposition.

Proposition 2.8 (Canonical decomposition and splitting) *Let us assume s,h,h_1,h_2 such that $h = h_1 \uplus h_2$. With the notation $\overline{X}$ for $\mathbb{N}\backslash X$, the identities*

$$\heartsuit(s,h) \cap \mathrm{dom}(h_1) = \heartsuit(s,h_1) \uplus \Delta(s,h_1,h_2)$$
$$\heartsuit(s,h) \cap \mathrm{dom}(h_2) = \heartsuit(s,h_2) \uplus \Delta(s,h_2,h_1)$$

hold with $\Delta(s,h_1,h_2) \stackrel{\text{def}}{=} \mathrm{dom}(h_1) \cap h_2(s(\mathcal{V})) \cap \overline{s(\mathcal{V})} \cap \overline{h_1(s(\mathcal{V}))}$.

The proof is by Boolean computations.

The set $\Delta(s, h_1, h_2)$ contains the locations belonging to the core of h and to the domain of h_1, without being in the core of h_1. Its expression in Proposition 2.8 uses only basic set-theoretical operations. From Proposition 2.8, we conclude that $\heartsuit(s, h_1 \uplus h_2)$ can be different from $\heartsuit(s, h_1) \uplus \heartsuit(s, h_2)$.

Proposition 2.9 *Let s, h, h_1, h_2 be such that $h = h_1 \uplus h_2$ and let $i \in [1, q]$. The following identities hold:*

1. $\mathrm{pred}_{\overline{\heartsuit}}(s, h_1, i) = (\mathrm{pred}_{\overline{\heartsuit}}(s, h, i) \cap \mathrm{dom}(h_1)) \uplus (\mathrm{pred}(s, h, i) \cap \Delta(s, h_1, h_2))$;
2. $\mathrm{loop}_{\overline{\heartsuit}}(s, h_1) = (\mathrm{loop}_{\overline{\heartsuit}}(s, h) \cap \mathrm{dom}(h_1)) \uplus (\mathrm{loop}(s, h) \cap \Delta(s, h_1, h_2))$;
3. $\mathrm{rem}_{\overline{\heartsuit}}(s, h_1) = (\mathrm{rem}_{\overline{\heartsuit}}(s, h) \cap \mathrm{dom}(h_1)) \uplus (\mathrm{rem}(s, h) \cap \Delta(s, h_1, h_2))$.

The proof can be found in Appendix A starting at page 52.

Remark that $\uplus$ is commutative, hence symmetric identities hold for the subsets $\mathrm{pred}_{\overline{\heartsuit}}(s, h_2, i)$, $\mathrm{loop}_{\overline{\heartsuit}}(s, h_2)$ and $\mathrm{rem}_{\overline{\heartsuit}}(s, h_2)$. The following propositions describe the changes that occur in the canonical decomposition of a memory state when exactly one memory cell is added to the heap. Recall that we write $[l_1 \mapsto l_2]$ for the unique atomic heap h such that $\mathrm{dom}(h) = \{l_1\}$ and $h(l_1) = l_2$. Below, we write $\mathfrak{p}\heartsuit(s, h)$ to denote the set $s(\mathcal{V}) \cup h(s(\mathcal{V}))$.

Proposition 2.10 *Let (s, h) be a memory state, $l_1 \in \mathbb{N} \backslash \mathrm{dom}(h)$ and $l_2 \in \mathbb{N}$. Let us write $h_{1\to 2}$ for $h \uplus [l_1 \mapsto l_2]$ and let i be in $[1, q]$. The following identities hold:*

$$
\begin{aligned}
\mathrm{dom}(h_{1\to 2}) &= \mathrm{dom}(h) \uplus \{l_1\} \\
\mathrm{pred}(s, h_{1\to 2}, i) &= \mathrm{pred}(s, h, i) \uplus \begin{cases} \{l_1\} & \text{if } l_2 = s(\mathtt{x}_i) \\ \varnothing & \text{if } l_2 \neq s(\mathtt{x}_i) \end{cases} \\
\mathrm{loop}(s, h_{1\to 2}) &= \mathrm{loop}(s, h) \uplus \begin{cases} \{l_1\} & \text{if } l_1 = l_2 \\ \varnothing & \text{if } l_1 \neq l_2 \end{cases} \\
\mathrm{rem}(s, h_{1\to 2}) &= \mathrm{rem}(s, h) \uplus \begin{cases} \{l_1\} & \text{if } l_2 \notin s(\mathcal{V}) \cup \{l_1\} \\ \varnothing & \text{if } l_2 \in s(\mathcal{V}) \cup \{l_1\} \end{cases}
\end{aligned}
$$

$$
\heartsuit(s, h_{1\to 2}) = \heartsuit(s, h) \uplus \begin{cases} \{l_1, l_2\} & \text{if } l_1 \in s(\mathcal{V}),\ l_2 \in \mathrm{dom}(h) \text{ and } l_2 \notin \heartsuit(s, h) \\ \{l_1\} & \text{if } l_1 \in s(\mathcal{V}) \text{ and } (l_2 \notin \mathrm{dom}(h) \text{ or } l_2 \in \heartsuit(s, h)) \\ \{l_1\} & \text{if } l_1 \notin s(\mathcal{V}) \text{ and } l_1 \in h(s(\mathcal{V})) \\ \varnothing & \text{if } l_1 \notin \mathfrak{p}\heartsuit(s, h). \end{cases}
$$

The proof can be found in Appendix A starting at page 52.

Proposition 2.11 *Let (s, h) be a memory state, $l_1 \in \mathbb{N} \backslash \mathrm{dom}(h)$ and $l_2 \in \mathbb{N}$. Let us write $h_{1\to 2}$ for $h \uplus [l_1 \mapsto l_2]$ and let i be in $[1, q]$. The following identities hold:*

$$\mathrm{pred}_{\overline{\heartsuit}}(s,h_{1\to 2},i) = \begin{cases} \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \uplus \{l_1\} & \textit{if } l_1 \notin \mathfrak{p}\heartsuit(s,h) \textit{ and } l_2 = s(\mathtt{x}_i) \\ \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) - \{l_2\} & \textit{if } l_1 \in s(\mathcal{V}) \textit{ and } l_2 \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \\ \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) & \textit{otherwise} \end{cases}$$

$$\mathrm{loop}_{\overline{\heartsuit}}(s,h_{1\to 2}) = \begin{cases} \mathrm{loop}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\} & \textit{if } l_1 \notin \mathfrak{p}\heartsuit(s,h) \textit{ and } l_1 = l_2 \\ \mathrm{loop}_{\overline{\heartsuit}}(s,h) - \{l_2\} & \textit{if } l_1 \in s(\mathcal{V}) \textit{ and } l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h) \\ \mathrm{loop}_{\overline{\heartsuit}}(s,h) & \textit{otherwise} \end{cases}$$

$$\mathrm{rem}_{\overline{\heartsuit}}(s,h_{1\to 2}) = \begin{cases} \mathrm{rem}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\} & \textit{if } l_1 \notin \mathfrak{p}\heartsuit(s,h) \textit{ and } l_2 \notin s(\mathcal{V}) \cup \{l_1\} \\ \mathrm{rem}_{\overline{\heartsuit}}(s,h) - \{l_2\} & \textit{if } l_1 \in s(\mathcal{V}) \textit{ and } l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h) \\ \mathrm{rem}_{\overline{\heartsuit}}(s,h) & \textit{otherwise} \end{cases}$$

where $X - \{l_2\}$ means that the location l_2 already belongs to the set X and is (strictly) removed from it.

The proof can be found in Appendix A starting at page 53.

2.4 How to Count in 1SL1

In this section, let us consider a fixed memory state (s,h) and a fixed location l. We explain how to measure the cardinal of some finite sets of locations using 1SL1 formulæ, in particular those of the form $X\backslash\heartsuit(s,h)$ where X is one of the sets among $\mathrm{pred}(s,h,j)$, $\mathrm{loop}(s,h)$ and $\mathrm{rem}(s,h)$. The ground idea is the following: using the identity

$$X = \big(X \cap \heartsuit(s,h)\big) \uplus \big(X\backslash\heartsuit(s,h)\big)$$

the cardinal of $X\backslash\heartsuit(s,h)$ can be obtained from the cardinal of X and the cardinal of $X \cap \heartsuit(s,h)$ (by expressing their difference in 1SL1).

In 1SL1, it is easy to detect if there is one element in either $\mathrm{pred}(s,h,j)$ or $\mathrm{loop}(s,h)$ using the formulæ

$$\#\,\mathtt{pred}(\mathtt{x}_j) \geqslant 1 \stackrel{\text{def}}{=} \exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{x}_j \qquad \text{and} \qquad \#\,\mathtt{loop} \geqslant 1 \stackrel{\text{def}}{=} \exists \mathtt{u}\, \mathtt{u} \hookrightarrow \mathtt{u}.$$

Hence, using the separating conjunction $*$, we can measure the cardinal of $\mathrm{pred}(s,h,j)$ or $\mathrm{loop}(s,h)$ with

$$\begin{array}{rcll} \#\,\mathtt{pred}(\mathtt{x}_j) \geqslant k & \stackrel{\text{def}}{=} & \#\,\mathtt{pred}(\mathtt{x}_j) \geqslant 1 * \cdots * \#\,\mathtt{pred}(\mathtt{x}_j) \geqslant 1 & \text{repeated } k \text{ times} \\ \#\,\mathtt{loop} \geqslant k & \stackrel{\text{def}}{=} & \#\,\mathtt{loop} \geqslant 1 * \cdots * \#\,\mathtt{loop} \geqslant 1 & \text{repeated } k \text{ times} \end{array}$$

This encoding works smoothly thanks to the identities $\mathrm{pred}(s, h_1 \uplus h_2, j) = \mathrm{pred}(s,h_1,j) \uplus \mathrm{pred}(s,h_2,j)$, $\mathrm{loop}(s, h_1 \uplus h_2) = \mathrm{loop}(s,h_1) \uplus \mathrm{loop}(s,h_2)$.

Let us explain how to evaluate the cardinal of $\mathrm{rem}(s,h)$. To do so, we define the following 1SL1 formulæ for every $\mathtt{e} \in \{\mathtt{x}_1, \ldots, \mathtt{x}_q\} \cup \{\mathtt{u}\}$:

$$\begin{array}{rcl} \mathtt{inrem}(\mathtt{e}) & \stackrel{\text{def}}{=} & \mathtt{alloc}(\mathtt{e}) \wedge \neg \mathtt{e} \hookrightarrow \mathtt{e} \wedge \bigwedge_{j \in [1,q]} \neg \mathtt{e} \hookrightarrow \mathtt{x}_j \\ \mathtt{torem}(\mathtt{x}_i) & \stackrel{\text{def}}{=} & \mathtt{toalloc}(\mathtt{x}_i) \wedge \neg \mathtt{toloop}(\mathtt{x}_i) \wedge \bigwedge_{j \in [1,q]} \neg \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \end{array}$$

Proposition 2.12 *With* $[\![\mathtt{u}]\!] = l$ *and* $[\![\mathtt{x}_i]\!] = s(\mathtt{x}_i)$*, we have:*

$$\begin{array}{lll}(s,h) \models_l \mathtt{inrem(e)} & \textit{iff} & [\![\mathtt{e}]\!] \in \mathrm{rem}(s,h)\\ (s,h) \models_l \mathtt{torem}(\mathtt{x}_i) & \textit{iff} & h(s(\mathtt{x}_i)) \in \mathrm{rem}(s,h)\end{array}$$

The proof is left to the reader.

Hence, the formula $\#\,\mathtt{rem} \geqslant 1 \stackrel{def}{=} \exists \mathtt{u}\, \mathtt{inrem}(\mathtt{u})$ detects if $\mathrm{rem}(s,h)$ is non-empty. Using the separating conjunction $*$, we can measure the cardinal of $\mathrm{rem}(s,h)$ in 1SL1 by

$$\#\,\mathtt{rem} \geqslant k \stackrel{def}{=} \#\,\mathtt{rem} \geqslant 1 * \cdots * \#\,\mathtt{rem} \geqslant 1 \quad \text{repeated } k \text{ times}$$

because the identity $\mathrm{rem}(s, h_1 \uplus h_2) = \mathrm{rem}(s,h_1) \uplus \mathrm{rem}(s,h_2)$ holds too.

To count the number of elements in e.g. $\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$, it is thus sufficient to count the number of elements in $\mathrm{pred}_{\heartsuit}(s,h,j)$ and then use the identity

$$\mathrm{pred}(s,h,j) = \mathrm{pred}_{\heartsuit}(s,h,j) \uplus \mathrm{pred}_{\overline{\heartsuit}}(s,h,j).$$

However, splitting $\mathrm{pred}_{\heartsuit}(s,h,j)$ into k parts using the separating conjunction like we did for $\mathrm{pred}(s,h,j)$ will not work because $\heartsuit(s, h_1 \uplus h_2)$ is not necessarily equal to $\heartsuit(s,h_1) \uplus \heartsuit(s,h_2)$; see Proposition 2.8 for instance.

By contrast, one possible trick consists in directly enumerating the elements in $\mathrm{pred}(s,h,j) \cap \mathrm{ref}(s,h)$ and in $\mathrm{pred}(s,h,j) \cap \big(\mathrm{acc}(s,h) \backslash \mathrm{ref}(s,h)\big)$ and then to use the identity

$$\mathrm{pred}_{\heartsuit}(s,h,j) = \big(\mathrm{pred}(s,h,j) \cap \mathrm{ref}(s,h)\big) \uplus \big(\mathrm{pred}(s,h,j) \cap (\mathrm{acc}(s,h) \backslash \mathrm{ref}(s,h))\big).$$

For any subset $I \subseteq [1,q]$, let us define the 1SL1 formulæ below

$$\begin{array}{l}\mathtt{ref}_I \stackrel{def}{=} \bigwedge_{i \neq j \in I} \neg(\mathtt{x}_i = \mathtt{x}_j) \wedge \bigwedge_{i \in I} \mathtt{alloc}(\mathtt{x}_i)\\ \mathtt{acc}_I \stackrel{def}{=} \bigwedge_{i \neq j \in I} \neg\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \wedge \bigwedge_{i \in I} \mathtt{toalloc}(\mathtt{x}_i) \wedge \bigwedge_{i \in I, j \in [1,q]} \neg(\mathtt{x}_i \hookrightarrow \mathtt{x}_j)\end{array}$$

Proposition 2.13 *For* $I \subseteq [1,q]$ *with* $sI \stackrel{def}{=} \{s(\mathtt{x}_i) \mid i \in I\}$*, we have*

$$\begin{array}{lll}(s,h) \models_l \mathtt{ref}_I & \textit{iff} & sI \subseteq \mathrm{ref}(s,h) \textit{ and } \mathrm{card}(sI) = \mathrm{card}(I)\\ (s,h) \models_l \mathtt{acc}_I & \textit{iff} & h(sI) \subseteq \mathrm{acc}(s,h) \backslash \mathrm{ref}(s,h) \textit{ and } \mathrm{card}(h(sI)) = \mathrm{card}(I)\end{array}$$

The proof is left to the reader.

Hence, $\mathtt{ref}_I$ holds iff sI is a subset of $\mathrm{ref}(s,h)$ of size $\mathrm{card}(I)$ and thus $\mathrm{card}(I)$ gives a lower bound for the cardinal of $\mathrm{ref}(s,h)$. Moreover, $\mathtt{acc}_I$ provides us a way to give a lower bound for the cardinal of $\mathrm{acc}(s,h) \backslash \mathrm{ref}(s,h)$.

To illustrate the usefulness of $\mathtt{ref}_I$ and $\mathtt{acc}_I$, we show how to measure the cardinal of the core. Using the identity $\heartsuit(s,h) = \mathrm{ref}(s,h) \uplus \big(\mathrm{acc}(s,h) \backslash \mathrm{ref}(s,h)\big)$ we can express the fact that $\heartsuit(s,h)$ has cardinal at least k by

$$\#\,\mathtt{core} \geqslant k \stackrel{def}{=} \bigvee \big\{ \mathtt{ref}_R \wedge \mathtt{acc}_A \;\big|\; R \cup A \subseteq [1,q] \text{ and } \mathrm{card}(R) + \mathrm{card}(A) \geqslant k \big\}.$$

But we can easily measure the cardinal of subsets of the core such as e.g. $\mathrm{pred}_{\heartsuit}(s,h,j)$. Its cardinal is at least k iff the following formula is satisfied

$$\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_j) \geqslant k \stackrel{\text{def}}{=} \bigvee \left\{ \begin{array}{l} \mathtt{ref}_R \wedge \bigwedge_{r\in R} \mathtt{x}_r \hookrightarrow \mathtt{x}_j \wedge \\ \mathtt{acc}_A \wedge \bigwedge_{a\in A} \mathtt{btwn}(\mathtt{x}_a, \mathtt{x}_j) \end{array} \middle| \begin{array}{c} R \cup A \subseteq [1,q] \text{ and} \\ \mathrm{card}(R) + \mathrm{card}(A) \geqslant k \end{array} \right\}$$

and, using $\mathrm{pred}(s,h,j) = \mathrm{pred}_{\heartsuit}(s,h,j) \uplus \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ we conclude that the cardinal of $\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ is at least k iff the following formula is satisfied

$$\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant k \stackrel{\text{def}}{=} \bigvee_{p\leqslant 2q} \#\,\mathtt{pred}(\mathtt{x}_j) \geqslant k+p \wedge \neg \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_j) \geqslant p+1.$$

Notice that we can stop at $2q$ because of the inclusion $\mathrm{pred}_{\heartsuit}(s,h,j) \subseteq \heartsuit(s,h)$ and the upper bound $\mathrm{card}(\heartsuit(s,h)) \leqslant 2q$.

Similarly, measuring the sizes of $\mathrm{loop}_{\heartsuit}(s,h)$ and $\mathrm{rem}_{\heartsuit}(s,h)$ is done with

$$\#\,\mathtt{loop}_{\heartsuit} \geqslant k \stackrel{\text{def}}{=} \bigvee \left\{ \begin{array}{l} \mathtt{ref}_R \wedge \bigwedge_{r\in R} \mathtt{x}_r \hookrightarrow \mathtt{x}_r \wedge \\ \mathtt{acc}_A \wedge \bigwedge_{a\in A} \mathtt{toloop}(\mathtt{x}_a) \end{array} \middle| \begin{array}{c} R \cup A \subseteq [1,q] \text{ and} \\ \mathrm{card}(R) + \mathrm{card}(A) \geqslant k \end{array} \right\}$$

$$\#\,\mathtt{rem}_{\heartsuit} \geqslant k \stackrel{\text{def}}{=} \bigvee \left\{ \begin{array}{l} \mathtt{ref}_R \wedge \bigwedge_{r\in R} \mathtt{inrem}(\mathtt{x}_r) \wedge \\ \mathtt{acc}_A \wedge \bigwedge_{a\in A} \mathtt{torem}(\mathtt{x}_a) \end{array} \middle| \begin{array}{c} R \cup A \subseteq [1,q] \text{ and} \\ \mathrm{card}(R) + \mathrm{card}(A) \geqslant k \end{array} \right\}$$

and using the partitions $\mathrm{loop}(s,h) = \mathrm{loop}_{\heartsuit}(s,h) \uplus \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ and $\mathrm{rem}(s,h) = \mathrm{rem}_{\heartsuit}(s,h) \uplus \mathrm{rem}_{\overline{\heartsuit}}(s,h)$, we can measure the sizes of $\mathrm{loop}_{\overline{\heartsuit}}(s,h)$ and $\mathrm{rem}_{\overline{\heartsuit}}(s,h)$ with

$$\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k \stackrel{\text{def}}{=} \bigvee_{p\leqslant 2q} \#\,\mathtt{loop} \geqslant k+p \wedge \neg \#\,\mathtt{loop}_{\heartsuit} \geqslant p+1$$

$$\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k \stackrel{\text{def}}{=} \bigvee_{p\leqslant 2q} \#\,\mathtt{rem} \geqslant k+p \wedge \neg \#\,\mathtt{rem}_{\heartsuit} \geqslant p+1.$$

Lemma 2.14 *For any $k \geqslant 1$ and for any $i \in [1,q]$, there exist 1SL1 formulæ denoted $\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant k$, $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ and $\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k$ respectively such that, for any memory state (s,h) and for any location $l \in \mathbb{N}$ the following equivalences hold:*

1. $(s,h) \models_l \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant k$ *iff* $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant k$;
2. $(s,h) \models_l \#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ *iff* $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant k$;
3. $(s,h) \models_l \#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k$ *iff* $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant k$.

The proof can be found in Appendix A starting at page 55.

We conclude this section with a remark about the encodings that we propose for the test formulæ $\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant k$, $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ and $\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k$. Their sizes are obviously exponential in both q and k. However neither the way they are encoded nor their sizes have an influence on the complexity characterisations that we establish in Sections 4.4 and 5. We could even have used test formulæ which are outside the expressive power of 1SL1 to obtain a complexity upper bound, as soon as the added test formulæ do not prevent the proofs of the lemmas of distributivity 3.33, of compositionality 3.34 and of

existence 3.35.[2] This means that it is still possible to strictly extend 1SL1 with specific atomic formulæ that would not trigger the decidability bottleneck that occurs with two quantified variables in 1SL2.

2.5 Equipotence for Comparing Cardinalities

We introduce the notion of *equipotence* and we state several properties about it. This will be useful in the forthcoming developments.

Definition 2.15 We say that x/y *respect* X/Y if $(x \in X \text{ iff } y \in Y)$ holds.

Proposition 2.16 *Let us consider four sets $X \subseteq X'$ and $Y \subseteq Y'$. Let $R \subseteq X' \times Y'$ be a bijective relation between X' and Y'. Let us assume that for any x, y, if $x\ R\ y$ then x/y respect X/Y. Then $R \cap X \times Y$ is a bijective relation between X and Y.*

The proof is left to the reader.

Definition 2.17 (α-equipotence) Let $\alpha \in \mathbb{N}$. We say that two finite sets X and Y are α-*equipotent* and we write $X \sim_\alpha Y$ if, either $\mathrm{card}(X) = \mathrm{card}(Y)$ or both $\mathrm{card}(X)$ and $\mathrm{card}(Y)$ are greater than α. We say that two finite sets X and Y are *equipotent* and we write $X \sim_\infty Y$ when $\mathrm{card}(X) = \mathrm{card}(Y)$.

Proposition 2.18 *For any $\alpha \in \mathbb{N}$ and any finite sets X and Y, the following conditions are equivalent:*

1. $X \sim_\alpha Y$;
2. $(\mathrm{card}(X) \geqslant k$ *iff* $\mathrm{card}(Y) \geqslant k)$ *for any* $k \leqslant \alpha$;
3. $(\mathrm{card}(X) = \mathrm{card}(Y) < \alpha)$ *or* $(\mathrm{card}(X) \geqslant \alpha$ *and* $\mathrm{card}(Y) \geqslant \alpha)$;
4. $\min(\mathrm{card}(X), \alpha) = \min(\mathrm{card}(Y), \alpha)$.

The proof is left to the reader.

It is thus obvious that $X \sim_\alpha Y$ holds whenever $\mathrm{card}(X) = \mathrm{card}(Y)$, i.e. $\sim_\infty \subseteq \sim_\alpha$. The equipotence relation is also decreasing, i.e. $\sim_{\alpha_2} \subseteq \sim_{\alpha_1}$ holds for all $\alpha_1 \leqslant \alpha_2$. It is easy to verify $\sim_\infty = \bigcap_\alpha \sim_\alpha$. Hence the notation $\sim_\infty$ is consistent with the intuitive idea of a downward limit. We state below two lemmas that will be helpful in the sequel.

Lemma 2.19 *Let $\alpha \in \mathbb{N}$ and X, X', Y, Y' be finite sets such that $X \cap X' = \varnothing$, $Y \cap Y' = \varnothing$, $X \sim_\alpha Y$ and $X' \sim_\infty Y'$ hold. Then $X \uplus X' \sim_{\alpha+n} Y \uplus Y'$ holds where $n = \mathrm{card}(X')$.*

The proof is left to the reader.

Proposition 2.20 *Let X and Y be two finite sets and let $x \notin X$ and $y \notin Y$. If the equipotence $X \uplus \{x\} \sim_{\alpha+1} Y \uplus \{y\}$ holds then $X \sim_\alpha Y$ holds.*

[2] But in that case, we would have lost the expressivity characterisation of Theorem 4.11.

The proof is left to the reader.

Lemma 2.21 *Let $\alpha_1, \alpha_2 \in \mathbb{N}$ and X, X', Y_0 be finite sets such that $X \uplus X' \sim_{\alpha_1+\alpha_2} Y_0$ holds. Then there are two finite sets Y, Y' such that $Y_0 = Y \uplus Y'$, $X \sim_{\alpha_1} Y$ and $X' \sim_{\alpha_2} Y'$ hold.*

The proof can be found in Appendix A starting at page 55.

Lemma 2.22 *Let $\alpha_1, \alpha_2 \geqslant 1$ and X, X', Y_0 be finite sets and x, y be elements such that $X \uplus X' \sim_{\alpha_1+\alpha_2} Y_0$ holds and x/y respect $X \uplus X'/Y_0$. Then there are two finite sets Y and Y' such that $Y_0 = Y \uplus Y'$, $X \sim_{\alpha_1} Y$ and $X' \sim_{\alpha_2} Y'$ hold, and x/y respect both X/Y and X'/Y'.*

Proof There are three cases:

- if $x \in X$ then we have $x \in X \uplus X'$ and as a consequence $y \in Y_0$. Let us define $X'' = X\backslash\{x\}$ and $Y_0' = Y_0\backslash\{y\}$. As $\alpha_1 \geqslant 1$, by Proposition 2.20 we deduce $X'' \uplus X' \sim_{(\alpha_1-1)+\alpha_2} Y_0'$. By Lemma 2.21, we obtain Y'' and Y' such that $Y_0' = Y'' \uplus Y'$, $X'' \sim_{\alpha_1-1} Y''$ and $X' \sim_{\alpha_2} Y'$. Observe that $y \notin Y''$ and $y \notin Y'$ because $Y'' \uplus Y' \subseteq Y_0\backslash\{y\}$. We define $Y = Y'' \uplus \{y\}$. By Lemma 2.19, we have $X = X'' \uplus \{x\} \sim_{\alpha_1} Y'' \uplus \{y\} = Y$. Since $x \in X$ and $y \in Y$, x/y respect X/Y; and since $x \notin X'$ (because $X \cap X' = \varnothing$) and $y \notin Y'$, x/y respect X'/Y';
- the case $x \in X'$ is the symmetric case;
- if $x \notin X \uplus X'$ then $y \notin Y_0$. Using Lemma 2.21 we choose Y, Y' such that $Y_0 = Y \uplus Y'$, $X \sim_{\alpha_1} Y$ and $X' \sim_{\alpha_2} Y'$. From $x \notin X \uplus X'$ and $y \notin Y \uplus Y'$, we deduce $x \notin X$, $x \notin X'$, $y \notin Y$ and $y \notin Y'$ hence x/y respect both X/Y and X'/Y'. □

Proposition 2.23 *Let X and Y be two 2-equipotent finite sets, i.e. $X \sim_2 Y$. Let x and y be such that x/y respect X/Y. For any $u \in X\backslash\{x\}$ there exists $v \in Y\backslash\{y\}$.*

Proof Let $u \in X\backslash\{x\}$. We have two cases. If $x \in X$, then we have $u \neq x \in X$ thus $\mathrm{card}(X) \geqslant 2$. Using $X \sim_2 Y$ we deduce $\mathrm{card}(Y) \geqslant 2$ and thus $\mathrm{card}(Y\backslash\{y\}) \geqslant 1$. Otherwise ($x \notin X$), we have $y \notin Y$ and thus $X\backslash\{x\} = X \sim_2 Y = Y\backslash\{y\}$. Hence if $X\backslash\{x\}$ is non-empty then so is $Y\backslash\{y\}$. □

Proposition 2.24 *For any $\alpha \geqslant 0$ and any finite set X, there exists a partition $X = X_1 \uplus X_2$ of X such that $X_1 \sim_\alpha X_1 \uplus X_2$ and $\mathrm{card}(X_1) \leqslant \alpha$.*

Proof If $\mathrm{card}(X) \leqslant \alpha$ then, we define $X_1 = X$ and $X_2 = \varnothing$. If $\mathrm{card}(X) > \alpha$ then choose $X_1 \subseteq X$ such that $\mathrm{card}(X_1) = \alpha$ (e.g. X_1 is composed of the least α elements of X) and $X_2 = X\backslash X_1$. □

3 Test formulæ and Pointed Memory States

In this section, we consider a fixed set $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$ of $q \geqslant 1$ distinct program variables. The value q can always be chosen large enough to accommodate a formula that contains many program variables. Below, we introduce test formulæ stating simple properties and we show that every formula in 1SL1 is equivalent to a Boolean combination of test formulæ.

3.1 Test Formulæ for 1SL1

Test formulæ express simple properties about the memory states; this includes properties about program variables but also global properties about numbers of predecessors or loops, following the decomposition in Section 2.3. These test formulæ allow us to characterize the expressive power of 1SL1, similarly to what has been done in (Lozes 2004a,b; Brochenin et al 2009) for 1SL0. Moreover, we aim at defining the class of test formulæ as small as possible in order to nail down the very expressive power of 1SL1.

Since every formula in 1SL1 is shown equivalent to a Boolean combination of test formulæ (forthcoming Theorem 4.11), this process can be viewed as a means to eliminate separating connectives in a controlled way; elimination is not total since the test formulæ require such separating connectives. However, this is analogous to quantifier elimination in Presburger arithmetic (Presburger 1929) for which simple modulo constraints need to be introduced in order to eliminate the quantifiers (of course, modulo constraints are defined with quantifiers but in a controlled way too).

In Sections 2.2 and 2.4, we explained how to express the following test formulæ in 1SL1. In particular in Lemma 2.14, we show how to define the formulæ $\#\,\mathtt{loop}_{\heartsuit} \geqslant k$ (resp. $\#\,\mathtt{rem}_{\heartsuit} \geqslant k$ and $\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant k$) in 1SL1. However, the precise way they are actually defined in 1SL1 does not impact our developments.

Definition 3.1 (Test formulæ) Given $\alpha \geqslant 0$, we define several sets of test formulæ:

$$
\begin{array}{rcl}
\mathsf{Equality} & \stackrel{\text{def}}{=} & \big\{\mathtt{x}_i = \mathtt{x}_j \;\big|\; i,j \in [1,q]\big\} \\
\mathsf{Pattern} & \stackrel{\text{def}}{=} & \big\{\mathtt{x}_i \hookrightarrow \mathtt{x}_j, \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j), \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j) \;\big|\; i,j \in [1,q]\big\} \\
& \cup & \big\{\mathtt{toalloc}(\mathtt{x}_i), \mathtt{toloop}(\mathtt{x}_i) \;\big|\; i \in [1,q]\big\} \\
\mathsf{Extra}^{\mathtt{u}} & \stackrel{\text{def}}{=} & \big\{\mathtt{u} = \mathtt{u}, \mathtt{u} \hookrightarrow \mathtt{u}, \mathtt{alloc}(\mathtt{u})\big\} \\
& \cup & \big\{\mathtt{x}_i = \mathtt{u}, \mathtt{u} = \mathtt{x}_i, \mathtt{x}_i \hookrightarrow \mathtt{u}, \mathtt{u} \hookrightarrow \mathtt{x}_i \;\big|\; i \in [1,q]\big\} \\
\mathsf{Size}_\alpha & \stackrel{\text{def}}{=} & \big\{\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant k \;\big|\; i \in [1,q], k \in [1,\alpha]\big\} \\
& \cup & \big\{\#\,\mathtt{loop}_{\heartsuit} \geqslant k, \#\,\mathtt{rem}_{\heartsuit} \geqslant k \;\big|\; k \in [1,\alpha]\big\}
\end{array}
$$

$$
\begin{array}{rcl@{\qquad}rcl}
\mathsf{Basic} & \stackrel{\text{def}}{=} & \mathsf{Equality} \cup \mathsf{Pattern} & \mathsf{Test}_\alpha & \stackrel{\text{def}}{=} & \mathsf{Basic} \cup \mathsf{Size}_\alpha \cup \{\bot\} \\
\mathsf{Basic}^{\mathtt{u}} & \stackrel{\text{def}}{=} & \mathsf{Basic} \cup \mathsf{Extra}^{\mathtt{u}} & \mathsf{Test}^{\mathtt{u}}_\alpha & \stackrel{\text{def}}{=} & \mathsf{Test}_\alpha \cup \mathsf{Extra}^{\mathtt{u}}.
\end{array}
$$

We observe that Size_0 is an empty set of formulæ. The formula $\mathtt{alloc}(\mathtt{x}_i)$ is not included because we use the logically equivalent $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)$. Notice however that $\mathtt{alloc}(\mathtt{u})$ (defined by $(\mathtt{u} \hookrightarrow \mathtt{u}) \mathrel{-\!\!*} \bot$) cannot be replaced because $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$ needs the quantifier $\exists \mathtt{u}$ to be defined in 1SL1. Unlike $\mathtt{alloc}(\mathtt{x}_i)$, the test formula $\mathtt{alloc}(\mathtt{u})$ cannot be defined from $\mathtt{conv}$.

It is important to note that the sets $\mathsf{Basic}^{\mathtt{u}}$, Size_α and $\mathsf{Test}^{\mathtt{u}}_\alpha$ depend on a particular choice of q and $\mathcal{V}$; and again our notation does not reflect that dependency. By way of example, the definition of $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ and its semantics inherently depend on q (and on $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$); see Proposition 2.14. As an illustration, the conjunction of literals

$$\mathtt{x}_1 \hookrightarrow \mathtt{x}_2 \wedge \mathtt{x}_2 \hookrightarrow \mathtt{x}_1 \wedge \neg(\mathtt{x}_1 = \mathtt{x}_2) \wedge \mathtt{u} \hookrightarrow \mathtt{u} \wedge \neg \#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1$$

is satisfiable if $q \geqslant 3$ and unsatisfiable if $q = 2$. Indeed, in any model, the interpretation of $\mathtt{u}$ (a self-loop) must be equal to the interpretation of some program variable in $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$. But $\mathtt{u}$ cannot be interpreted by either $\mathtt{x}_1$ or $\mathtt{x}_2$ unless $\mathtt{x}_1 = \mathtt{x}_2$ is satisfied as well.

Proposition 3.2 (Monotonicity of $\mathsf{Basic}^{\mathtt{u}}$) *Let s, h_1, h_2 and $l \in \mathbb{N}$ be such that $h_1 \sqsubseteq h_2$. For any formula $\mathcal{B} \in \mathsf{Basic}^{\mathtt{u}}$, if $(s, h_1) \models_l \mathcal{B}$ then $(s, h_2) \models_l \mathcal{B}$.*

The proof is left to the reader.

3.2 Satisfiability of Boolean Combinations of Test Formulæ

Definition 3.3 (Literals and atoms) A *literal* is a test formula in $\mathsf{Test}^{\mathtt{u}}_\alpha$ or the negation of a test formula in $\mathsf{Test}^{\mathtt{u}}_\alpha$. An *atom* is a saturated conjunction of literals from $\mathsf{Test}^{\mathtt{u}}_\alpha$, i.e. each formula from $\mathsf{Test}^{\mathtt{u}}_\alpha$ occurs exactly once, possibly negated.

Hence atoms are either inconsistent (and equivalent to $\bot$) or minimal elements in the Boolean algebra generated by $\mathsf{Test}^{\mathtt{u}}_\alpha$; any Boolean combination of formulæ of $\mathsf{Test}^{\mathtt{u}}_\alpha$ is equivalent to a disjunction of atoms.

Definition 3.4 The *satisfiability problem for Boolean combinations of test formulæ* is defined by:

INPUT: a set of program variables $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$; a Boolean combination $\mathcal{A}$ of formulæ from $\bigcup_{\alpha \geqslant 1} \mathsf{Test}^{\mathtt{u}}_\alpha$ built on $\mathcal{V}$ (the bounds k in the Size_α formulæ are encoded in binary).
QUESTION: is $\mathcal{A}$ satisfiable with test formulæ understood using the set $\mathcal{V}$?

Indeed, we need to specify the set $\{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$ as an input because the meaning of test formulæ like $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1$ depends on $\{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$; see the previous section.

Theorem 3.5 *The satisfiability problem for Boolean combinations of test formulæ is* NP*-complete.*

Proof NP-hardness follows from a reduction from SAT, assuming that the number of program variables is unbounded. Indeed, each propositional variable p_i can be encoded by the equality $\mathtt{x}_{2i} = \mathtt{x}_{2i+1}$ where the program variables $\mathtt{x}_{2i}$ and $\mathtt{x}_{2i+1}$ are dedicated to p_i only. The proof of the NP upper bound is given in Section 5. □

Checking the satisfiability status of a Boolean combination of test formulæ is typically the kind of tasks that could be performed by an SMT solver, see e.g. (de Moura and Björner 2008; Barrett et al 2011).

3.3 Observing Pointed Memory States with Test Formulæ

We introduce the notion of pointed memory state and two kinds of equivalence relations between pointed memory states: α-equivalence denoted $\simeq_\alpha$ and basic equivalence denoted $\simeq_b$.

Definition 3.6 (Pointed memory state, maxval) The triple $\mathfrak{m} = (s, h, l)$ is a *pointed memory state* if (s, h) is a memory state and $l \in \mathbb{N}$ is a location. We define

$$\begin{array}{rcl} \mathrm{maxval}(h) & \stackrel{def}{=} & \max\big(\mathrm{dom}(h) \cup \mathrm{ran}(h)\big) \\ \mathrm{maxval}(s, h) & \stackrel{def}{=} & \max\big(s(\mathcal{V}) \cup \mathrm{dom}(h) \cup \mathrm{ran}(h)\big) \\ \mathrm{maxval}(\mathfrak{m}) & \stackrel{def}{=} & \max\big(s(\mathcal{V}) \cup \mathrm{dom}(h) \cup \mathrm{ran}(h) \cup \{l\}\big). \end{array}$$

Definition 3.7 (Pseudo-core) The *pseudo-core of a memory state* (s, h), written $\mathfrak{p}\heartsuit(s, h)$, is defined as $\mathfrak{p}\heartsuit(s, h) \stackrel{def}{=} s(\mathcal{V}) \cup h(s(\mathcal{V}))$. The *pseudo-core of a pointed memory state* $\mathfrak{m} = (s, h, l)$ is defined by $\mathfrak{p}\heartsuit(\mathfrak{m}) \stackrel{def}{=} s(\mathcal{V}) \cup h(s(\mathcal{V})) \cup \{l\}$.

We observe that for any $u > \mathrm{maxval}(s, h)$, we have $u \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s, h)$ and if $u > \mathrm{maxval}(s, h, l)$ then $u \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s, h, l)$. Note also that the identity $\heartsuit(s, h) = \mathfrak{p}\heartsuit(s, h) \cap \mathrm{dom}(h)$ holds. Moreover, $\heartsuit(s, h) \subseteq \mathfrak{p}\heartsuit(s, h)$ and $\mathfrak{p}\heartsuit(s, h)$ may contain locations that are not in $\mathrm{dom}(h)$, unlike the core $\heartsuit(s, h)$.

Below, we introduce equivalence relations depending on whether memory states are indistinguishable with respect to some sets of test formulæ.

Definition 3.8 (Equivalences) Given pointed memory states $\mathfrak{m} = (s, h, l)$ and $\mathfrak{m}' = (s', h', l')$, we say that $\mathfrak{m}$ and $\mathfrak{m}'$ are *α-equivalent* and we write $\mathfrak{m} \simeq_\alpha \mathfrak{m}'$ when the equivalence

$$(s, h) \models_l \mathcal{B} \text{ iff } (s', h') \models_{l'} \mathcal{B} \text{ holds for any } \mathcal{B} \in \mathsf{Test}^{\mathtt{u}}_{\alpha}.$$

We also define the *basic equivalence*, written $\simeq_b$, by using $\mathsf{Basic}^{\mathtt{u}}$ instead of $\mathsf{Test}^{\mathtt{u}}_{\alpha}$.

It is obvious that $\simeq_\alpha$ and $\simeq_b$ are indeed equivalence relations between pointed memory states. The pointed memory states $\mathfrak{m}$ and $\mathfrak{m}'$ are basically equivalent (resp. α-equivalent) if and only if they cannot be distinguished by

the formulæ of $\mathsf{Basic}^{\mathsf{u}}$ (resp. $\mathsf{Test}^{\mathsf{u}}_{\alpha}$). Since the inclusion $\mathsf{Basic}^{\mathsf{u}} \subseteq \mathsf{Test}^{\mathsf{u}}_{\alpha}$ holds, it is obvious that the inclusion $\simeq_\alpha \subseteq \simeq_b$ holds. Also observe that the identity $\simeq_b = \simeq_0$ holds because the set of formulæ $\mathsf{Test}^{\mathsf{u}}_{0}$ is identical to $\mathsf{Basic}^{\mathsf{u}} \cup \{\bot\}$. Nevertheless, we think it is clearer to keep a separate notation for $\simeq_b$.

Proposition 3.9 *Let (s,h,l) and (s',h',l') be two pointed memory states such that $(s,h,l) \simeq_b (s',h',l')$. Then $(s,\square,l) \simeq_b (s',\square,l')$ holds.*

The proof is left to the reader; remember that $\square$ denotes the empty heap. Proposition 3.10 below states that α-equivalence corresponds to basic equivalence together with α-equipotence of the sets $\mathrm{pred}_{\overline{\heartsuit}}(\cdot,\cdot,i)$, $\mathrm{loop}_{\overline{\heartsuit}}(\cdot,\cdot)$ and $\mathrm{rem}_{\overline{\heartsuit}}(\cdot,\cdot)$.

Proposition 3.10 *For any $\alpha \geqslant 0$, the relation $(s,h,l) \simeq_\alpha (s',h',l')$ holds if and only if the four following conditions hold:*

1. $(s,h,l) \simeq_b (s',h',l')$;
2. $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \sim_\alpha \mathrm{pred}_{\overline{\heartsuit}}(s',h',i)$ *for any* $i \in [1,q]$;
3. $\mathrm{loop}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{loop}_{\overline{\heartsuit}}(s',h')$;
4. $\mathrm{rem}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{rem}_{\overline{\heartsuit}}(s',h')$.

The proof follows from $\mathsf{Test}^{\mathsf{u}}_{\alpha} = \{\bot\} \cup \mathsf{Basic}^{\mathsf{u}} \cup \mathsf{Size}_{\alpha}$ and Lemma 2.14.

Proposition 3.11 *Let $\mathfrak{m} = (s,h,l)$ and $\mathfrak{m}' = (s',h',l')$ be two pointed memory states such that $\mathrm{dom}(h) \subseteq \heartsuit(s,h)$ and $\mathrm{dom}(h') \subseteq \heartsuit(s',h')$. For any $\alpha \geqslant 0$, $\mathfrak{m} \simeq_\alpha \mathfrak{m}'$ iff $\mathfrak{m} \simeq_b \mathfrak{m}'$.*

This is a direct consequence of Proposition 3.10.

3.4 A Relational View of Basic Equivalence

In this section, we assume two pointed memory states $\mathfrak{m} = (s,h,l)$ and $\mathfrak{m}' = (s',h',l')$. We name some properties which will be used to define binary relations between locations.

Definition 3.12 For $u, v \in \mathbb{N}$, we define the following properties:

($\mathfrak{R}1$) $u = l$ and $v = l'$;
($\mathfrak{R}2$) $u = s(\mathtt{x}_i)$ and $v = s'(\mathtt{x}_i)$ for some $i \in [1,q]$;
($\mathfrak{R}3$) $u = h(s(\mathtt{x}_i))$ and $v = h'(s'(\mathtt{x}_i))$ for some $i \in [1,q]$;

($\mathfrak{T}1$) $u = l$ iff $v = l'$;
($\mathfrak{T}2$) $u = s(\mathtt{x}_i)$ iff $v = s'(\mathtt{x}_i)$ for any $i \in [1,q]$;
($\mathfrak{T}3$) $u = h(s(\mathtt{x}_i))$ iff $v = h'(s'(\mathtt{x}_i))$ for any $i \in [1,q]$;
($\mathfrak{T}4$) $u \in \mathrm{dom}(h)$ iff $v \in \mathrm{dom}(h')$;
($\mathfrak{T}5$) $h(u) = s(\mathtt{x}_i)$ iff $h'(v) = s'(\mathtt{x}_i)$ for any $i \in [1,q]$;
($\mathfrak{T}6$) $h(u) = u$ iff $h'(v) = v$.

We say that e.g. u/v *verify* ($\mathfrak{T}2$) if this property holds.

We emphasize the fact that Properties ($\mathfrak{R}$1-3) and ($\mathfrak{T}$1-6) depend on $\mathfrak{m}/\mathfrak{m}'$. More precisely, ($\mathfrak{R}$1) and ($\mathfrak{T}$1) depend on l/l', ($\mathfrak{R}$2) and ($\mathfrak{T}$2) depend on s/s', ($\mathfrak{T}$4,6) depend on h/h', and the remaining ($\mathfrak{R}$3) and ($\mathfrak{T}$3,5) depend on $(s,h)/(s',h')$. When the context does not single out a unique choice for $\mathfrak{m}/\mathfrak{m}'$, we will explicitly say that e.g. u/v *verify* ($\mathfrak{T}$3) *with respect to* $\mathfrak{m}/\mathfrak{m}'$.

The Properties ($\mathfrak{T}$2–6) characterize precisely when u and v have a similar situation with regard to the canonical decompositions of $\mathfrak{m}$ and $\mathfrak{m}'$ respectively. The Property ($\mathfrak{T}$1) characterizes the situations of u/v with respect to l/l'. As witnessed in upcoming Lemma 3.25 and Propositions 3.29 and 3.31, we do not need all this precision to establish that basic or α-equivalence is preserved by atomic extensions.

Proposition 3.13 *Let* $u, v \in \mathbb{N}$. *For* ($\mathfrak{T}$10)–($\mathfrak{T}$20) *defined as*

($\mathfrak{T}$10) $u \in s(\mathcal{V})$ *iff* $v \in s'(\mathcal{V})$;
($\mathfrak{T}$11) $u \in h(s(\mathcal{V}))$ *iff* $v \in h'(s'(\mathcal{V}))$;
($\mathfrak{T}$12) $u \in \mathfrak{p}\heartsuit(s,h)$ *iff* $v \in \mathfrak{p}\heartsuit(s',h')$;
($\mathfrak{T}$13) $u \in \heartsuit(s,h)$ *iff* $v \in \heartsuit(s',h')$;
($\mathfrak{T}$14) $u \in \mathrm{pred}(s,h,i)$ *iff* $v \in \mathrm{pred}(s',h',i)$ *for any* $i \in [1,q]$;
($\mathfrak{T}$15) $u \in \mathrm{pred}(s,h)$ *iff* $v \in \mathrm{pred}(s',h')$;
($\mathfrak{T}$16) $u \in \mathrm{loop}(s,h)$ *iff* $v \in \mathrm{loop}(s',h')$;
($\mathfrak{T}$17) $u \in \mathrm{rem}(s,h)$ *iff* $v \in \mathrm{rem}(s',h')$;
($\mathfrak{T}$18) $u \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ *iff* $v \in \mathrm{pred}_{\overline{\heartsuit}}(s',h',i)$ *for any* $i \in [1,q]$;
($\mathfrak{T}$19) $u \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ *iff* $v \in \mathrm{loop}_{\overline{\heartsuit}}(s',h')$;
($\mathfrak{T}$20) $u \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ *iff* $v \in \mathrm{rem}_{\overline{\heartsuit}}(s',h')$;
($\mathfrak{T}$21) $u \in \mathfrak{p}\heartsuit(\mathfrak{m})$ *iff* $v \in \mathfrak{p}\heartsuit(\mathfrak{m}')$.

the following propositions hold:

1. ($\mathfrak{T}$2) *implies* ($\mathfrak{T}$10);
2. ($\mathfrak{T}$3) *implies* ($\mathfrak{T}$11);
3. ($\mathfrak{T}$2–3) *imply* ($\mathfrak{T}$12);
4. ($\mathfrak{T}$2–4) *imply* ($\mathfrak{T}$13);
5. ($\mathfrak{T}$2–6) *imply* ($\mathfrak{T}$10–20);
6. ($\mathfrak{T}$1–3) *imply* ($\mathfrak{T}$21).

The proof can be found in Appendix B starting at page 56.

Definition 3.14 (Binary relations between locations) We define binary relations between locations denoted $\mathfrak{R}_{\mathfrak{m},\mathfrak{m}'}$, $\mathfrak{R}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}$, $\mathfrak{D}_{\mathfrak{m},\mathfrak{m}'}$, $\mathfrak{T}_{\mathfrak{m},\mathfrak{m}'}$ and $\mathfrak{T}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}$ by

$u\ \mathfrak{R}_{\mathfrak{m},\mathfrak{m}'}\ v$ iff ($\mathfrak{R}$2) or ($\mathfrak{R}$3)
$u\ \mathfrak{R}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}\ v$ iff ($\mathfrak{R}$1) or ($\mathfrak{R}$2) or ($\mathfrak{R}$3)

$u\ \mathfrak{D}_{\mathfrak{m},\mathfrak{m}'}\ v$ iff ($\mathfrak{T}$4)
$u\ \mathfrak{T}_{\mathfrak{m},\mathfrak{m}'}\ v$ iff ($\mathfrak{T}$2) and ... and ($\mathfrak{T}$6)
$u\ \mathfrak{T}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}\ v$ iff ($\mathfrak{T}$1) and ... and ($\mathfrak{T}$6)

In the rest of this section, we simply denote $\mathfrak{R}$, $\mathfrak{R}^{\mathsf{l}}$,... instead of $\mathfrak{R}_{\mathfrak{m},\mathfrak{m}'}$, $\mathfrak{R}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}$,... Obviously the inclusions $\mathfrak{R} \subseteq \mathfrak{R}^{\mathsf{l}}$ and $\mathfrak{T}^{\mathsf{l}} \subseteq \mathfrak{T} \subseteq \mathfrak{D}$ hold.

Proposition 3.15 *The following inclusions hold:*

1. $\mathfrak{R} \subseteq \mathfrak{p}\heartsuit(s,h) \times \mathfrak{p}\heartsuit(s',h')$
2. $\mathfrak{T} \cap \mathfrak{p}\heartsuit(s,h) \times \mathbb{N} \subseteq \mathfrak{R}$
3. $\mathfrak{T} \cap \mathbb{N} \times \mathfrak{p}\heartsuit(s',h') \subseteq \mathfrak{R}$
4. $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{p}\heartsuit(\mathfrak{m}) \times \mathfrak{p}\heartsuit(\mathfrak{m}')$
5. $\mathfrak{T}^{\mathsf{l}} \cap \mathfrak{p}\heartsuit(\mathfrak{m}) \times \mathbb{N} \subseteq \mathfrak{R}^{\mathsf{l}}$
6. $\mathfrak{T}^{\mathsf{l}} \cap \mathbb{N} \times \mathfrak{p}\heartsuit(\mathfrak{m}') \subseteq \mathfrak{R}^{\mathsf{l}}$

The proof can be found in Appendix B starting at page 57.

Proposition 3.16 *The following properties hold:*

1. *The relation* $\mathfrak{T}$ *restricted to* $\mathfrak{p}\heartsuit(s,h)\times\mathfrak{p}\heartsuit(s',h')$ *is functional and injective.*
2. *The relation* $\mathfrak{T}^{\mathsf{I}}$ *restricted to* $\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')$ *is functional and injective.*
3. *For any* $u\notin \mathrm{dom}(h)\cup\mathfrak{p}\heartsuit(s,h)$, $v\notin\mathrm{dom}(h')\cup\mathfrak{p}\heartsuit(s',h')$, *we have* $u\,\mathfrak{T}\,v$.
4. *For any* $u\notin \mathrm{dom}(h)\cup\mathfrak{p}\heartsuit(\mathfrak{m})$, $v\notin\mathrm{dom}(h')\cup\mathfrak{p}\heartsuit(\mathfrak{m}')$, *we have* $u\,\mathfrak{T}^{\mathsf{I}}\,v$.

The proof can be found in Appendix B starting at page 57.

We get a characterisation of the basic equivalence of $\mathfrak{m}$ and $\mathfrak{m}'$ in terms of the inclusion of the relation $\mathfrak{R}^{\mathsf{I}}$ into $\mathfrak{T}^{\mathsf{I}}$.

Theorem 3.17 $\mathfrak{m}\simeq_b\mathfrak{m}'$ *if and only if* $\mathfrak{R}^{\mathsf{I}}\subseteq\mathfrak{T}^{\mathsf{I}}$.

The proof can be found in Appendix B starting at page 57.

Proposition 3.18 *If* $\mathfrak{m}\simeq_b\mathfrak{m}'$ *then the following properties hold:*

1. *The relation* $\mathfrak{R}$ *is total and surjective between* $\mathfrak{p}\heartsuit(s,h)$ *and* $\mathfrak{p}\heartsuit(s',h')$;
2. *The relation* $\mathfrak{R}'$ *is total and surjective between* $\mathfrak{p}\heartsuit(\mathfrak{m})$ *and* $\mathfrak{p}\heartsuit(\mathfrak{m}')$.

The proof can be found in Appendix B starting at page 58.

Lemma 3.19 (Bijections between pseudo-cores) *When* $\mathfrak{m}$ *and* $\mathfrak{m}'$ *are basically equivalent, i.e.* $\mathfrak{m}\simeq_b\mathfrak{m}'$, *the following properties hold:*

1. *the inclusions* $\mathfrak{R}\subseteq\mathfrak{T}$ *and* $\mathfrak{R}^{\mathsf{I}}\subseteq\mathfrak{T}^{\mathsf{I}}$ *hold;*
2. *both* $\mathfrak{R}=\mathfrak{T}\cap\mathfrak{p}\heartsuit(s,h)\times\mathfrak{p}\heartsuit(s',h')$ *and* $\mathfrak{R}^{\mathsf{I}}=\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')$ *hold;*
3. *the relation* $\mathfrak{R}$ *is bijective between* $\mathfrak{p}\heartsuit(s,h)$ *and* $\mathfrak{p}\heartsuit(s',h')$;
4. *the relation* $\mathfrak{R}^{\mathsf{I}}$ *is bijective between* $\mathfrak{p}\heartsuit(\mathfrak{m})$ *and* $\mathfrak{p}\heartsuit(\mathfrak{m}')$;
5. *the relation* $\mathfrak{R}^{\mathsf{I}}\cap\heartsuit(s,h)\times\heartsuit(s',h')$ *is bijective between* $\heartsuit(s,h)$ *and* $\heartsuit(s',h')$.

Proof We have proved $\mathfrak{R}^{\mathsf{I}}\subseteq\mathfrak{T}^{\mathsf{I}}$ in Theorem 3.17. Hence we deduce $\mathfrak{R}\subseteq\mathfrak{R}^{\mathsf{I}}\subseteq\mathfrak{T}^{\mathsf{I}}\subseteq\mathfrak{T}$. Hence Property 1 holds.

By Proposition 3.15 item 5, we derive $\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')\subseteq\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathbb{N}\subseteq\mathfrak{R}^{\mathsf{I}}$. By Proposition 3.15 item 4 and $\mathfrak{R}^{\mathsf{I}}\subseteq\mathfrak{T}^{\mathsf{I}}$, we derive $\mathfrak{R}^{\mathsf{I}}\subseteq\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')$. Hence we obtain the identity $\mathfrak{R}^{\mathsf{I}}=\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')$. The identity $\mathfrak{R}=\mathfrak{T}\cap\mathfrak{p}\heartsuit(s,h)\times\mathfrak{p}\heartsuit(s',h')$ can then be established with similar arguments, i.e. Proposition 3.15 items 1 and 2. Hence Property 2 holds.

By Proposition 3.18 item 2, $\mathfrak{R}^{\mathsf{I}}$ is total and surjective. By Proposition 3.16 item 2, $\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')$ is functional and injective. From the identity $\mathfrak{R}^{\mathsf{I}}=\mathfrak{T}^{\mathsf{I}}\cap\mathfrak{p}\heartsuit(\mathfrak{m})\times\mathfrak{p}\heartsuit(\mathfrak{m}')$ we deduce that $\mathfrak{R}^{\mathsf{I}}$ is a bijective relation between $\mathfrak{p}\heartsuit(\mathfrak{m})$ and $\mathfrak{p}\heartsuit(\mathfrak{m}')$. The same reasoning applies to $\mathfrak{R}$. Properties 3 and 4 hold.

We have $\heartsuit(s,h)\subseteq\mathfrak{p}\heartsuit(s,h,l)$ and $\heartsuit(s',h')\subseteq\mathfrak{p}\heartsuit(s',h',l')$. Moreover if u and v are such that $u\;\mathfrak{R}^{\mathsf{I}}\;v$ then u/v respect $\heartsuit(s,h)/\heartsuit(s',h')$ using $u\;\mathfrak{T}^{\mathsf{I}}\;v$ with ($\mathfrak{T}$13). Hence by Proposition 2.16, $\mathfrak{R}^{\mathsf{I}}\cap\heartsuit(s,h)\times\heartsuit(s',h')$ is a bijection between $\heartsuit(s,h)$ and $\heartsuit(s',h')$. □

When we atomically extend a heap (see forthcoming Propositions 3.29 and 3.31), we use the totality of the relations $\mathfrak{T}^!$ (and $\mathfrak{T}$). To get these results, we need slightly stronger assumptions. With $\simeq_2$ instead of $\simeq_b$, the relation $\mathfrak{T}^!$ is total from $\mathbb{N}$ to $[0, m+1]$ with $m = \mathrm{maxval}(\mathfrak{m}')$.

Proposition 3.20 *If $\mathfrak{m}$ and $\mathfrak{m}'$ are 2-equivalent (i.e. $\mathfrak{m} \simeq_2 \mathfrak{m}'$), then $\mathfrak{T}^!$ is a total relation on $\mathbb{N}$: for any $u \in \mathbb{N}$, there exists $v \leqslant \mathrm{maxval}(\mathfrak{m}') + 1$ such that $u\ \mathfrak{T}^!\ v$.*

Proof Since $\simeq_2 \subseteq \simeq_b$ we have $\mathfrak{R}^! \subseteq \mathfrak{T}^!$ by Theorem 3.17. Let us consider $u \in \mathbb{N}$. We have to show that there exists $v \in \mathbb{N}$ such that $u\ \mathfrak{T}^!\ v$ holds. We determine the value of v according to the first condition satisfied in the list below.

- If $u \in \mathfrak{p}\heartsuit(\mathfrak{m})$ then we define v as the unique location in $\mathfrak{p}\heartsuit(\mathfrak{m}')$ such that $u\ \mathfrak{R}^!\ v$, see Lemma 3.19 item 4. We derive $u\ \mathfrak{T}^!\ v$. The relation $v \leqslant \mathrm{maxval}(\mathfrak{m}') + 1$ holds because $v \in \mathfrak{p}\heartsuit(\mathfrak{m}')$.
- If $u \in \mathrm{pred}_{\overline{\heartsuit}}(s, h, j)$ for some $j \in [1, q]$ then we know that $u \neq l$ because the case $u = l \in \mathfrak{p}\heartsuit(\mathfrak{m})$ occurs earlier in the list.
  Hence we have $u \in \mathrm{pred}_{\overline{\heartsuit}}(s, h, j) \backslash \{l\}$. From $l\ \mathfrak{R}^!\ l'$ we deduce $l\ \mathfrak{T}^!\ l'$. Hence by Proposition 3.13 ($\mathfrak{T}18$), l/l' respect $\mathrm{pred}_{\overline{\heartsuit}}(s, h, j)/\mathrm{pred}_{\overline{\heartsuit}}(s', h', j)$. We also have $\mathrm{pred}_{\overline{\heartsuit}}(s, h, j) \sim_2 \mathrm{pred}_{\overline{\heartsuit}}(s', h', j)$ by Proposition 3.10. Hence by Proposition 2.23, we can choose a location $v \in \mathrm{pred}_{\overline{\heartsuit}}(s', h', j) \backslash \{l'\}$.
  The relation $v \leqslant \mathrm{maxval}(\mathfrak{m}') + 1$ holds because $v \in \mathrm{dom}(h')$. Let us establish $u\ \mathfrak{T}^!\ v$. We have $u \in \mathrm{pred}_{\overline{\heartsuit}}(s, h, j) \backslash \{l\}$ and $v \in \mathrm{pred}_{\overline{\heartsuit}}(s', h', j) \backslash \{l'\}$. As a consequence, we deduce $u \notin \mathfrak{p}\heartsuit(\mathfrak{m})$ and $v \notin \mathfrak{p}\heartsuit(\mathfrak{m}')$. Hence, Properties ($\mathfrak{T}1$–3) hold. We also have $u \in \mathrm{dom}(h)$ and $v \in \mathrm{dom}(h')$, whence Property ($\mathfrak{T}4$) holds. We have $h(u) = s(\mathtt{x}_j)$ and $h'(v) = s'(\mathtt{x}_j)$. We deduce $h(u)\ \mathfrak{R}^!\ h'(v)$ and thus $h(u)\ \mathfrak{T}^!\ h'(v)$. Let us prove Property ($\mathfrak{T}5$) for u/v: we use ($\mathfrak{T}2$) for $h(u)/h'(v)$ and get $h(u) = s(\mathtt{x}_i)$ iff $h'(v) = s'(\mathtt{x}_i)$ for any $i \in [1, q]$. Let us prove Property ($\mathfrak{T}6$): the identity $u = h(u)$ implies $u = s(\mathtt{x}_j)$ which contradicts $u \notin \mathfrak{p}\heartsuit(\mathfrak{m})$. Hence $u \neq h(u)$ and for similar reasons, $v \neq h'(v)$.
- If $u \in \mathrm{loop}_{\overline{\heartsuit}}(s, h)$ then we know that $u \neq l$ because the case $u = l \in \mathfrak{p}\heartsuit(\mathfrak{m})$ occurs earlier in the list. Hence we have $u \in \mathrm{loop}_{\overline{\heartsuit}}(s, h) \backslash \{l\}$. Since $l\ \mathfrak{T}^!\ l'$, by Proposition 3.13 ($\mathfrak{T}19$) we deduce that l/l' respect $\mathrm{loop}_{\overline{\heartsuit}}(s, h)/\mathrm{loop}_{\overline{\heartsuit}}(s', h')$. We also have $\mathrm{loop}_{\overline{\heartsuit}}(s, h) \sim_2 \mathrm{loop}_{\overline{\heartsuit}}(s', h')$ by Proposition 3.10. Hence by Proposition 2.23, we can choose a location $v \in \mathrm{loop}_{\overline{\heartsuit}}(s', h') \backslash \{l'\}$.
  The relation $v \leqslant \mathrm{maxval}(\mathfrak{m}') + 1$ holds because $v \in \mathrm{dom}(h')$. Let us check that $u\ \mathfrak{T}^!\ v$ holds. We have $u \in \mathrm{loop}_{\overline{\heartsuit}}(s, h) \backslash \{l\}$ and $v \in \mathrm{loop}_{\overline{\heartsuit}}(s', h') \backslash \{l'\}$. As a consequence, we deduce $u \notin \mathfrak{p}\heartsuit(\mathfrak{m})$ and $v \notin \mathfrak{p}\heartsuit(\mathfrak{m}')$. Hence Properties ($\mathfrak{T}1$–3) hold. We also have $u \in \mathrm{dom}(h)$ and $v \in \mathrm{dom}(h')$; hence Property ($\mathfrak{T}4$) holds. We have $h(u) = u$ and $h'(v) = v$, whence Property ($\mathfrak{T}6$) holds. We have already proved that Property ($\mathfrak{T}2$) holds for u/v. As $h(u) = u$ and $h'(v) = v$ we deduce that Property ($\mathfrak{T}2$) holds for $h(u)/h'(v)$. Hence Property ($\mathfrak{T}5$) holds for u/v.
- If $u \in \mathrm{rem}_{\overline{\heartsuit}}(s, h)$ then $u \in \mathrm{rem}_{\overline{\heartsuit}}(s, h) \backslash \{l\}$. By Proposition 3.13 ($\mathfrak{T}20$), l/l' respect $\mathrm{rem}_{\overline{\heartsuit}}(s, h)/\mathrm{rem}_{\overline{\heartsuit}}(s', h')$. We have $\mathrm{rem}_{\overline{\heartsuit}}(s, h) \sim_2 \mathrm{rem}_{\overline{\heartsuit}}(s', h')$

by Proposition 3.10. Hence by Proposition 2.23, we can choose a location $v \in \mathrm{rem}_{\heartsuit}(s', h') \backslash \{l'\}$.
The relation $v \leqslant \mathrm{maxval}(\mathfrak{m}') + 1$ holds because $v \in \mathrm{dom}(h')$. Let us check that $u \; \mathfrak{T}^{\mathsf{I}} \; v$ holds. We have $u \in \mathrm{rem}_{\heartsuit}(s, h) \backslash \{l\}$ and $v \in \mathrm{rem}_{\heartsuit}(s', h') \backslash \{l'\}$. So, we deduce $u \notin \mathfrak{p}^{\heartsuit}(\mathfrak{m})$ and $v \notin \mathfrak{p}^{\heartsuit}(\mathfrak{m}')$. Hence Properties ($\mathfrak{T}$1–3) hold. We also have $u \in \mathrm{dom}(h)$ and $v \in \mathrm{dom}(h')$ hence Property ($\mathfrak{T}$4) holds. We have $h(u) \notin s(\mathcal{V}) \cup \{u\}$ and $h'(v) \notin s'(\mathcal{V}) \cup \{v\}$ hence Properties ($\mathfrak{T}$5–6) hold.

– In the remaining cases we have $u \notin \mathrm{dom}(h) \cup \mathfrak{p}^{\heartsuit}(\mathfrak{m})$. Let $v = \mathrm{maxval}(\mathfrak{m}') + 1$. Then we have $v \notin \mathrm{dom}(h') \cup \mathfrak{p}^{\heartsuit}(\mathfrak{m}')$ and by Proposition 3.16 item 4, we conclude $u \; \mathfrak{T}^{\mathsf{I}} \; v$. □

If we do not require Property ($\mathfrak{T}$1), i.e. we work with $\mathfrak{T}$ instead of $\mathfrak{T}^{\mathsf{I}}$, then only $\simeq_1$ is needed to establish that $\mathfrak{T}$ is total from $\mathbb{N}$ to $[0, m+1]$ with $m = \mathrm{maxval}(s', h')$, see Proposition 3.21 below.

Proposition 3.21 *If* $\mathfrak{m}$ *and* $\mathfrak{m}'$ *satisfy* $\mathfrak{m} \simeq_1 \mathfrak{m}'$*, then* $\mathfrak{T}$ *is a total relation on* $\mathbb{N}$*: for any* $u \in \mathbb{N}$*, there exists* $v \leqslant \mathrm{maxval}(s', h') + 1$ *such that* $u \; \mathfrak{T} \; v$*.*

The proof can be found in Appendix B starting at page 58.

3.5 Basic Equivalence and Heap Splitting

In this section, we consider two stores s and s', two locations l and l', and two heaps $h = h_1 \uplus h_2$ and $h' = h'_1 \uplus h'_2$ that are divided into two disjoint subheaps. Let us denote

$$\begin{array}{llll} \mathfrak{m} = (s, h, l) & \mathfrak{m}' = (s', h', l') & \mathfrak{R}^{\mathsf{I}} = \mathfrak{R}^{\mathsf{I}}_{\mathfrak{m},\mathfrak{m}'} & \mathfrak{T}^{\mathsf{I}} = \mathfrak{T}^{\mathsf{I}}_{\mathfrak{m},\mathfrak{m}'} \\ \mathfrak{m}_1 = (s, h_1, l) & \mathfrak{m}'_1 = (s', h'_1, l') & \mathfrak{R}^{\mathsf{I}}_1 = \mathfrak{R}^{\mathsf{I}}_{\mathfrak{m}_1,\mathfrak{m}'_1} & \mathfrak{T}^{\mathsf{I}}_1 = \mathfrak{T}^{\mathsf{I}}_{\mathfrak{m}_1,\mathfrak{m}'_1} \quad \mathfrak{D}_1 = \mathfrak{D}_{\mathfrak{m}_1,\mathfrak{m}'_1} \\ \mathfrak{m}_2 = (s, h_2, l) & \mathfrak{m}'_2 = (s', h'_2, l') & \mathfrak{R}^{\mathsf{I}}_2 = \mathfrak{R}^{\mathsf{I}}_{\mathfrak{m}_2,\mathfrak{m}'_2} & \mathfrak{T}^{\mathsf{I}}_2 = \mathfrak{T}^{\mathsf{I}}_{\mathfrak{m}_2,\mathfrak{m}'_2} \quad \mathfrak{D}_2 = \mathfrak{D}_{\mathfrak{m}_2,\mathfrak{m}'_2} \end{array}$$

It is trivial to check the inclusion $\mathfrak{R}^{\mathsf{I}}_1 \subseteq \mathfrak{R}^{\mathsf{I}}$ because $h_1 \sqsubseteq h$ and $h'_1 \sqsubseteq h'$. The inclusion $\mathfrak{R}^{\mathsf{I}}_2 \subseteq \mathfrak{R}^{\mathsf{I}}$ holds by symmetry.

Let us study under which conditions the splits $h = h_1 \uplus h_2$ and $h' = h'_1 \uplus h'_2$ preserve basic equivalence, i.e. when do $\mathfrak{m}_1 \simeq_b \mathfrak{m}'_1$ and $\mathfrak{m}_2 \simeq_b \mathfrak{m}'_2$ hold, provided that $\mathfrak{m} \simeq_b \mathfrak{m}'$ already holds.

Proposition 3.22 *Let us assume* $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{T}^{\mathsf{I}}$ *(or equivalently* $\mathfrak{m} \simeq_b \mathfrak{m}'$*). Then the following statements are equivalent:*

1. $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{D}_1 \cap \mathfrak{D}_2$*;*
2. $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{T}^{\mathsf{I}}_1 \cap \mathfrak{T}^{\mathsf{I}}_2$*;*
3. $\mathfrak{R}^{\mathsf{I}}_1 \subseteq \mathfrak{T}^{\mathsf{I}}_1$ *and* $\mathfrak{R}^{\mathsf{I}}_2 \subseteq \mathfrak{T}^{\mathsf{I}}_2$*;*
4. $\mathfrak{m}_1 \simeq_b \mathfrak{m}'_1$ *and* $\mathfrak{m}_2 \simeq_b \mathfrak{m}'_2$*.*

The proof can be found in Appendix B starting at page 59.

If the splits $h = h_1 \uplus h_2$ and $h' = h'_1 \uplus h'_2$ preserve basic equivalence, then some subsets of the core are equipotent.

Proposition 3.23 *Let us assume* $\mathfrak{m} \simeq_b \mathfrak{m}'$, $\mathfrak{m}_1 \simeq_b \mathfrak{m}'_1$ *and* $\mathfrak{m}_2 \simeq_b \mathfrak{m}'_2$. *With the notation* $\overline{X}$ *for* $\mathbb{N}\backslash X$, *the following properties hold:*

1. $\mathrm{pred}(s,h,i) \cap \Delta_c \sim_\infty \mathrm{pred}(s',h',i) \cap \Delta'_c$ *for any* $i \in [1,q]$,
2. $\mathrm{loop}(s,h) \cap \Delta_c \sim_\infty \mathrm{loop}(s',h') \cap \Delta'_c$,
3. $\mathrm{rem}(s,h) \cap \Delta_c \sim_\infty \mathrm{rem}(s',h') \cap \Delta'_c$,

where $c \in \{1,2\}$ *and* $\begin{cases} \Delta_c = \mathrm{dom}(h_c) \cap h_{3-c}(s(\mathcal{V})) \cap \overline{s(\mathcal{V})} \cap \overline{h_c(s(\mathcal{V}))} \\ \Delta'_c = \mathrm{dom}(h'_c) \cap h'_{3-c}(s'(\mathcal{V})) \cap \overline{s'(\mathcal{V})} \cap \overline{h'_c(s'(\mathcal{V}))}. \end{cases}$

Proof The definition of $\Delta(s,h_1,h_2)$ corresponds to that of Propositions 2.8 and 2.9. We have $\Delta_1 = \Delta(s,h_1,h_2)$, $\Delta_2 = \Delta(s,h_2,h_1)$, $\Delta'_1 = \Delta(s',h'_1,h'_2)$ and $\Delta'_2 = \Delta(s',h'_2,h'_1)$. We easily verify that the inclusions $\Delta_1 \cup \Delta_2 \subseteq \heartsuit(s,h)$ and $\Delta'_1 \cup \Delta'_2 \subseteq \heartsuit(s',h')$ hold. We invite the reader to check the following equivalences:

$$\begin{array}{rcl} u \in \mathrm{pred}(s,h,j) \cap \Delta_1 & \text{iff} & h_1(u) = s(\mathtt{x}_j) \text{ and} \\ & & u \in h_2(s(\mathcal{V})) \text{ and } u \notin s(\mathcal{V}) \text{ and } u \notin h_1(s(\mathcal{V})) \\ u \in \mathrm{loop}(s,h) \cap \Delta_1 & \text{iff} & h_1(u) = u \text{ and} \\ & & u \in h_2(s(\mathcal{V})) \text{ and } u \notin s(\mathcal{V}) \text{ and } u \notin h_1(s(\mathcal{V})) \\ u \in \mathrm{rem}(s,h) \cap \Delta_1 & \text{iff} & u \in \mathrm{dom}(h_1) \text{ and } h_1(u) \notin s(\mathcal{V}) \text{ and } h(u) \neq u \text{ and} \\ & & u \in h_2(s(\mathcal{V})) \text{ and } u \notin s(\mathcal{V}) \text{ and } u \notin h_1(s(\mathcal{V})). \end{array}$$

From Proposition 3.22 and Theorem 3.19 we deduce $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{T}^{\mathsf{I}} \cap \mathfrak{T}^{\mathsf{I}}_1 \cap \mathfrak{T}^{\mathsf{I}}_2$. Moreover by Lemma 3.19, the relation $\mathfrak{R}^{\mathsf{I}} \cap \heartsuit(s,h) \times \heartsuit(s',h')$ is a bijection between $\heartsuit(s,h)$ and $\heartsuit(s',h')$.

Let us prove Property 1 with $c = 1$ for instance. We use Proposition 2.16: we have $\mathrm{pred}(s,h,i) \cap \Delta_1 \subseteq \heartsuit(s,h)$ and $\mathrm{pred}(s',h',i) \cap \Delta'_1 \subseteq \heartsuit(s',h')$. Hence let us show that if $u \in \heartsuit(s,h)$ and $v \in \heartsuit(s',h')$ verify $u\ \mathfrak{R}^{\mathsf{I}}\ v$ then u/v respect $\mathrm{pred}(s,h,i) \cap \Delta_1/\mathrm{pred}(s',h',i) \cap \Delta'_1$: we use the first of the three above equivalences and $u\ \mathfrak{T}^{\mathsf{I}}_1\ v$ with ($\mathfrak{T}$5), $u\ \mathfrak{T}^{\mathsf{I}}_2\ v$ with ($\mathfrak{T}$11), $u\ \mathfrak{T}^{\mathsf{I}}\ v$ with ($\mathfrak{T}$10), and $u\ \mathfrak{T}^{\mathsf{I}}_1\ v$ with ($\mathfrak{T}$11).

Hence by Proposition 2.16, there is a bijection between $\mathrm{pred}(s,h,i) \cap \Delta_1$ and $\mathrm{pred}(s',h',i) \cap \Delta'_1$ and thus $\mathrm{pred}(s,h,i) \cap \Delta_1 \sim_\infty \mathrm{pred}(s',h',i) \cap \Delta'_1$ holds. We use similar arguments for Property 1 ($c = 2$) and Properties 2–3. □

3.6 Basic Equivalence and Location Update

In this section, we study under which conditions an update of the location l in the pointed memory state (s,h,l) preserves basic equivalence.

Proposition 3.24 *Let* (s,h,l) *and* (s',h',l') *be basically equivalent pointed memory states, i.e.* $(s,h,l) \simeq_b (s',h',l')$. *For any* $l_0, l'_0 \in \mathbb{N}$, *if* l_0/l'_0 *verify* ($\mathfrak{T}$2–6) *then* $(s,h,l_0) \simeq_b (s',h',l'_0)$ *holds.*

Proof Since any formula $\mathcal{B} \in \mathsf{Basic}$ contains no free occurence of $\mathtt{u}$, by Proposition 2.2 we have $(s,h) \models_{l_0} \mathcal{B}$ iff $(s,h) \models_l \mathcal{B}$ iff $(s',h') \models_{l'} \mathcal{B}$ iff $(s',h') \models_{l'_0} \mathcal{B}$.

As the identity $\mathsf{Basic}^{\mathtt{u}} = \mathsf{Basic} \cup \mathsf{Extra}^{\mathtt{u}}$ holds, to get $(s,h,l_0) \simeq_b (s',h',l'_0)$ it is sufficient to prove the property $(s,h) \models_{l_0} \mathcal{B}$ iff $(s',h') \models_{l'_0} \mathcal{B}$ for any formula $\mathcal{B} \in \mathsf{Extra}^{\mathtt{u}}$. We proceed by a case analysis on $\mathcal{B}$; we display the *only if case*, the *if case* being proved in a symmetric way:

- $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$: from $(s,h) \models_{l_0} \mathtt{u} \hookrightarrow \mathtt{u}$ we get $h(l_0) = l_0$. Since l_0/l'_0 verify $(\mathfrak{T}6)$, we deduce $h'(l'_0) = l'_0$ and thus $(s',h') \models_{l'_0} \mathtt{u} \hookrightarrow \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{alloc}(\mathtt{u})$: from $(s,h) \models_{l_0} \mathtt{alloc}(\mathtt{u})$ we get $l_0 \in \mathrm{dom}(h)$. Since l_0/l'_0 verify $(\mathfrak{T}4)$, we deduce $l'_0 \in \mathrm{dom}(h')$ and thus $(s',h') \models_{l'_0} \mathtt{alloc}(\mathtt{u})$;
- $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$: from $(s,h) \models_{l_0} \mathtt{x}_i = \mathtt{u}$ we get $s(\mathtt{x}_i) = l_0$. Since l_0/l'_0 verify $(\mathfrak{T}2)$, we deduce $s'(\mathtt{x}_i) = l'_0$ and thus $(s',h') \models_{l'_0} \mathtt{x}_i = \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$: from $(s,h) \models_{l_0} \mathtt{x}_i \hookrightarrow \mathtt{u}$ we get $h(s(\mathtt{x}_i)) = l_0$. Since l_0/l'_0 verify $(\mathfrak{T}3)$, we deduce $h'(s'(\mathtt{x}_i)) = l'_0$ and thus $(s',h') \models_{l'_0} \mathtt{x}_i \hookrightarrow \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_j$: from $(s,h) \models_{l_0} \mathtt{u} \hookrightarrow \mathtt{x}_j$ we get $h(l_0) = s(\mathtt{x}_i)$. Since l_0/l'_0 verify $(\mathfrak{T}5)$, we deduce $h'(l'_0) = s'(\mathtt{x}_i)$ and thus $(s',h') \models_{l'_0} \mathtt{u} \hookrightarrow \mathtt{x}_j$. □

3.7 Atomic Extensions and α-Equivalence

Recall that we write $[l_1 \mapsto l_2]$ to denote the (atomic) heap h such that $\mathrm{dom}(h) = \{l_1\}$, $h(l_1) = l_2$ and $\mathrm{ran}(h) = \{l_2\}$. We study under which conditions atomic extensions preserve α-equivalence.

Lemma 3.25 *Let $\alpha \geqslant 1$ and let (s,h,l) and (s',h',l') be two pointed memory states. Let $l_1, l_2, l'_1, l'_2 \in \mathbb{N}$ be such that $l_1 \notin \mathrm{dom}(h)$ and $l'_1 \notin \mathrm{dom}(h')$. We assume that one of the conditions below holds:*

(C1) l_1/l'_1 verify $(\mathfrak{T}1$–$3)$, l_2/l'_2 verify $(\mathfrak{T}1$–$6)$, and $l_2 = l_1$ iff $l'_2 = l'_1$;
(C2) $l_1 \notin s(\mathcal{V})$, l_1/l'_1 verify $(\mathfrak{T}1$–$3)$, l_2/l'_2 verify $(\mathfrak{T}2)$, and $l_2 = l_1$ iff $l'_2 = l'_1$.

If $(s,h,l) \simeq_\alpha (s',h',l')$ then $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\beta (s', h' \uplus [l'_1 \mapsto l'_2], l')$ where $\beta = \alpha - 1$ if $l_1 \in s(\mathcal{V})$, and $\beta = \alpha$ otherwise.

The proof can be found in Appendix B starting at page 60.

Now let us present sufficient conditions under which an atomic extension does not change a pointed memory state up to α-equivalence.

Proposition 3.26 *Let $\mathfrak{m} = (s,h,l)$ be a pointed memory state and $l_1, l_2 \in \mathbb{N}$ be such that $l_1 \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})$. We have $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_b (s,h,l)$. Moreover, given $\alpha \geqslant 0$, if we assume that one of the following conditions hold*

(C1) $l_2 = s(\mathtt{x}_i)$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant \alpha$ for some $i \in [1,q]$;
(C2) $l_2 = l_1$ and $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$;
(C3) $l_2 \notin s(\mathcal{V}) \cup \{l_1\}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$.

then we have $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s,h,l)$.

The proof can be found in Appendix B starting at page 63.

We extend the previous result to more general extensions that avoid adding locations in the pseudo-core.

Corollary 3.27 *Let $\alpha \geqslant 0$. Let $\mathfrak{m} = (s, h, l)$ be a pointed memory state and h' be a heap such that $\mathrm{dom}(h') \cap (\mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})) = \varnothing$. If for any $u \in \mathrm{dom}(h')$ one of the following conditions holds*

(C1) $h'(u) = s(\mathtt{x}_i)$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h, i)) \geqslant \alpha$ for some $i \in [1, q]$;
(C2) $h'(u) = u$ and $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h)) \geqslant \alpha$;
(C3) $h'(u) \notin s(\mathcal{V}) \cup \{u\}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h)) \geqslant \alpha$.

then we have $(s, h \uplus h', l) \simeq_\alpha (s, h, l)$.

The proof can be found in Appendix B starting at page 64.

3.8 Transposing Heap Extensions through α-Equivalence

In this section, we assume s, s', h_0, h'_0, h, h', l and l' such that $h_0 \perp h$ and $h'_0 \perp h'$. We denote

$$\begin{array}{llll} \mathfrak{m} = (s, h, l) & \mathfrak{m}_0 = (s, h_0 \uplus h, l) & \mathfrak{R}^{\mathsf{I}} = \mathfrak{R}^{\mathsf{I}}_{\mathfrak{m},\mathfrak{m}'} & \mathfrak{T}^{\mathsf{I}} = \mathfrak{T}^{\mathsf{I}}_{\mathfrak{m},\mathfrak{m}'} \\ \mathfrak{m}' = (s', h', l') & \mathfrak{m}'_0 = (s', h'_0 \uplus h', l') & \mathfrak{R}^{\mathsf{I}}_0 = \mathfrak{R}^{\mathsf{I}}_{\mathfrak{m}_0,\mathfrak{m}'_0} & \mathfrak{T}^{\mathsf{I}}_0 = \mathfrak{T}^{\mathsf{I}}_{\mathfrak{m}_0,\mathfrak{m}'_0} \end{array}$$

We insist that the heap of $\mathfrak{m}_0$ is $h_0 \uplus h$, not h_0: the short notation might be a bit confusing here. Because $h \sqsubseteq h_0 \uplus h$ and $h' \sqsubseteq h'_0 \uplus h'$, it is trivial to check that the inclusion $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{R}^{\mathsf{I}}_0$ holds.

Proposition 3.28 *We assume $\mathfrak{m} \simeq_b \mathfrak{m}'$, $\mathfrak{m}_0 \simeq_b \mathfrak{m}'_0$ and $u, v \in \mathbb{N}$ such that $u \; \mathfrak{T}^{\mathsf{I}}_0 \; v$. If either $u \in \mathfrak{p}\heartsuit(\mathfrak{m})$ or $v \in \mathfrak{p}\heartsuit(\mathfrak{m}')$ then $u \; \mathfrak{R}^{\mathsf{I}} \; v$.*

Proof We assume $u \in \mathfrak{p}\heartsuit(\mathfrak{m})$ and we show $u \; \mathfrak{R}^{\mathsf{I}} \; v$. From $\mathfrak{p}\heartsuit(\mathfrak{m}) \subseteq \mathfrak{p}\heartsuit(\mathfrak{m}_0)$, we deduce $u \in \mathfrak{p}\heartsuit(\mathfrak{m}_0)$. By Proposition 3.15 item 5, from $u \; \mathfrak{T}^{\mathsf{I}}_0 \; v$ we deduce $u \; \mathfrak{R}^{\mathsf{I}}_0 \; v$ and thus $v \in \mathfrak{p}\heartsuit(\mathfrak{m}'_0)$ by Proposition 3.15 item 4. Since $u \in \mathfrak{p}\heartsuit(\mathfrak{m})$, by Lemma 3.19 there exists a unique location $w \in \mathfrak{p}\heartsuit(\mathfrak{m}')$ such that $u \; \mathfrak{R}^{\mathsf{I}} \; w$. From $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{R}^{\mathsf{I}}_0$ we deduce $u \; \mathfrak{R}^{\mathsf{I}}_0 \; w$. Hence, we have ($u \; \mathfrak{R}^{\mathsf{I}}_0 \; v$ and $u \; \mathfrak{R}^{\mathsf{I}}_0 \; w$) and by Proposition 3.15 item 4 and Lemma 3.19, $\mathfrak{R}^{\mathsf{I}}_0$ is a bijection. We deduce $v = w$, and then $u \; \mathfrak{R}^{\mathsf{I}} \; v$. □

Proposition 3.29 *Let $\alpha \geqslant 1$. We assume that the following conditions hold:*

(a) $\mathfrak{m} \simeq_{\alpha+1} \mathfrak{m}'$;
(b) $\mathfrak{m}_0 \simeq_{\alpha+1} \mathfrak{m}'_0$;
(c) $\mathrm{dom}(h) \subseteq \mathfrak{p}\heartsuit(\mathfrak{m})$;
(d) $\mathrm{dom}(h') \subseteq \mathfrak{p}\heartsuit(\mathfrak{m}')$.

Let $l_1 \in s(\mathcal{V}) \backslash \mathrm{dom}(h_0 \uplus h)$ and $l_2 \in \mathbb{N}$. There exist $l'_1, l'_2 \in \mathbb{N}$ such that

1. *$l'_1 \in s'(\mathcal{V}) \backslash \mathrm{dom}(h'_0 \uplus h')$;*
2. *$l'_1, l'_2 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 1$;*
3. *$(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h' \uplus [l'_1 \mapsto l'_2], l')$;*
4. *$(s, h_0 \uplus h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h'_0 \uplus h' \uplus [l'_1 \mapsto l'_2], l')$.*

The proof can be found in Appendix B starting at page 65.

Corollary 3.30 *Let $\alpha \geqslant 1$. Let $h_0 \perp h_1$, $\mathrm{dom}(h_1) \subseteq s(\mathcal{V})$ and $(s, h_0, l) \simeq_{p+\alpha} (s', h'_0, l')$ with $p = \mathrm{card}(\mathrm{dom}(h_1))$. Then there exists a heap h'_1 such that $h'_0 \perp h'_1$, $\mathrm{dom}(h'_1) \subseteq s'(\mathcal{V})$, $(s, h_1, l) \simeq_\alpha (s', h'_1, l')$, $(s, h_0 \uplus h_1, l) \simeq_\alpha (s', h'_0 \uplus h'_1, l')$ and $\mathrm{maxval}(s', h'_1) \leqslant \mathrm{maxval}(s', h'_0, l') + p$.*

The proof is by induction on the cardinality of $\mathrm{dom}(h_1)$ using Proposition 3.29. Proposition 3.31 below is a slight variant of Proposition 3.29.

Proposition 3.31 *Let $\alpha \geqslant 1$. We assume that the following conditions hold:*

(a) $\mathfrak{m} \simeq_\alpha \mathfrak{m}'$;
(b) $\mathfrak{m}_0 \simeq_\alpha \mathfrak{m}'_0$.

Let $l_1 \notin \mathrm{dom}(h_0 \uplus h) \cup s(\mathcal{V})$ and $l_2 \in \mathbb{N}$. There exist $l'_1, l'_2 \in \mathbb{N}$ such that

1. $l'_1 \notin \mathrm{dom}(h'_0 \uplus h') \cup s'(\mathcal{V})$
2. $l'_1, l'_2 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 2$;
3. $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h' \uplus [l'_1 \mapsto l'_2], l')$;
4. $(s, h_0 \uplus h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h'_0 \uplus h' \uplus [l'_1 \mapsto l'_2], l')$.

The proof can be found in Appendix B starting at page 66.

Corollary 3.32 *Let $\alpha \geqslant 1$. Let $\mathrm{dom}(h_1) \cap (\mathrm{dom}(h_0 \uplus h) \cup s(\mathcal{V})) = \varnothing$, $(s, h, l) \simeq_\alpha (s', h', l')$ and $(s, h_0 \uplus h, l) \simeq_\alpha (s', h'_0 \uplus h', l')$. Then there exists a heap h'_1 such that $\mathrm{dom}(h'_1) \cap (\mathrm{dom}(h'_0 \uplus h') \cup s'(\mathcal{V})) = \varnothing$, $(s, h \uplus h_1, l) \simeq_\alpha (s', h' \uplus h'_1, l')$, $(s, h_0 \uplus h \uplus h_1, l) \simeq_\alpha (s', h'_0 \uplus h' \uplus h'_1, l')$ and $\mathrm{maxval}(s', h'_1) \leqslant \mathrm{maxval}(s', h'_0 \uplus h, l') + 2.\,\mathrm{card}(\mathrm{dom}(h_1))$.*

The proof is by induction on the cardinality of $\mathrm{dom}(h_1)$ using Proposition 3.31.

3.9 Correctness of the Abstraction

Lemmas 3.33, 3.34 and 3.35 below roughly state that the relation $\simeq_\alpha$ (and therefore the set of test formulæ we have introduced) behaves properly. Each lemma corresponds to a given quantifier, respectively separating conjunction $*$, separating implication $-\!\!*$ and first-order quantifier $\exists \mathfrak{u}$. We combine these three lemmas in the proof of Correctness Theorem 4.3.

Lemma 3.33 below states how two equivalent memory states can be split. The precision is split accordingly.

Lemma 3.33 (Distributivity) *Let $\alpha, \alpha_1, \alpha_2 \geqslant 1$ such that $\alpha = \alpha_1 + \alpha_2$. Let us consider two α-equivalent pointed memory states (s, h, l) and (s', h', l'), i.e. $(s, h, l) \simeq_\alpha (s', h', l')$. For every split $h = h_1 \uplus h_2$ of h, there exists a split $h' = h'_1 \uplus h'_2$ of h' such that $(s, h_1, l) \simeq_{\alpha_1} (s', h'_1, l')$ and $(s, h_2, l) \simeq_{\alpha_2} (s', h'_2, l')$.*

Proof Let $\mathfrak{m} = (s, h, l)$ and $\mathfrak{m}' = (s', h', l')$ be such that $(s, h, l) \simeq_\alpha (s', h', l')$. Let us denote by $\mathfrak{R}^{\mathsf{l}}$ (resp. $\mathfrak{T}^{\mathsf{l}}$) the relation $\mathfrak{R}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}$ (resp. $\mathfrak{T}^{\mathsf{l}}_{\mathfrak{m},\mathfrak{m}'}$) from Definition 3.14. From $\mathfrak{m} \simeq_\alpha \mathfrak{m}'$ we deduce $\mathfrak{m} \simeq_b \mathfrak{m}'$ and then by Theorem 3.17, we have the inclusion $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$. Moreover by Lemma 3.19, we know that $\mathfrak{R}^{\mathsf{l}}_{\heartsuit} \stackrel{def}{=} \mathfrak{R}^{\mathsf{l}} \cap \heartsuit(s, h) \times \heartsuit(s', h')$ is a bijective relation between $\heartsuit(s, h)$ and $\heartsuit(s', h')$.

Let us define $J = \{j \in [1, q] \mid \text{ for all } k \in [1, q], s(\mathtt{x}_j) = s(\mathtt{x}_k) \text{ implies } j \leqslant k\}$. Since $s(\mathtt{x}_i)\ \mathfrak{R}^{\mathsf{l}}\ s'(\mathtt{x}_i)$ and $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$, using ($\mathfrak{T}$2) we deduce $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ iff $s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$ for all $i, j \in [1, q]$. Hence J is a subset of $[1, q]$ that verifies

(J1) for any $i \in [1, q]$, there exists $j \in J$ s.t. $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ and $s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$;
(J2) for all $i, j \in J$, $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ or $s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$ implies $i = j$.

For every $c \in \{1, 2\}$ and for every $j \in J$, let us consider the following notations:

$$\begin{array}{ccc}
D = \mathrm{dom}(h) & D_c = \mathrm{dom}(h_c) & D' = \mathrm{dom}(h') \\
C = \heartsuit(s, h) & C_c = C \cap D_c & C' = \heartsuit(s', h') \\
P(j) = \mathrm{pred}_{\heartsuit}(s, h, j) & P_c(j) = P(j) \cap D_c & P'(j) = \mathrm{pred}_{\heartsuit}(s', h', j) \\
L = \mathrm{loop}_{\heartsuit}(s, h) & L_c = L \cap D_c & L' = \mathrm{loop}_{\heartsuit}(s', h') \\
R = \mathrm{rem}_{\heartsuit}(s, h) & R_c = R \cap D_c & R' = \mathrm{rem}_{\heartsuit}(s', h')
\end{array}$$

According to Lemma 2.6 and Properties (J1) and (J2), we have the following canonical decompositions:

$$D = C \uplus \biguplus_{j \in J} P(j) \uplus L \uplus R \qquad D' = C' \uplus \biguplus_{j \in J} P'(j) \uplus L' \uplus R'.$$

We know that $\mathfrak{R}^{\mathsf{l}}_{\heartsuit}$ is a one-to-one relation between C and C', and from Proposition 3.10, we have $P(j) \sim_\alpha P'(j)$, $L \sim_\alpha L'$ and $R \sim_\alpha R'$.

Using the bijection $\mathfrak{R}^{\mathsf{l}}_{\heartsuit}$, for $c = 1$ or 2, let us define $C'_c = \mathfrak{R}^{\mathsf{l}}_{\heartsuit}(C_c)$. Then, we have $C = C_1 \uplus C_2$ and $C' = C'_1 \uplus C'_2$. Let us show that l/l' respect both C_1/C'_1 and C_2/C'_2. We have $l\ \mathfrak{R}^{\mathsf{l}}\ l'$ by definition and hence $l\ \mathfrak{T}^{\mathsf{l}}\ l'$. By ($\mathfrak{T}$13), we deduce that l/l' respect C/C'. Hence if $l \in C_1$ then $l \in C$ and thus $l' \in C'$. As a consequence, $l\ \mathfrak{R}^{\mathsf{l}}_{\heartsuit}\ l'$ and thus as $C'_1 = \mathfrak{R}^{\mathsf{l}}_{\heartsuit}(C_1)$, we deduce $l' \in C'_1$. For similar reasons, if $l \in C_2$ then $l' \in C'_2$. Now, if $l' \in C'_1$ then $l' \in C'$ hence $l \in C = C_1 \uplus C_2$. The case $l \in C_2$ would lead to $l' \in C'_2$ hence $l' \in C'_1 \cap C'_2$ which is impossible. Hence we have $l \in C_1$. For similar reasons, if $l' \in C'_2$ then $l \in C_2$. We conclude that l/l' respect both C_1/C'_1 and C_2/C'_2.

Let us verify that l/l' respect both $P(j)/P'(j)$: $l \in P(j)$ iff $h(l) = s(\mathtt{x}_j)$ and $l \notin \heartsuit(s, h)$ iff $h'(l') = s'(\mathtt{x}_j)$ and $l' \notin \heartsuit(s', h')$ iff $l' \in P'(j)$ using $l\ \mathfrak{T}^{\mathsf{l}}\ l'$ with ($\mathfrak{T}$5) and ($\mathfrak{T}$13). By Lemma 2.22, from $P(j) \sim_{\alpha_1+\alpha_2} P'(j)$ and $\alpha_1, \alpha_2 \geqslant 1$, we compute $P'_c(j)$ such that $P'(j) = P'_1(j) \uplus P'_2(j)$, $P_1(j) \sim_{\alpha_1} P'_1(j)$, $P_2(j) \sim_{\alpha_2} P'_2(j)$ and l/l' respect both $P_1(j)/P'_1(j)$ and $P_2(j)/P'_2(j)$.

By a similar argument, we get L'_1 and L'_2 (resp. R'_1 and R'_2) such that $L' = L'_1 \uplus L'_2$, $L_1 \sim_{\alpha_1} L'_1$, $L_2 \sim_{\alpha_2} L'_2$ (resp. $R' = R'_1 \uplus R'_2$, $R_1 \sim_{\alpha_1} R'_1$, $R_2 \sim_{\alpha_2} R'_2$) and l/l' respect both L_1/L'_1 and L_2/L'_2 (resp. R_1/R'_1 and R_2/R'_2).

Now let us define a partition $D' = D'_1 \uplus D'_2$ by

$$D'_1 = C'_1 \uplus \biguplus_{j \in J} P'_1(j) \uplus L'_1 \uplus R'_1 \qquad D'_2 = C'_2 \uplus \biguplus_{j \in J} P'_2(j) \uplus L'_2 \uplus R'_2$$

and h'_1, h'_2 such that $h' = h'_1 \uplus h'_2$, $\mathrm{dom}(h'_1) = D'_1$ and $\mathrm{dom}(h'_2) = D'_2$. We point out that the defining equation of D'_c is not necessarily the canonical decomposition of $D'_c = \mathrm{dom}(h'_c)$ according to (s', h'_c). Since the identities

$$D_1 = C_1 \uplus \biguplus_{j\in J} P_1(j) \uplus L_1 \uplus R_1 \qquad D_2 = C_2 \uplus \biguplus_{j\in J} P_2(j) \uplus L_c \uplus R_2$$

hold, we observe that l/l' respect both D_1/D'_1 and D_2/D'_2.

Using Proposition 3.22, we check that the basic equivalences $(s, h_1, l) \simeq_b (s', h'_1, l')$ and $(s, h_2, l) \simeq_b (s', h'_2, l')$ hold. For $c = 1$ or $c = 2$, let us prove $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{D}_c$:

- we already proved that l/l' respect $\mathrm{dom}(h_c)/\mathrm{dom}(h'_c)$;
- if $s(\mathtt{x}_i) \in \mathrm{dom}(h_c) = D_c$ then $s(\mathtt{x}_i) \in \mathrm{dom}(h)$ and thus $s(\mathtt{x}_i) \in C$. We derive $s(\mathtt{x}_i) \in C_c = C \cap D_c$. Since $s(\mathtt{x}_i) \in \heartsuit(s, h)$ we derive $s'(\mathtt{x}_i) \in \heartsuit(s', h')$ using $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{I}}\ s'(\mathtt{x}_i)$ and ($\mathfrak{T}$13). Thus $s(\mathtt{x}_i)\ \mathfrak{R}^{\mathsf{I}}_{\heartsuit}\ s'(\mathtt{x}_i)$ holds and we deduce $s'(\mathtt{x}_i) \in C'_c$ hence $s'(\mathtt{x}_i) \in D'_c = \mathrm{dom}(h'_c)$. The reverse implication "$s'(\mathtt{x}_i) \in \mathrm{dom}(h'_c)$ implies $s(\mathtt{x}_i) \in \mathrm{dom}(h_c)$" is proved by symmetric arguments;
- if $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h_c) = D_c$ then $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h)$ and thus $h(s(\mathtt{x}_i)) \in C$. We derive $h(s(\mathtt{x}_i)) \in C_c = C \cap D_c$. From $h(s(\mathtt{x}_i))\ \mathfrak{R}^{\mathsf{I}}_{\heartsuit}\ h'(s'(\mathtt{x}_i))$, we deduce $h'(s'(\mathtt{x}_i)) \in C'_c$ hence $h'(s'(\mathtt{x}_i)) \in D'_c = \mathrm{dom}(h'_c)$. The reverse implication "$h'(s'(\mathtt{x}_i)) \in \mathrm{dom}(h'_c)$ implies $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h_c)$" is proved by symmetric arguments.

Hence we have $(s, h, l) \simeq_b (s', h', l')$, $(s, h_1, l) \simeq_b (s', h'_1, l')$ and $(s, h_2, l) \simeq_b (s', h'_2, l')$. According to the above definitions and Proposition 2.9, we get the following identities:

$$\begin{array}{l}\mathrm{pred}_{\overline{\heartsuit}}(s, h_c, j) = P_c(j) \uplus \big(\mathrm{pred}(s, h, j) \cap \Delta(s, h_c, h_{3-c})\big)\\ \mathrm{pred}_{\overline{\heartsuit}}(s', h'_c, j) = P'_c(j) \uplus \big(\mathrm{pred}(s', h', j) \cap \Delta(s', h'_c, h'_{3-c})\big).\end{array}$$

According to Proposition 3.23 item 1 we have

$$\mathrm{pred}(s, h, j) \cap \Delta(s, h_c, h_{3-c}) \sim_\infty \mathrm{pred}(s', h', j) \cap \Delta(s', h'_c, h'_{3-c}).$$

For any $j \in J$, from $P_c(j) \sim_{\alpha_c} P'_c(j)$ we get $\mathrm{pred}_{\overline{\heartsuit}}(s, h_c, j) \sim_{\alpha_c} \mathrm{pred}_{\overline{\heartsuit}}(s', h'_c, j)$ by Lemma 2.19. In fact, we get more precision but we do not need it here.

By similar arguments using Proposition 3.23 items 2 and 3, we establish $\mathrm{loop}_{\overline{\heartsuit}}(s, h_c) \sim_{\alpha_c} \mathrm{loop}_{\overline{\heartsuit}}(s', h'_c)$ and $\mathrm{rem}_{\overline{\heartsuit}}(s, h_c) \sim_{\alpha_c} \mathrm{rem}_{\overline{\heartsuit}}(s', h'_c)$.

Using Properties (J1), we have $\mathrm{pred}_{\overline{\heartsuit}}(s, h_c, i) \sim_{\alpha_c} \mathrm{pred}_{\overline{\heartsuit}}(s', h'_c, i)$ for any $i \in [1, q]$. Hence, by Proposition 3.10, we deduce $(s, h_1, l) \simeq_{\alpha_1} (s', h'_1, l')$ and $(s, h_2, l) \simeq_{\alpha_2} (s', h'_2, l')$. □

Lemma 3.34 below states how to extend a memory state with a heap while preserving equivalence. Some precision (not exceeding q) is lost in the process.

Lemma 3.34 (Compositionality) *Let us consider $\alpha \geqslant 1$, two pointed memory states (s, h_0, l) and (s', h'_0, l') such that $(s, h_0, l) \simeq_{q+\alpha} (s', h'_0, l')$. For any h such that $h \perp h_0$ there exists h' such that $h' \perp h'_0$ and*

1. *$(s, h, l) \simeq_\alpha (s', h', l')$;*

2. $(s, h_0 \uplus h, l) \simeq_\alpha (s', h'_0 \uplus h', l')$;
3. $\mathrm{maxval}(s', h') \leqslant \mathrm{maxval}(s', h'_0, l') + (2\alpha + 3)(q + 2) - 4$.

Proof Let $J \stackrel{\text{def}}{=} \{j \in [1, q] \mid \text{for all } k \in [1, q], s(\mathtt{x}_j) = s(\mathtt{x}_k) \text{ implies } j \leqslant k\}$. So J is a subset of $[1, q]$ that verifies:

(J1) for any $i \in [1, q]$, there exists $j \in J$ such that $s(\mathtt{x}_i) = s(\mathtt{x}_j)$;
(J2) for all $i, j \in J$, $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ implies $i = j$.

We define the following subsets of $\mathrm{dom}(h)$:

- $S = \mathrm{dom}(h) \cap s(\mathcal{V})$;
- $H = \big(\mathrm{dom}(h) \cap \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)\big) \backslash s(\mathcal{V})$;
- $P_j \uplus P'_j = \mathrm{pred}(s, h, j) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$ and $P_j \sim_\alpha P_j \uplus P'_j$ and $\mathrm{card}(P_j) \leqslant \alpha$ for any $j \in J$;
- $L \uplus L' = \mathrm{loop}(s, h) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$ and $L \sim_\alpha L \uplus L'$ and $\mathrm{card}(L) \leqslant \alpha$;
- $R \uplus R' = \mathrm{rem}(s, h) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$ and $R \sim_\alpha R \uplus R'$ and $\mathrm{card}(R) \leqslant \alpha$,

where $(P_j / P'_j)_{j \in J}$, L / L' and R / R' are obtained using Proposition 2.24.[3] From e.g. $L \sim_\alpha L \uplus L'$ and $\mathrm{card}(L) \leqslant \alpha$ we deduce either $\mathrm{card}(L) = \alpha$ or $L' = \varnothing$. Let us check that

$$\mathrm{dom}(h) = S \uplus H \uplus \biguplus_{j \in J} \big(P_j \uplus P'_j\big) \uplus (L \uplus L') \uplus (R \uplus R') \tag{3.1}$$

is indeed a partition of the domain of h. Obviously $S \uplus H = \mathrm{dom}(h) \cap \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. Then $\mathrm{pred}(s, h) = \biguplus_{j \in J} \mathrm{pred}(s, h, j)$ because of Properties (J1) and (J2). From $\mathrm{dom}(h) = (\mathrm{pred}(s, h) \cup \mathrm{loop}(s, h)) \uplus \mathrm{rem}(s, h)$, we deduce

$$\mathrm{dom}(h) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l) = \Big(\biguplus_{j \in J} \big(P_j \uplus P'_j\big) \cup (L \uplus L')\Big) \uplus (R \uplus R').$$

Then the only remaining point is to show that $(P_j \uplus P'_j) \cap (L \uplus L') = \varnothing$. If $u \in (P_j \uplus P'_j) \cap (L \uplus L')$ then we have $u \in \mathrm{pred}(s, h, j)$ and $u \in \mathrm{loop}(s, h)$. Hence $h(u) = s(\mathtt{x}_j)$ and $h(u) = u$. We deduce $u = s(\mathtt{x}_j) \in \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$ which contradicts $u \in L \uplus L'$.

We observe that $\mathrm{card}(S) \leqslant q$ because $S \subseteq s(\mathcal{V})$ and that $\mathrm{card}(H) \leqslant q + 1$ because $H \subseteq (h_0 \uplus h)(s(\mathcal{V})) \cup \{l\}$. Let us define

- h_1 as the restriction of h to S, i.e. $h_1 \sqsubseteq h$ and $\mathrm{dom}(h_1) = S$;
- h_2 as the restriction of h to $H \cup \bigcup_j P_j \cup L \cup R$;
- h_3 as the restriction of $\bigcup_j P'_j \cup L' \cup R'$.

Then we have $h = h_1 \uplus h_2 \uplus h_3$, $\mathrm{card}(\mathrm{dom}(h_1)) \leqslant q$, and $\mathrm{card}(\mathrm{dom}(h_2)) \leqslant \mathrm{card}(H) + \sum_{j \in J} \mathrm{card}(P_j) + \mathrm{card}(L) + \mathrm{card}(R) \leqslant (q + 1) + q.\alpha + \alpha + \alpha = (\alpha + 1)(q + 2) - 1$.

Let us write $p = \mathrm{card}(S) = \mathrm{card}(\mathrm{dom}(h_1)) \leqslant q$ and $m = \mathrm{maxval}(s', h'_0, l')$. We deduce $(s, h_0, l) \simeq_{p+\alpha} (s', h'_0, l')$. By Corollary 3.30, we get h'_1 such that:

[3] In the case of L / L' for instance, we define $L_0 = \mathrm{loop}(s, h) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. Then, by Proposition 2.24 applied to α and L_0, we choose L and L' such that $L_0 = L \uplus L'$, $L \sim_\alpha L \uplus L'$ and $\mathrm{card}(L) \leqslant \alpha$.

- $\mathrm{dom}(h'_1) \subseteq s'(\mathcal{V})$;
- $\mathrm{maxval}(s', h'_1) \leqslant \mathrm{maxval}(s', h'_0, l') + p$;
- $(s, h_1, l) \simeq_\alpha (s', h'_1, l')$;
- $(s, h_0 \uplus h_1, l) \simeq_\alpha (s', h'_0 \uplus h'_1, l')$.

We deduce $\mathrm{maxval}(s', h'_1) \leqslant m + q$ and thus also $\mathrm{maxval}(s', h'_0 \uplus h'_1) \leqslant m + q$.

Then we use Corollary 3.32 for h_2. We have indeed $\mathrm{dom}(h_2) \cap (\mathrm{dom}(h_0 \uplus h_1) \cup s(\mathcal{V})) \subseteq \mathrm{dom}(h_2) \cap (\mathrm{dom}(h_0) \cup s(\mathcal{V})) \subseteq (\mathrm{dom}(h_2) \cap \mathrm{dom}(h_0)) \cup (\mathrm{dom}(h_2) \cap s(\mathcal{V})) \subseteq \varnothing \cup \varnothing \subseteq \varnothing$, $(s, h_1, l) \simeq_\alpha (s', h'_1, l')$ and $(s, h_0 \uplus h_1, l) \simeq_\alpha (s', h'_0 \uplus h'_1, l')$. We obtain a heap h'_2 such that:

- $\mathrm{dom}(h'_1) \cap (\mathrm{dom}(h'_0 \uplus h'_1) \cup s'(\mathcal{V})) = \varnothing$;
- $\mathrm{maxval}(s', h'_2) \leqslant \mathrm{maxval}(s', h'_0 \uplus h'_1, l') + 2.\mathrm{card}(\mathrm{dom}(h_2))$;
- $(s, h_1 \uplus h_2, l) \simeq_\alpha (s', h'_1 \uplus h'_2, l')$;
- $(s, h_0 \uplus h_1 \uplus h_2, l) \simeq_\alpha (s', h'_0 \uplus h'_1 \uplus h'_2, l')$.

We deduce $\mathrm{maxval}(s', h'_2) \leqslant (m+q)+2((\alpha+1)(q+2)-1) = m+(2\alpha+3)(q+2)-4$ and thus also $\mathrm{maxval}(s', h'_1 \uplus h'_2) \leqslant m + (2\alpha + 3)(q + 2) - 4$.

Then, we use Corollary 3.27 to show that $(s, h_1 \uplus h_2 \uplus h_3, l) \simeq_\alpha (s, h_1 \uplus h_2, l)$ holds. By construction of h_3, it is clear that $\mathrm{dom}(h_3) \cap (\mathrm{dom}(h_1 \uplus h_2) \cup \mathfrak{p}\heartsuit(s, h_1 \uplus h_2, l)) = \varnothing$ because $\mathfrak{p}\heartsuit(s, h_1 \uplus h_2, l) \subseteq \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. It is thus sufficient to verify either (C1) or (C2) or (C3) for any $l_1 \in \mathrm{dom}(h_3) = \bigcup_j P'_j \cup L' \cup R'$. We have three cases for $l_1 \in \mathrm{dom}(h_3)$:

- if $l_1 \in P'_j$ for some $j \in J$. Then $h(l_1) = s(\mathtt{x}_j)$ and thus $h_3(l_1) = s(\mathtt{x}_j)$. Moreover $P'_j \neq \varnothing$ and thus we must have $\mathrm{card}(P_j) = \alpha$ (because $P_j \sim_\alpha P_j \uplus P'_j$ and $\mathrm{card}(P_j) \leqslant \alpha$). Let us prove $P_j \subseteq \mathrm{pred}_{\overline{\heartsuit}}(s, h_1 \uplus h_2, j)$.
  We have $P_j \subseteq (\mathrm{pred}(s, h, j) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)) \cap \mathrm{dom}(h_2)$ hence we deduce $P_j \subseteq (\mathrm{pred}(s, h, j) \cap \mathrm{dom}(h_2)) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. But $\mathrm{pred}(s, h, j) \cap \mathrm{dom}(h_2) \subseteq \mathrm{pred}(s, h_1 \uplus h_2, j)$ because $h_2 \sqsubseteq h_1 \uplus h_2 \sqsubseteq h$; and $\heartsuit(s, h_1 \uplus h_2) \subseteq \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$ because $h_1 \uplus h_2 \sqsubseteq h_0 \uplus h$. We get $P_j \subseteq \mathrm{pred}(s, h_1 \uplus h_2, j) \backslash \heartsuit(s, h_1 \uplus h_2) = \mathrm{pred}_{\overline{\heartsuit}}(s, h_1 \uplus h_2, j)$.
  We deduce $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h_1 \uplus h_2, j)) \geqslant \alpha$. Condition (C1) holds for l_1;
- if $l_1 \in L'$ then $h(l_1) = l_1$ hence $h_3(l_1) = l_1$. Since L' is not empty, we have $\mathrm{card}(L) = \alpha$ and we show that $L \subseteq \mathrm{loop}_{\overline{\heartsuit}}(s, h_1 \uplus h_2)$.
  We have $L \subseteq (\mathrm{loop}(s, h) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)) \cap \mathrm{dom}(h_2)$ hence $L \subseteq (\mathrm{loop}(s, h) \cap \mathrm{dom}(h_2)) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. But $\mathrm{loop}(s, h) \cap \mathrm{dom}(h_2) \subseteq \mathrm{loop}(s, h_1 \uplus h_2)$ and $\heartsuit(s, h_1 \uplus h_2) \subseteq \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. We get $L \subseteq \mathrm{loop}(s, h_1 \uplus h_2) \backslash \heartsuit(s, h_1 \uplus h_2)$.
  We deduce $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h_1 \uplus h_2)) \geqslant \alpha$. Condition (C2) holds for l_1;
- if $l_1 \in R'$ then $h(l_1) \notin s(\mathcal{V}) \cup \{l_1\}$ hence $h_3(l_1) \notin s(\mathcal{V}) \cup \{l_1\}$. Since R' is not empty, we have $\mathrm{card}(R) = \alpha$ and we show that $R \subseteq \mathrm{rem}_{\overline{\heartsuit}}(s, h_1 \uplus h_2)$.
  We have $R \subseteq (\mathrm{rem}(s, h) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)) \cap \mathrm{dom}(h_2)$ hence $R \subseteq (\mathrm{rem}(s, h) \cap \mathrm{dom}(h_2)) \backslash \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. But $\mathrm{rem}(s, h) \cap \mathrm{dom}(h_2) \subseteq \mathrm{rem}(s, h_1 \uplus h_2)$ and $\heartsuit(s, h_1 \uplus h_2) \subseteq \mathfrak{p}\heartsuit(s, h_0 \uplus h, l)$. We get $R \subseteq \mathrm{rem}(s, h_1 \uplus h_2) \backslash \heartsuit(s, h_1 \uplus h_2)$.
  We deduce $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h_1 \uplus h_2)) \geqslant \alpha$. Condition (C3) holds for l_1.

Hence, by Corollary 3.27, we deduce that $(s, h_1 \uplus h_2 \uplus h_3, l) \simeq_\alpha (s, h_1 \uplus h_2, l)$ holds. By similar arguments, we show that $(s, h_0 \uplus h_1 \uplus h_2 \uplus h_3, l) \simeq_\alpha (s, h_0 \uplus h_1 \uplus h_2, l)$ holds as well.

Let us finally show that $h' = h'_1 \uplus h'_2$ satisfies the required conditions. We have already proved $\mathrm{maxval}(s', h') \leqslant \mathrm{maxval}(s', h'_0, l') + (2\alpha + 3)(q + 2) - 4$. Then we have $(s, h, l) = (s, h_1 \uplus h_2 \uplus h_3, l) \simeq_\alpha (s, h_1 \uplus h_2, l) \simeq_\alpha (s', h'_1 \uplus h'_2, l') = (s', h', l')$ and $(s, h_0 \uplus h, l) = (s, h_0 \uplus h_1 \uplus h_2 \uplus h_3, l) \simeq_\alpha (s, h_0 \uplus h_1 \uplus h_2, l) \simeq_\alpha (s', h'_0 \uplus h'_1 \uplus h'_2, l') = (s', h'_0 \uplus h', l')$. □

Lemma 3.35 below states how to update the location of the quantified variable while preserving equivalence. No precision is lost here.

Lemma 3.35 (Existence) *Let $\alpha \geqslant 1$ and (s, h, l) and (s', h', l') be two α-equivalent pointed memory states, i.e. $(s, h, l) \simeq_\alpha (s', h', l')$. For every $l_0 \in \mathbb{N}$, there exists $l'_0 \leqslant \mathrm{maxval}(s', h') + 1$ such that $(s, h, l_0) \simeq_\alpha (s', h', l'_0)$.*

Proof By Proposition 3.10, we have $(s, h, l) \simeq_b (s', h', l')$ together with three α-equipotence constraints: $\mathrm{pred}_{\overline{\heartsuit}}(s, h, i) \sim_\alpha \mathrm{pred}_{\overline{\heartsuit}}(s', h', i)$ for any $i \in [1, q]$, $\mathrm{loop}_{\overline{\heartsuit}}(s, h) \sim_\alpha \mathrm{loop}_{\overline{\heartsuit}}(s', h')$, and $\mathrm{rem}_{\overline{\heartsuit}}(s, h) \sim_\alpha \mathrm{rem}_{\overline{\heartsuit}}(s', h')$.

Let $l_0 \in \mathbb{N}$. Let us consider the relation $\mathfrak{T}_{\mathfrak{m},\mathfrak{m}'}$ of Definition 3.14 where $\mathfrak{m} = (s, h, l)$ and $\mathfrak{m}' = (s', h', l')$. By Proposition 3.21, there exists $l'_0 \in \mathbb{N}$ such that $l'_0 \leqslant \mathrm{maxval}(s', h') + 1$ and $l_0 \; \mathfrak{T}_{\mathfrak{m},\mathfrak{m}'} \; l'_0$ holds. By definition of $\mathfrak{T}_{\mathfrak{m},\mathfrak{m}'}$, this means that l_0/l'_0 verify ($\mathfrak{T}$2–6). Hence by Proposition 3.24, we get $(s, h, l_0) \simeq_b (s', h', l'_0)$, and by Proposition 3.10, we conclude $(s, h, l_0) \simeq_\alpha (s', h', l'_0)$. □

4 Decidability, Expressiveness and Complexity

In this section, we show that two α-equivalent pointed memory states cannot be distinguished by 1SL1 formulæ of memory threshold less than α. We introduce model compression results and deduce that the unbounded quantifications used in the definitions of $(s, h) \models_l \exists \mathtt{u}\, \mathcal{A}$ and $(s, h) \models_l \mathcal{A} \mathrel{-\!\!*} \mathcal{B}$ can be replaced by bounded quantifications. We then derive decidability results for the model-checking and satisfiability problems in 1SL1 and a quantifier elimination result for 1SL1: any formula of 1SL1 is equivalent to a Boolean combination of test formulæ in $\mathsf{Test}^{\mathtt{u}}_\alpha$ for some threshold $\alpha \geqslant 1$. Then we provide a PSPACE complexity characterisation for both model-checking and satisfiability in 1SL1 using a bounded model-checking algorithm.

4.1 Correctness of the Abstraction

Given the three previous results, Lemma 3.33 for distributivity, Lemma 3.34 for compositionality and Lemma 3.35 for existence, we design a notion of memory threshold that matches the loss of precision induced by the separating conjunction, the separating implication and the first-order quantification.

Definition 4.1 (Memory Threshold) Given $q \geqslant 1$ and a 1SL1 formula $\mathcal{A}$ built over the program variables $\mathtt{x}_1, \ldots, \mathtt{x}_q$, we define its *memory threshold*

$\mathsf{th}(q, \mathcal{A})$ inductively as follows:

$$\begin{array}{rcl@{\qquad}rcl}
\mathsf{th}(q, \neg\mathcal{A}_1) & \stackrel{def}{=} & \mathsf{th}(q, \mathcal{A}_1) & \mathsf{th}(q, \mathcal{A}_1 \wedge \mathcal{A}_2) & \stackrel{def}{=} & \max\big(\mathsf{th}(q, \mathcal{A}_1), \mathsf{th}(q, \mathcal{A}_2)\big) \\
\mathsf{th}(q, \exists \mathtt{u}\, \mathcal{A}_1) & \stackrel{def}{=} & \mathsf{th}(q, \mathcal{A}_1) & \mathsf{th}(q, \mathcal{A}_1 * \mathcal{A}_2) & \stackrel{def}{=} & \mathsf{th}(q, \mathcal{A}_1) + \mathsf{th}(q, \mathcal{A}_2) \\
 & & & \mathsf{th}(q, \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2) & \stackrel{def}{=} & q + \max\big(\mathsf{th}(q, \mathcal{A}_1), \mathsf{th}(q, \mathcal{A}_2)\big)
\end{array}$$

$$\mathsf{th}(q, \mathcal{A}_1) \stackrel{def}{=} 1 \quad \text{for every atomic formula } \mathcal{A}_1 \text{ in } \pi \cup \{\bot, \mathtt{emp}\}.$$

For instance, $\mathsf{th}(3, (\mathtt{x}_1 \hookrightarrow \mathtt{x}_1) \mathrel{-\!\!*} \bot) = 3 + \max(1, 1) = 4$. The rationale for these inductive definitions comes from the proof of the upcoming correctness result and how Lemmas 3.33, 3.34 and 3.35 are used there. In the case of $\mathcal{A}_1 * \mathcal{A}_2$, the use of the Distributivity Lemma 3.33 implies that precision is split in half. In the case of $\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$, the use of the Compositionality Lemma 3.34 implies a loss of precision bounded by q.

Proposition 4.2 *Given $q \geqslant 1$ and a formula $\mathcal{A}$ in 1SL1 with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$, we have $1 \leqslant \mathsf{th}(q, \mathcal{A}) \leqslant q.|\mathcal{A}|$.*

The proof is left to the reader.

Now we state the correctness result which means that test formulæ in $\mathsf{Test}^{\mathtt{u}}_{\alpha}$ provide the proper abstraction for the formulæ of 1SL1 with a memory threshold bounded by α.

Theorem 4.3 (Abstraction Correctness) *Let $q \geqslant 1$. For any 1SL1 formula $\mathcal{A}$ with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$, for any $\alpha \geqslant 1$, if $\mathsf{th}(q, \mathcal{A}) \leqslant \alpha$ and $(s, h, l) \simeq_\alpha (s', h', l')$ hold then $(s, h) \models_l \mathcal{A}$ iff $(s', h') \models_{l'} \mathcal{A}$.*

Proof The proof is by induction on the structure of $\mathcal{A}$. Suppose that $(s, h, l) \simeq_\alpha (s', h', l')$ and $\mathcal{A}$ be a formula with $\mathsf{th}(q, \mathcal{A}) \leqslant \alpha$. By structural induction, we show that $(s, h) \models_l \mathcal{A}$ if and only if $(s', h') \models_{l'} \mathcal{A}$: but we only display the proof of the *only if* implication, the converse implication is obtained by symmetry.

- $\mathcal{A}$ is $\mathtt{e} \hookrightarrow \mathtt{e}'$ which is covered by one of the following cases: $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$, $\mathtt{x}_i \hookrightarrow \mathtt{u}$, $\mathtt{u} \hookrightarrow \mathtt{x}_i$ and $\mathtt{u} \hookrightarrow \mathtt{u}$. All of these formulæ belong to $\mathsf{Basic}^{\mathtt{u}} \subseteq \mathsf{Test}^{\mathtt{u}}_{\alpha}$;
- $\mathcal{A}$ is $\mathtt{e} = \mathtt{e}'$ which is covered by one of the following cases: $\mathtt{x}_i = \mathtt{x}_j$, $\mathtt{x}_i = \mathtt{u}$, $\mathtt{u} = \mathtt{x}_i$ and $\mathtt{u} = \mathtt{u}$. The two first belong to $\mathsf{Basic}^{\mathtt{u}} \subseteq \mathsf{Test}^{\mathtt{u}}_{\alpha}$, $\mathtt{u} = \mathtt{x}_i$ is logically equivalent to $\mathtt{x}_i = \mathtt{u}$ and $\mathtt{u} = \mathtt{u}$ is a tautology;
- $\mathcal{A}$ is $\mathtt{emp}$ which is logically equivalent to

  $$\neg\big(\bigvee_i \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \vee \bigvee_i \mathtt{toalloc}(\mathtt{x}_i) \vee \bigvee_i \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant 1 \vee \#\,\mathtt{loop}_{\heartsuit} \geqslant 1 \vee \#\,\mathtt{rem}_{\heartsuit} \geqslant 1\big)$$

  by the canonical decomposition of Lemma 2.6. This formula is a Boolean combination of formulæ of $\mathsf{Test}^{\mathtt{u}}_1 \subseteq \mathsf{Test}^{\mathtt{u}}_{\alpha}$. Remember that $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)$ is used in place of $\mathtt{alloc}(\mathtt{x}_i)$;
- $\mathcal{A}$ is $\exists \mathtt{u}\, \mathcal{A}_1$ with $(s, h) \models_l \exists \mathtt{u}\, \mathcal{A}_1$ and $(s, h, l) \simeq_\alpha (s', h', l')$ and $\mathsf{th}(q, \exists \mathtt{u}\, \mathcal{A}_1) \leqslant \alpha$. There exists l_0 such that $(s, h) \models_{l_0} \mathcal{A}_1$. By Lemma 3.35, there is l_1 such that $(s, h, l_0) \simeq_\alpha (s', h', l_1)$. Since we have $(s, h) \models_{l_0} \mathcal{A}_1$, by induction hypothesis we get $(s', h') \models_{l_1} \mathcal{A}_1$ (note that $\mathsf{th}(q, \mathcal{A}_1) = \mathsf{th}(q, \exists \mathtt{u}\, \mathcal{A}_1) \leqslant \alpha$). Thus we conclude $(s', h') \models_{l'} \exists \mathtt{u}\, \mathcal{A}_1$;

– $\mathcal{A}$ is $\mathcal{A}_1 * \mathcal{A}_2$ with $(s,h) \models_l \mathcal{A}_1 * \mathcal{A}_2$, $(s,h,l) \simeq_\alpha (s',h',l')$, $\mathsf{th}(q, \mathcal{A}_1 * \mathcal{A}_2) \leqslant \alpha$. There are heaps h_1 and h_2 such that $h = h_1 \uplus h_2$ and $(s,h_1) \models_l \mathcal{A}_1$ and $(s,h_2) \models_l \mathcal{A}_2$. As $\alpha \geqslant \mathsf{th}(q,\mathcal{A}) = \mathsf{th}(q,\mathcal{A}_1) + \mathsf{th}(q,\mathcal{A}_2)$, there exist α_1 and α_2 such that $\alpha = \alpha_1 + \alpha_2$ and $\alpha_1 \geqslant \mathsf{th}(q,\mathcal{A}_1)$ and $\alpha_2 \geqslant \mathsf{th}(q,\mathcal{A}_2)$. By Lemma 3.33, there exist heaps h'_1 and h'_2 such that $h' = h'_1 \uplus h'_2$ and $(s,h_1,l) \simeq_{\alpha_1} (s',h'_1,l')$ and $(s,h_2,l) \simeq_{\alpha_2} (s',h'_2,l')$. By the induction hypothesis, we get $(s',h'_1) \models_{l'} \mathcal{A}_1$ and $(s',h'_2) \models_{l'} \mathcal{A}_2$ (since $\mathsf{th}(q,\mathcal{A}_1) \leqslant \alpha_1$ and $\mathsf{th}(q,\mathcal{A}_2) \leqslant \alpha_2$). Consequently we obtain $(s',h') \models_{l'} \mathcal{A}_1 * \mathcal{A}_2$;

– $\mathcal{A}$ is $\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ with $(s,h) \models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ and $(s,h,l) \simeq_\alpha (s',h',l')$ and

$$q + \beta = \mathsf{th}(q, \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2) \leqslant \alpha \quad \text{with} \quad \beta = \max(\mathsf{th}(q,\mathcal{A}_1), \mathsf{th}(q,\mathcal{A}_2)).$$

We deduce $(s,h,l) \simeq_{q+\beta} (s',h',l')$. Let us prove $(s',h') \models_{l'} \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$. We pick h'_1 such that $h'_1 \perp h'$ and $(s',h'_1) \models_{l'} \mathcal{A}_1$ and show that $(s', h' \uplus h'_1) \models_{l'} \mathcal{A}_2$. By Lemma 3.34, there is a heap h_1 such that $h_1 \perp h$ and $(s,h_1,l) \simeq_\beta (s',h'_1,l')$ and $(s, h \uplus h_1, l) \simeq_\beta (s', h' \uplus h'_1, l')$. We deduce that $(s,h_1) \models_l \mathcal{A}_1$ holds by the induction hypothesis (since $\mathsf{th}(q,\mathcal{A}_1) \leqslant \beta$). As $h_1 \perp h$ and $(s,h) \models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ hold, we deduce $(s, h \uplus h_1) \models_l \mathcal{A}_2$. By induction hypothesis, we deduce $(s', h' \uplus h'_1) \models_{l'} \mathcal{A}_2$ (since $\mathsf{th}(q,\mathcal{A}_2) \leqslant \beta$). Hence we have $(s',h') \models_{l'} \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$.

The induction step for a Boolean outermost connective is straightforward and therefore this concludes the proof. □

4.2 Model Checking, Satisfiability and Expressive Completeness

Lemma 3.34 and the proof of Lemma 3.35 suggest a "compressor" lemma that can operate at any precision and compress either the location or the heap of a pointed memory state.

Lemma 4.4 (Compressor) *Let $q \geqslant 1$ and let (s,h,l) be a pointed memory state. The two following statements hold:*

1. *there exists $l' \leqslant \mathrm{maxval}(s,h) + 1$ s.t. $(s,h,l) \simeq_\alpha (s,h,l')$ holds for any α;*
2. *for any $\alpha \geqslant 1$, any $h_1 \perp h$, there exists $h'_1 \perp h$ s.t. $(s,h_1,l) \simeq_\alpha (s,h'_1,l)$ and $(s, h \uplus h_1, l) \simeq_\alpha (s, h \uplus h'_1, l)$ and $\mathrm{maxval}(h'_1) \leqslant \mathrm{maxval}(s,h,l) + 15q\alpha$.*

Proof Let $\mathfrak{m} = (s,h,l)$ be a pointed memory state.

Let us start with Statement 1. Let us define l' in the following way: if $l \in \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s,h)$ then $l' = l$; and $l' = \mathrm{maxval}(s,h)+1$ if $l \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s,h)$. In the former case, we obviously have $l \; \mathfrak{T}_{\mathfrak{m},\mathfrak{m}} \; l'$ because $\mathfrak{T}_{\mathfrak{m},\mathfrak{m}}$ is reflexive. In the later case, we get $l \; \mathfrak{T}_{\mathfrak{m},\mathfrak{m}} \; l'$ by Proposition 3.16 item 3. Hence l/l' verify ($\mathfrak{T}$2–6) with respect to $\mathfrak{m}/\mathfrak{m}$. In both cases, the relation $l' \leqslant \mathrm{maxval}(s,h) + 1$ is obvious. Since $(s,h,l) \simeq_b (s,h,l)$ holds by reflexivity and l/l' verify ($\mathfrak{T}$2–6) then by Proposition 3.24, we get $(s,h,l) \simeq_b (s,h,l')$. Let $\alpha \geqslant 0$. By Proposition 3.10, only the α-equipotence constraints $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \sim_\alpha \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$,

$\mathrm{loop}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ and $\mathrm{rem}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ remain. By reflexivity, they hold trivially.

Now let us prove Statement 2. Let $\alpha \geqslant 1$ and $h_1 \perp h$. By reflexivity we have $(s,h,l) \simeq_{q+\alpha} (s,h,l)$. Hence by Lemma 3.34, there exists a heap h'_1 such that $(s,h_1,l) \simeq_\alpha (s,h'_1,l)$, $(s, h \uplus h_1, l) \simeq_\alpha (s, h \uplus h'_1, l)$ and $\mathrm{maxval}(s,h'_1) \leqslant \mathrm{maxval}(s,h,l) + (2\alpha+3)(q+2) - 4$. As $\alpha \geqslant 1$ and $q \geqslant 1$, we get the relation $(2\alpha+3)(q+2)-4 \leqslant 15q\alpha$. We deduce $\mathrm{maxval}(h'_1) \leqslant \mathrm{maxval}(s,h,l)+15q\alpha$. □

We derive two corollaries that aim at replacing unbounded/infinite quantification with bounded/finite quantification in the respective definitions of $(s,h) \models_l \exists \mathtt{u}\, \mathcal{A}$ and $(s,h) \models_l \mathcal{A} \mathrel{-\!\!*} \mathcal{B}$.

Corollary 4.5 *Let $q \geqslant 1$. For any pointed memory state (s,h,l) and for any 1SL1 formula $\mathcal{A}$ with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$, we have:*

$$(s,h) \models_l \exists \mathtt{u}\, \mathcal{A} \quad \textit{iff} \quad \textit{there exists } l' \leqslant \mathrm{maxval}(s,h)+1 \textit{ such that } (s,h) \models_{l'} \mathcal{A}.$$

Proof The *if case* is trivial. For the *only if case*, let us assume $(s,h) \models_l \exists \mathtt{u}\, \mathcal{A}$. Then there exists $l_0 \in \mathbb{N}$ such that $(s,h) \models_{l_0} \mathcal{A}$. By Lemma 4.4, there exists $l' \leqslant \mathrm{maxval}(s,h)+1$ such that $(s,h,l_0) \simeq_\alpha (s,h,l')$ holds for any α. Choosing e.g. $\alpha = \mathsf{th}(q,\mathcal{A})$, we deduce $(s,h) \models_{l'} \mathcal{A}$ by Theorem 4.3. □

Corollary 4.6 *Let $q \geqslant 1$. Let (s,h,l) be a pointed memory state and $\mathcal{A}, \mathcal{B}$ be two 1SL1 formulæ with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$. Let us define $m = \mathrm{maxval}(s,h,l) + 15|\mathcal{A} \mathrel{-\!\!*} \mathcal{B}|q^2$. Then $(s,h) \models_l \mathcal{A} \mathrel{-\!\!*} \mathcal{B}$ if and only if*

$$\textit{for any } h_1 \perp h \textit{ with } \mathrm{maxval}(h_1) \leqslant m,\ (s,h_1) \models_l \mathcal{A} \textit{ implies } (s, h \uplus h_1) \models_l \mathcal{B}.$$

Proof The *only if case* is trivial. For the *if case*, let us assume that $(s,h_1) \models_l \mathcal{A}$ implies $(s, h \uplus h_1) \models_l \mathcal{B}$ for any $h_1 \perp h$ such that $\mathrm{maxval}(h_1) \leqslant m$ and show that $(s,h) \models_l \mathcal{A} \mathrel{-\!\!*} \mathcal{B}$ holds. Let us consider a heap h' such that $h' \perp h$ and $(s,h') \models_l \mathcal{A}$ and prove $(s, h \uplus h') \models_l \mathcal{B}$. By Lemma 4.4, for $\alpha = q|\mathcal{A} \mathrel{-\!\!*} \mathcal{B}|$ there exists a heap h'' such that $h'' \perp h$, $(s,h',l) \simeq_\alpha (s,h'',l)$, $(s, h \uplus h', l) \simeq_\alpha (s, h \uplus h'', l)$ and $\mathrm{maxval}(h'') \leqslant \mathrm{maxval}(s,h,l) + 15q\alpha$. We deduce $\mathrm{maxval}(h'') \leqslant m$.

But we have $\mathsf{th}(q,\mathcal{A}) \leqslant q|\mathcal{A}| \leqslant \alpha$. Hence by Theorem 4.3, from $(s,h') \models_l \mathcal{A}$ and $(s,h',l) \simeq_\alpha (s,h'',l)$ we deduce $(s,h'') \models_l \mathcal{A}$. Then by hypothesis with h_1 as h'', we deduce $(s, h \uplus h'') \models_l \mathcal{B}$. By Theorem 4.3, from $\mathsf{th}(q,\mathcal{B}) \leqslant q|\mathcal{B}| \leqslant \alpha$ and $(s, h \uplus h', l) \simeq_\alpha (s, h \uplus h'', l)$ we conclude $(s, h \uplus h') \models_l \mathcal{B}$. □

We also deduce a *model compression* result under α-equivalence and then we derive a *small model property* as a consequence of Theorem 4.3.

Corollary 4.7 *Let $q, \alpha \geqslant 1$. Every pointed memory state is α-equivalent to a pointed memory state (s,h,l) such that $\mathrm{maxval}(s,h,l) \leqslant 16q\alpha$.*

Proof Let (s_0,h_0,l_0) be a pointed memory state. The set $s_0(\mathcal{V}) \cup \{l_0\}$ is a subset of $\mathbb{N}$ of cardinal less than $q+1$. Hence there exists a bijection $\varphi : \mathbb{N} \to \mathbb{N}$ such that $\varphi(s_0(\mathcal{V}) \cup \{l_0\}) \subseteq [0,q]$. We define $s = \varphi \circ s_0$, $h_1 = \varphi \circ h_0 \circ \varphi^{-1}$ and $l =$

$\varphi(l_0)$. By Proposition 2.3, the pointed memory states (s_0, h_0, l_0) and (s, h_1, l) do not distinguish any formula of 1SL1, hence we have $(s_0, h_0, l_0) \simeq_\alpha (s, h_1, l)$.

Now we consider the pointed memory state $(s, \Box, l)$. As $\varphi(s_0(\mathcal{V}) \cup \{l_0\}) \subseteq [0, q]$ we deduce $s(\mathcal{V}) \cup \{l\} \subseteq [0, q]$ hence $\mathrm{maxval}(s, \Box, l) \leqslant q$. We use Lemma 4.4 item 2 with $\alpha \geqslant 1$ and $h_1 \perp \Box$, hence there exists h such that $(s, h_1, l) \simeq_\alpha (s, h, l)$ (and $(s, \Box \uplus h_1, l) \simeq_\alpha (s, \Box \uplus h, l)$) and $\mathrm{maxval}(h) \leqslant \mathrm{maxval}(s, \Box, l) + 15q\alpha$. We deduce that (s_0, h_0, l_0) and (s, h, l) are α-equivalent. Moreover, $\mathrm{maxval}(s, h, l) \leqslant \max(q, q + 15q\alpha) \leqslant 16q\alpha$. □

Corollary 4.8 (Small Model Property) *Let $\mathcal{A}$ be a 1SL1 formula with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$. If $\mathcal{A}$ is satisfiable then there exists a pointed memory state (s, h, l) such that $(s, h) \models_l \mathcal{A}$ and $\mathrm{maxval}(s, h, l) \leqslant 16q.\mathsf{th}(q, \mathcal{A})$.*

Proof Let $\alpha = \mathsf{th}(q, \mathcal{A})$. Let (s', h', l') be a pointed memory state that satisfies $\mathcal{A}$, i.e. $(s', h') \models_{l'} \mathcal{A}$. By Corollary 4.7, there exists a pointed memory state such that $(s', h', l') \simeq_\alpha (s, h, l)$ and $\mathrm{maxval}(s, h, l) \leqslant 16q\alpha = 16q.\mathsf{th}(q, \mathcal{A})$. By Theorem 4.3, from $\mathsf{th}(q, \mathcal{A}) \leqslant \alpha$ we deduce $(s, h) \models_l \mathcal{A}$. □

Now we can give proofs of high-level decidability results as easy consequences of the previous Compressor Lemma 4.4 and its corollaries.

Theorem 4.9 (Decidability of Model-Checking) *The problem of checking whether a 1SL1 formula $\mathcal{A}$ and a pointed memory state (s, h, l) verify $(s, h) \models_l \mathcal{A}$, is decidable.*

Proof We fix $q \geqslant 1$ and $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$. We show that the predicate

$$\big(\mathcal{A}, (s, h, l)\big) \mapsto (s, h) \models_l \mathcal{A}$$

is decidable for any formula $\mathcal{A}$ with program variables in $\mathcal{V}$. By choosing q and $\mathcal{V}$ large enough, we can then ensure the decidability of model-checking for any formula of 1SL1.

Given a formula $\mathcal{A}$ with program variables in $\mathcal{V}$ and a pointed memory state (s, h, l), we define a function $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big)$ which returns a value in $\{\mathtt{ff}, \mathtt{tt}\}$, by structural induction on the formula $\mathcal{A}$:

if $\mathcal{A}$ is either $\mathtt{e} = \mathtt{e}'$ or $\mathtt{e} \hookrightarrow \mathtt{e}'$ or $\mathtt{emp}$, then we define $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big)$ such that $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big) = \mathtt{tt}$ iff $(s, h) \models_l \mathcal{A}$ (the details are left to the reader, same as upcoming function $\mathtt{amc}$ in Figure 4.1 page 41);

if the principal connective of $\mathcal{A}$ is Boolean, for instance if $\mathcal{A} = \mathcal{A}_1 \wedge \mathcal{A}_2$ then $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big) = \mathtt{tt}$ if both

$$\mathtt{mc}\big(\mathcal{A}_1, (s, h, l)\big) = \mathtt{tt} \quad \text{and} \quad \mathtt{mc}\big(\mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$$

hold; otherwise $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big) = \mathtt{ff}$;

if $\mathcal{A}$ is $\exists \mathtt{u}\, \mathcal{A}_1$ then $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big) = \mathtt{tt}$ if $\mathtt{mc}\big(\mathcal{A}_1, (s, h, l')\big) = \mathtt{tt}$ holds for some $l' \leqslant \mathrm{maxval}(s, h) + 1$; otherwise $\mathtt{mc}\big(\mathcal{A}, (s, h, l)\big) = \mathtt{ff}$;

if $\mathcal{A}$ is $\mathcal{A}_1 * \mathcal{A}_2$ then $\mathtt{mc}\big(\mathcal{A}, (s,h,l)\big) = \mathtt{tt}$ if both

$$\mathtt{mc}\big(\mathcal{A}_1, (s,h_1,l)\big) = \mathtt{tt} \quad \text{and} \quad \mathtt{mc}\big(\mathcal{A}_2, (s,h_2,l)\big) = \mathtt{tt}$$

hold for some heaps h_1 and h_2 such that $h = h_1 \uplus h_2$ (there are only finitely many such splits); otherwise $\mathtt{mc}\big(\mathcal{A}, (s,h,l)\big) = \mathtt{ff}$;

if $\mathcal{A}$ is $\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ then $\mathtt{mc}\big(\mathcal{A}, (s,h,l)\big) = \mathtt{ff}$ if both

$$\mathtt{mc}\big(\mathcal{A}_1, (s,h_1,l)\big) = \mathtt{tt} \quad \text{and} \quad \mathtt{mc}\big(\mathcal{A}_2, (s,h \uplus h_1,l)\big) = \mathtt{ff}$$

hold for some heap h_1 such that $h_1 \perp h$ and

$$\mathrm{maxval}(h_1) \leqslant \mathrm{maxval}(s,h,l) + 15|\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2|q^2$$

(there are only finitely many such h_1); otherwise $\mathtt{mc}\big(\mathcal{A}, (s,h,l)\big) = \mathtt{tt}$.

The termination of $\mathtt{mc}$ is by induction on $\mathcal{A}$: any call to $\mathtt{mc}\big(\mathcal{A}, (s,h,l)\big)$ only generates finitely many recursive sub-calls on the immediate subformulæ of $\mathcal{A}$.

Let us prove the correctness of the function $\mathtt{mc}$. We show by induction on $\mathcal{A}$ that if $\mathcal{A}$ has its program variables in $\mathcal{V}$ then for any pointed memory state (s,h,l), the equivalence

$$(s,h) \models_l \mathcal{A} \quad \text{if and only if} \quad \mathtt{mc}\big(\mathcal{A}, (s,h,l)\big) = \mathtt{tt}$$

holds. The proof uses Corollaries 4.5 and 4.6 in an obvious way. For instance, let us consider the case where $\mathcal{A} = \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$:

- let us assume $(s,h) \models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ and let us show $\mathtt{mc}\big(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s,h,l)\big) = \mathtt{tt}$. We show that $\mathtt{mc}\big(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s,h,l)\big) = \mathtt{ff}$ leads to a contradiction. Indeed, in that case, there exists a heap h_1 such that $h_1 \perp h$ and $\mathtt{mc}\big(\mathcal{A}_1, (s,h_1,l)\big) = \mathtt{tt}$ and $\mathtt{mc}\big(\mathcal{A}_2, (s,h \uplus h_1,l)\big) = \mathtt{ff}$. By the induction hypothesis, we deduce $(s,h_1) \models_l \mathcal{A}_1$ and $(s,h \uplus h_1) \not\models_l \mathcal{A}_2$. As a consequence we get $(s,h) \not\models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ which contradicts the hypothesis. Hence $\mathtt{mc}\big(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s,h,l)\big) = \mathtt{ff}$ is impossible so we must have $\mathtt{mc}\big(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s,h,l)\big) = \mathtt{tt}$;
- let us assume $\mathtt{mc}\big(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s,h,l)\big) = \mathtt{tt}$. We show $(s,h) \models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$. We use Corollary 4.6. Let us consider a heap h_1 such that $h_1 \perp h$, $\mathrm{maxval}(h_1) \leqslant \mathrm{maxval}(s,h,l) + 15|\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2|q^2$ and $(s,h_1) \models_l \mathcal{A}_1$. Let us show $(s,h \uplus h_1) \models_l \mathcal{A}_2$. By contradiction, if $(s,h \uplus h_1) \not\models_l \mathcal{A}_2$ then by induction, we have $\mathtt{mc}\big(\mathcal{A}_1, (s,h_1,l)\big) = \mathtt{tt}$ and $\mathtt{mc}\big(\mathcal{A}_2, (s,h \uplus h_1,l)\big) = \mathtt{ff}$. From the definition of $\mathtt{mc}$, we derive $\mathtt{mc}\big(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s,h,l)\big) = \mathtt{ff}$ which leads to a contradiction. Hence we must have $(s,h \uplus h_1) \models_l \mathcal{A}_2$. □

Later we show how to transform the decidability proof into a bounded model-checking algorithm that runs in PSPACE.

Theorem 4.10 (Decidability of Satisfiability) *The problem of checking whether an 1SL1 formula $\mathcal{A}$ admits a pointed memory state (s,h,l) such that $(s,h) \models_l \mathcal{A}$, is decidable.*

This is a direct consequence of Corollary 4.8 and Theorem 4.9. Here is our main result characterising the expressive power of 1SL1 in terms of Boolean combination of test formulæ.

Theorem 4.11 (Quantifier Admissibility) *Let $q \geqslant 1$. Every formula $\mathcal{A}$ in 1SL1 with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$ is logically equivalent to a Boolean combination of test formulæ in $\mathsf{Test}^{\mathrm{u}}_{\alpha}$ with $\alpha = \mathsf{th}(q, \mathcal{A})$.*

Proof Let $\alpha = \mathsf{th}(q, \mathcal{A})$ and consider the (saturated) set of literals

$$\mathcal{S}_\alpha(s,h,l) \stackrel{\text{def}}{=} \left[\begin{array}{c} \{\mathcal{B} \mid \mathcal{B} \in \mathsf{Test}^{\mathrm{u}}_{\alpha} \text{ and } (s,h) \models_l \mathcal{B}\} \\ \cup\ \{\neg\mathcal{B} \mid \mathcal{B} \in \mathsf{Test}^{\mathrm{u}}_{\alpha} \text{ and } (s,h) \not\models_l \mathcal{B}\} \end{array} \right]$$

As $\mathsf{Test}^{\mathrm{u}}_{\alpha}$ is finite, the set $\mathcal{S}_\alpha(s,h,l)$ is finite and let us consider the well-defined atom $\bigwedge \mathcal{S}_\alpha(s,h,l)$. It is obvious to check the equivalence

$$(s',h') \models_{l'} \bigwedge \mathcal{S}_\alpha(s,h,l) \quad \text{iff} \quad (s,h,l) \simeq_\alpha (s',h',l').$$

The disjunction

$$\mathcal{T}_{\mathcal{A}} \stackrel{\text{def}}{=} \bigvee \{ \textstyle\bigwedge \mathcal{S}_\alpha(s,h,l) \mid (s,h) \models_l \mathcal{A} \}$$

is a Boolean combination of test formulæ in $\mathsf{Test}^{\mathrm{u}}_{\alpha}$ because $\bigwedge \mathcal{S}_\alpha(s,h,l)$ ranges over the finite set of atoms built from $\mathsf{Test}^{\mathrm{u}}_{\alpha}$. As usual, the empty disjunction is understood as $\bot$. By Theorem 4.3, we deduce that $\mathcal{A}$ is logically equivalent to $\mathcal{T}_{\mathcal{A}}$ which finishes the proof. □

The proof of Theorem 4.11 can lead to an algorithmic way to eliminate quantifiers thanks to the decidability of model-checking (Theorem 4.9). Indeed, by Corollary 4.7, for every pointed memory state (s,h,l), there is a pointed memory state (s',h',l') such that (s,h,l) and (s',h',l') are α-equivalent and $\mathrm{maxval}(s',h',l') \leqslant 16q\alpha$. So, in the construction of $\mathcal{T}_{\mathcal{A}}$, we can restrict ourselves to pointed memory states whose maximal value is bounded by $16q\alpha$. However, the procedure described in the proof is not really suited for an effective elimination of quantifiers. Theorem 4.11 provides a characterisation of the expressive power of 1SL1, which is now easy to differenciate from 1SL2. No big deal here since undecidability of 1SL2 is established in (Demri and Deters 2014) (but strictly speaking, this does not entail that 1SL2 is strictly more expressive than 1SL1 in case there would be a non-computable translation from 1SL2 into 1SL1).

Corollary 4.12 *1SL2 is strictly more expressive than 1SL1.*

The proof can be found in Appendix C starting at page 67.

4.3 Discussion

The Boolean combination equivalent to any formula in 1SL1 is made of test formulæ that depend on the memory threshold of the formula and on the set of program variables occurring in it, which seems fair enough.

When $\mathcal{A}$ in 1SL1 has no free occurrence of $\mathtt{u}$, one can show that $\mathcal{A}$ is equivalent to a Boolean combination of formulæ in $\mathsf{Test}_{\mathsf{th}(q,\mathcal{A})}$. Similarly, when $\mathcal{A}$ in 1SL1 has no occurrence of $\mathtt{u}$ at all, $\mathcal{A}$ is equivalent to a Boolean combination of formulæ of the form $\mathtt{x}_i = \mathtt{x}_j$, $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$, $\mathtt{alloc}(\mathtt{x}_i)$ and $\#\,\mathtt{rem}_{\heartsuit} \geqslant k$ with the *alternative* definition $\heartsuit(s,h) = s(\mathcal{V}) \cap \mathrm{dom}(h)$; see also (Lozes 2004a,b; Brochenin et al 2009).

Theorem 4.11 witnesses that the test formulæ we introduced properly abstract memory states when 1SL1 formulæ are involved. Test formulæ from Definition 3.1 were not given to us and we had to design such formulæ to conclude Theorem 4.11. All the test formulæ can be expressed in 1SL1, see developments in Section 2.4 and Lemma 2.14.

Last but not least, we need to prove that the set of test formulæ is expressively complete to get Theorem 4.11. Lemmas 3.33, 3.34 and 3.35 are helpful to obtain the Correctness Theorem 4.3, taking care of the different first- or second-order quantifiers. It is in their proofs that the completeness of the set $\mathsf{Test}^{\mathtt{u}}_{\alpha}$ is best illustrated. Nevertheless, to apply these lemmas in the proof of Theorem 4.3, we designed the adequate definition for the function $\mathsf{th}(\cdot,\cdot)$ and we arranged different thresholds in their statements. Then, there is a real interplay between the definition of $\mathsf{th}(\cdot,\cdot)$ and how Lemmas 3.33, 3.34 and 3.35 are used in the proof of Theorem 4.3.

4.4 A PSPACE Upper Bound for Model Checking and Satisfiability

In this section, we assume that the set of program variables $\mathtt{PVAR}$ is countably infinite and equal to $\{\mathtt{x}_1, \mathtt{x}_2, \ldots\}$. Any given formula $\mathcal{A}$ of 1SL1 contains at most $|\mathcal{A}|+1$ different program variables. Without any loss of generality, we can assume that $\mathcal{A}$ uses only program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$ with $1 \leqslant q \leqslant |\mathcal{A}|+1$. We say that the program variables of $\mathcal{A}$ can be chosen small. If not, large program variables can always be replaced by small ones: indeed, thanks to Proposition 2.3, a renaming can be performed in polynomial time, possibly reported to the store too, and the satisfaction and satisfiability status remain unchanged by renaming operations.

Contrary to previous sections where we insisted that q and $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$ where fixed, in this section $q \geqslant 1$ is not fixed. Instead, it is a parameter that bounds the largest program variable that can be used inside 1SL1 formulæ.

Proposition 4.13 *We consider the function φ_q that maps formulæ of 1SL1 to $\mathbb{N}$ defined by $\varphi_q(\mathcal{A}) \stackrel{\text{def}}{=} 15q^2|\mathcal{A}|^2$. The following relations hold for any $\mathcal{A}, \mathcal{B}$:*

1. $\varphi_q(\neg\mathcal{A}) \geqslant \varphi_q(\mathcal{A})$;
2. $\varphi_q(\exists \mathtt{u}\,\mathcal{A}) \geqslant 1 + \varphi_q(\mathcal{A})$;
3. $\varphi_q(\mathcal{A} \wedge \mathcal{B}) \geqslant \max(\varphi_q(\mathcal{A}), \varphi_q(\mathcal{B}))$;
4. $\varphi_q(\mathcal{A} * \mathcal{B}) \geqslant \max(\varphi_q(\mathcal{A}), \varphi_q(\mathcal{B}))$;
5. $\varphi_q(\mathcal{A} \mathrel{-\!\!*} \mathcal{B}) \geqslant 15|\mathcal{A} \mathrel{-\!\!*} \mathcal{B}|q^2 + \max(\varphi_q(\mathcal{A}), \varphi_q(\mathcal{B}))$.

1: **if** $\mathcal{B}$ is $\mathtt{emp}$ **then** return $\mathtt{tt}$ iff $\mathrm{dom}(h)$ is empty;
2: **if** $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{x}_j$ **then** return $\mathtt{tt}$ iff $s(\mathtt{x}_i)$ and $s(\mathtt{x}_j)$ are equal;
3: **if** $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$ or $\mathtt{u} = \mathtt{x}_i$ **then** return $\mathtt{tt}$ iff $s(\mathtt{x}_i)$ and l are equal;
4: **if** $\mathcal{B}$ is $\mathtt{u} = \mathtt{u}$ **then** return $\mathtt{tt}$;
5: **if** $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$ **then** return $\mathtt{tt}$ iff $s(\mathtt{x}_i) \in \mathrm{dom}(h)$, and $h(s(\mathtt{x}_i))$ and $s(\mathtt{x}_j)$ are equal;
6: **if** $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$ **then** return $\mathtt{tt}$ iff $s(\mathtt{x}_i) \in \mathrm{dom}(h)$, and $h(s(\mathtt{x}_i))$ and l are equal;
7: **if** $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_j$ **then** return $\mathtt{tt}$ iff $l \in \mathrm{dom}(h)$, and $h(l)$ and $s(\mathtt{x}_j)$ are equal;
8: **if** $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$ **then** return $\mathtt{tt}$ iff $l \in \mathrm{dom}(h)$, and $h(l)$ and l are equal.

Fig. 4.1 Function $\mathtt{amc}\big(\mathcal{B}, (s,h,l)\big) \in \{\mathtt{ff}, \mathtt{tt}\}$ for any atomic formula $\mathcal{B}$ of 1SL1.

1: **if** $\mathcal{A}$ is atomic **then** return $\mathtt{amc}\big(\mathcal{A}, (s,h,l)\big)$;
2: **if** $\mathcal{A}$ is $\neg\mathcal{A}_1$ **then** return $\mathtt{not}\,\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s,h,l)\big)$;
3: **if** $\mathcal{A}$ is $\mathcal{A}_1 \wedge \mathcal{A}_2$ **then** return $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s,h,l)\big)$ $\mathtt{and}$ $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s,h,l)\big)$;
4: **if** $\mathcal{A}$ is $\exists \mathtt{u}\, \mathcal{A}_1$ **then** return

$$\mathtt{fin\_exst}\,\big\{\, l_0 \in [0,m] \;\big|\; l_0 + \varphi_q(\mathcal{A}_1) \leqslant m \;\mathtt{and}\; \mathtt{bmc}\big(q, m, \mathcal{A}_1, (s,h,l_0)\big) \,\big\}$$

5: **if** $\mathcal{A}$ is $\mathcal{A}_1 * \mathcal{A}_2$ **then** return

$$\mathtt{fin\_exst}\left\{ h_1 : [0,m] \rightharpoonup [0,m] \;\middle|\; \begin{array}{l} \mathtt{maxval}(h_1) + \mathtt{max}(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m \\ \mathtt{and}\; \mathtt{subheap}(h_1, h) \\ \mathtt{and}\; \mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) \\ \mathtt{and}\; \mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h - h_1, l)\big) \end{array} \right\}$$

6: **if** $\mathcal{A}$ is $\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ **then** return

$$\mathtt{fin\_fall}\left\{ h_1 : [0,m] \rightharpoonup [0,m] \;\middle|\; \begin{array}{l} \mathtt{not}\,(\mathtt{maxval}(h_1) + \mathtt{max}(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m) \\ \mathtt{or}\; \mathtt{not}\,\mathtt{ortho}(h_1, h) \\ \mathtt{or}\; \mathtt{not}\,\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) \\ \mathtt{or}\; \mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h \uplus h_1, l)\big) \end{array} \right\}$$

where $\mathtt{not}$, $\mathtt{or}$, $\mathtt{and}$ are *Boolean* operators; $\mathtt{fin\_exst}$, $\mathtt{fin\_fall}$ are *finite quantification* operators; $\mathtt{subheap}$, $\mathtt{ortho}$, $\uplus$ and $-$ are *heap related* functions; $\mathtt{maxval}$, $\mathtt{max}$, $+$ and $\leqslant$ are *natural number* related functions; and φ_q is implemented as defined in Proposition 4.13

Fig. 4.2 Function $\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big) \in \{\mathtt{ff}, \mathtt{tt}\}$ for any formula $\mathcal{A}$ of 1SL1.

The proof is left to the reader. The exact definition of $\varphi_q(\mathcal{A})$ is a bit arbitrary and other definitions are possible. What really matters is the satisfaction of the relations 1–5 of Proposition 4.13 because they are used in the correctness proof for the forthcoming algorithm $\mathtt{bmc}$. However, $\varphi_q(\mathcal{A})$ should also be designed so that it is smoothly computable, i.e. in our case, with a low space-complexity bound in $\mathcal{O}(\log q + \log|\mathcal{A}|)$. The function φ_q depends on q, as stressed by the notation.

In Figure 4.2, we describe the recursive function $\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big)$ with the following inputs:

- $q \geqslant 1$ and $m \in \mathbb{N}$ are two natural numbers;
- $\mathcal{A}$ is a 1SL1 formula with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$;
- (s,h,l) is a pointed memory state.

Such an input is called *legitimate* for $\mathtt{bmc}$. The function $\mathtt{bmc}$ is defined by structural induction on the formula $\mathcal{A}$ and (eventually) returns a Boolean value in

$\{\mathtt{ff}, \mathtt{tt}\}$. There is a special treatment for atomic formulæ $\mathcal{B}$ of 1SL1 described by the (non-recursive) function $\mathtt{amc}\big(\mathcal{B}, (s,h,l)\big) \in \{\mathtt{ff}, \mathtt{tt}\}$; see Figure 4.1.

On a legitimate input, the functions `amc` and thus `bmc` only use the values $s(\mathtt{x}_1), \ldots, s(\mathtt{x}_q)$ of the store s. The other values $s(\mathtt{x}_i)$ for $i > q$ of the store s have no influence on the result. Hence we may as well describe s as a finite data structure $s : \{\mathtt{x}_1, \ldots, \mathtt{x}_q\} \to \mathbb{N}$.

Proposition 4.14 *On a legitimate input,* $\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big)$ *always terminates and returns either* `ff` *or* `tt`*. If the relation* $\mathrm{maxval}(s,h,l) \leqslant m$ *holds then the call* $\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big)$ *runs in space* $\mathcal{O}\big(q \log m + |\mathcal{A}| m \log m\big)$.

Proof We remind the reader that the definition of

$$\mathrm{maxval}(s,h,l) = \max\big(\{s(\mathtt{x}_1), \ldots s(\mathtt{x}_q)\} \cup \mathrm{dom}(h) \cup \mathrm{ran}(h) \cup \{l\}\big)$$

depends on q (for the part concerning the store s). The parameters q, m and s remain unchanged during a run of `bmc` and can be stored in space $\mathcal{O}(q \log m)$ (because $\mathrm{maxval}(s,h,l) \leqslant m$ hence $s(\mathtt{x}_1), \ldots, s(\mathtt{x}_q) \leqslant m$). More importantly, running $\mathtt{amc}\big(\mathcal{A}, (s,h,l)\big)$ requires then only space in $\mathcal{O}(q \log m)$, assuming a reasonably succinct encoding of (s,h,l).

The call $\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big)$ terminates because it is defined by structural induction on $\mathcal{A}$. The depth of recursion is thus bounded by the depth of the syntactic tree of $\mathcal{A}$, which is itself bounded by $|\mathcal{A}|$. Notice that the recursive sub-calls generated by a legitimate call are also legitimate calls.

The heap parameter h or the location l are modified during recursive sub-calls but the invariant $\mathrm{maxval}(s,h,l) \leqslant m$ still holds as it can be checked in Figure 4.2. Each recursive sub-call involves storing new data in the stack:

- when $\mathtt{bmc}\big(q, m, \exists \mathtt{u}\, \mathcal{A}, (s,h,l)\big)$ makes the recursive call, the new data is l_0 (size bounded by $\mathcal{O}(\log m)$);
- when $\mathtt{bmc}\big(q, m, \mathcal{A} * \mathcal{B}, (s,h,l)\big)$ makes recursive calls, the new data is either h_1 or $h_2 = h - h_1$ (both of size bounded by $\mathcal{O}(m \log m)$);
- when $\mathtt{bmc}\big(q, m, \mathcal{A} \mathrel{-\!\!*} \mathcal{B}, (s,h,l)\big)$ makes recursive sub-calls, the new data is either h_1 or $h \uplus h_1$ (both of size bounded by $\mathcal{O}(m \log m)$). □

We now prove the following correctness lemma for $\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big)$: on a legitimate input, it returns the correct value of the predicate $(s,h) \models_l \mathcal{A}$ provided m is chosen large enough.

Lemma 4.15 *Let* $q \geqslant 1$ *and* $m \in \mathbb{N}$*. Let* $\mathcal{A}$ *be a 1SL1 formula with program variables in* $\mathtt{x}_1, \ldots, \mathtt{x}_q$ *and* (s,h,l) *be a pointed memory state. If we assume* $\mathrm{maxval}(s,h,l) + \varphi_q(\mathcal{A}) \leqslant m$ *then*

$$\mathtt{bmc}\big(q, m, \mathcal{A}, (s,h,l)\big) = \mathtt{tt} \quad \textit{iff} \quad (s,h) \models_l \mathcal{A}.$$

The proof can be found in Appendix C starting at page 67.

Theorem 4.16 *Let $q \geqslant 1$. Given a 1SL1 formula $\mathcal{A}$ with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$, a pointed memory state (s, h, l) and $m \in \mathbb{N}$ such that $\max\big(q, \mathrm{maxval}(s, h, l), |\mathcal{A}|\big) \leqslant m$, we can decide the predicate $(s, h) \models_l \mathcal{A}$ in space $\mathcal{O}(m^5 \log m)$.*

Proof Let $m' = 16m^4$. We have both $\mathrm{maxval}(s, h, l) \leqslant m \leqslant m'$ and

$$\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) = \mathrm{maxval}(s, h, l) + 15q^2|\mathcal{A}|^2 \leqslant m + 15m^4 \leqslant m'$$

By Proposition 4.14, the computation of $\mathtt{bmc}\big(q, m', \mathcal{A}, (s, h, l)\big)$ terminates and runs in space $\mathcal{O}(q \log m' + |\mathcal{A}|m' \log m')$. By Lemma 4.15, it returns the correct value of the predicate $(s, h) \models_l \mathcal{A}$. From $q \leqslant m$ and $|\mathcal{A}| \leqslant m$, we deduce the space upper bound $\mathcal{O}(m^5 \log m)$. □

Theorem 4.17 *The satisfiability status of a 1SL1 formula $\mathcal{A}$ can be solved in space $\mathcal{O}\big(|\mathcal{A}|^{15} \log |\mathcal{A}|\big)$.*

Proof The first step is to collect the program variables of $\mathcal{A}$. Since each atomic formula contains at most two different program variables, $\mathcal{A}$ contains at most $|\mathcal{A}| + 1$ different program variables. Let $q = |\mathcal{A}| + 1$. Using Proposition 2.3, we rename the program variables of $\mathcal{A}$ so that $\mathcal{A}$ uses only program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$. The satisfiability status is unchanged by this renaming operation. Computing q and renaming the program variables of $\mathcal{A}$ can be done in deterministic time $\mathcal{O}(|\mathcal{A}| \log |\mathcal{A}|)$ with a sorting algorithm. Hence it can also be done in space $\mathcal{O}(|\mathcal{A}| \log |\mathcal{A}|)$.

Corollary 4.8 states that if $\mathcal{A}$ is satisfiable, then there is a pointed memory state (s, h, l) such that $(s, h) \models_l \mathcal{A}$ and $\mathrm{maxval}(s, h, l) \leqslant m$ with $m = 16q.\mathtt{th}(q, \mathcal{A})$. From $q = |\mathcal{A}|+1$ and $\mathtt{th}(q, \mathcal{A}) \leqslant q|\mathcal{A}|$, we deduce $m \leqslant 16(|\mathcal{A}|+1)^3$.

To check for the satisfiability of $\mathcal{A}$, it is thus sufficient to test the predicate $(s, h) \models_l \mathcal{A}$ for all pointed memory states (s, h, l) such that $\mathrm{maxval}(s, h, l) \leqslant m$. For such a pointed memory state we have

$$\max\big(q, \mathrm{maxval}(s, h, l), |\mathcal{A}|\big) \leqslant \max\big(q, m, |\mathcal{A}|\big) \leqslant 16\big(|\mathcal{A}| + 1\big)^3.$$

Hence by Theorem 4.16, checking for satisfiability of $\mathcal{A}$ can be done in space $\mathcal{O}\big(|\mathcal{A}|^{15} \log |\mathcal{A}|\big)$. □

Before we prove the PSPACE upper bound for the model-checking problem, we have to describe the composition of the input of this problem (and most importantly, how its size is measured) because, not only it does contain a formula but also a model. Hence the input of the model-checking problem is of the form $\big((s, h, l), \mathcal{A}\big)$ where (s, h, l) is a pointed memory state and $\mathcal{A}$ is a 1SL1 formula; and the problem is to determine whether $(s, h) \models_l \mathcal{A}$ holds. More precisely, because the input cannot be of infinite size, s is a partial map from $\mathtt{PVAR}$ to $\mathbb{N}$ with finite domain so that every program variable in $\mathcal{A}$ belongs to the domain of s. Let $\mathcal{V}$ be the finite set of program variables that occur in $\mathcal{A}$ and let q be the cardinal of $\mathcal{V}$. The store s must be defined on at least the program variables of $\mathcal{V}$; and the values of $s(\mathtt{x})$ for $\mathtt{x}$ outside of $\mathcal{V}$ have

no influence on the predicate $(s,h) \models_l \mathcal{A}$ (see Proposition 2.2). Let n be the cardinality of $\mathrm{dom}(h)$. The heap h is defined on exactly n locations. Let m be the maximal value of the numerical inputs, i.e. $s(\mathcal{V}) \cup \mathrm{dom}(h) \cup \mathrm{ran}(h) \cup \{l\}$. So, the size of (s,h,l) is polynomial in $(q+2n+1)\log m$ for any reasonably succinct encoding (natural numbers are encoded with a binary representation) and $(q+2n+1)$ is linearly bounded by the size of (s,h,l).

Theorem 4.18 *Model-checking for 1SL1 can be solved in polynomial space.*

Proof First, we assume that $\mathcal{A}$ uses small program variables, i.e. we assume $\mathcal{V} = \{\mathtt{x}_1, \ldots, \mathtt{x}_q\}$ where $1 \leqslant q \leqslant |\mathcal{A}|+1$. This is always possible upto a renaming of the program variables that occur in $\mathcal{A}$. That renaming needs to be reported in s as well.

Then, from the input $\big((s,h,l),\mathcal{A}\big)$, we compute in polynomial space a pointed memory state (s',h',l') of small size which is equivalent to (s,h,l): we construct (s',h',l') such that $\mathrm{maxval}(s',h',l') \leqslant q+2n+1$ and $((s,h) \models_l \mathcal{A}$ iff $(s',h') \models_{l'} \mathcal{A})$. Let us define the set $D = s(\mathcal{V}) \cup \mathrm{dom}(h) \cup \mathrm{ran}(h) \cup \{l\}$. We know that $d = \mathrm{card}(D) \leqslant q+2n+1$. In deterministic polynomial time (and thus also polynomial space), we compute a bijection $\varphi : D \to [0, d-1]$, for instance using a sorting algorithm. Then we compute $s' = \varphi \circ s$, $h' = \varphi \circ h \circ \varphi^{-1}$ and $l' = \varphi(l)$ in polynomial time. We obviously have $\mathrm{maxval}(s',h',l') \leqslant q+2n+1$ and by Proposition 2.3, we have the equivalence $(s,h) \models_l \mathcal{A}$ iff $(s',h') \models_{l'} \mathcal{A}$.

We have $\max\big(q, \mathrm{maxval}(s',h',l'), |\mathcal{A}|\big) \leqslant (q+2n+1) + |\mathcal{A}| \stackrel{\text{def}}{=} k$, hence by Theorem 4.16, we can check the predicate $(s',h') \models_{l'} \mathcal{A}$ in space $\mathcal{O}(k^5 \log k)$ which is thus polynomial in the size of the input. □

Theorem 4.19 *The model-checking and satisfiability problems for 1SL1 are* PSPACE*-complete.*

Proof PSPACE-hardness for both problems is a consequence of (Calcagno et al 2001) since the problems for propositional separation logic SL0 are already PSPACE-hard. Note that the PSPACE-hardness proof can be adapted with a single record field and that *nil* can be simulated by a dedicated program variable, see (Calcagno et al 2001). The PSPACE upper bound for model-checking (resp. satisfiability) is stated in Theorem 4.18 (resp. Theorem 4.17). □

Corollary 4.20 *Let $q \geqslant 1$. Let $\mathcal{A}$ be a 1SL1 formula with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$ and let $\alpha = \mathsf{th}(q,\mathcal{A})$. Computing a Boolean combination of test formulæ in $\mathsf{Test}^{\mathtt{u}}_{\alpha}$ logically equivalent to $\mathcal{A}$ can be done in polynomial space (even though the outcome formula can be of exponential size).*

Proof Let $\mathcal{A}$ be a 1SL1 formula with $\alpha = \mathsf{th}(q,\mathcal{A})$. By the proof of Theorem 4.11 and Corollary 4.7, $\mathcal{A}$ is logically equivalent to the formula below

$$\bigvee \big\{ \bigwedge \mathcal{S}_\alpha(s,h,l) \;\big|\; (s,h) \models_l \mathcal{A} \text{ and } \mathrm{maxval}(s,h,l) \leqslant 16q\alpha \big\}.$$

The non-isomorphic copies of pointed memory states (s,h,l) such that

$$\mathrm{maxval}(s,h,l) \leqslant 16q\alpha$$

$$\frac{}{\mathtt{x}=\mathtt{x}} \qquad \frac{\mathtt{x}=\mathtt{y}}{\mathtt{y}=\mathtt{x}} \qquad \frac{\mathtt{x}=\mathtt{y} \quad \mathtt{y}=\mathtt{z}}{\mathtt{x}=\mathtt{z}} \qquad \frac{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)}{\mathtt{conv}(\mathtt{x}_j,\mathtt{x}_i)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j) \quad \mathtt{conv}(\mathtt{x}_j,\mathtt{x}_k)}{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_k)}$$

$$\frac{\mathtt{x}=\mathtt{y} \quad \mathtt{x}\hookrightarrow\mathtt{z}}{\mathtt{y}\hookrightarrow\mathtt{z}} \qquad \frac{\mathtt{x}=\mathtt{y} \quad \mathtt{z}\hookrightarrow\mathtt{x}}{\mathtt{z}\hookrightarrow\mathtt{y}} \qquad \frac{\mathtt{x}_i=\mathtt{x}_j \quad \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_k)}{\mathtt{conv}(\mathtt{x}_j,\mathtt{x}_k)}$$

$$\frac{\mathtt{x}_i=\mathtt{x}_j \quad \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_k)}{\mathtt{btwn}(\mathtt{x}_j,\mathtt{x}_k)} \qquad \frac{\mathtt{x}_i=\mathtt{x}_j \quad \mathtt{btwn}(\mathtt{x}_k,\mathtt{x}_i)}{\mathtt{btwn}(\mathtt{x}_k,\mathtt{x}_j)} \qquad \frac{\mathtt{u}=\mathtt{x}_i \quad \mathtt{alloc}(\mathtt{u})}{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)}$$

$$\frac{\mathtt{x}_i\hookrightarrow\mathtt{z} \quad \mathtt{x}_j\hookrightarrow\mathtt{z}}{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)} \qquad \frac{\mathtt{x}\hookrightarrow\mathtt{y} \quad \mathtt{x}\hookrightarrow\mathtt{z}}{\mathtt{y}=\mathtt{z}} \qquad \frac{\mathtt{x}_i\hookrightarrow\mathtt{y} \quad \mathtt{y}\hookrightarrow\mathtt{x}_j}{\mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)} \qquad \frac{\mathtt{x}_i\hookrightarrow\mathtt{y} \quad \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)}{\mathtt{y}\hookrightarrow\mathtt{x}_j}$$

$$\frac{\mathtt{x}_i\hookrightarrow\mathtt{x}_j \quad \mathtt{conv}(\mathtt{x}_j,\mathtt{x}_j)}{\mathtt{toalloc}(\mathtt{x}_i)} \qquad \frac{\mathtt{x}_i\hookrightarrow\mathtt{u} \quad \mathtt{alloc}(\mathtt{u})}{\mathtt{toalloc}(\mathtt{x}_i)} \qquad \frac{\mathtt{x}_i\hookrightarrow\mathtt{x}_j \quad \mathtt{toalloc}(\mathtt{x}_i)}{\mathtt{conv}(\mathtt{x}_j,\mathtt{x}_j)} \qquad \frac{\mathtt{x}_i\hookrightarrow\mathtt{y} \quad \mathtt{y}\hookrightarrow\mathtt{y}}{\mathtt{toloop}(\mathtt{x}_i)}$$

$$\frac{\mathtt{x}_i\hookrightarrow\mathtt{y} \quad \mathtt{toloop}(\mathtt{x}_i)}{\mathtt{y}\hookrightarrow\mathtt{y}} \qquad \frac{\mathtt{u}\hookrightarrow\mathtt{x}}{\mathtt{alloc}(\mathtt{u})} \qquad \frac{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j) \quad \mathtt{x}_i\hookrightarrow\mathtt{z}}{\mathtt{x}_j\hookrightarrow\mathtt{z}} \qquad \frac{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j) \quad \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_k)}{\mathtt{btwn}(\mathtt{x}_j,\mathtt{x}_k)}$$

$$\frac{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j) \quad \mathtt{toloop}(\mathtt{x}_i)}{\mathtt{toloop}(\mathtt{x}_j)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j) \quad \mathtt{toalloc}(\mathtt{x}_i)}{\mathtt{toalloc}(\mathtt{x}_j)} \qquad \frac{\mathtt{x}_i=\mathtt{u} \quad \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)}{\mathtt{alloc}(\mathtt{u})}$$

$$\frac{\mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j) \quad \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_k)}{\mathtt{x}_j=\mathtt{x}_k} \qquad \frac{\mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)}{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)} \qquad \frac{\mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)}{\mathtt{toalloc}(\mathtt{x}_i)} \qquad \frac{\mathtt{toloop}(\mathtt{x}_i) \quad \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)}{\mathtt{x}_i\hookrightarrow\mathtt{x}_j}$$

$$\frac{\mathtt{toloop}(\mathtt{x}_i)}{\mathtt{toalloc}(\mathtt{x}_i)} \qquad \frac{\mathtt{toalloc}(\mathtt{x}_i)}{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)} \qquad \frac{\mathtt{toalloc}(\mathtt{x}_i) \quad \mathtt{x}_i\hookrightarrow\mathtt{u}}{\mathtt{alloc}(\mathtt{u})}$$

Fig. 5.1 Saturation rules for basic formulæ with $\mathtt{x},\mathtt{y},\mathtt{z} \in \mathtt{PVAR} \cup \{\mathtt{u}\}$ and $\mathtt{x}_i,\mathtt{x}_j,\mathtt{x}_k \in \mathtt{PVAR}$.

can be enumerated in polynomial space. Moreover, the model-checking problem for 1SL1 can be solved in polynomial space, whence the above formula can be built in polynomial space (even though its size may be exponential in the size of $\mathcal{A}$ due to a possible exponential amount of disjuncts). Note also that $S_\alpha(s,h,l)$ can be computed in polynomial space too. □

5 Complexity of Satisfiability of Test Formulæ

The goal of this section is to complete the proof of Theorem 3.5. We decide the conjunctions of literals in deterministic polynomial-time using a saturation algorithm based on a set of deduction rules for basic test formulæ. Then, we describe the NP procedure for Boolean combinations of test formulæ.

Remember that $\mathsf{Basic}^{\mathtt{u}}$ (resp. $\mathsf{Test}^{\mathtt{u}}_{\alpha}$) denotes the set of basic formulæ (resp. test formulæ) built from the program variables $\mathtt{x}_1,\ldots,\mathtt{x}_q$.

Proposition 5.1 (Soundness of the saturation rules) *Let* P *be a subset of* $\mathsf{Basic}^{\mathtt{u}}$ *and let* (s,h,l) *be a pointed memory state such that* $(s,h) \models_l \mathcal{B}$ *holds for any* $\mathcal{B} \in \mathsf{P}$. *If* $\mathcal{C} \in \mathsf{Basic}^{\mathtt{u}}$ *is derivable from the formulæ of* P *using the rules of Figure 5.1 then* $(s,h) \models_l \mathcal{C}$ *holds.*

The proof is by induction on the derivation height and it is left to the reader.

If a set of basic formulæ is satisfied in a model, then all the basic logical consequences of those formulæ are also satisfied in that model. Hence, in gen-

eral, it is not possible to satisfy exactly a given set of basic formulæ. But if a set P of basic formulæ is closed under the rules of Figure 5.1, then it is possible to satisfy exactly the formulæ of P in the canonical pre-model defined below.

Definition 5.2 (Canonical pre-model) Let $q \geqslant 1$. The *canonical pre-model* of a finite set P of formulæ of $\mathsf{Basic}^{\mathtt{u}}$ is built the following way: we define two partial functions $s : \mathcal{V} \rightharpoonup [1,q]$ and $\mathfrak{h} : [1,q] \rightharpoonup [q+1, 2q]$, a finite graph $H \subseteq [0,2q] \times [0,2q+1]$ and a finite subset $L \subseteq [0,2q]$ by:

- $s(\mathtt{x}_i) \stackrel{\text{def}}{=} \min\{j \mid \mathtt{x}_i = \mathtt{x}_j \in \mathsf{P}\}$ and $\mathfrak{h}_i \stackrel{\text{def}}{=} \min\{q+j \mid \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}\}$;
- $\mathsf{Fm}_i \stackrel{\text{def}}{=} \{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\}$ and $\mathsf{Bw}_i \stackrel{\text{def}}{=} \{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q)\}$;
- $\mathsf{Fm}_{\mathtt{u}} \stackrel{\text{def}}{=} \{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\}$ and $\mathsf{To}_{\mathtt{u}} \stackrel{\text{def}}{=} \{\mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\}$;
- $\mathsf{Eq}_{\mathtt{u}} \stackrel{\text{def}}{=} \{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}\}$;
- for $\mathsf{B} \subseteq \mathsf{Basic}^{\mathtt{u}}$ the notation $\mathsf{B} \perp \mathsf{P}$ is a shortcut for $\mathsf{B} \cap \mathsf{P} = \varnothing$;
- H is defined as the least set (w.r.t. set inclusion) such that:
  H_1. $(s(\mathtt{x}_i), s(\mathtt{x}_j)) \in H$ if $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$;
  H_2. $(s(\mathtt{x}_i), \mathfrak{h}_j) \in H$ if $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ and $\mathsf{Fm}_i \perp \mathsf{P}$;
  H_3. $(\mathfrak{h}_i, s(\mathtt{x}_j)) \in H$ if $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ and $\mathsf{Fm}_i \perp \mathsf{P}$;
  H_4. $(\mathfrak{h}_i, \mathfrak{h}_j) \in H$ if $\{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j), \mathtt{toloop}(\mathtt{x}_i)\} \subseteq \mathsf{P}$ and $\mathsf{Fm}_i \perp \mathsf{P}$;
  H_5. $(\mathfrak{h}_i, 0) \in H$ if $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$ and $(\mathsf{Fm}_i \cup \mathsf{Bw}_i \cup \{\mathtt{toloop}(\mathtt{x}_i)\}) \perp \mathsf{P}$;
  H_6. $(0, s(\mathtt{x}_i)) \in H$ if $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ and $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}) \perp \mathsf{P}$;
  H_7. $(0,0) \in H$ if $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}}) \perp \mathsf{P}$;
  H_8. $(0, 2q+1) \in H$ if $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$ and $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}) \perp \mathsf{P}$;
- L is defined to be the least set such that:
  L_1. $s(\mathtt{x}_i) \in L$ if $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$;
  L_2. $\mathfrak{h}_i \in L$ if $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\mathsf{Eq}_{\mathtt{u}} \perp \mathsf{P}$;
  L_3. $0 \in L$ if $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}) \perp \mathsf{P}$.

Remark that when we write e.g. $(s(\mathtt{x}_i), \mathfrak{h}_j) \in H$, we assume that $s(\mathtt{x}_i)$ and $\mathfrak{h}_j$ are both defined, otherwise the pair is not added to H. The same remark holds for the definition of L.

Remark that Fm abbreviates "from," Bw abbreviates "between," To abbreviates "to" and Eq abbreviates "equal." We prove that the canonical pre-model, which might not even be a memory state in general, becomes a model of exactly those formulæ of P when P is saturated under logical consequence.

Proposition 5.3 (Completeness of the saturation rules) *If the (finite) subset $\mathsf{P} \subseteq \mathsf{Basic}^{\mathtt{u}}$ is closed under the rules of Figure 5.1 and (s, H, L) is the canonical pre-model of P then:*

- *s is a total function $s : \mathcal{V} \to [1,q]$, hence s is a store;*
- *H is a finite and functional graph, hence H is the graph of some heap h;*
- *L is a singleton subset of $\mathbb{N}$, i.e. $L = \{l\}$ for some location l;*
- *the inclusion $\mathrm{dom}(h) \subseteq \heartsuit(s,h) \cup \{l\}$ holds;*
- *for any formula $\mathcal{B} \in \mathsf{Basic}^{\mathtt{u}}$ we have $(s,h) \models_l \mathcal{B}$ iff $\mathcal{B} \in \mathsf{P}$.*

The proof can be found in Appendix D starting at page 68.

Hence if P is closed under the rules of Figure 5.1 then the canonical pre-model of P is a model of exactly those formulæ in P. As a consequence, any conjunction of basic test formulæ is satisfiable. Let us see what happens when we add negations of basic test formulæ and cardinality constraints like for instance $\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant k$ or $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k'$.

Let us write $\mathrm{cl}(\mathsf{P})$ to denote the closure of a (finite) set P of basic formulæ of $\mathsf{Basic}^{\mathtt{u}}$ under the rules of Figure 5.1, i.e. all the formulæ that can be deduced from those of P. We can compute $\mathrm{cl}(\mathsf{P})$ by an obvious saturation algorithm. Note that $\mathrm{cl}(\cdot)$ is a *closure operator*, i.e. it satisfies the properties below for any $\mathsf{P}, \mathsf{P}' \subseteq \mathsf{Basic}^{\mathtt{u}}$:

$$\mathsf{P} \subseteq \mathrm{cl}(\mathsf{P}) \qquad \mathsf{P} \subseteq \mathsf{P}' \Rightarrow \mathrm{cl}(\mathsf{P}) \subseteq \mathrm{cl}(\mathsf{P}') \qquad \mathrm{cl}\big(\mathrm{cl}(\mathsf{P})\big) \subseteq \mathrm{cl}(\mathsf{P}).$$

Since $\mathsf{Basic}^{\mathtt{u}}$ is stable under the rules of Figure 5.1, the cardinal of $\mathrm{cl}(\mathsf{P})$ is at most the cardinal of $\mathsf{Basic}^{\mathtt{u}}$, i.e. $4q^2 + 6q + 3$. Hence, as the rules of Figure 5.1 have a number of premises/sub-goals bounded by 2, the saturation algorithm runs in polynomial time in q.

The expression $\neg\mathsf{B}^-$ is defined as usual by $\neg\mathsf{B}^- \stackrel{\text{def}}{=} \{\neg\mathcal{B} \mid \mathcal{B} \in \mathsf{B}^-\}$. Given $\mathsf{B}^+, \mathsf{B}^- \subseteq \mathsf{Basic}^{\mathtt{u}}$, checking for the satisfiability of the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^-$ is easy. It is sufficient to compute the closure $\mathrm{cl}(\mathsf{B}^+)$:

- if $\mathsf{B}^- \cap \mathrm{cl}(\mathsf{B}^+) = \varnothing$ then the conjunction of $\mathsf{B}^+ \cup \neg\mathsf{B}^-$ is satisfied in the canonical model of $\mathrm{cl}(\mathsf{B}^+)$ by Proposition 5.3;
- otherwise $\mathsf{B}^- \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$ holds and $\mathsf{B}^+ \cup \neg\mathsf{B}^-$ is unsatisfiable. Indeed, let us consider $\mathcal{B} \in \mathsf{B}^- \cap \mathrm{cl}(\mathsf{B}^+)$. Then $\mathcal{B}$ should be both satisfied (by Proposition 5.1) and unsatisfied in any model of $\mathsf{B}^+ \cup \neg\mathsf{B}^-$, which leads to a contradiction.

More generally, for checking the satisfiability status of a conjunction of literals obtained from $\mathsf{Basic}^{\mathtt{u}} \cup \mathsf{Size}_\alpha$, there is a subtlety related to the interpretation of the quantified variable $\mathtt{u}$. Indeed, using only literals from $\mathsf{Basic}^{\mathtt{u}}$ it is possible to require that the interpretation of $\mathtt{u}$ belongs to $\mathrm{dom}(h)\backslash\heartsuit(s,h)$. For instance, any model of

$$\mathsf{B} \cup \big\{\neg(\mathtt{x}_1 = \mathtt{u}), \ldots, \neg(\mathtt{x}_q = \mathtt{u}), \neg(\mathtt{x}_1 \hookrightarrow \mathtt{u}), \ldots, \neg(\mathtt{x}_q \hookrightarrow \mathtt{u}), \mathtt{alloc}(\mathtt{u})\big\}$$

would meet such a requirement. Hence, one of the test formulæ

$$\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_1) \geqslant 1, \ldots, \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_q) \geqslant 1, \#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1, \#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant 1$$

has to be satisfied in such a model. That is why, we develop a more involved argument in order to check the satisfiability status of conjunctions of literals in $\mathsf{Test}^{\mathtt{u}}_\alpha$ in polynomial time.

Definition 5.4 A *cardinality assignment* is a tuple $(\mathsf{p}_1, \ldots, \mathsf{p}_q, \mathsf{l}, \mathsf{r})$ of $q + 2$ elements of $\mathbb{N}$.

Proposition 5.5 *Let $q \geqslant 1$. Let $s : \mathcal{V} \to \mathbb{N}$ be a store, $h : \mathbb{N} \rightharpoonup \mathbb{N}$ be a heap and $l \in \mathbb{N}$ be a location. Let $(\mathsf{p}_1, \ldots, \mathsf{p}_q, \mathsf{l}, \mathsf{r})$ be a cardinality assignment such that:*

1. *$s(\mathtt{x}_i) = s(\mathtt{x}_j)$ implies $\mathsf{p}_i = \mathsf{p}_j$ for all $i, j \in [1, q]$;*
2. *$\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h, i)) \leqslant \mathsf{p}_i$ for any $i \in [1, q]$;*
3. *$\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h)) \leqslant \mathsf{l}$;*
4. *$\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h)) \leqslant \mathsf{r}$.*

There exists a heap h' such that:

- *$(s, h, l) \simeq_b (s, h', l)$;*
- *$\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', i)) = \mathsf{p}_i$ for any $i \in [1, q]$;*
- *$\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h')) = \mathsf{l}$;*
- *$\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h')) = \mathsf{r}$.*

The proof can be found in Appendix D starting at page 78.

It is always possible to add elements outside the core of a heap as much as we wish while preserving basic equivalence.

Definition 5.6 (1-, 2- and 3-consistency) Let $q, \alpha \geqslant 1$. Let $\mathsf{B}^+, \mathsf{B}^-$ be two (finite) sets of formulæ of $\mathsf{Basic}^{\mathsf{u}}$ and S be a (finite) set of literals from Size_α. We say that *the triple* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *is 1-consistent* when for all $i, j \in [1, q]$ and for all $a, b \in \mathbb{N}$ the following conditions hold:

C1.1 $\mathsf{B}^- \cap \mathrm{cl}(\mathsf{B}^+) = \varnothing$;
C1.2 if $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$ and $\{\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant a, \neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant b\} \subseteq \mathsf{S}$ then $a < b$;
C1.3 if $\{\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a, \neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant b\} \subseteq \mathsf{S}$ then $a < b$;
C1.4 if $\{\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant a, \neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant b\} \subseteq \mathsf{S}$ then $a < b$.

We say that *the triple* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *is 2-consistent* when it is 1-consistent and for all $i, j \in [1, q]$ the following conditions hold:

C2.1 if $\{\mathtt{x}_i = \mathtt{x}_j, \mathtt{u} \hookrightarrow \mathtt{x}_i\} \subseteq \mathrm{cl}(\mathsf{B}^+)$ and $\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant 1 \in \mathsf{S}$ then $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 1-consistent for some $\mathcal{B} \in \mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}$;
C2.2 if $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathrm{cl}(\mathsf{B}^+)$ and $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1 \in \mathsf{S}$ then $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 1-consistent for some $\mathcal{B} \in \mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}$.

We say that *the triple* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *is 3-consistent* when it is 2-consistent and the following conditions hold:

C3.1 if $\mathtt{alloc}(\mathtt{u}) \in \mathrm{cl}(\mathsf{B}^+)$ and $\neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant 1 \in \mathsf{S}$ then $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 2-consistent for some $\mathcal{B} \in \mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}$.

where $\mathsf{Eq}_{\mathtt{u}}$, $\mathsf{To}_{\mathtt{u}}$ and $\mathsf{Fm}_{\mathtt{u}}$ are from Definition 5.2.

Proposition 5.7 *If the conjunction of the formulæ in $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable then the triple $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is n-consistent for any $n \in \{1, 2, 3\}$.*

The proof can be found in Appendix D starting at page 79.

Proposition 5.8 *If the triple* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *is 3-consistent then the conjunction of the formulæ in* $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ *is satisfiable.*

The proof can be found in Appendix D starting at page 79.

Proposition 5.9 *For any* $n \in \{1, 2, 3\}$*, the* n*-consistency of* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *can be checked in polynomial time in the size of* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$*.*

The proof is left to the reader.

Let us conclude this section by explaining why the satisfiability problem for Boolean combinations of test formulæ (see Definition 3.4) can be solved in NP, establishing the upper bound that was postponed from the proof of Theorem 3.5. Let $\mathcal{A}$ be a Boolean combination of test formulæ over $\mathtt{x}_1, \ldots, \mathtt{x}_q$. An NP procedure for checking the satisfiability of $\mathcal{A}$ goes as follows:

1. non-deterministically guess a conjunction of literals contained in $\mathcal{A}$ that makes true $\mathcal{A}$ propositionnally, say $\mathcal{A}_1 \wedge \cdots \wedge \mathcal{A}_n$. Then check for the satisfiability of $\mathcal{A}_1 \wedge \cdots \wedge \mathcal{A}_n$ starting at step 2. If it is satisfiable return "true." If no such conjunction is satisfiable return "false";
2. none of the $\mathcal{A}_i$ can be $\bot$; if $\mathcal{A}_i$ is $\neg\bot$ then remove it; hence all the $\mathcal{A}_i$ are literals from $\mathsf{Basic}^{\mathtt{u}} \cup \mathsf{Size}_\alpha$;
3. sort $\mathcal{A}_1, \ldots, \mathcal{A}_n$ and find $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ so that $\{\mathcal{A}_1, \ldots, \mathcal{A}_n\} = \mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$;
4. Return the 3-consistency status of $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$.

From Propositions 5.7 and 5.8, the 3-consistency of $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is equivalent to satisfiability of the conjunction of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$, hence this is equivalent to the satisfiability of $\mathcal{A}_1 \wedge \cdots \wedge \mathcal{A}_n$. Step 1 is the nondeterministic polynomial-time step, all the other steps can be performed in deterministic polynomial time; use Proposition 5.9 for step 4.

6 Conclusion

In (Brochenin et al 2012), the undecidability of first-order separation logic 1SL with a unique record field is shown. Propositional separation logic 1SL0 is also known to be PSPACE-complete (Calcagno et al 2001). In this paper, we provided an extension with a unique quantified variable (and with both separating connectives) and we show that the satisfiability problem for 1SL1 is PSPACE-complete by presenting an original and fine-tuned abstraction of memory states. We proved that in 1SL1 separating connectives can be eliminated in a controlled way as well as first-order quantification over the single variable. In that way, we show a quantifier elimination property similar to what is known with Presburger arithmetic. Last but not least, we have established that satisfiability problem for Boolean combinations of test formulæ is NP-complete thanks to a saturation algorithm to deal with conjunctions. This is reminiscent of decision procedures used in SMT solvers and it is a challenging question to take advantage of these features to decide 1SL1 with an

SMT solver. Finally, the design of decidable fragments between 1SL1 and undecidable 1SL2 that admit decision procedures by adapting our method would be worth being further investigated. Indeed, even though the extension with strictly more than one record field might preserve decidability (which remains to be formally proved), it is open whether the addition of the reachability predicate remains decidable.

References

Antonopoulos T, Gorogiannis N, Haase C, Kanovich M, Ouaknine J (2014) Foundations for decision problems in separation logic with general inductive predicates. In: FOSSACS'14, Springer, Lecture Notes in Computer Science, vol 8412, pp 411–425, to appear

Bansal K, Reynolds A, King T, Barrett C, Wies T (2015) Deciding local theory extensions via E-matching. In: CAV'15, Springer, Lecture Notes in Computer Science, vol 9207, pp 87–105

Barrett C, Conway C, Deters M, Hadarean L, Jovanovic D, King T, Reynolds A, Tinelli C (2011) CVC4. In: CAV'11, Springer, Lecture Notes in Computer Science, vol 8606, pp 171–177

Berdine J, Calcagno C, O'Hearn P (2005) Smallfoot: Modular Automatic Assertion Checking with Separation Logic. In: FMCO'05, Springer, Lecture Notes in Computer Science, vol 4111, pp 115–137

Brochenin R, Demri S, Lozes E (2009) Reasoning about sequences of memory states. Annals of Pure and Applied Logic 161(3):305–323

Brochenin R, Demri S, Lozes E (2012) On the almighty wand. Information and Computation 211:106–137

Brotherston J, Kanovich M (2010) Undecidability of Propositional Separation Logic and Its Neighbours. In: LICS'10, IEEE, pp 130–139

Brotherston J, Fuhs C, Gorogiannis N, Navarro Perez J (2014) A decision procedure for satisfiability in separation logic with inductive predicates. In: CSL-LICS'14

Calcagno C, O'Hearn P, Yang H (2001) Computability and Complexity Results for a Spatial Assertion Language for Data Structures. In: FSTTCS'01, Springer, Lecture Notes in Computer Science, vol 2245, pp 108–119

Cook B, Haase C, Ouaknine J, Parkinson M, Worrell J (2011) Tractable Reasoning in a Fragment of Separation Logic. In: CONCUR'11, Springer, Lecture Notes in Computer Science, vol 6901, pp 235–249

Dawar A, Gardner P, Ghelli G (2007) Expressiveness and complexity of graph logic. Information and Computation 205(3):263–310

Demri S, Deters M (2014) Expressive Completeness of Separation Logic with Two Variables and No Separating Conjunction. In: CSL-LICS'14, ACM Press

Demri S, Galmiche D, Larchey-Wendling D, Méry D (2014) Separation Logic with One Quantified Variable. In: CSR'14, Springer, Lecture Notes in Computer Science, vol 8476, pp 125–138

Galmiche D, Méry D (2010) Tableaux and Resource Graphs for Separation Logic. Journal of Logic and Computation 20(1):189–231

Haase C, Ishtiaq S, Ouaknine J, Parkinson M (2013) SeLoger: A Tool for Graph-Based Reasoning in Separation Logic. In: CAV'13, Springer, Lecture Notes in Computer Science, vol 8044, pp 790–795

Iosif R, Rogalewicz A, Simacek J (2013) The Tree Width of Separation Logic with Recursive Definitions. In: CADE'13, Springer, Lecture Notes in Computer Science, vol 7898, pp 21–38

Ishtiaq S, O'Hearn P (2001) BI as an Assertion Language for Mutable Data Structures. In: Hankin C, Schmidt D (eds) POPL'01, ACM, pp 14–26

Larchey-Wendling D, Galmiche D (2010) The Undecidability of Boolean BI through Phase Semantics. In: LICS'10, IEEE, pp 140–149

Lozes E (2004a) Expressivité des logiques spatiales. PhD thesis, LIP, ENS Lyon, France

Lozes E (2004b) Separation Logic preserves the expressive power of classical logic. In: 2nd Workshop on Semantics, Program Analysis, and Computing Environments for Memory Management (SPACE'04)

de Moura L, Björner N (2008) Z3: An Efficient SMT Solver. In: TACAS'08, Springer, Lecture Notes in Computer Science, vol 4963, pp 337–340

Pérez JN, Rybalchenko A (2013) Separation Logic Modulo Theories. In: APLAS'13, Lecture Notes in Computer Science, vol 8301, pp 90–106

Piskac R, Wies T, Zufferey D (2013) Automating Separation Logic using SMT. In: CAV'13, Springer, Lecture Notes in Computer Science, vol 2013, pp 773–789

Piskac R, Wies T, Zufferey D (2014) GRASShopper - complete heap verification with mixed specifications. In: TACAS'14, Springer, Lecture Notes in Computer Science, vol 8413, pp 124–139

Presburger M (1929) Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In: Comptes Rendus du premier congrès de mathématiciens des Pays Slaves, Warszawa, pp 92–101

Reynolds J (2002) Separation Logic: A Logic for Shared Mutable Data Structures. In: LICS'02, IEEE, pp 55–74

A Proofs of Section 2

Proposition 2.9 *Let s, h, h_1, h_2 be such that $h = h_1 \uplus h_2$ and let $i \in [1, q]$. The following identities hold:*

1. $\mathrm{pred}_{\overline{\heartsuit}}(s, h_1, i) = (\mathrm{pred}_{\overline{\heartsuit}}(s, h, i) \cap \mathrm{dom}(h_1)) \uplus (\mathrm{pred}(s, h, i) \cap \Delta(s, h_1, h_2))$;
2. $\mathrm{loop}_{\overline{\heartsuit}}(s, h_1) = (\mathrm{loop}_{\overline{\heartsuit}}(s, h) \cap \mathrm{dom}(h_1)) \uplus (\mathrm{loop}(s, h) \cap \Delta(s, h_1, h_2))$;
3. $\mathrm{rem}_{\overline{\heartsuit}}(s, h_1) = (\mathrm{rem}_{\overline{\heartsuit}}(s, h) \cap \mathrm{dom}(h_1)) \uplus (\mathrm{rem}(s, h) \cap \Delta(s, h_1, h_2))$.

Proof First, observe that we have the following identities:

$$\begin{array}{rcl} \mathrm{pred}(s, h_1, i) & = & \mathrm{pred}(s, h, i) \cap \mathrm{dom}(h_1) \\ \mathrm{loop}(s, h_1) & = & \mathrm{loop}(s, h) \cap \mathrm{dom}(h_1) \\ \mathrm{rem}(s, h_1) & = & \mathrm{rem}(s, h) \cap \mathrm{dom}(h_1) \\ \overline{\heartsuit(s, h_1)} & = & \overline{\heartsuit(s, h)} \cup \overline{\mathrm{dom}(h_1)} \cup \Delta(s, h_1, h_2) \end{array}$$

By definition, we have

$$\mathrm{pred}_{\overline{\heartsuit}}(s, h_1, i) = \mathrm{pred}(s, h_1, i) \backslash \heartsuit(s, h_1) = \mathrm{pred}(s, h_1, i) \cap \overline{\heartsuit(s, h_1)}.$$

Hence,

$$\mathrm{pred}_{\overline{\heartsuit}}(s, h_1, i) = (\mathrm{pred}(s, h, i) \cap \mathrm{dom}(h_1) \cap \overline{\heartsuit(s, h)}) \cup$$

$$(\mathrm{pred}(s, h, i) \cap \mathrm{dom}(h_1) \cap \overline{\mathrm{dom}(h_1)}) \cup (\mathrm{pred}(s, h, i) \cap \mathrm{dom}(h_1) \cap \Delta(s, h_1, h_2)).$$

Consequently,

$$\mathrm{pred}_{\overline{\heartsuit}}(s, h_1, i) = (\mathrm{pred}_{\overline{\heartsuit}}(s, h, i) \cap \mathrm{dom}(h_1)) \cup (\mathrm{pred}(s, h, i) \cap \Delta(s, h_1, h_2))$$

since $\Delta(s, h_1, h_2) \subseteq \mathrm{dom}(h_1)$. The other identities are established in a similar fashion. □

Proposition 2.10 *Let (s, h) be a memory state, $l_1 \in \mathbb{N} \backslash \mathrm{dom}(h)$ and $l_2 \in \mathbb{N}$. Let us write $h_{1\to 2}$ for $h \uplus [l_1 \mapsto l_2]$ and let i be in $[1, q]$. The following identities hold:*

$$\begin{array}{lll} \mathrm{dom}(h_{1\to 2}) & = \mathrm{dom}(h) & \uplus \ \{l_1\} \\ \mathrm{pred}(s, h_{1\to 2}, i) & = \mathrm{pred}(s, h, i) & \uplus \begin{cases} \{l_1\} & \text{if } l_2 = s(\mathtt{x}_i) \\ \varnothing & \text{if } l_2 \neq s(\mathtt{x}_i) \end{cases} \\ \mathrm{loop}(s, h_{1\to 2}) & = \mathrm{loop}(s, h) & \uplus \begin{cases} \{l_1\} & \text{if } l_1 = l_2 \\ \varnothing & \text{if } l_1 \neq l_2 \end{cases} \\ \mathrm{rem}(s, h_{1\to 2}) & = \mathrm{rem}(s, h) & \uplus \begin{cases} \{l_1\} & \text{if } l_2 \notin s(\mathcal{V}) \cup \{l_1\} \\ \varnothing & \text{if } l_2 \in s(\mathcal{V}) \cup \{l_1\} \end{cases} \end{array}$$

$$\heartsuit(s, h_{1\to 2}) = \heartsuit(s, h) \uplus \begin{cases} \{l_1, l_2\} & \text{if } l_1 \in s(\mathcal{V}),\ l_2 \in \mathrm{dom}(h) \text{ and } l_2 \notin \heartsuit(s, h) \\ \{l_1\} & \text{if } l_1 \in s(\mathcal{V}) \text{ and } (l_2 \notin \mathrm{dom}(h) \text{ or } l_2 \in \heartsuit(s, h)) \\ \{l_1\} & \text{if } l_1 \notin s(\mathcal{V}) \text{ and } l_1 \in h(s(\mathcal{V})) \\ \varnothing & \text{if } l_1 \notin \mathfrak{p}\heartsuit(s, h). \end{cases}$$

Proof The proof of the first four identities is left to the reader. For the identity that describes $\heartsuit(s, h_{1\to 2})$, we notice that

$$\mathrm{ref}(s, h_{1\to 2}) = \mathrm{ref}(s, h) \cup (\{l_1\} \cap s(\mathcal{V}))$$

holds. For $\mathrm{acc}(s, h_{1\to 2})$, it is a bit more complicated. We have

$$\mathrm{acc}(s, h_{1\to 2}) = \left(h(s(\mathcal{V})) \cup \begin{cases} \{l_2\} & \text{if } l_1 \in s(\mathcal{V}) \\ \varnothing & \text{if } l_1 \notin s(\mathcal{V}) \end{cases} \right) \cap \big(\mathrm{dom}(h) \cup \{l_1\}\big)$$

Hence we deduce the properties:

(P1) $\mathrm{acc}(s, h) \subseteq \mathrm{acc}(s, h_{1\to 2}) \subseteq \mathrm{acc}(s, h) \cup \{l_1, l_2\}$;
(P2) $l_1 \in \mathrm{acc}(s, h_{1\to 2}) \backslash \mathrm{acc}(s, h)$ iff $l_1 \in h(s(\mathcal{V}))$ or $l_1 = l_2 \in s(\mathcal{V})$;

(P3) $l_2 \in \mathrm{acc}(s, h_{1\to 2})\backslash\mathrm{acc}(s,h)$ iff $l_1 = l_2 \in h(s(\mathcal{V}))$ or $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{dom}(h) \cup \{l_1\}$

From $\heartsuit(s, h_{1\to 2}) = \mathrm{ref}(s, h_{1\to 2}) \cup \mathrm{acc}(s, h_{1\to 2})$, it is then easy to deduce the inclusions

$$\heartsuit(s,h) \subseteq \heartsuit(s, h_{1\to 2}) \subseteq \heartsuit(s,h) \cup \{l_1, l_2\}.$$

Then we study the statements $l_1 \in \heartsuit(s, h_{1\to 2})$ and $l_2 \in \heartsuit(s, h_{1\to 2})$ according to the four supplementary conditions on the right-hand side of the fifth identity:

- if $l_1 \in s(\mathcal{V})$, $l_2 \in \mathrm{dom}(h)$ and $l_2 \notin \heartsuit(s,h)$ then $l_1 \neq l_2$ (because $l_1 \notin \mathrm{dom}(h)$). Then $l_1, l_2 \notin \heartsuit(s,h)$. Hence the union $\heartsuit(s,h) \uplus \{l_1, l_2\}$ is indeed disjoint. From $l_1 \in s(\mathcal{V})$ we deduce $l_1 \in \mathrm{ref}(s, h_{1\to 2})$. From $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{dom}(h)$ we deduce $l_2 \in \mathrm{acc}(s, h_{1\to 2})$. We obtain $\heartsuit(s, h_{1\to 2}) = \heartsuit(s,h) \uplus \{l_1, l_2\}$;
- if $l_1 \in s(\mathcal{V})$ and ($l_2 \notin \mathrm{dom}(h)$ or $l_2 \in \heartsuit(s,h)$) then we already have $l_1 \in \mathrm{ref}(s, h_{1\to 2})$ and $l_1 \notin \heartsuit(s,h)$. Hence $l_1 \in \heartsuit(s, h_{1\to 2})\backslash\heartsuit(s,h)$.
  Let us show that if $l_1 \neq l_2$ and $l_2 \in \heartsuit(s, h_{1\to 2})$ then $l_2 \in \heartsuit(s,h)$. By contradiction, let us assume $l_1 \neq l_2$ and $l_2 \in \heartsuit(s, h_{1\to 2})\backslash\heartsuit(s,h)$. Then we have $l_2 \in \mathrm{ref}(s, h_{1\to 2}) \cup \mathrm{acc}(s, h_{1\to 2})$. Then either $l_2 \in \mathrm{ref}(s, h_{1\to 2})$ or $l_2 \in \mathrm{acc}(s, h_{1\to 2})$. In the former case, from $l_1 \neq l_2$ we deduce $l_2 \in \mathrm{ref}(s,h) \subseteq \heartsuit(s,h)$, a contradiction. In the later case, we deduce $l_2 \in \mathrm{acc}(s, h_{1\to 2})\backslash\heartsuit(s,h) \subseteq \mathrm{acc}(s, h_{1\to 2})\backslash\mathrm{acc}(s,h)$ hence, by (P3) either $l_1 = l_2 \in h(s(\mathcal{V}))$ (a contradiction) or $l_2 \in \mathrm{dom}(h) \cup \{l_1\}$. From $l_1 \neq l_2$, we get $l_2 \in \mathrm{dom}(h)$. Since we also have $l_2 \notin \heartsuit(s,h)$, we get a contradiction with $l_2 \notin \mathrm{dom}(h)$ or $l_2 \in \heartsuit(s,h)$. We deduce $\heartsuit(s, h_{1\to 2}) = \heartsuit(s,h) \uplus \{l_1\}$;
- if $l_1 \notin s(\mathcal{V})$ and $l_1 \in h(s(\mathcal{V}))$ then we have $l_1 \in \mathrm{acc}(s, h_{1\to 2})$ and $l_1 \notin \heartsuit(s,h)$. Hence $l_1 \in \heartsuit(s, h_{1\to 2})\backslash\heartsuit(s,h)$.
  Let us show that if $l_1 \neq l_2$ and $l_2 \in \heartsuit(s, h_{1\to 2})$ then $l_2 \in \heartsuit(s,h)$. By contradiction, let us assume $l_1 \neq l_2$ and $l_2 \in \heartsuit(s, h_{1\to 2})\backslash\heartsuit(s,h)$. Then we have $l_2 \in \mathrm{ref}(s, h_{1\to 2}) \cup \mathrm{acc}(s, h_{1\to 2})$. Then either $l_2 \in \mathrm{ref}(s, h_{1\to 2})$ or $l_2 \in \mathrm{acc}(s, h_{1\to 2})$. But $l_2 \in \mathrm{acc}(s, h_{1\to 2})$ implies $l_2 \in \mathrm{acc}(s, h_{1\to 2})\backslash\heartsuit(s,h) \subseteq \mathrm{acc}(s, h_{1\to 2})\backslash\mathrm{acc}(s,h)$ and thus, by (P3) we get either $l_1 = l_2$ (a contradiction) or $l_1 \notin s(\mathcal{V})$ (a contradiction). From $l_2 \in \mathrm{ref}(s, h_{1\to 2})$ and $l_1 \neq l_2$ we deduce $l_2 \in \mathrm{ref}(s,h) \subseteq \heartsuit(s,h)$ (a contradiction).
  We obtain $\heartsuit(s, h_{1\to 2}) = \heartsuit(s,h) \uplus \{l_1\}$;
- if $l_1 \notin \mathfrak{p}\heartsuit(s,h)$ then neither l_1 nor l_2 belong to $\mathrm{acc}(s, h_{1\to 2})\backslash\mathrm{acc}(s,h)$. Then $l_1 \in \mathrm{ref}(s, h_{1\to 2})$ implies $l_1 \in s(\mathcal{V})$ which contradicts $l_1 \notin \mathfrak{p}\heartsuit(s,h)$. Hence we get $l_1 \notin \heartsuit(s, h_{1\to 2})$. Finally, $l_2 \in \mathrm{ref}(s, h_{1\to 2})$ implies either $l_2 \in \mathrm{ref}(s,h)$ or $l_1 = l_2$. In the former case, we get $l_2 \in \heartsuit(s,h)$. In the later case, we have already proved $l_2 = l_1 \notin \heartsuit(s, h_{1\to 2})$. Hence in any case, ($l_1 \notin \heartsuit(s, h_{1\to 2})$ and $l_2 \in \heartsuit(s, h_{1\to 2})$) imply $l_2 \in \heartsuit(s,h)$.
  We obtain $\heartsuit(s, h_{1\to 2}) = \heartsuit(s,h)$. □

Proposition 2.11 *Let (s,h) be a memory state, $l_1 \in \mathbb{N}\backslash\mathrm{dom}(h)$ and $l_2 \in \mathbb{N}$. Let us write $h_{1\to 2}$ for $h \uplus [l_1 \mapsto l_2]$ and let i be in $[1,q]$. The following identities hold:*

$$\mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i) = \begin{cases} \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \uplus \{l_1\} & \textit{if } l_1 \notin \mathfrak{p}\heartsuit(s,h) \textit{ and } l_2 = s(\mathtt{x}_i) \\ \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) - \{l_2\} & \textit{if } l_1 \in s(\mathcal{V}) \textit{ and } l_2 \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \\ \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) & \textit{otherwise} \end{cases}$$

$$\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \begin{cases} \mathrm{loop}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\} & \textit{if } l_1 \notin \mathfrak{p}\heartsuit(s,h) \textit{ and } l_1 = l_2 \\ \mathrm{loop}_{\overline{\heartsuit}}(s,h) - \{l_2\} & \textit{if } l_1 \in s(\mathcal{V}) \textit{ and } l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h) \\ \mathrm{loop}_{\overline{\heartsuit}}(s,h) & \textit{otherwise} \end{cases}$$

$$\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \begin{cases} \mathrm{rem}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\} & \textit{if } l_1 \notin \mathfrak{p}\heartsuit(s,h) \textit{ and } l_2 \notin s(\mathcal{V}) \cup \{l_1\} \\ \mathrm{rem}_{\overline{\heartsuit}}(s,h) - \{l_2\} & \textit{if } l_1 \in s(\mathcal{V}) \textit{ and } l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h) \\ \mathrm{rem}_{\overline{\heartsuit}}(s,h) & \textit{otherwise} \end{cases}$$

where $X - \{l_2\}$ means that the location l_2 already belongs to the set X and is (strictly) removed from it.

Proof Let us first establish the two following properties:

(P1) $l_1 \in \heartsuit(s, h_{1\to 2})$ iff $l_1 \in \mathfrak{p}\heartsuit(s,h)$;
(P2) if $l_2 \in \mathrm{dom}(h)\backslash\heartsuit(s,h)$ then $(l_2 \in \heartsuit(s, h_{1\to 2})$ iff $l_1 \in s(\mathcal{V}))$.

Property (P1) is a direct consequence of the last equation of Proposition 2.10 and the fact that $l_1 \notin \heartsuit(s,h)$ (remember $l_1 \notin \mathrm{dom}(h)$). Let us prove Property (P2):

- for the *only if part*, we assume $l_2 \in \heartsuit(s, h_{1\to 2})$ and prove $l_1 \in s(\mathcal{V})$ by contradiction. Indeed, if $l_1 \notin s(\mathcal{V})$, then we have the inclusion $l_2 \in \heartsuit(s, h_{1\to 2}) \subseteq \heartsuit(s,h) \cup \{l_1\}$ according to Proposition 2.10. Hence either $l_2 \in \heartsuit(s,h)$ which contradicts $l_2 \in \mathrm{dom}(h)\backslash\heartsuit(s,h)$ or $l_2 = l_1$ which implies $l_2 \notin \mathrm{dom}(h)$ and contradicts $l_2 \in \mathrm{dom}(h)\backslash\heartsuit(s,h)$;
- for the *if part*, if $l_1 \in s(\mathcal{V})$ then $l_2 = h_{1\to 2}(l_1) \in h_{1\to 2}(s(\mathcal{V}))$. Since $l_2 \in \mathrm{dom}(h) \subseteq \mathrm{dom}(h_{1\to 2})$ we deduce $l_2 \in \heartsuit(s, h_{1\to 2})$.

  By Proposition 2.10, there are only three possible values for $\heartsuit(s, h_{1\to 2})$:

$$\heartsuit(s, h_{1\to 2}) \in \big\{\heartsuit(s,h), \heartsuit(s,h) \uplus \{l_1\}, \heartsuit(s,h) \uplus \{l_1, l_2\}\big\} \tag{A.1}$$

Let us now consider the case of $\mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)$. According to Proposition 2.10 and $l_1 \notin \mathrm{dom}(h)$, we know that

$$\mathrm{pred}(s, h_{1\to 2}, i) \in \big\{\mathrm{pred}(s,h,i), \mathrm{pred}(s,h,i) \uplus \{l_1\}\big\}.$$

Hence there are only three possible values for $\mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)$ which are $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \uplus \{l_1\}$, $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) - \{l_2\}$ and $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ and we study those three cases:

- $l_1 \in \mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)\backslash\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ iff $l_1 \in \mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)$ iff $l_1 \in \mathrm{pred}(s, h_{1\to 2}, i)$ and $l_1 \notin \heartsuit(s, h_{1\to 2})$ iff $l_2 = s(\mathtt{x}_i)$ and $l_1 \notin \mathfrak{p}\heartsuit(s,h)$;
- if $l_2 \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)\backslash\mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)$ then $l_2 \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ and $(l_2 \notin \mathrm{pred}(s, h_{1\to 2}, i)$ or $l_2 \in \heartsuit(s, h_{1\to 2}))$. But $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \subseteq \mathrm{pred}(s,h,i) \subseteq \mathrm{pred}(s, h_{1\to 2}, i)$. Hence we know that $l_2 \in \mathrm{pred}(s, h_{1\to 2}, i)$ and thus we must have $l_2 \in \heartsuit(s, h_{1\to 2})$. Since $l_2 \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \subseteq \mathrm{dom}(h)\backslash\heartsuit(s,h)$ we deduce $l_1 \in s(\mathcal{V})$ by Property (P2);
- if $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ then $l_2 \in h_{1\to 2}(s(\mathcal{V})) \cap \mathrm{dom}(h) \subseteq \heartsuit(s, h_{1\to 2})$. Hence $l_2 \notin \mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)$.

Let us now consider the case of $\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2})$. According to Proposition 2.10, we know that

$$\mathrm{loop}(s, h_{1\to 2}) \in \big\{\mathrm{loop}(s,h), \mathrm{loop}(s,h) \uplus \{l_1\}\big\}$$

and by inclusion (A.1), we deduce that there are only three possible values for $\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2})$ which are $\mathrm{loop}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\}$, $\mathrm{loop}_{\overline{\heartsuit}}(s,h) - \{l_2\}$ and $\mathrm{loop}_{\overline{\heartsuit}}(s,h)$.

- $l_1 \in \mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2})\backslash\mathrm{loop}_{\overline{\heartsuit}}(s,h)$ iff $l_1 \in \mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2})$ iff $l_1 \in \mathrm{loop}(s, h_{1\to 2})$ and $l_1 \notin \heartsuit(s, h_{1\to 2})$ iff $l_1 = l_2$ and $l_1 \notin \mathfrak{p}\heartsuit(s,h)$;
- if $l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)\backslash\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2})$ then $l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ and either $l_2 \notin \mathrm{loop}(s, h_{1\to 2})$ or $l_2 \in \heartsuit(s, h_{1\to 2})$. But $\mathrm{loop}_{\overline{\heartsuit}}(s,h) \subseteq \mathrm{loop}(s, h_{1\to 2})$. Hence we get $l_2 \in \mathrm{loop}(s, h_{1\to 2})$ and thus we must have $l_2 \in \heartsuit(s, h_{1\to 2})$. Since $l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h) \subseteq \mathrm{dom}(h)\backslash\heartsuit(s,h)$ we deduce $l_1 \in s(\mathcal{V})$ by Property (P2);
- if $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ then $l_2 \in h_{1\to 2}(s(\mathcal{V})) \cap \mathrm{dom}(h) \subseteq \heartsuit(s, h_{1\to 2})$. Hence $l_2 \notin \mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2})$.

Let us now consider the case of $\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2}, i)$. According to Proposition 2.10, we know that

$$\mathrm{rem}(s, h_{1\to 2}) \in \big\{\mathrm{rem}(s,h), \mathrm{rem}(s,h) \uplus \{l_1\}\big\}$$

and by inclusion (A.1), we deduce that there are only three possible values for $\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2})$ which are $\mathrm{rem}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\}$, $\mathrm{rem}_{\overline{\heartsuit}}(s,h) - \{l_2\}$ and $\mathrm{rem}_{\overline{\heartsuit}}(s,h)$.

- $l_1 \in \mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2})\backslash\mathrm{rem}_{\overline{\heartsuit}}(s,h)$ iff $l_1 \in \mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2})$ iff $l_1 \in \mathrm{rem}(s, h_{1\to 2})$ and $l_1 \notin \heartsuit(s, h_{1\to 2})$ iff $l_2 \notin s(\mathcal{V}) \cup \{l_1\}$ and $l_1 \notin \mathfrak{p}\heartsuit(s,h)$;
- if $l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)\backslash\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2})$ then $l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ and either $l_2 \notin \mathrm{rem}(s, h_{1\to 2})$ or $l_2 \in \heartsuit(s, h_{1\to 2})$. Because of the inclusions $\mathrm{rem}_{\overline{\heartsuit}}(s,h) \subseteq \mathrm{rem}(s,h) \subseteq \mathrm{rem}(s, h_{1\to 2})$, we have $l_2 \in \mathrm{rem}(s, h_{1\to 2})$ and thus we deduce $l_2 \in \heartsuit(s, h_{1\to 2})$. Since $l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h) \subseteq \mathrm{dom}(h)\backslash\heartsuit(s,h)$ we deduce $l_1 \in s(\mathcal{V})$ by Property (P2);

– if $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ then $l_2 \in h_{1\to 2}(s(\mathcal{V})) \cap \mathrm{dom}(h) \subseteq \heartsuit(s, h_{1\to 2})$. Hence $l_2 \notin \mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2})$. □

Lemma 2.14 *For any $k \geqslant 1$ and for any $i \in [1,q]$, there exist 1SL1 formulæ denoted $\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant k$, $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ and $\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k$ respectively such that, for any memory state (s,h) and for any location $l \in \mathbb{N}$ the following equivalences hold:*

1. $(s,h) \models_l \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant k$ *iff* $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant k$;
2. $(s,h) \models_l \#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ *iff* $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant k$;
3. $(s,h) \models_l \#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k$ *iff* $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant k$.

Proof Let us first establish the equivalence

$$(s,h) \models_l \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_j) \geqslant k \quad \text{iff} \quad \mathrm{card}(\mathrm{pred}_{\heartsuit}(s,h,j)) \geqslant k$$

For the *if part*, let us assume $\mathrm{card}(\mathrm{pred}_{\heartsuit}(s,h,j)) \geqslant k$. Then, then let us define

$$R = \left\{ i \in [1,q] \;\middle|\; h(s(\mathtt{x}_i)) = s(\mathtt{x}_j) \text{ and } \forall r \in [1,q],\, s(\mathtt{x}_r) = s(\mathtt{x}_i) \Rightarrow i \leqslant r \right\}$$
$$A = \left\{ i \in [1,q] \;\middle|\; \begin{array}{c} h(s(\mathtt{x}_i)) \notin s(\mathcal{V}) \text{ and } h^2(s(\mathtt{x}_i)) = s(\mathtt{x}_j) \\ \text{and } \forall r \in [1,q],\, h(s(\mathtt{x}_r)) = h(s(\mathtt{x}_i)) \Rightarrow i \leqslant r \end{array} \right\}$$

We also recall the notations $sR = \{s(\mathtt{x}_i) \mid i \in R\}$ and $sA = \{s(\mathtt{x}_i) \mid i \in A\}$ from Proposition 2.13. We check the identities $\mathrm{pred}_{\heartsuit}(s,h,j) = sR \uplus h(sA)$, $\mathrm{card}(sR) = \mathrm{card}(R)$ and $\mathrm{card}(h(sA)) = \mathrm{card}(A)$ hold. We deduce $\mathrm{card}(R)+\mathrm{card}(A) \geqslant k$. By Proposition 2.13, we have $(s,h) \models_l \mathtt{ref}_R$ and $(s,h) \models_l \mathtt{acc}_A$. For any $r \in R$ we have $h(s(\mathtt{x}_r)) = s(\mathtt{x}_j)$ hence $(s,h) \models_l \bigwedge_{r\in R} \mathtt{x}_r \hookrightarrow \mathtt{x}_j$. For any $a \in A$ we have $h^2(s(\mathtt{x}_a)) = s(\mathtt{x}_j)$ hence $(s,h) \models_l \bigwedge_{a\in A} \mathtt{btwn}(\mathtt{x}_a, \mathtt{x}_j)$.

For the *only if part*, let us assume $(s,h) \models_l \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_j) \geqslant k$. By definition, there exist $R, A \subseteq [1,q]$ such that $\mathrm{card}(R) + \mathrm{card}(A) \geqslant k$, $(s,h) \models_l \mathtt{ref}_R$, $(s,h) \models_l \mathtt{acc}_A$, $\forall r \in R$, $h(s(\mathtt{x}_r)) = s(\mathtt{x}_j)$ and $\forall a \in A$, $h^2(s(\mathtt{x}_a)) = s(\mathtt{x}_j)$. We deduce $sR \subseteq \mathrm{pred}(s,h,j) \cap \mathrm{ref}(s,h)$ and $h(sA) \subseteq \mathrm{pred}(s,h,j) \cap (\mathrm{acc}(s,h) \backslash \mathrm{ref}(s,h))$ and $\mathrm{card}(sR) = \mathrm{card}(R)$ and $\mathrm{card}(h(sA)) = \mathrm{card}(A)$. Hence $sR \uplus h(sA) \subseteq \mathrm{pred}_{\heartsuit}(s,h,j)$ and $\mathrm{card}(sR \uplus h(sA)) = \mathrm{card}(R) + \mathrm{card}(A) \geqslant k$. As a consequence, $\mathrm{card}(\mathrm{pred}_{\heartsuit}(s,h,j)) \geqslant k$ holds.

Let us now establish the equivalence

$$(s,h) \models_l \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant k \quad \text{iff} \quad \mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)) \geqslant k$$

For the *if part*, let us assume $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)) \geqslant k$. Let us define $p = \mathrm{card}(\mathrm{pred}_{\heartsuit}(s,h,j))$. From $\mathrm{pred}_{\heartsuit}(s,h,j) \subseteq \heartsuit(s,h)$ we deduce $p \leqslant 2q$. We have $\mathrm{card}(\mathrm{pred}_{\heartsuit}(s,h,j)) = p < p+1$ and as a consequence, we deduce $(s,h) \not\models_l \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_j) \geqslant p+1$. From $\mathrm{pred}(s,h,j) = \mathrm{pred}_{\heartsuit}(s,h,j) \uplus \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ we get $\mathrm{pred}(s,h,j) \geqslant k+p$ and thus the relation $(s,h) \models_l \#\,\mathtt{pred}(\mathtt{x}_j) \geqslant k+p$ holds. We deduce $(s,h) \models_l \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant k$.

For the *only if part*, let us assume $(s,h) \models_l \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant k$. There exists $p \leqslant 2q$ such that $(s,h) \models_l \#\,\mathtt{pred}(\mathtt{x}_j) \geqslant k+p$ and $(s,h) \not\models_l \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_j) \geqslant p+1$. We deduce the upper bound $\mathrm{card}(\mathrm{pred}_{\heartsuit}(s,h,j)) \leqslant p$ and the lower bound $\mathrm{card}(\mathrm{pred}(s,h,j)) \geqslant k+p$. Using the partition $\mathrm{pred}(s,h,j) = \mathrm{pred}_{\heartsuit}(s,h,j) \uplus \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$, we derive the lower bound $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)) \geqslant k$.

The cases of the test formulæ $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant k$ and $\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant k$ can be treated in a similar way after slight modifications in the definitions of R and A. □

Lemma 2.21 *Let $\alpha_1, \alpha_2 \in \mathbb{N}$ and X, X', Y_0 be finite sets such that $X \uplus X' \sim_{\alpha_1+\alpha_2} Y_0$ holds. Then there are two finite sets Y, Y' such that $Y_0 = Y \uplus Y'$, $X \sim_{\alpha_1} Y$ and $X' \sim_{\alpha_2} Y'$ hold.*

Proof By Proposition 2.18 item 3, we have two cases: either $\mathrm{card}(X \uplus X') = \mathrm{card}(Y_0) < \alpha_1 + \alpha_2$ or $\mathrm{card}(X \uplus X') \geqslant \alpha_1 + \alpha_2$ and $\mathrm{card}(Y_0) \geqslant \alpha_1 + \alpha_2$.

The case $\mathrm{card}(X \uplus X') = \mathrm{card}(Y_0) < \alpha_1 + \alpha_2$ is easy: we have $\mathrm{card}(X) \leqslant \mathrm{card}(X) + \mathrm{card}(X') = \mathrm{card}(Y_0)$; then for Y, we choose any subset of Y_0 such that $\mathrm{card}(Y) = \mathrm{card}(X)$. Then we define $Y' = Y_0 \backslash Y$ and we get $\mathrm{card}(Y') = \mathrm{card}(Y_0) - \mathrm{card}(Y) = \mathrm{card}(X \uplus X') - \mathrm{card}(X) = \mathrm{card}(X')$. Then we have both $X \sim_{\alpha_1} Y$ and $X' \sim_{\alpha_2} Y'$.

Let us consider the case $\mathrm{card}(X \uplus X') \geqslant \alpha_1 + \alpha_2$ and $\mathrm{card}(Y_0) \geqslant \alpha_1 + \alpha_2$. We have four sub-cases:

- the case $\mathrm{card}(X) < \alpha_1$ and $\mathrm{card}(X') < \alpha_2$ is impossible because it contradicts $\mathrm{card}(X \uplus X') \geqslant \alpha_1 + \alpha_2$;
- in the case $\mathrm{card}(X) \geqslant \alpha_1$ and $\mathrm{card}(X') < \alpha_2$, let Y' be any subset of Y_0 such that $\mathrm{card}(Y') = \mathrm{card}(X')$ and $Y = Y_0 \backslash Y'$. We have $\mathrm{card}(X') = \mathrm{card}(Y')$ hence $X' \sim_{\alpha_2} Y'$. We have $\mathrm{card}(X) \geqslant \alpha_1$ and $\mathrm{card}(Y) = \mathrm{card}(Y_0) - \mathrm{card}(Y') \geqslant (\alpha_1 + \alpha_2) - \alpha_2 = \alpha_1$ hence $X \sim_{\alpha_1} Y$;
- the case $\mathrm{card}(X) < \alpha_1$ and $\mathrm{card}(X') \geqslant \alpha_2$ is obtained by symmetry from the previous case;
- in the case $\mathrm{card}(X) \geqslant \alpha_1$ and $\mathrm{card}(X') \geqslant \alpha_2$, let Y be any subset of Y_0 s.t. $\mathrm{card}(Y) = \alpha_1$ and $Y' = Y_0 \backslash Y$. We have $\mathrm{card}(X) \geqslant \alpha_1$ and $\mathrm{card}(Y) = \alpha_1$ hence $X \sim_{\alpha_1} Y$. We have $\mathrm{card}(X') \geqslant \alpha_2$ and $\mathrm{card}(Y') = \mathrm{card}(Y_0) - \mathrm{card}(Y) \geqslant (\alpha_1 + \alpha_2) - \alpha_1 = \alpha_2$ hence $X' \sim_{\alpha_2} Y'$. □

B Proofs of Section 3

Proposition 3.13 *Let $u, v \in \mathbb{N}$. For* (𝔗10)–(𝔗20) *defined as*

(𝔗10) $u \in s(\mathcal{V})$ *iff* $v \in s'(\mathcal{V})$;
(𝔗11) $u \in h(s(\mathcal{V}))$ *iff* $v \in h'(s'(\mathcal{V}))$;
(𝔗12) $u \in \mathfrak{p}\heartsuit(s,h)$ *iff* $v \in \mathfrak{p}\heartsuit(s',h')$;
(𝔗13) $u \in \heartsuit(s,h)$ *iff* $v \in \heartsuit(s',h')$;
(𝔗14) $u \in \mathrm{pred}(s,h,i)$ *iff* $v \in \mathrm{pred}(s',h',i)$ *for any* $i \in [1,q]$;
(𝔗15) $u \in \mathrm{pred}(s,h)$ *iff* $v \in \mathrm{pred}(s',h')$;
(𝔗16) $u \in \mathrm{loop}(s,h)$ *iff* $v \in \mathrm{loop}(s',h')$;
(𝔗17) $u \in \mathrm{rem}(s,h)$ *iff* $v \in \mathrm{rem}(s',h')$;
(𝔗18) $u \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ *iff* $v \in \mathrm{pred}_{\overline{\heartsuit}}(s',h',i)$ *for any* $i \in [1,q]$;
(𝔗19) $u \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ *iff* $v \in \mathrm{loop}_{\overline{\heartsuit}}(s',h')$;
(𝔗20) $u \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ *iff* $v \in \mathrm{rem}_{\overline{\heartsuit}}(s',h')$;
(𝔗21) $u \in \mathfrak{p}\heartsuit(\mathfrak{m})$ *iff* $v \in \mathfrak{p}\heartsuit(\mathfrak{m}')$.

the following propositions hold:

1. (𝔗2) *implies* (𝔗10);
2. (𝔗3) *implies* (𝔗11);
3. (𝔗2–3) *imply* (𝔗12);
4. (𝔗2–4) *imply* (𝔗13);
5. (𝔗2–6) *imply* (𝔗10–20);
6. (𝔗1–3) *imply* (𝔗21).

Proof The proofs are easy. Here is a summary of the arguments:

(𝔗10) is a direct consequence of (𝔗2);
(𝔗11) is a direct consequence of (𝔗3);
(𝔗12) we use the identity $\mathfrak{p}\heartsuit(s,h) = s(\mathcal{V}) \cup h(s(\mathcal{V}))$, (𝔗10) and (𝔗11);
(𝔗13) we use the identity $\heartsuit(s,h) = \mathfrak{p}\heartsuit(s,h) \cap \mathrm{dom}(h)$, (𝔗12) and (𝔗4);
(𝔗14) just another way to write (𝔗5);
(𝔗15) we use the identity $\mathrm{pred}(s,h) = \bigcup_i \mathrm{pred}(s,h,i)$ and (𝔗14);
(𝔗16) just another way to write (𝔗6);
(𝔗17) with $\mathrm{rem}(s,h) = \mathrm{dom}(h) \backslash (\mathrm{pred}(s,h) \cup \mathrm{loop}(s,h))$, (𝔗4), (𝔗15), (𝔗16);
(𝔗18) we use $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) = \mathrm{pred}(s,h,i) \backslash \mathfrak{p}\heartsuit(s,h)$ and (𝔗14), (𝔗12);
(𝔗19) we use $\mathrm{loop}_{\overline{\heartsuit}}(s,h) = \mathrm{loop}(s,h) \backslash \mathfrak{p}\heartsuit(s,h)$ and (𝔗16), (𝔗12);
(𝔗20) we use $\mathrm{rem}_{\overline{\heartsuit}}(s,h) = \mathrm{rem}(s,h) \backslash \mathfrak{p}\heartsuit(s,h)$ and (𝔗17), (𝔗12);
(𝔗21) we use the identity $\mathfrak{p}\heartsuit(\mathfrak{m}) = \mathfrak{p}\heartsuit(s,h) \cup \{l\}$, (𝔗12) and (𝔗1). □

Proposition 3.15 *The following inclusions hold:*

1. $\mathfrak{R} \subseteq \mathfrak{p}\heartsuit(s,h) \times \mathfrak{p}\heartsuit(s',h')$
2. $\mathfrak{T} \cap \mathfrak{p}\heartsuit(s,h) \times \mathbb{N} \subseteq \mathfrak{R}$
3. $\mathfrak{T} \cap \mathbb{N} \times \mathfrak{p}\heartsuit(s',h') \subseteq \mathfrak{R}$
4. $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{p}\heartsuit(\mathfrak{m}) \times \mathfrak{p}\heartsuit(\mathfrak{m}')$
5. $\mathfrak{T}^{\mathsf{l}} \cap \mathfrak{p}\heartsuit(\mathfrak{m}) \times \mathbb{N} \subseteq \mathfrak{R}^{\mathsf{l}}$
6. $\mathfrak{T}^{\mathsf{l}} \cap \mathbb{N} \times \mathfrak{p}\heartsuit(\mathfrak{m}') \subseteq \mathfrak{R}^{\mathsf{l}}$

Proof Inclusions 1 and 4 are trivial. Let us consider Inclusion 2, hence let u and v be such that $u\ \mathfrak{T}\ v$ and $u \in \mathfrak{p}\heartsuit(s,h)$. Since $\mathfrak{p}\heartsuit(s,h) = s(\mathcal{V}) \cup h(s(\mathcal{V}))$ we have two cases:

- either $u = s(\mathtt{x}_i)$ for some $i \in [1,q]$ and then $v = s'(\mathtt{x}_i)$ by ($\mathfrak{T}2$). Hence u and v satisfy ($\mathfrak{R}2$) and thus we deduce $u\ \mathfrak{R}\ v$;
- or $u = h(s(\mathtt{x}_i))$ for some $i \in [1,q]$ and then $v = h'(s'(\mathtt{x}_i))$ by ($\mathfrak{T}3$). Hence u and v satisfy ($\mathfrak{R}3$) and thus we deduce $u\ \mathfrak{R}\ v$.

Inclusions 3, 5 and 6 are proved in a similar way. □

Proposition 3.16 *The following properties hold:*

1. *The relation $\mathfrak{T}$ restricted to $\mathfrak{p}\heartsuit(s,h) \times \mathfrak{p}\heartsuit(s',h')$ is functional and injective.*
2. *The relation $\mathfrak{T}^{\mathsf{l}}$ restricted to $\mathfrak{p}\heartsuit(\mathfrak{m}) \times \mathfrak{p}\heartsuit(\mathfrak{m}')$ is functional and injective.*
3. *For any $u \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s,h)$, $v \notin \mathrm{dom}(h') \cup \mathfrak{p}\heartsuit(s',h')$, we have $u\ \mathfrak{T}\ v$.*
4. *For any $u \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})$, $v \notin \mathrm{dom}(h') \cup \mathfrak{p}\heartsuit(\mathfrak{m}')$, we have $u\ \mathfrak{T}^{\mathsf{l}}\ v$.*

Proof To show that $\mathfrak{T}$ is functional, we prove that for all $u \in \mathfrak{p}\heartsuit(s,h)$, for all $v, w \in \mathfrak{p}\heartsuit(s',h')$, $u\ \mathfrak{T}\ v$ and $u\ \mathfrak{T}\ w$ imply $v = w$. If $u = s(\mathtt{x}_i)$ then we must have $v = s'(\mathtt{x}_i) = w$; and if $u = h(s(\mathtt{x}_i))$ then we must have $v = h'(s'(\mathtt{x}_i)) = w$. By symmetric arguments, $\mathfrak{T}$ is injective. The proof that $\mathfrak{T}^{\mathsf{l}}$ restricted $\mathfrak{p}\heartsuit(\mathfrak{m}) \times \mathfrak{p}\heartsuit(\mathfrak{m}')$ is functional and injective (Property 2) is similar.

Let us now establish Property 3. We consider $u, v \in \mathbb{N}$ such that $u \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s,h)$ and $v \notin \mathrm{dom}(h') \cup \mathfrak{p}\heartsuit(s',h')$. Conditions ($\mathfrak{T}2$)–($\mathfrak{T}3$) hold because $u \notin s(\mathcal{V}) \cup h(s(\mathcal{V}))$ and $v \notin s'(\mathcal{V}) \cup h'(s'(\mathcal{V}))$. Conditions ($\mathfrak{T}4$)–($\mathfrak{T}6$) hold because $u \notin \mathrm{dom}(h)$ and $v \notin \mathrm{dom}(h')$. The proof of Property 4 follows a similar pattern. □

Theorem 3.17 $\mathfrak{m} \simeq_b \mathfrak{m}'$ *if and only if* $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$.

Proof Let us show the *only if part* first. We assume $\mathfrak{m} \simeq_b \mathfrak{m}'$ and we prove $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$ by case analysis on $u\ \mathfrak{R}^{\mathsf{l}}\ v$:

($\mathfrak{R}1$) let us check $l\ \mathfrak{T}^{\mathsf{l}}\ l'$: Condition ($\mathfrak{T}1$) is trivial; for Condition ($\mathfrak{T}2$), $l = s(\mathtt{x}_j)$ iff $(s,h) \models_l \mathtt{x}_j = \mathtt{u}$ iff $(s',h') \models_{l'} \mathtt{x}_j = \mathtt{u}$ iff $l' = s'(\mathtt{x}_j)$; for Condition ($\mathfrak{T}3$), $l = h(s(\mathtt{x}_j))$ iff $(s,h) \models_l \mathtt{x}_j \hookrightarrow \mathtt{u}$ iff $(s',h') \models_{l'} \mathtt{x}_j \hookrightarrow \mathtt{u}$ iff $l' = h'(s'(\mathtt{x}_j))$; for Condition ($\mathfrak{T}4$), $l \in \mathrm{dom}(h)$ iff $(s,h) \models_l \mathtt{alloc(u)}$ iff $(s',h') \models_{l'} \mathtt{alloc(u)}$ iff $l' \in \mathrm{dom}(h')$; for Condition ($\mathfrak{T}5$), $h(l) = s(\mathtt{x}_j)$ iff $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_j$ iff $(s',h') \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_j$ iff $h'(l') = s'(\mathtt{x}_j)$; for Condition ($\mathfrak{T}6$), $h(l) = l$ iff $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ iff $(s',h') \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$ iff $h'(l') = l'$;

($\mathfrak{R}2$) let us check $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{l}}\ s'(\mathtt{x}_i)$: for Condition ($\mathfrak{T}1$), $s(\mathtt{x}_i) = l$ iff $s'(\mathtt{x}_i) = l'$ (as previously); for Condition ($\mathfrak{T}2$), $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ iff $(s,h) \models_l \mathtt{x}_i = \mathtt{x}_j$ iff $(s',h') \models_{l'} \mathtt{x}_i = \mathtt{x}_j$ iff $s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$; for Condition ($\mathfrak{T}3$), $s(\mathtt{x}_i) = h(s(\mathtt{x}_j))$ iff $(s,h) \models_l \mathtt{x}_j \hookrightarrow \mathtt{x}_i$ iff $(s',h') \models_{l'} \mathtt{x}_j \hookrightarrow \mathtt{x}_i$ iff $s'(\mathtt{x}_i) = h'(s'(\mathtt{x}_j))$; for Condition ($\mathfrak{T}4$), $s(\mathtt{x}_i) \in \mathrm{dom}(h)$ iff $(s,h) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)$ iff $(s',h') \models_{l'} \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)$ iff $s'(\mathtt{x}_i) \in \mathrm{dom}(h')$; for Condition ($\mathfrak{T}5$), $h(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$ iff $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ iff $(s',h') \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ iff $h'(s'(\mathtt{x}_i)) = s'(\mathtt{x}_j)$; for Condition ($\mathfrak{T}6$), $h(s(\mathtt{x}_i)) = s(\mathtt{x}_i)$ iff $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_i$ iff $(s',h') \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_i$ iff $h'(s'(\mathtt{x}_i)) = s'(\mathtt{x}_i)$;

($\mathfrak{R}3$) let us check $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{l}}\ h'(s'(\mathtt{x}_i))$: for Condition ($\mathfrak{T}1$), $h(s(\mathtt{x}_i)) = l$ iff $h'(s'(\mathtt{x}_i)) = l'$ (as previously); for Condition ($\mathfrak{T}2$), $h(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$ iff $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ iff $(s',h') \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ iff $h'(s'(\mathtt{x}_i)) = s'(\mathtt{x}_j)$; for Condition ($\mathfrak{T}3$), $h(s(\mathtt{x}_i)) = h(s(\mathtt{x}_j))$ iff $(s,h) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$ iff $(s',h') \models_{l'} \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$ iff $h'(s'(\mathtt{x}_i)) = h'(s'(\mathtt{x}_j))$; for Condition ($\mathfrak{T}4$), $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h)$ iff $(s,h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ iff $(s',h') \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$ iff $h'(s'(\mathtt{x}_i)) \in \mathrm{dom}(h')$; for Condition ($\mathfrak{T}5$), $h(h(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$ iff $(s,h) \models_l \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$ iff $(s',h') \models_{l'} \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$ iff $h'(h'(s'(\mathtt{x}_i))) = s'(\mathtt{x}_j)$; for Condition ($\mathfrak{T}6$), $h(h(s(\mathtt{x}_i))) = h(s(\mathtt{x}_i))$ iff $(s,h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ iff $(s',h') \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$ iff $h'(h'(s'(\mathtt{x}_i))) = h'(s'(\mathtt{x}_i))$.

Let us now tackle the *if part*. We assume $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{T}^{\mathsf{I}}$. Hence we have $l\ \mathfrak{T}^{\mathsf{I}}\ l'$, $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{I}}\ s'(\mathtt{x}_i)$ for any $i \in [1,q]$, and $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$ for any $i \in [1,q]$ such that $s(\mathtt{x}_i) \in \mathrm{dom}(h)$ (and $s'(\mathtt{x}_i) \in \mathrm{dom}(h')$). To establish $\mathfrak{m} \simeq_b \mathfrak{m}'$, we consider a formula $\mathcal{B} \in \mathsf{Basic}^{\mathsf{u}}$ and we show that $(s,h) \models_l \mathcal{B}$ implies $(s',h') \models_{l'} \mathcal{B}$. The reverse implication can be proved by symmetric arguments. We proceed by a case analysis on $\mathcal{B}$:

- $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{x}_j$: if $(s,h) \models_l \mathtt{x}_i = \mathtt{x}_j$ then $s(\mathtt{x}_i) = s(\mathtt{x}_j)$. Using the instance of $(\mathfrak{T}2)$ for $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{I}}\ s'(\mathtt{x}_i)$ with parameter j, we get $s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$. We deduce $(s',h') \models_{l'} \mathtt{x}_i = \mathtt{x}_j$;
- $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$: if $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ then $h(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$. Using the instance of $(\mathfrak{T}2)$ for $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$ with parameter j, we get $h'(s'(\mathtt{x}_i)) = s'(\mathtt{x}_j)$. We deduce $(s',h') \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$;
- $\mathcal{B}$ is $\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$: if $(s,h) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$ then $h(s(\mathtt{x}_i)) = h(s(\mathtt{x}_j))$. Using the instance of $(\mathfrak{T}3)$ for $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$ with parameter j, we get $h'(s'(\mathtt{x}_i)) = h'(s'(\mathtt{x}_j))$. We deduce $(s',h') \models_{l'} \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$;
- $\mathcal{B}$ is $\mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$: if $(s,h) \models_l \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$ then $h(h(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$. Using the instance of $(\mathfrak{T}5)$ for $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$ with parameter j, we get $h'(h'(s'(\mathtt{x}_i))) = s'(\mathtt{x}_j)$. We deduce $(s',h') \models_{l'} \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$;
- $\mathcal{B}$ is $\mathtt{toalloc}(\mathtt{x}_i)$: if $(s,h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ then $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h)$. Using $(\mathfrak{T}4)$ for $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$, we get $h'(s'(\mathtt{x}_i)) \in \mathrm{dom}(h')$. We deduce $(s',h') \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$;
- $\mathcal{B}$ is $\mathtt{toloop}(\mathtt{x}_i)$: if $(s,h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ then $h(h(s(\mathtt{x}_i))) = h(s(\mathtt{x}_i))$. By $(\mathfrak{T}6)$ for $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$, we get $h'(h'(s'(\mathtt{x}_i))) = h'(s'(\mathtt{x}_i))$. We deduce $(s',h') \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$;
- $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$: if $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ then $h(l) = l$. Using $(\mathfrak{T}6)$ for $l\ \mathfrak{T}^{\mathsf{I}}\ l'$, we get $h'(l') = l'$. We deduce $(s',h') \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{alloc}(\mathtt{u})$: if $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$ then $l \in \mathrm{dom}(h)$. Using $(\mathfrak{T}4)$ for $l\ \mathfrak{T}^{\mathsf{I}}\ l'$, we get $l' \in \mathrm{dom}(h')$. We deduce $(s',h') \models_{l'} \mathtt{alloc}(\mathtt{u})$;
- $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$: if $(s,h) \models_l \mathtt{x}_i = \mathtt{u}$ then $s(\mathtt{x}_i) = l$. Using $(\mathfrak{T}1)$ for $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{I}}\ s'(\mathtt{x}_i)$, we get $s'(\mathtt{x}_i) = l'$. We deduce $(s',h') \models_{l'} \mathtt{x}_i = \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$: if $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ then $h(s(\mathtt{x}_i)) = l$. Using $(\mathfrak{T}1)$ for $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{I}}\ h'(s'(\mathtt{x}_i))$, we get $h'(s'(\mathtt{x}_i)) = l'$. We deduce $(s',h') \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_i$: if $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$ then $h(l) = s(\mathtt{x}_i)$. Using $(\mathfrak{T}5)$ for $l\ \mathfrak{T}^{\mathsf{I}}\ l'$ with parameter i, we get $h'(l') = s'(\mathtt{x}_i)$. We deduce $(s',h') \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_i$. □

Proposition 3.18 *If $\mathfrak{m} \simeq_b \mathfrak{m}'$ then the following properties hold:*

1. *The relation $\mathfrak{R}$ is total and surjective between $\mathfrak{p}\heartsuit(s,h)$ and $\mathfrak{p}\heartsuit(s',h')$;*
2. *The relation $\mathfrak{R}'$ is total and surjective between $\mathfrak{p}\heartsuit(\mathfrak{m})$ and $\mathfrak{p}\heartsuit(\mathfrak{m}')$.*

Proof Let us consider Property 1. To show that $\mathfrak{R}$ is total, we prove that for all $u \in \mathfrak{p}\heartsuit(s,h)$, there is $v \in \mathfrak{p}\heartsuit(s',h')$ such that $u\ \mathfrak{R}\ v$. If $u = s(\mathtt{x}_i)$ then choose $v = s'(\mathtt{x}_i)$; and if $u = h(s(\mathtt{x}_i))$ then $s(\mathtt{x}_i) \in \mathrm{dom}(h)$. But we have $s(\mathtt{x}_i)\ \mathfrak{R}^{\mathsf{I}}\ s'(\mathtt{x}_i)$ by definition of $\mathfrak{R}^{\mathsf{I}}$, hence by Theorem 3.17 we deduce $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{I}}\ s'(\mathtt{x}_i)$. As a consequence $s(\mathtt{x}_i)/s'(\mathtt{x}_i)$ verify $(\mathfrak{T}4)$ and thus $s'(\mathtt{x}_i) \in \mathrm{dom}(h')$. We choose $v = h'(s'(\mathtt{x}_i))$ and we get $u\ \mathfrak{R}^{\mathsf{I}}\ v$ and $v \in \mathfrak{p}\heartsuit(s',h')$.

By symmetric arguments, $\mathfrak{R}$ is surjective. The proof that $\mathfrak{R}^{\mathsf{I}}$ is total and surjective (Property 2) is similar.

Proposition 3.21 *If $\mathfrak{m}$ and $\mathfrak{m}'$ satisfy $\mathfrak{m} \simeq_1 \mathfrak{m}'$, then $\mathfrak{T}$ is a total relation on $\mathbb{N}$: for any $u \in \mathbb{N}$, there exists $v \leqslant \mathrm{maxval}(s',h') + 1$ such that $u\ \mathfrak{T}\ v$.*

Proof Since $\simeq_1 \subseteq \simeq_b$ we have $\mathfrak{R} \subseteq \mathfrak{T}$ by Lemma 3.19. Let us consider $u \in \mathbb{N}$. We have to show that there exists $v \in \mathbb{N}$ such that $u\ \mathfrak{T}\ v$ holds. We determine the value of v according to the first condition that holds in the following list:

- if $u \in \mathfrak{p}\heartsuit(s,h)$ then let us define v to be the unique location in $\mathfrak{p}\heartsuit(s',h')$ such that $u\ \mathfrak{R}\ v$, see Lemma 3.19. We deduce $u\ \mathfrak{T}\ v$. The relation $v \leqslant \mathrm{maxval}(s',h') + 1$ holds because $v \in \mathfrak{p}\heartsuit(s',h')$;

if $u \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ for some $j \in [1,q]$ then we know that $\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ is not empty. We have $\mathrm{pred}_{\overline{\heartsuit}}(s,h,j) \sim_1 \mathrm{pred}_{\overline{\heartsuit}}(s',h',j)$ by Proposition 3.10 hence $\mathrm{pred}_{\overline{\heartsuit}}(s',h',j)$ is not empty either. We choose any $v \in \mathrm{pred}_{\overline{\heartsuit}}(s',h',j)$.
The relation $v \leqslant \mathrm{maxval}(s',h')+1$ holds because $v \in \mathrm{dom}(h')$. Let us check that $u \mathfrak{T} v$ holds by establishing Properties ($\mathfrak{T}$2–6) for u/v. We have $u \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ and $v \in \mathrm{pred}_{\overline{\heartsuit}}(s',h',j)$. As a consequence we deduce $u \notin \mathfrak{p}\heartsuit(s,h)$ and $v \notin \mathfrak{p}\heartsuit(s',h')$. Hence Properties ($\mathfrak{T}$2–3) hold. We also have $u \in \mathrm{dom}(h)$ and $v \in \mathrm{dom}(h')$ hence Property ($\mathfrak{T}$4) holds. We have $h(u) = s(\mathtt{x}_j)$ and $h'(v) = s'(\mathtt{x}_j)$. We deduce $h(u)\ \mathfrak{R}\ h'(v)$ and thus $h(u)\ \mathfrak{T}\ h'(v)$. Since ($\mathfrak{T}$2) holds for $h(u)/h'(v)$, we deduce that Property ($\mathfrak{T}$5) holds for u/v. Let us prove Property ($\mathfrak{T}$6) for u/v: the identity $u = h(u)$ implies $u = s(\mathtt{x}_j)$ which contradicts $u \notin \mathfrak{p}\heartsuit(\mathtt{m})$. Hence $u \neq h(u)$ and for the similar reasons, $v \neq h'(v)$;

if $u \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ then we proceed in a way similar to the previous case;

if $u \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ then we proceed in a way similar to the previous case.

In the remaining cases we have $u \notin (\mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(s,h))$. Let us define $v = \mathrm{maxval}(s',h')+1$. We have $v \notin (\mathrm{dom}(h') \cup \mathfrak{p}\heartsuit(s',h'))$ and by Proposition 3.16 item 3, we deduce $u\ \mathfrak{T}\ v$. □

Proposition 3.22 *Let us assume $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$ (or equivalently $\mathtt{m} \simeq_b \mathtt{m}'$). Then the following statements are equivalent:*

1. $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{D}_1 \cap \mathfrak{D}_2$;
2. $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}_1 \cap \mathfrak{T}^{\mathsf{l}}_2$;
3. $\mathfrak{R}^{\mathsf{l}}_1 \subseteq \mathfrak{T}^{\mathsf{l}}_1$ *and* $\mathfrak{R}^{\mathsf{l}}_2 \subseteq \mathfrak{T}^{\mathsf{l}}_2$;
4. $\mathtt{m}_1 \simeq_b \mathtt{m}'_1$ *and* $\mathtt{m}_2 \simeq_b \mathtt{m}'_2$.

Proof Let us review the easy implications first. Obviously, statement 3 and 4 are equivalent by Theorem 3.17. Then statement 2 implies statement 3 by the two following deductions: $\mathfrak{R}^{\mathsf{l}}_1 \subseteq \mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}_1 \cap \mathfrak{T}^{\mathsf{l}}_2 \subseteq \mathfrak{T}^{\mathsf{l}}_1$ and $\mathfrak{R}^{\mathsf{l}}_2 \subseteq \mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}_1 \cap \mathfrak{T}^{\mathsf{l}}_2 \subseteq \mathfrak{T}^{\mathsf{l}}_2$.

Let us now show that statement 1 implies statement 2. So we assume $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{D}_1 \cap \mathfrak{D}_2$. We show that $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}_1$, the case $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}_2$ being treated in a similar way. So let us assume u, v such that $u\ \mathfrak{R}^{\mathsf{l}}\ v$ and let us show that $u\ \mathfrak{T}^{\mathsf{l}}_1\ v$. Because $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$ we have $u\ \mathfrak{T}^{\mathsf{l}}\ v$. Because $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{D}_1 \cap \mathfrak{D}_2$ we have $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{D}_1$. Let us show that u/v verify ($\mathfrak{T}$1–6) with respect to $\mathtt{m}_1/\mathtt{m}'_1$:

Properties ($\mathfrak{T}$1,2) hold because $u\ \mathfrak{T}^{\mathsf{l}}\ v$;
Property ($\mathfrak{T}$3): $u = h_1(s(\mathtt{x}_i))$ iff $(s(\mathtt{x}_i) \in \mathrm{dom}(h_1)$ and $u = h(s(\mathtt{x}_i)))$ iff $(s'(\mathtt{x}_i) \in \mathrm{dom}(h'_1)$ and $v = h'(s'(\mathtt{x}_i)))$ iff $v = h'_1(s'(\mathtt{x}_i))$ because $s(\mathtt{x}_i)\ \mathfrak{D}_1\ s'(\mathtt{x}_i)$ (which comes from $s(\mathtt{x}_i)\ \mathfrak{R}^{\mathsf{l}}\ s'(\mathtt{x}_i)$) and $u\ \mathfrak{T}^{\mathsf{l}}\ v$;
Property ($\mathfrak{T}$4): $u \in \mathrm{dom}(h_1)$ iff $v \in \mathrm{dom}(h'_1)$ because $u\ \mathfrak{D}_1\ v$;
Property ($\mathfrak{T}$5): $h_1(u) = s(\mathtt{x}_i)$ iff $(u \in \mathrm{dom}(h_1)$ and $h(u) = s(\mathtt{x}_i))$ iff $(v \in \mathrm{dom}(h'_1)$ and $h'(v) = s'(\mathtt{x}_i))$ iff $h'_1(v) = s'(\mathtt{x}_i)$ because $u\ \mathfrak{D}_1\ v$ and $u\ \mathfrak{T}^{\mathsf{l}}\ v$;
Property ($\mathfrak{T}$6): $h_1(u) = u$ iff $(u \in \mathrm{dom}(h_1)$ and $h(u) = u)$ iff $(v \in \mathrm{dom}(h'_1)$ and $h'(v) = v)$ iff $h'_1(v) = v$ because $u\ \mathfrak{D}_1\ v$ and $u\ \mathfrak{T}^{\mathsf{l}}\ v$.

Let us finish by showing that statement 3 implies statement 1. So we assume $\mathfrak{R}^{\mathsf{l}}_1 \subseteq \mathfrak{T}^{\mathsf{l}}_1$ and $\mathfrak{R}^{\mathsf{l}}_2 \subseteq \mathfrak{T}^{\mathsf{l}}_2$. Let us show the inclusion $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{D}_1 \cap \mathfrak{D}_2$. We assume u, v such that $u\ \mathfrak{R}^{\mathsf{l}}\ v$. Let us show that $u\ \mathfrak{D}_1\ v$, the case of $u\ \mathfrak{D}_2\ v$ being treated in a similar way. Notice the inclusion $\mathfrak{T}^{\mathsf{l}}_1 \subseteq \mathfrak{D}_1$ that always holds by Definition 3.14. For $u\ \mathfrak{R}^{\mathsf{l}}\ v$ there are three cases:

- either $u = l$ and $v = l'$. We have $l\ \mathfrak{R}^{\mathsf{l}}_1\ l'$, $\mathfrak{R}^{\mathsf{l}}_1 \subseteq \mathfrak{T}^{\mathsf{l}}_1$ and $\mathfrak{T}^{\mathsf{l}}_1 \subseteq \mathfrak{D}_1$, thus we get $l\ \mathfrak{D}_1\ l'$ hence $u\ \mathfrak{D}_1\ v$;
- or $u = s(\mathtt{x}_i)$ and $v = s'(\mathtt{x}_i)$ for some $i \in [1,q]$. We have $s(\mathtt{x}_i)\ \mathfrak{R}^{\mathsf{l}}_1\ s'(\mathtt{x}_i)$, $\mathfrak{R}^{\mathsf{l}}_1 \subseteq \mathfrak{T}^{\mathsf{l}}_1 \subseteq \mathfrak{D}_1$, thus we get $u\ \mathfrak{D}_1\ v$;
- or $u = h(s(\mathtt{x}_i))$ and $v = h'(s'(\mathtt{x}_i))$ for some $i \in [1,q]$. Let us assume $u \in \mathrm{dom}(h_1)$ and let us prove $v \in \mathrm{dom}(h'_1)$. From $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h_1)$ deduce $s(\mathtt{x}_i) \in \mathrm{dom}(h) = \mathrm{dom}(h_1) \uplus \mathrm{dom}(h_2)$. Hence we have two cases:
  - either $s(\mathtt{x}_i) \in \mathrm{dom}(h_1)$. In this case we have $h_1(s(\mathtt{x}_i)) = h(s(\mathtt{x}_i)) = u \in \mathrm{dom}(h_1)$ and $s(\mathtt{x}_i) \in \mathrm{dom}(h_1)$. From $\mathfrak{R}^{\mathsf{l}}_1 \subseteq \mathfrak{T}^{\mathsf{l}}_1$ we deduce $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{l}}_1\ s'(\mathtt{x}_i)$ and $h_1(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{l}}_1\ h'_1(s'(\mathtt{x}_i))$. By ($\mathfrak{T}$4) (twice) we get $h'_1(s'(\mathtt{x}_i)) \in \mathrm{dom}(h'_1)$ and $s'(\mathtt{x}_i) \in \mathrm{dom}(h'_1)$. We deduce $v = h'(s'(\mathtt{x}_i)) = h'_1(s'(\mathtt{x}_i)) \in \mathrm{dom}(h'_1)$;

– or $s(\mathtt{x}_i) \in \mathrm{dom}(h_2)$. In this case we have $h_2(s(\mathtt{x}_i)) = h(s(\mathtt{x}_i)) = u \in \mathrm{dom}(h_1)$ and $s(\mathtt{x}_i) \in \mathrm{dom}(h_2)$. Since $\mathrm{dom}(h) = \mathrm{dom}(h_1) \uplus \mathrm{dom}(h_2)$ we deduce $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h)$ and $h_2(s(\mathtt{x}_i)) \notin \mathrm{dom}(h_2)$. Because $\mathfrak{R}^{\mathsf{l}}_2 \subseteq \mathfrak{T}^{\mathsf{l}}_2$ and $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$ we have $s(\mathtt{x}_i)\ \mathfrak{T}^{\mathsf{l}}_2\ s'(\mathtt{x}_i)$, $h(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{l}}\ h'(s'(\mathtt{x}_i))$ and $h_2(s(\mathtt{x}_i))\ \mathfrak{T}^{\mathsf{l}}_2\ h'_2(s'(\mathtt{x}_i))$. Hence by ($\mathfrak{T}4$) (three times) we deduce $s'(\mathtt{x}_i) \in \mathrm{dom}(h'_2)$, $h'(s'(\mathtt{x}_i)) \in \mathrm{dom}(h')$ and $h'_2(s'(\mathtt{x}_i)) \notin \mathrm{dom}(h'_2)$. As a consequence $v = h'(s'(\mathtt{x}_i)) = h'_2(s'(\mathtt{x}_i)) \notin \mathrm{dom}(h'_2)$ and $v = h'(s'(\mathtt{x}_i)) \in \mathrm{dom}(h')$. We conclude $v \in \mathrm{dom}(h'_1)$. □

Lemma 3.25 *Let $\alpha \geqslant 1$ and let (s,h,l) and (s',h',l') be two pointed memory states. Let $l_1, l_2, l'_1, l'_2 \in \mathbb{N}$ be such that $l_1 \notin \mathrm{dom}(h)$ and $l'_1 \notin \mathrm{dom}(h')$. We assume that one of the conditions below holds:*

(C1) l_1/l'_1 verify ($\mathfrak{T}$1–3), l_2/l'_2 verify ($\mathfrak{T}$1–6), and $l_2 = l_1$ iff $l'_2 = l'_1$;
(C2) $l_1 \notin s(\mathcal{V})$, l_1/l'_1 verify ($\mathfrak{T}$1–3), l_2/l'_2 verify ($\mathfrak{T}$2), and $l_2 = l_1$ iff $l'_2 = l'_1$.

If $(s,h,l) \simeq_\alpha (s',h',l')$ then $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\beta (s', h' \uplus [l'_1 \mapsto l'_2], l')$ where $\beta = \alpha - 1$ if $l_1 \in s(\mathcal{V})$, and $\beta = \alpha$ otherwise.

Proof We denote $h \uplus [l_1 \mapsto l_2]$ by $h_{1\to 2}$ and $h' \uplus [l'_1 \mapsto l'_2]$ by $h'_{1\to 2}$. According to Proposition 3.10, we have to establish

$$(s, h_{1\to 2}, l) \simeq_b (s', h'_{1\to 2}, l') \tag{B.1}$$

together with β-equipotence constraints ($i \in [1,q]$):

$$\mathrm{pred}_{\overline{\heartsuit}}(s, h_{1\to 2}, i) \sim_\beta \mathrm{pred}_{\overline{\heartsuit}}(s', h'_{1\to 2}, i) \tag{B.2a}$$

$$\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2}) \sim_\beta \mathrm{loop}_{\overline{\heartsuit}}(s', h'_{1\to 2}) \tag{B.2b}$$

$$\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2}) \sim_\beta \mathrm{rem}_{\overline{\heartsuit}}(s', h'_{1\to 2}) \tag{B.2c}$$

We start with basic Equivalence (B.1). We have to show that for any $\mathcal{B} \in \mathsf{Basic}^{\mathrm{u}}$, $(s, h_{1\to 2}) \models_l \mathcal{B}$ iff $(s', h'_{1\to 2}) \models_{l'} \mathcal{B}$. We prove that $(s, h_{1\to 2}) \models_l \mathcal{B}$ implies $(s', h'_{1\to 2}) \models_{l'} \mathcal{B}$. The reverse implication can be established in a symmetric way. Note that all hypotheses are symmetric: when l_1/l'_1 verify ($\mathfrak{T}2$), we have $l_1 \notin s(\mathcal{V})$ iff $l'_1 \notin s'(\mathcal{V})$. We proceed by a case analysis on $\mathcal{B}$:

$\mathcal{B}$ is $\mathtt{x}_i = \mathtt{x}_j$: using $(s,h,l) \simeq_b (s',h',l')$, we derive $(s, h_{1\to 2}) \models_l \mathtt{x}_i = \mathtt{x}_j$ iff $(s,h) \models_l \mathtt{x}_i = \mathtt{x}_j$ iff $(s',h') \models_{l'} \mathtt{x}_i = \mathtt{x}_j$ iff $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i = \mathtt{x}_j$;

$\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$: let us suppose $(s, h_{1\to 2}) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and let us show $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$. For $(s, h_{1\to 2}) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ we have two cases:
  – $h(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$. We derive $h'(s'(\mathtt{x}_i)) = s'(\mathtt{x}_j)$ (because $(s,h,l) \simeq_b (s',h',l')$) and thus we also have $h'_{1\to 2}(s'(\mathtt{x}_i)) = s'(\mathtt{x}_j)$ hence $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$;
  – $l_1 = s(\mathtt{x}_i)$ and $l_2 = s(\mathtt{x}_j)$. Since l_1/l'_1 and l_2/l'_2 verify ($\mathfrak{T}2$) in both (C1) and (C2), we get $l'_1 = s'(\mathtt{x}_i)$ and $l'_2 = s'(\mathtt{x}_j)$ and thus $h'_{1\to 2}(s'(\mathtt{x}_i)) = s'(\mathtt{x}_j)$ and finally $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$.
  In both cases we obtain $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{x}_j$;

$\mathcal{B}$ is $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$: let us suppose $(s, h_{1\to 2}) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$ and prove that $(s', h'_{1\to 2}) \models_{l'} \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$. From $h_{1\to 2} = h \uplus [l_1 \mapsto l_2]$ and $(s, h_{1\to 2}) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$, we get four cases:
  – $s(\mathtt{x}_i), s(\mathtt{x}_j) \in \mathrm{dom}(h)$ and $h(s(\mathtt{x}_i)) = h(s(\mathtt{x}_j))$. Then $(s,h) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$ from which we get $(s',h') \models_{l'} \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$ and thus also $(s', h'_{1\to 2}) \models_{l'} \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$;
  – $h(s(\mathtt{x}_i)) = l_2$ and $s(\mathtt{x}_j) = l_1$. If (C1) holds then l_2/l'_2 verify ($\mathfrak{T}3$) and l_1/l'_1 verify ($\mathfrak{T}2$) and we get $h'(s'(\mathtt{x}_i)) = l'_2$ and $s'(\mathtt{x}_j) = l'_1$. Hence $(s', h'_{1\to 2}) \models_{l'} \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$. If (C2) holds then $s(\mathtt{x}_j) = l_1$ contradicts $l_1 \notin s(\mathcal{V})$;
  – the case $s(\mathtt{x}_i) = l_1$ and $h(s(\mathtt{x}_j)) = l_2$ is symmetric to the previous one;
  – $s(\mathtt{x}_i) = s(\mathtt{x}_j) = l_1$. In case of (C1), l_1/l'_1 verify ($\mathfrak{T}2$) and thus we get $s'(\mathtt{x}_i) = s'(\mathtt{x}_j) = l'_1$ and then $(s', h'_{1\to 2}) \models_{l'} \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$. (C2) implies $l_1 \notin s(\mathcal{V})$ which contradicts $s(\mathtt{x}_i) = l_1$.
  In all four cases we have $(s', h'_{1\to 2}) \models_{l'} \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$;

$\mathcal{B}$ is $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$: let us assume $(s, h_{1\to 2}) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$ and let us prove that $(s', h'_{1\to 2}) \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$. We have four cases:
- $h(h(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$. Then $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$ from which we get $(s', h') \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$ then $(s', h'_{1\to 2}) \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$;
- $h(s(\mathtt{x}_i)) = l_1$ and $l_2 = s(\mathtt{x}_j)$. In both (C1) and (C2), l_1/l'_1 verify ($\mathfrak{T}3$) and l_2/l'_2 verify ($\mathfrak{T}2$). Hence we get $h'(s'(\mathtt{x}_i)) = l'_1$ and $l'_2 = s'(\mathtt{x}_j)$. Thus $(s', h'_{1\to 2}) \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$;
- $l_1 = s(\mathtt{x}_i)$ and $h(l_2) = s(\mathtt{x}_j)$. In (C1), l_1/l'_1 verify ($\mathfrak{T}2$) and l_2/l'_2 verify ($\mathfrak{T}5$), hence we get $l'_1 = s'(\mathtt{x}_i)$ and $h'(l'_2) = s'(\mathtt{x}_j)$, Thus $(s', h'_{1\to 2}) \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$. In case (C2), $l_1 = s(\mathtt{x}_i)$ contradicts $l_1 \notin s(\mathcal{V})$;
- $l_1 = l_2 = s(\mathtt{x}_i) = s(\mathtt{x}_j)$. l_1/l'_1 and l_2/l'_2 verify ($\mathfrak{T}2$) in both (C1) and (C2), hence we get $l'_1 = l'_2 = s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$. We deduce $(s', h'_{1\to 2}) \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$.

In all four cases we have $(s', h'_{1\to 2}) \models_{l'} \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$;

$\mathcal{B}$ is $\mathtt{toalloc}(\mathtt{x}_i)$: let us assume $(s, h_{1\to 2}) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ and let us prove $(s', h'_{1\to 2}) \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$. We get four cases:
- $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h)$. Then $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ from which we get $(s', h') \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$ then $(s', h'_{1\to 2}) \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$;
- $h(s(\mathtt{x}_i)) = l_1$. l_1/l'_1 verify ($\mathfrak{T}3$) in both (C1) and (C2), thus we get $h'(s'(\mathtt{x}_i)) = l'_1$ and thus $(s', h'_{1\to 2}) \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$;
- $l_1 = s(\mathtt{x}_i)$ and $l_2 \in \mathrm{dom}(h)$. In case (C1), l_1/l'_1 verify ($\mathfrak{T}2$) and l_2/l'_2 verify ($\mathfrak{T}4$). Then we get $l'_1 = s'(\mathtt{x}_i)$ and $l'_2 \in \mathrm{dom}(h')$ and we deduce $(s', h'_{1\to 2}) \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$. (C2) implies $l_1 \notin s(\mathcal{V})$ which contradicts $s(\mathtt{x}_i) = l_1$;
- $l_1 = l_2 = s(\mathtt{x}_i)$. l_1/l'_1 and l_2/l'_2 verify ($\mathfrak{T}2$) in both (C1) and (C2), hence $l'_1 = l'_2 = s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$. We deduce $(s', h'_{1\to 2}) \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$.

In all four cases we have $(s', h'_{1\to 2}) \models_{l'} \mathtt{toalloc}(\mathtt{x}_i)$;

$\mathcal{B}$ is $\mathtt{toloop}(\mathtt{x}_i)$: let us suppose $(s, h_{1\to 2}) \models_l \mathtt{toloop}(\mathtt{x}_i)$ and let us prove $(s', h'_{1\to 2}) \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$. We get four cases:
- $h(s(\mathtt{x}_i)) = h(h(s(\mathtt{x}_i)))$. Then $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ from which we get $(s', h') \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$ then $(s', h'_{1\to 2}) \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$;
- $h(s(\mathtt{x}_i)) = l_1 = l_2$. In both (C1) and (C2), l_1/l'_1 verify ($\mathfrak{T}3$), and $l_2 = l_1$ iff $l'_2 = l'_1$. We get $h'(s'(\mathtt{x}_i)) = l'_1$ and $l'_1 = l'_2$ and thus $(s', h'_{1\to 2}) \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$;
- $l_1 = s(\mathtt{x}_i)$ and $h(l_2) = l_2$. In case (C1), l_1/l'_1 verify ($\mathfrak{T}2$) and l_2/l'_2 verify ($\mathfrak{T}6$), hence we get $l'_1 = s'(\mathtt{x}_i)$ and $h'(l'_2) = l'_2$ and then $(s', h'_{1\to 2}) \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$ In case (C2), $l_1 = s(\mathtt{x}_i)$ contradicts $l_1 \notin s(\mathcal{V})$;
- $l_1 = l_2 = s(\mathtt{x}_i)$. l_1/l'_1 and l_2/l'_2 verify ($\mathfrak{T}2$) in both (C1) and (C2), hence $l'_1 = l'_2 = s'(\mathtt{x}_i) = s'(\mathtt{x}_j)$. We deduce $(s', h'_{1\to 2}) \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$.

In all four cases we have $(s', h'_{1\to 2}) \models_{l'} \mathtt{toloop}(\mathtt{x}_i)$;

$\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$: let us suppose $(s, h_{1\to 2}) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ and let us prove that $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$. We get two cases.
- $h(l) = l$. We derive $(s, h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ then $(s', h') \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$ and hence $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$;
- $l = l_1 = l_2$. l_1/l'_1 verify ($\mathfrak{T}1$), and $l_2 = l_1$ iff $l'_2 = l'_1$ holds in both (C1) and (C2). Hence we get $l' = l'_1$ and $l'_1 = l'_2$ and thus $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$.

In both cases we obtain $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{u}$;

$\mathcal{B}$ is $\mathtt{alloc}(\mathtt{u})$: let us assume $(s, h_{1\to 2}) \models_l \mathtt{alloc}(\mathtt{u})$ and let us prove that $(s', h'_{1\to 2}) \models_{l'} \mathtt{alloc}(\mathtt{u})$. We get two cases.
- $l \in \mathrm{dom}(h)$. We derive $(s, h) \models_l \mathtt{alloc}(\mathtt{u})$ then $(s', h') \models_{l'} \mathtt{alloc}(\mathtt{u})$ and hence $(s', h'_{1\to 2}) \models_{l'} \mathtt{alloc}(\mathtt{u})$;
- $l = l_1$. l_1/l'_1 verify ($\mathfrak{T}1$) in both (C1) and (C2), hence we get $l' = l'_1$. We deduce $(s', h'_{1\to 2}) \models_{l'} \mathtt{alloc}(\mathtt{u})$.

In both cases we obtain $(s', h'_{1\to 2}) \models_{l'} \mathtt{alloc}(\mathtt{u})$;

$\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$: using $(s, h, l) \simeq_b (s', h', l')$, we derive the equivalences $(s, h_{1\to 2}) \models_l \mathtt{x}_i = \mathtt{u}$ iff $(s, h) \models_l \mathtt{x}_i = \mathtt{u}$ iff $(s', h') \models_{l'} \mathtt{x}_i = \mathtt{u}$ iff $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i = \mathtt{u}$;

$\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$: let us suppose $(s, h_{1\to 2}) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ and let us prove that $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{u}$. We get two cases.
- $h(s(\mathtt{x}_i)) = l$. We derive $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ then $(s', h') \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{u}$ and hence $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{u}$;

– $l_1 = s(\mathtt{x}_i)$ and $l_2 = l$. In case (C1), l_1/l'_1 verify ($\mathfrak{T}2$) and l_2/l'_2 verify ($\mathfrak{T}1$), hence we get $l'_1 = s'(\mathtt{x}_i)$ and $l'_2 = l'$ and thus $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{u}$. In case (C2), $l_1 = s(\mathtt{x}_i)$ contradicts $l_1 \notin s(\mathcal{V})$.

In both cases we obtain $(s', h'_{1\to 2}) \models_{l'} \mathtt{x}_i \hookrightarrow \mathtt{u}$;

$\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_j$: let us suppose $(s, h_{1\to 2}) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_j$ and let us show $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_j$. We get two cases.

– $h(l) = s(\mathtt{x}_j)$. We derive $(s, h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_j$ then $(s', h') \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_j$ and hence $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_j$;

– $l_1 = l$ and $l_2 = s(\mathtt{x}_i)$. l_1/l'_1 verify ($\mathfrak{T}1$) and l_2/l'_2 verify ($\mathfrak{T}2$) in both (C1) and (C2), hence we get $l'_1 = l'$ and $l'_2 = s'(\mathtt{x}_i)$. We deduce $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_j$.

In both cases we obtain $(s', h'_{1\to 2}) \models_{l'} \mathtt{u} \hookrightarrow \mathtt{x}_j$.

This ends the proof of the basic equivalence $(s, h_{1\to 2}, l) \simeq_b (s', h'_{1\to 2}, l')$.

Let us consider β-Equipotence (B.2a). By Proposition 2.11, there are 3 possible values for $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2})$:

if $l_1 \notin \mathfrak{p}\heartsuit(s, h)$ and $l_2 = s(\mathtt{x}_i)$ then the identity

$$\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, i) = \mathrm{pred}_{\heartsuit}(s, h, i) \uplus \{l_1\}$$

holds. We can treat the case of (C1) and (C2) simulaneously. As l_1/l'_1 verify ($\mathfrak{T}2$–3), l_1/l'_1 also verify ($\mathfrak{T}12$) and we deduce $l'_1 \notin \mathfrak{p}\heartsuit(s', h')$. As l_2/l'_2 verify ($\mathfrak{T}2$), we get $l'_2 = s'(\mathtt{x}_i)$. Thus by Proposition 2.11 again, we have

$$\mathrm{pred}_{\heartsuit}(s', h'_{1\to 2}, i) = \mathrm{pred}_{\heartsuit}(s', h', i) \uplus \{l'_1\}.$$

From $(s, h, l) \simeq_\alpha (s', h', l)$, we have $\mathrm{pred}_{\heartsuit}(s, h, i) \sim_\alpha \mathrm{pred}_{\heartsuit}(s', h', i)$ by Proposition 2.11 and we derive Equipotence (B.2a) using Lemma 2.19 and $\beta \leqslant \alpha + 1$;

if $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{pred}_{\heartsuit}(s, h, i)$ then the identity

$$\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, i) = \mathrm{pred}_{\heartsuit}(s, h, i) - \{l_2\}$$

holds. We treat the case (C1) and (C2) separately.

On the one hand, if (C1) holds then, l_1/l'_1 verify ($\mathfrak{T}2$) and thus also ($\mathfrak{T}10$). Thus we get $l'_1 \in s'(\mathcal{V})$. Moreover, l_2/l'_2 verify ($\mathfrak{T}1$–6) and thus also ($\mathfrak{T}18$). Thus we get $l'_2 \in \mathrm{pred}_{\heartsuit}(s', h', i)$. We deduce

$$\mathrm{pred}_{\heartsuit}(s', h'_{1\to 2}, i) = \mathrm{pred}_{\heartsuit}(s', h', i) - \{l'_2\}$$

by Proposition 2.11. Since $l_1 \in s(\mathcal{V})$, we have $\beta + 1 = \alpha$. Thus by Proposition 2.20, we obtain Equipotence (B.2a);

On the other hand, (C2) contradicts $l_1 \in s(\mathcal{V})$;

in the *otherwise* case we have

$$\big(l_1 \in \mathfrak{p}\heartsuit(s, h) \text{ or } l_2 \neq s(\mathtt{x}_i)\big) \text{ and } \big(l_1 \notin s(\mathcal{V}) \text{ or } l_2 \notin \mathrm{pred}_{\heartsuit}(s, h, i)\big)$$

and $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, i) = \mathrm{pred}_{\heartsuit}(s, h, i)$. By a combination of the arguments of two previous cases, in both (C1) and (C2), we have

$$\big(l'_1 \in \mathfrak{p}\heartsuit(s', h') \text{ or } l'_2 \neq s'(\mathtt{x}_i)\big) \text{ and } \big(l'_1 \notin s'(\mathcal{V}) \text{ or } l'_2 \notin \mathrm{pred}_{\heartsuit}(s', h', i)\big)$$

and thus we get $\mathrm{pred}_{\heartsuit}(s', h'_{1\to 2}, i) = \mathrm{pred}_{\heartsuit}(s', h', i)$ by Proposition 2.11. Equipotence (B.2a) is immediate.

Let us consider β-Equipotence (B.2b). We have $\mathrm{loop}_{\heartsuit}(s, h) \sim_\alpha \mathrm{loop}_{\heartsuit}(s', h')$. By Proposition 2.11, there are three cases for the value of $\mathrm{loop}_{\heartsuit}(s, h_{1\to 2})$:

if $l_1 \notin \mathfrak{p}\heartsuit(s,h)$ and $l_1 = l_2$ then $\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \mathrm{loop}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\}$. As l_1/l_1' verify ($\mathfrak{T}$12), and $l_2 = l_1$ iff $l_2' = l_1'$, we deduce $l_1' \notin \mathfrak{p}\heartsuit(s',h')$ and $l_1' = l_2'$. Thus by Proposition 2.11, we have $\mathrm{loop}_{\overline{\heartsuit}}(s', h'_{1\to 2}) = \mathrm{loop}_{\overline{\heartsuit}}(s',h') \uplus \{l_1'\}$. Since we have $\mathrm{loop}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{loop}_{\overline{\heartsuit}}(s',h')$, we deduce Equipotence (B.2b) using Lemma 2.19 and $\beta \leqslant \alpha + 1$;

if $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$ then $\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \mathrm{loop}_{\overline{\heartsuit}}(s,h) - \{l_2\}$.
On the one hand, if (C1) holds then l_1/l_1' verify ($\mathfrak{T}$10) and l_2/l_2' verify ($\mathfrak{T}$19). Hence we get $l_1' \in s'(\mathcal{V})$ and $l_2' \in \mathrm{loop}_{\overline{\heartsuit}}(s',h')$. We deduce $\mathrm{loop}_{\overline{\heartsuit}}(s', h'_{1\to 2}) = \mathrm{loop}_{\overline{\heartsuit}}(s',h') - \{l_2'\}$. Since $l_1 \in s(\mathcal{V})$, we have $\beta + 1 = \alpha$ and thus by Proposition 2.20 we get Equipotence (B.2b).
On the other hand, (C2) contradicts $l_1 \in s(\mathcal{V})$;

in the *otherwise* case we have

$$\big(l_1 \in \mathfrak{p}\heartsuit(s,h) \text{ or } l_1 \neq l_2\big) \text{ and } \big(l_1 \notin s(\mathcal{V}) \text{ or } l_2 \notin \mathrm{loop}_{\overline{\heartsuit}}(s,h)\big)$$

and $\mathrm{loop}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \mathrm{loop}_{\overline{\heartsuit}}(s,h)$. By a combination of the arguments of two previous cases, in both (C1) and (C2), we have

$$\big(l_1' \in \mathfrak{p}\heartsuit(s',h') \text{ or } l_1' \neq l_2'\big) \text{ and } \big(l_1' \notin s'(\mathcal{V}) \text{ or } l_2' \notin \mathrm{loop}_{\overline{\heartsuit}}(s',h')\big)$$

thus $\mathrm{loop}_{\overline{\heartsuit}}(s', h'_{1\to 2}) = \mathrm{loop}_{\overline{\heartsuit}}(s',h')$. Equipotence (B.2b) is immediate.

Let us consider β-Equipotence (B.2c). We have $\mathrm{rem}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{rem}_{\overline{\heartsuit}}(s',h')$. By Proposition 2.11, there are three cases for the value of $\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2})$:

if $l_1 \notin \mathfrak{p}\heartsuit(s,h) \cup \{l_2\}$ and $l_2 \notin s(\mathcal{V})$ then $\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \mathrm{rem}_{\overline{\heartsuit}}(s,h) \uplus \{l_1\}$. As l_1/l_1' verify ($\mathfrak{T}$12), $l_2 = l_1$ iff $l_2' = l_1'$, and l_2/l_2' verify ($\mathfrak{T}$10), we deduce $l_1' \notin \mathfrak{p}\heartsuit(s',h')$, $l_1' \neq l_2'$ and $l_2' \notin s'(\mathcal{V})$. Thus by Proposition 2.11, we have $\mathrm{rem}_{\overline{\heartsuit}}(s', h'_{1\to 2}) = \mathrm{rem}_{\overline{\heartsuit}}(s',h') \uplus \{l_1'\}$. Since we have $\mathrm{rem}_{\overline{\heartsuit}}(s,h) \sim_\alpha \mathrm{rem}_{\overline{\heartsuit}}(s',h')$, we deduce Equipotence (B.2c) using Lemma 2.19 and $\beta \leqslant \alpha + 1$;

if $l_1 \in s(\mathcal{V})$ and $l_2 \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$ then $\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \mathrm{rem}_{\overline{\heartsuit}}(s,h) - \{l_2\}$.
On the one hand, if Hypothesis (C1) holds then l_1/l_1' verify ($\mathfrak{T}$10) and l_2/l_2' verify ($\mathfrak{T}$20). Hence we get $l_1' \in s'(\mathcal{V})$ and $l_2' \in \mathrm{rem}_{\overline{\heartsuit}}(s',h')$. We deduce $\mathrm{rem}_{\overline{\heartsuit}}(s', h'_{1\to 2}) = \mathrm{rem}_{\overline{\heartsuit}}(s',h') - \{l_2'\}$. Since $l_1 \in s(\mathcal{V})$, we have $\beta + 1 = \alpha$ and thus by Proposition 2.20 we get Equipotence (B.2c).
On the other hand, (C2) contradicts $l_1 \in s(\mathcal{V})$;

in the *otherwise* case we have

$$\big(l_1 \in \mathfrak{p}\heartsuit(s,h) \cup \{l_2\} \text{ or } l_2 \in s(\mathcal{V})\big) \text{ and } \big(l_1 \notin s(\mathcal{V}) \text{ or } l_2 \notin \mathrm{rem}_{\overline{\heartsuit}}(s,h)\big)$$

and $\mathrm{rem}_{\overline{\heartsuit}}(s, h_{1\to 2}) = \mathrm{rem}_{\overline{\heartsuit}}(s,h)$. By a combination of the arguments of two previous cases, in both (C1) and (C2), we have

$$\big(l_1' \in \mathfrak{p}\heartsuit(s',h') \cup \{l_2'\} \text{ or } l_2' \in s'(\mathcal{V})\big) \text{ and } \big(l_1' \notin s'(\mathcal{V}) \text{ or } l_2' \notin \mathrm{rem}_{\overline{\heartsuit}}(s',h')\big)$$

thus $\mathrm{rem}_{\overline{\heartsuit}}(s', h'_{1\to 2}) = \mathrm{rem}_{\overline{\heartsuit}}(s',h')$. Equipotence (B.2c) is immediate. □

Proposition 3.26 *Let $\mathfrak{m} = (s,h,l)$ be a pointed memory state and $l_1, l_2 \in \mathbb{N}$ be such that $l_1 \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})$. We have $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_b (s,h,l)$. Moreover, given $\alpha \geqslant 0$, if we assume that one of the following conditions hold*

(C1) $l_2 = s(\mathtt{x}_i)$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant \alpha$ for some $i \in [1,q]$;
(C2) $l_2 = l_1$ and $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$;
(C3) $l_2 \notin s(\mathcal{V}) \cup \{l_1\}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$.

then we have $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s,h,l)$.

Proof We write $h_{1\to 2}$ to denote $h \uplus [l_1 \mapsto l_2]$. First, without assuming any of (C1–3), let us prove that $(s, h_{1\to 2}, l) \simeq_b (s, h, l)$ holds. By Proposition 3.2 (monotonicity), we only need to prove that $(s, h_{1\to 2}) \models_l \mathcal{B}$ implies $(s, h) \models_l \mathcal{B}$ for any formula $\mathcal{B} \in \mathsf{Basic}^{\mathtt{u}}$. We proceed by a case analysis on $\mathcal{B}$:

$\mathcal{B}$ is $\mathtt{x}_i = \mathtt{x}_j$: This only depends on the value of s and therefore we are done;

$\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$: from $(s, h_{1\to 2}) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$, we get $h_{1\to 2}(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$. But since $l_1 \notin \mathtt{p}\heartsuit(\mathtt{m})$, we deduce $l_1 \neq s(\mathtt{x}_i)$ and thus $h(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$. We get $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$;

$\mathcal{B}$ is $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$: from $(s, h_{1\to 2}) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$, the identity $h_{1\to 2}(s(\mathtt{x}_i)) = h_{1\to 2}(s(\mathtt{x}_j))$ holds. But since $l_1 \notin \{s(\mathtt{x}_i), s(\mathtt{x}_j)\}$, we deduce $h(s(\mathtt{x}_i)) = h(s(\mathtt{x}_j))$ and thus $(s, h) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$;

$\mathcal{B}$ is $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$: from $(s, h_{1\to 2}) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$, we get $h_{1\to 2}(h_{1\to 2}(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$. But since $l_1 \notin \{s(\mathtt{x}_i), h(s(\mathtt{x}_i))\}$ (remember $h(s(\mathtt{x}_i)) \in \mathtt{p}\heartsuit(\mathtt{m})$), we get $h(h(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$ and thus $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$;

$\mathcal{B}$ is $\mathtt{toalloc}(\mathtt{x}_i)$: from $(s, h_{1\to 2}) \models_l \mathtt{toalloc}(\mathtt{x}_i)$, we deduce $h_{1\to 2}(s(\mathtt{x}_i))) \in \mathrm{dom}(h_{1\to 2})$. Since $l_1 \neq s(\mathtt{x}_i)$, we get $h(s(\mathtt{x}_i))) \in \mathrm{dom}(h) \cup \{l_1\}$. Since $h(s(\mathtt{x}_i)) \neq l_1$, we deduce $h(s(\mathtt{x}_i)) \in \mathrm{dom}(h)$ and thus $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$;

$\mathcal{B}$ is $\mathtt{toloop}(\mathtt{x}_i)$: from $(s, h_{1\to 2}) \models_l \mathtt{toloop}(\mathtt{x}_i)$ we get $h_{1\to 2}(h_{1\to 2}(s(\mathtt{x}_i))) = h_{1\to 2}(s(\mathtt{x}_i))$. But since $l_1 \notin \{s(\mathtt{x}_i), h(s(\mathtt{x}_i))\}$, we get $h(h(s(\mathtt{x}_i))) = h(s(\mathtt{x}_i))$ and thus $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$;

$\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$: from $(s, h_{1\to 2}) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ we get $h_{1\to 2}(l) = l$. But since $l_1 \neq l$, we deduce $h(l) = l$ and thus $(s, h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$;

$\mathcal{B}$ is $\mathtt{alloc}(\mathtt{u})$: from $(s, h_{1\to 2}) \models_l \mathtt{alloc}(\mathtt{u})$ we get $l \in \mathrm{dom}(h_{1\to 2})$. But since $l_1 \neq l$ and $\mathrm{dom}(h_{1\to 2}) = \mathrm{dom}(h) \cup \{l_1\}$, we deduce $l \in \mathrm{dom}(h)$ and thus $(s, h) \models_l \mathtt{alloc}(\mathtt{u})$;

$\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$: only depends on the values of s and l;

$\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$: from $(s, h_{1\to 2}) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ we get $h_{1\to 2}(s(\mathtt{x}_i)) = l$. But since $l_1 \neq s(\mathtt{x}_i)$, we deduce $h(s(\mathtt{x}_i)) = l$ and thus $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$;

$\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_j$: from $(s, h_{1\to 2}) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_j$ we get $h_{1\to 2}(l) = s(\mathtt{x}_i)$. But since $l_1 \neq l$, we deduce $h(l) = s(\mathtt{x}_i)$ and thus $(s, h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_j$.

Now we assume $\alpha \geqslant 0$ such that one of either (C1), (C2) or (C3) holds. Since we already have $(s, h_{1\to 2}, l) \simeq_b (s, h, l)$, according to Proposition 3.10, we have to establish three α-equipotence constraints:

$$\begin{aligned} \mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) &\sim_\alpha \mathrm{pred}_{\heartsuit}(s, h, j) \quad \text{for any } j \in [1, q] \\ \mathrm{loop}_{\heartsuit}(s, h_{1\to 2}) &\sim_\alpha \mathrm{loop}_{\heartsuit}(s, h) \\ \mathrm{rem}_{\heartsuit}(s, h_{1\to 2}) &\sim_\alpha \mathrm{rem}_{\heartsuit}(s, h) \end{aligned}$$

If (C1) holds then by Proposition 2.11, we have $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) = \mathrm{pred}_{\heartsuit}(s, h, j) \uplus \{l_1\}$ if $s(\mathtt{x}_i) = s(\mathtt{x}_j)$, $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) = \mathrm{pred}_{\heartsuit}(s, h, j)$ if $s(\mathtt{x}_i) \neq s(\mathtt{x}_j)$, $\mathrm{loop}_{\heartsuit}(s, h_{1\to 2}) = \mathrm{loop}_{\heartsuit}(s, h)$ and $\mathrm{rem}_{\heartsuit}(s, h_{1\to 2}) = \mathrm{rem}_{\heartsuit}(s, h)$. Then we already have $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) \sim_\alpha \mathrm{pred}_{\heartsuit}(s, h, j)$ when $s(\mathtt{x}_i) \neq s(\mathtt{x}_j)$, $\mathrm{loop}_{\heartsuit}(s, h_{1\to 2}) \sim_\alpha \mathrm{loop}_{\heartsuit}(s, h)$ and $\mathrm{rem}_{\heartsuit}(s, h_{1\to 2}) \sim_\alpha \mathrm{rem}_{\heartsuit}(s, h)$. If $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ holds then we have $\mathrm{pred}_{\heartsuit}(s, h, j) = \mathrm{pred}_{\heartsuit}(s, h, i)$. As a consequence, we get $\mathrm{card}(\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j)) \geqslant \alpha + 1$ and $\mathrm{card}(\mathrm{pred}_{\heartsuit}(s, h, j)) \geqslant \alpha$. Hence $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) \sim_\alpha \mathrm{pred}_{\heartsuit}(s, h, j)$ holds as well.

If (C2) holds then by Proposition 2.11, we have $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) = \mathrm{pred}_{\heartsuit}(s, h, j)$. Indeed, $l_2 = s(\mathtt{x}_j)$ implies $l_1 = l_2 \in s(\mathcal{V})$ which contradicts $l_1 \notin \mathrm{dom}(h) \cup \mathtt{p}\heartsuit(\mathtt{m})$. We also get $\mathrm{loop}_{\heartsuit}(s, h_{1\to 2}) = \mathrm{loop}_{\heartsuit}(s, h) \uplus \{l_1\}$ and $\mathrm{rem}_{\heartsuit}(s, h_{1\to 2}) = \mathrm{rem}_{\heartsuit}(s, h)$. The three α-equipotence constraints follow.

If (C3) holds then by Proposition 2.11, we have $\mathrm{pred}_{\heartsuit}(s, h_{1\to 2}, j) = \mathrm{pred}_{\heartsuit}(s, h, j)$, $\mathrm{loop}_{\heartsuit}(s, h_{1\to 2}) = \mathrm{loop}_{\heartsuit}(s, h)$ and $\mathrm{rem}_{\heartsuit}(s, h_{1\to 2}) = \mathrm{rem}_{\heartsuit}(s, h) \uplus \{l_1\}$. The α-equipotence constraints follow. □

Corollary 3.27 *Let $\alpha \geqslant 0$. Let $\mathtt{m} = (s, h, l)$ be a pointed memory state and h' be a heap such that $\mathrm{dom}(h') \cap (\mathrm{dom}(h) \cup \mathtt{p}\heartsuit(\mathtt{m})) = \varnothing$. If for any $u \in \mathrm{dom}(h')$ one of the following conditions holds*

(C1) $h'(u) = s(\mathtt{x}_i)$ *and* $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant \alpha$ *for some* $i \in [1,q]$*;*
(C2) $h'(u) = u$ *and* $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$*;*
(C3) $h'(u) \notin s(\mathcal{V}) \cup \{u\}$ *and* $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$.

then we have $(s, h \uplus h', l) \simeq_\alpha (s,h,l)$.

Proof We prove the result by induction on (the size of the domain of) h'. If $h' = \square$ then the result it trivial by reflexivity of $\simeq_\alpha$. Otherwise, we can write $h' = [l_1 \mapsto l_2] \uplus h''$. From $\mathrm{dom}(h') \cap (\mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})) = \varnothing$ we deduce $l_1 \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})$. We apply Proposition 3.26 to $\mathfrak{m}$, l_1 and l_2 and we get $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s,h,l)$.

We then use the induction hypothesis on h'' (with $h \uplus [l_1 \mapsto l_2]$ replacing h). Let us verify the requirements:

- We have $\mathrm{dom}(h'') \cap (\mathrm{dom}(h \uplus [l_1 \mapsto l_2]) \cup \mathfrak{p}\heartsuit(s, h \uplus [l_1 \mapsto l_2], l)) = \mathrm{dom}(h'') \cap (\mathrm{dom}(h) \cup \{l_1\} \cup \mathfrak{p}\heartsuit(s,h,l)) = \varnothing$ because $l_1 \notin \mathrm{dom}(h'')$ and $l_1 \notin s(\mathcal{V})$.
- Let $u \in \mathrm{dom}(h'')$. Let us show that either (C1), (C2) or (C3) holds for u. We have $u \in \mathrm{dom}(h')$ and $h'(u) = h''(u)$. By hypothesis, one of the following conditions holds:
  - $h'(u) = s(\mathtt{x}_i)$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant \alpha$ for some $i \in [1,q]$. From $s(\mathcal{V}) \subseteq \mathfrak{p}\heartsuit(s,h,l)$, we deduce $l_1 \notin s(\mathcal{V})$. Thus by Proposition 2.11 we have
  $$\mathrm{pred}_{\overline{\heartsuit}}(s, h \uplus [l_1 \mapsto l_2], i) \in \left\{\mathrm{pred}_{\overline{\heartsuit}}(s,h,i), \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \uplus \{l_1\}\right\}$$
  hence $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h \uplus [l_1 \mapsto l_2], i)) \geqslant \alpha$. We also have $h''(u) = h'(u) = s(\mathtt{x}_i)$ hence Condition (C1) holds;
  - $h'(u) = u$ and $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$. By Proposition 2.11 again, from $l_1 \notin s(\mathcal{V})$ we deduce $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h \uplus [l_1 \mapsto l_2])) \geqslant \alpha$. As $h''(u) = h'(u) = u$, Condition (C2) holds;
  - $h'(u) \notin s(\mathcal{V}) \cup \{u\}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant \alpha$. By Proposition 2.11 again, from $l_1 \notin s(\mathcal{V})$ we deduce $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h \uplus [l_1 \mapsto l_2])) \geqslant \alpha$. As $h''(u) = h'(u) \notin s(\mathcal{V}) \cup \{u\}$, Condition (C3) holds.

As a consequence, we obtain $(s, h \uplus [l_1 \mapsto l_2] \uplus h'', l) \simeq_\alpha (s, h \uplus [l_1 \mapsto l_2], l)$ by induction and thus $(s, h \uplus [l_1 \mapsto l_2] \uplus h'', l) \simeq_\alpha (s,h,l)$ by transitivity of $\simeq_\alpha$. □

Proposition 3.29 *Let* $\alpha \geqslant 1$*. We assume that the following conditions hold:*

(a) $\mathfrak{m} \simeq_{\alpha+1} \mathfrak{m}'$*;*
(b) $\mathfrak{m}_0 \simeq_{\alpha+1} \mathfrak{m}'_0$*;*
(c) $\mathrm{dom}(h) \subseteq \mathfrak{p}\heartsuit(\mathfrak{m})$*;*
(d) $\mathrm{dom}(h') \subseteq \mathfrak{p}\heartsuit(\mathfrak{m}')$.

Let $l_1 \in s(\mathcal{V}) \backslash \mathrm{dom}(h_0 \uplus h)$ *and* $l_2 \in \mathbb{N}$*. There exist* $l'_1, l'_2 \in \mathbb{N}$ *such that*

1. $l'_1 \in s'(\mathcal{V}) \backslash \mathrm{dom}(h'_0 \uplus h')$*;*
2. $l'_1, l'_2 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 1$*;*
3. $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h' \uplus [l'_1 \mapsto l'_2], l')$*;*
4. $(s, h_0 \uplus h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h'_0 \uplus h' \uplus [l'_1 \mapsto l'_2], l')$.

Proof According to Lemma 3.19, we have both $\mathfrak{R}^{\mathsf{I}} \subseteq \mathfrak{T}^{\mathsf{I}}$ and $\mathfrak{R}^{\mathsf{I}}_0 \subseteq \mathfrak{T}^{\mathsf{I}}_0$.

Let use define l'_1 to be the unique value such that $l_1 \; \mathfrak{R}^{\mathsf{I}} \; l'_1$. We also have $l_1 \; \mathfrak{R}^{\mathsf{I}}_0 \; l'_1$. Hence we get both $l_1 \; \mathfrak{T}^{\mathsf{I}} \; l'_1$ and $l_1 \; \mathfrak{T}^{\mathsf{I}}_0 \; l'_1$. From $l_1 \in \mathfrak{p}\heartsuit(\mathfrak{m})$ and $l_1 \; \mathfrak{T}^{\mathsf{I}} \; l'_1$ we deduce $l'_1 \in \mathfrak{p}\heartsuit(\mathfrak{m}')$. Since $l_1 \in s(\mathcal{V})$ and $l_1 \notin \mathrm{dom}(h_0 \uplus h)$ from $l_1 \; \mathfrak{T}^{\mathsf{I}}_0 \; l'_1$ we deduce $l'_1 \in s'(\mathcal{V}) \backslash \mathrm{dom}(h'_0 \uplus h')$ by ($\mathfrak{T}$10) and ($\mathfrak{T}$4). Hence Property 1 holds.

Let us define l'_2 by Proposition 3.20: since $\alpha \geqslant 1$, we have $\mathfrak{m}_0 \simeq_2 \mathfrak{m}'_0$, and thus there exists $l'_2 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 1$ such that $l_2 \; \mathfrak{T}^{\mathsf{I}}_0 \; l'_2$. Property 2 holds because $l'_1 \in s'(\mathcal{V})$ and $l'_2 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 1$.

Let us establish Property 4, i.e. $(s, h_0 \uplus h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h'_0 \uplus h' \uplus [l'_1 \mapsto l'_2], l')$. We use Lemma 3.25 (C1) with $\mathfrak{m}_0/\mathfrak{m}'_0$. We have both $l_1 \notin \mathrm{dom}(h_0 \uplus h)$ and $l'_1 \notin \mathrm{dom}(h'_0 \uplus h')$. l_1/l'_1 verify ($\mathfrak{T}$1–3) because $l_1 \; \mathfrak{T}^{\mathsf{I}}_0 \; l'_1$ holds. l_2/l'_2 verify ($\mathfrak{T}$1–6) because $l_2 \; \mathfrak{T}^{\mathsf{I}}_0 \; l'_2$ holds. Let us check $l_1 = l_2$ iff $l'_1 = l'_2$:

– if $l_2 \in \mathfrak{p}\heartsuit(\mathfrak{m}_0)$ then $l_2\ \mathfrak{R}^{\mathsf{l}}_0\ l'_2$ by Proposition 3.15 item 5. Since $l_1\ \mathfrak{R}^{\mathsf{l}}_0\ l'_1$, $l_2\ \mathfrak{R}^{\mathsf{l}}_0\ l'_2$ and $\mathfrak{R}^{\mathsf{l}}_0$ is a bijection (Lemma 3.19), we deduce $l_1 = l_2$ iff $l'_1 = l'_2$;
– if $l_2 \notin \mathfrak{p}\heartsuit(\mathfrak{m}_0)$ then $l'_2 \notin \mathfrak{p}\heartsuit(\mathfrak{m}'_0)$ by ($\mathfrak{T}$21) with $l_2\ \mathfrak{T}^{\mathsf{l}}_0\ l'_2$. But $l_1 \in s(\mathcal{V})$ and $l'_1 \in s'(\mathcal{V})$ hence $l_1 \neq l_2$ and $l'_1 \neq l'_2$ and we deduce $l_1 = l_2$ iff $l'_1 = l'_2$.

We apply Lemma 3.25 (C1) with Hypothesis (b) and we get Property 4.

Let us show Property 3, i.e. $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h' \uplus [l'_1 \mapsto l'_2], l')$. We use Lemma 3.25 (C1) with $\mathfrak{m}/\mathfrak{m}'$. Since $l_1 \notin \mathrm{dom}(h_0 \uplus h)$ and $l'_1 \notin \mathrm{dom}(h'_0 \uplus h')$ then $l_1 \notin \mathrm{dom}(h)$ and $l'_1 \notin \mathrm{dom}(h')$. l_1/l'_1 verify ($\mathfrak{T}$1–3) because $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$ holds. We already verified that $l_1 = l_2$ iff $l'_1 = l'_2$ holds. Let us check that l_2/l'_2 verify ($\mathfrak{T}$1–6), i.e. $l_2\ \mathfrak{T}^{\mathsf{l}}\ l'_2$:
– if $l_2 \in \mathfrak{p}\heartsuit(\mathfrak{m})$ then, as $l_2\ \mathfrak{T}^{\mathsf{l}}_0\ l'_2$ holds, by Proposition 3.28, we get $l_2\ \mathfrak{R}^{\mathsf{l}}\ l'_2$ and thus $l_2\ \mathfrak{T}^{\mathsf{l}}\ l'_2$;
– if $l_2 \notin \mathfrak{p}\heartsuit(\mathfrak{m})$ then we must have $l'_2 \notin \mathfrak{p}\heartsuit(\mathfrak{m}')$: otherwise if $l'_2 \in \mathfrak{p}\heartsuit(\mathfrak{m}')$ holds then we would have $l_2\ \mathfrak{R}^{\mathsf{l}}\ l'_2$ by Proposition 3.28, which contradicts $l_2 \notin \mathfrak{p}\heartsuit(\mathfrak{m})$. Hence by Hypotheses (c) and (d) we deduce $l_2 \notin \mathrm{dom}(h)$ and $l'_2 \notin \mathrm{dom}(h')$. By Proposition 3.16 item 4, we deduce $l_2\ \mathfrak{T}^{\mathsf{l}}\ l'_2$.

We apply Lemma 3.25 (C1) with Hypothesis (a) and we get Property 3. □

Proposition 3.31 *Let $\alpha \geqslant 1$. We assume that the following conditions hold:*
(a) $\mathfrak{m} \simeq_\alpha \mathfrak{m}'$;
(b) $\mathfrak{m}_0 \simeq_\alpha \mathfrak{m}'_0$.
Let $l_1 \notin \mathrm{dom}(h_0 \uplus h) \cup s(\mathcal{V})$ and $l_2 \in \mathbb{N}$. There exist $l'_1, l'_2 \in \mathbb{N}$ such that
1. $l'_1 \notin \mathrm{dom}(h'_0 \uplus h') \cup s'(\mathcal{V})$
2. $l'_1, l'_2 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 2$;
3. $(s, h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h' \uplus [l'_1 \mapsto l'_2], l')$;
4. $(s, h_0 \uplus h \uplus [l_1 \mapsto l_2], l) \simeq_\alpha (s', h'_0 \uplus h' \uplus [l'_1 \mapsto l'_2], l')$.

Proof According to Lemma 3.19, we have both $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$ and $\mathfrak{R}^{\mathsf{l}}_0 \subseteq \mathfrak{T}^{\mathsf{l}}_0$.

Let us define l'_1 and simultaneously check Property 1 and prove that $l'_1 \leqslant \mathrm{maxval}(\mathfrak{m}'_0)+1$, $l_1\ \mathfrak{T}^{\mathsf{l}}_0\ l'_1$ and $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$ hold:
– if $l_1 \in \mathfrak{p}\heartsuit(\mathfrak{m}_0)$ then let us define $l'_1 \in \mathfrak{p}\heartsuit(\mathfrak{m}'_0)$ as the unique value such that $l_1\ \mathfrak{R}^{\mathsf{l}}_0\ l'_1$. We immediately deduce $l_1\ \mathfrak{T}^{\mathsf{l}}_0\ l'_1$. As a consequence, $l'_1 \notin \mathrm{dom}(h'_0 \uplus h')$ by ($\mathfrak{T}$4) and $l'_1 \notin s'(\mathcal{V})$ by ($\mathfrak{T}$10). Hence Property 1 holds. As $l'_1 \in \mathfrak{p}\heartsuit(\mathfrak{m}'_0)$, the relation $l'_1 \leqslant \mathrm{maxval}(\mathfrak{m}'_0)+1$ holds trivially. Only $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$ remains. We use Proposition 3.28: if $l_1 \in \mathfrak{p}\heartsuit(\mathfrak{m})$ or $l'_1 \in \mathfrak{p}\heartsuit(\mathfrak{m}')$ then $l_1\ \mathfrak{R}^{\mathsf{l}}\ l'_1$, hence $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$; otherwise both $l_1 \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})$ and $l'_1 \notin \mathrm{dom}(h') \cup \mathfrak{p}\heartsuit(\mathfrak{m}')$ hold and we deduce $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$ by Proposition 3.16 item 4;
– if $l_1 \notin \mathfrak{p}\heartsuit(\mathfrak{m}_0)$ then we define $l'_1 = \mathrm{maxval}(\mathfrak{m}'_0) + 1$ and Property 1 holds in an obvious way. We also have $l_1 \notin \mathrm{dom}(h_0 \uplus h) \cup \mathfrak{p}\heartsuit(\mathfrak{m}_0)$ and $l'_1 \notin \mathrm{dom}(h'_0 \uplus h') \cup \mathfrak{p}\heartsuit(\mathfrak{m}'_0)$ and we deduce $l_1\ \mathfrak{T}^{\mathsf{l}}_0\ l'_1$ by Proposition 3.16 item 4. A fortiori we have $l_1 \notin \mathrm{dom}(h) \cup \mathfrak{p}\heartsuit(\mathfrak{m})$ and $l'_1 \notin \mathrm{dom}(h') \cup \mathfrak{p}\heartsuit(\mathfrak{m}')$ and we deduce $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$ by Proposition 3.16 item 4.

From $l'_1 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 1$, we obviously derive Property 2 for l'_1.

Let us define l'_2 by choosing the first possible choice in the following list. We simultaneously check Property 2 for l'_2 and prove that l_2/l'_2 verify ($\mathfrak{T}$2), and that $l_2 = l_1$ iff $l'_2 = l'_1$ holds:
– if $l_2 = l_1$ then we define $l'_2 = l'_1$. In case, Property 2 obviously holds for l'_2 since it holds for l'_1. Since $l_1\ \mathfrak{T}^{\mathsf{l}}_0\ l'_1$ holds then l_1/l'_1 verify ($\mathfrak{T}$2), and thus l_2/l'_2 verify ($\mathfrak{T}$2). Since $l_2 = l_1$ and $l'_2 = l'_1$, the property $l_2 = l_1$ iff $l'_2 = l'_1$ holds;
– if $l_2 \in s(\mathcal{V})$ then we define l'_2 to be the unique location such that $l_2\ \mathfrak{R}^{\mathsf{l}}\ l'_2$. Then $l'_2 \in \mathfrak{p}\heartsuit(\mathfrak{m}') \subseteq \mathfrak{p}\heartsuit(\mathfrak{m}'_0)$ and as a consequence, Property 2 holds for l'_2. From $\mathfrak{R}^{\mathsf{l}} \subseteq \mathfrak{T}^{\mathsf{l}}$ we deduce $l_2\ \mathfrak{T}^{\mathsf{l}}\ l'_2$ and as a consequence, l_2/l'_2 verify ($\mathfrak{T}$2). We have $l_1 \notin s(\mathcal{V})$ hence we deduce $l'_1 \notin s'(\mathcal{V})$ using $l_1\ \mathfrak{T}^{\mathsf{l}}\ l'_1$ and ($\mathfrak{T}$10). We have $l_2 \in s(\mathcal{V})$ hence we deduce $l'_2 \in s'(\mathcal{V})$ using $l_2\ \mathfrak{T}^{\mathsf{l}}\ l'_2$ and ($\mathfrak{T}$10). We derive both $l_1 \neq l_2$ and $l'_1 \neq l'_2$. Thus the property $l_2 = l_1$ iff $l'_2 = l'_1$ holds;
– otherwise we have $l_2 \notin s(\mathcal{V})$ and $l_1 \neq l_2$ an we define $l'_2 = \mathrm{maxval}(\mathfrak{m}'_0) + 2$. Hence Property 2 holds for l'_2. Moreover, as $l'_1 \leqslant \mathrm{maxval}(\mathfrak{m}'_0) + 1$, we deduce $l'_1 \neq l'_2$. Thus the property $l_2 = l_1$ iff $l'_2 = l'_1$ holds. Finally we have $l_2 \notin s(\mathcal{V})$ and $l'_2 \notin s'(\mathcal{V})$ (because $l'_2 > \mathrm{maxval}(\mathfrak{p}\heartsuit(m'_0))$). Hence l_2/l'_2 verify ($\mathfrak{T}$2).

We apply Lemma 3.25 (C2) with Hypothesis (a) and (b) and we get Property 3 and 4. □

C Proofs of Section 4

Corollary 4.12 *1SL2 is strictly more expressive than 1SL1.*

Proof Let $\mathcal{A}$ be the sentence in 1SL2 that states that there is a path of length 3 between $\mathtt{x}_1$ and $\mathtt{x}_2$ in the memory state and nothing else, for instance

$$\mathcal{A} \stackrel{def}{=} \mathtt{x}_1 \neq \mathtt{x}_2 \wedge \mathcal{C} \wedge \neg(\mathcal{C} * \neg\mathtt{emp})$$

with

$$\mathcal{C} \stackrel{def}{=} \exists \mathtt{u}_1, \mathtt{u}_2 \left(\begin{array}{c} \mathtt{u}_1 \neq \mathtt{u}_2 \wedge \mathtt{u}_1 \neq \mathtt{x}_1 \wedge \mathtt{u}_1 \neq \mathtt{x}_2 \wedge \mathtt{u}_2 \neq \mathtt{x}_1 \wedge \mathtt{u}_2 \neq \mathtt{x}_2 \\ \wedge\ \mathtt{x}_1 \hookrightarrow \mathtt{u}_1 \wedge \mathtt{u}_1 \hookrightarrow \mathtt{u}_2 \wedge \mathtt{u}_2 \hookrightarrow \mathtt{x}_2 \end{array} \right).$$

Suppose that there is a sentence $\mathcal{A}'$ in 1SL1 whose models are precisely the memory states defined by $\mathcal{A}$. Let us show that this leads to a contradiction.

Let $q \geqslant 1$ be such that $\mathtt{x}_1, \ldots, \mathtt{x}_q$ contains the program variables that occur in $\mathcal{A}'$. By Theorem 4.11, there is a Boolean combination $\mathcal{A}''$ of test formulae from $\mathsf{Test}^{\mathtt{u}}_{\alpha}$ such that $\mathcal{A}'$ and $\mathcal{A}''$ are equivalent, where with $\alpha = \mathsf{th}(q, \mathcal{A}')$. Let s be the store with $s(\mathtt{x}_1) = 0$ and $s(\mathtt{x}_2) = 3$. Let h_1 be the heap such that $h_1(0) = 1$, $h_1(1) = 2$ and $h_1(2) = 3$. Similarly, let h_2 be the heap such that $h_2(0) = 1$, $h_2(1) = 2$ and $h_2(4) = 3$. And let $l = 0$ for instance (any other value would fit). We note that $(s, h_1) \models_l \mathcal{A}$ and therefore $(s, h_1) \models_l \mathcal{A}'$ by assumption. Similarly, $(s, h_2) \not\models_l \mathcal{A}$ and therefore $(s, h_2) \not\models_l \mathcal{A}'$ by assumption.

Since $\mathcal{A}'$ and $\mathcal{A}''$ are two logically equivalent formulæ of 1SL1, we deduce that both $(s, h_1) \models_l \mathcal{A}''$ and $(s, h_2) \not\models_l \mathcal{A}''$ hold. However, it is worth noting that for every test formula $\mathcal{B}$ from $\mathsf{Test}^{\mathtt{u}}_{\alpha}$, we have $(s, h_1) \models_l \mathcal{B}$ iff $(s, h_2) \models_l \mathcal{B}$, which leads to a contradiction because $\mathcal{A}''$ is a Boolean combination of formulae from $\mathsf{Test}^{\mathtt{u}}_{\alpha}$. □

Lemma 4.15 *Let $q \geqslant 1$ and $m \in \mathbb{N}$. Let $\mathcal{A}$ be a 1SL1 formula with program variables in $\mathtt{x}_1, \ldots, \mathtt{x}_q$ and (s, h, l) be a pointed memory state. If we assume* $\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ *then*

$$\mathtt{bmc}\big(q, m, \mathcal{A}, (s, h, l)\big) = \mathtt{tt} \quad \textit{iff} \quad (s, h) \models_l \mathcal{A}.$$

Proof We proceed by induction on $\mathcal{A}$ and we prove the double implication, assuming that $\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ holds:

if $\mathcal{A}$ is atomic then $\mathtt{bmc}\big(q, m, \mathcal{A}, (s, h, l)\big) = \mathtt{amc}\big(\mathcal{A}, (s, h, l)\big)$. The correctness of $\mathtt{amc}$ is obvious and left to the reader;

if $\mathcal{A}$ is $\neg\mathcal{A}_1$ then

$$\mathtt{bmc}\big(q, m, \mathcal{A}, (s, h, l)\big) = \mathtt{not}\ \mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h, l)\big).$$

We deduce the equivalences $\mathtt{bmc}\big(q, m, \mathcal{A}, (s, h, l)\big) = \mathtt{tt}$ iff $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h, l)\big) \neq \mathtt{tt}$ iff $(s, h) \not\models_l \mathcal{A}_1$ iff $(s, h) \models_l \mathcal{A}$ using the induction hypothesis;

if $\mathcal{A}$ is $\mathcal{A}_1 \wedge \mathcal{A}_2$ then

$$\mathtt{bmc}\big(q, m, \mathcal{A}, (s, h, l)\big) = \mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h, l)\big)\ \mathtt{and}\ \mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h, l)\big).$$

We deduce the equivalences $\mathtt{bmc}\big(q, m, \mathcal{A}, (s, h, l)\big) = \mathtt{tt}$ iff $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h, l)\big) = \mathtt{tt}$ and $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$ iff $(s, h) \models_l \mathcal{A}_1$ and $(s, h) \models_l \mathcal{A}_2$ iff $(s, h) \models_l \mathcal{A}$ using the induction hypotheses;

if $\mathcal{A}$ is $\exists \mathtt{u}\, \mathcal{A}_1$, let us assume $\mathtt{bmc}\big(q, m, \exists \mathtt{u}\, \mathcal{A}_1, (s, h, l)\big) = \mathtt{tt}$ and prove $(s, h) \models_l \exists \mathtt{u}\, \mathcal{A}_1$. By definition, there exists $l_0 \leqslant m$ such that $l_0 + \varphi_q(\mathcal{A}_1) \leqslant m$ and $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h, l_0)\big) = \mathtt{tt}$. We have $\mathrm{maxval}(s, h) + \varphi_q(\mathcal{A}_1) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ by Proposition 4.13 item 1. Hence we deduce $\mathrm{maxval}(s, h, l_0) + \varphi_q(\mathcal{A}_1) \leqslant m$. By induction hypothesis we get $(s, h) \models_{l_0} \mathcal{A}_1$ and thus $(s, h) \models_l \exists \mathtt{u}\, \mathcal{A}_1$.
Now let us assume $(s, h) \models_l \exists \mathtt{u}\, \mathcal{A}_1$ and prove $\mathtt{bmc}\big(q, m, \exists \mathtt{u}\, \mathcal{A}_1, (s, h, l)\big) = \mathtt{tt}$. By Corollary 4.5, there exists $l_0 \leqslant \mathrm{maxval}(s, h) + 1$ s.t. $(s, h) \models_{l_0} \mathcal{A}_1$. We have $\mathrm{maxval}(s, h) + 1 + \varphi_q(\mathcal{A}_1) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ by Proposition 4.13 item 1. Hence we get both $l_0 + \varphi_q(\mathcal{A}_1) \leqslant m$ and $\mathrm{maxval}(s, h) + \varphi_q(\mathcal{A}_1) \leqslant m$ and we deduce $\mathrm{maxval}(s, h, l_0) + \varphi_q(\mathcal{A}_1) \leqslant m$. By induction we derive $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h, l_0)\big) = \mathtt{tt}$. By definition of $\mathtt{bmc}$, we conclude $\mathtt{bmc}\big(q, m, \exists \mathtt{u}\, \mathcal{A}_1, (s, h, l)\big) = \mathtt{tt}$;

if $\mathcal{A}$ is $\mathcal{A}_1 * \mathcal{A}_2$, let us assume $\mathtt{bmc}\big(q, m, \mathcal{A}_1 * \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$ and let us prove $(s, h) \models_l \mathcal{A}_1 * \mathcal{A}_2$. By definition of $\mathtt{bmc}$, there exists a heap $h_1 : [0, m] \rightharpoonup [0, m]$ such that $\mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m$ and $h_1 \sqsubseteq h$ and $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{tt}$ and $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h_2, l)\big) = \mathtt{tt}$ with $h_2 = h - h_1$. For each $c \in \{1, 2\}$, we have $\mathrm{maxval}(s, h_c, l) + \varphi_q(\mathcal{A}_c) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ by Proposition 4.13 item 4. Hence by induction hypotheses, we deduce $(s, h_1) \models_l \mathcal{A}_1$ and $(s, h_2) \models_l \mathcal{A}_2$. Given that the identity $h = h_1 \uplus h_2$ holds, we get $(s, h) \models_l \mathcal{A}_1 * \mathcal{A}_2$.
Now let us assume $(s, h) \models_l \mathcal{A}_1 * \mathcal{A}_2$ and prove $\mathtt{bmc}\big(q, m, \mathcal{A}_1 * \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$. There exists h_1 and h_2 such that $h = h_1 \uplus h_2$, $(s, h_1) \models_l \mathcal{A}_1$ and $(s, h_2) \models_l \mathcal{A}_2$. For each $c \in \{1, 2\}$, from $h_c \sqsubseteq h$ we deduce $\mathrm{maxval}(s, h_c, l) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ by Proposition 4.13 item 4. Hence we have the identities $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{tt}$ and $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h_2, l)\big) = \mathtt{tt}$ by induction hypothesis. Moreover, we have $\mathtt{subheap}(h_1, h) = \mathtt{tt}$ and $\mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m$ holds. As $h_2 = h - h_1$, by definition of $\mathtt{bmc}$, we get $\mathtt{bmc}\big(q, m, \mathcal{A}_1 * \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$;

if $\mathcal{A}$ is $\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$, let us assume $\mathtt{bmc}\big(q, m, \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$ and prove $(s, h) \models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$. For this, we use Corollary 4.6. Let us consider $h_1 \perp h$ such that $\mathrm{maxval}(h_1) \leqslant \mathrm{maxval}(s, h, l) + 15|\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2|q^2$ and $(s, h_1) \models_l \mathcal{A}_1$ and prove $(s, h \uplus h_1) \models_l \mathcal{A}_2$. We have

$$\begin{aligned} &\mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \\ \leqslant\ &\mathrm{maxval}(s, h, l) + 15|\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2|q^2 + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \\ \leqslant\ &\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2) \leqslant m \end{aligned}$$

by Proposition 4.13 item 5. Let us prove the identity $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{tt}$. We have $\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}_1) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ by Proposition 4.13 item 5. We also have $\mathrm{maxval}(h_1) + \varphi_q(\mathcal{A}_1) \leqslant \mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m$. Hence we get $\mathrm{maxval}(s, h_1, l) + \varphi_q(\mathcal{A}_1) \leqslant m$ and by induction hypothesis, from $(s, h_1) \models_l \mathcal{A}_1$, we get $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{tt}$. Since $\mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m$ holds, by definition of $\mathtt{bmc}$ we have $h_1 \perp h$ and $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{tt}$. Hence, to satisfy $\mathtt{bmc}\big(q, m, \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$, we must have $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h \uplus h_1, l)\big) = \mathtt{tt}$. Then $\mathrm{maxval}(s, h_1, l) + \varphi_q(\mathcal{A}_2) \leqslant \mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m$ holds and $\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}_2) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ holds by Proposition 4.13 item 5. We deduce $\mathrm{maxval}(s, h \uplus h_1, l) + \varphi_q(\mathcal{A}_2) \leqslant m$ and thus we have $(s, h \uplus h_1) \models_l \mathcal{A}_2$ by induction hypothesis.
Now let us assume $(s, h) \models_l \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2$ and prove $\mathtt{bmc}\big(q, m, \mathcal{A}_1 \mathrel{-\!\!*} \mathcal{A}_2, (s, h, l)\big) = \mathtt{tt}$. By definition of $\mathtt{bmc}$, we pick $h_1 : [0, m] \rightharpoonup [0, m]$ and we verify that either $\mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) > m$ or $h_1 \perp h$ does not hold or $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{ff}$ or $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h \uplus h_1, l)\big) = \mathtt{tt}$. So we assume $h_1 \perp h$ and

$$\mathrm{maxval}(h_1) + \max(\varphi_q(\mathcal{A}_1), \varphi_q(\mathcal{A}_2)) \leqslant m$$

and $\mathtt{bmc}\big(q, m, \mathcal{A}_1, (s, h_1, l)\big) = \mathtt{tt}$ and we prove the identity $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h \uplus h_1, l)\big) = \mathtt{tt}$. We have $\mathrm{maxval}(h_1) + \varphi_q(\mathcal{A}_1) \leqslant m$ and

$$\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}_1) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$$

by Proposition 4.13 item 5. Hence we derive $\mathrm{maxval}(s, h_1, l) + \varphi_q(\mathcal{A}_1) \leqslant m$ thus by induction hypothesis we get $(s, h_1) \models_l \mathcal{A}_1$. As $h_1 \perp h$ we deduce $(s, h \uplus h_1) \models_l \mathcal{A}_2$. We have $\mathrm{maxval}(h_1) + \varphi_q(\mathcal{A}_2) \leqslant m$ and $\mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}_2) \leqslant \mathrm{maxval}(s, h, l) + \varphi_q(\mathcal{A}) \leqslant m$ by Proposition 4.13 item 5. Hence we derive $\mathrm{maxval}(s, h \uplus h_1, l) + \varphi_q(\mathcal{A}_2) \leqslant m$ and thus by induction hypothesis we get $\mathtt{bmc}\big(q, m, \mathcal{A}_2, (s, h \uplus h_1, l)\big) = \mathtt{tt}$. □

D Proofs of Section 5

Proposition 5.3 (Completeness of the saturation rules) *If the (finite) subset* $\mathsf{P} \subseteq \mathsf{Basic}^{\mathsf{u}}$ *is closed under the rules of Figure 5.1 and* (s, H, L) *is the canonical pre-model of* P *then:*

- *s is a total function $s : \mathcal{V} \to [1, q]$, hence s is a store;*
- *H is a finite and functional graph, hence H is the graph of some heap h;*
- *L is a singleton subset of $\mathbb{N}$, i.e. $L = \{l\}$ for some location l;*
- *the inclusion $\mathrm{dom}(h) \subseteq \heartsuit(s, h) \cup \{l\}$ holds;*
- *for any formula $\mathcal{B} \in \mathsf{Basic}^{\mathtt{u}}$ we have $(s, h) \models_l \mathcal{B}$ iff $\mathcal{B} \in \mathsf{P}$.*

Proof Since P is closed under the three rules

$$\frac{}{\mathtt{x} = \mathtt{x}} \qquad \frac{\mathtt{x} = \mathtt{y}}{\mathtt{y} = \mathtt{x}} \qquad \frac{\mathtt{x} = \mathtt{y} \quad \mathtt{y} = \mathtt{z}}{\mathtt{x} = \mathtt{z}}$$

the relation $\{(\mathtt{x}, \mathtt{y}) \mid \mathtt{x} = \mathtt{y} \in \mathsf{P}\}$ is an equivalence relation. Hence the function s is total: indeed $\mathtt{x}_i = \mathtt{x}_i \in \mathsf{P}$ and the set $\{j \mid \mathtt{x}_i = \mathtt{x}_j \in \mathsf{P}\}$ contains at least i. Hence $s(\mathtt{x}_i)$ is always defined and we have $\mathtt{x}_i = \mathtt{x}_{s(\mathtt{x}_i)} \in \mathsf{P}$. Moreover we have

$$s(\mathtt{x}_i) = s(\mathtt{x}_j) \quad \text{iff} \quad \mathtt{x}_i = \mathtt{x}_j \in \mathsf{P} \quad \text{for all } i, j \in [1, q] \tag{D.1}$$

Since P is closed under the two rules

$$\frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)}{\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_i)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \quad \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_k)}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k)}$$

the relation $\{(i, j) \mid \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}\}$ is a partial equivalence relation and we have

$$\mathfrak{h}_i, \mathfrak{h}_j \text{ are both defined and } \mathfrak{h}_i = \mathfrak{h}_j \quad \text{iff} \quad \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P} \quad \text{for all } i, j \in [1, q] \tag{D.2}$$

It is obvious that H is a finite graph. An important remark for the rest of the proof is the following: by construction we have

$$\{0\} \uplus \{s(\mathtt{x}_i) \mid i \in [1, q]\} \uplus \{\mathfrak{h}_i \mid i \in [1, q] \text{ and } \mathfrak{h}_i \text{ is defined}\} \uplus \{2q + 1\} \subseteq \mathbb{N} \tag{D.3}$$

i.e. these sets are mutually disjoint. Let $u, w \in \mathbb{N}$ be such that $(u, w) \in H$ and let us check the following characteristic properties of the graph H:

P1 one of the three following properties holds:
- either $u = s(\mathtt{x}_i)$ for some $i \in [1, q]$;
- or $u = \mathfrak{h}_i$ for some i such that $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$;
- or $u = 0$;

P2 if $u = s(\mathtt{x}_i)$ then
- either $w = s(\mathtt{x}_j)$ and $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ for some $j \in [1, q]$;
- or $w = \mathfrak{h}_i$, $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$;

P3 if $\mathfrak{h}_i$ is defined and $u = \mathfrak{h}_i$ then $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$, $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$ and:
- either $w = s(\mathtt{x}_j)$ and $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ for some $j \in [1, q]$;
- or $w = \mathfrak{h}_i$ and $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$;
- or $w = 0$, $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q), \mathtt{toloop}(\mathtt{x}_i)\} \cap \mathsf{P} = \varnothing$;

P4 if $u = 0$ then $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$ and:
- either $w = s(\mathtt{x}_i)$ and $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ for some $i \in [1, q]$;
- or $w = 0$, $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$;
- or $w = 2q + 1$, $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$ and $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q, \mathtt{u} \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$.

We prove Properties P1 to P4 in that order:

- Property P1 holds by definition of H. We just have to check that when $u = \mathfrak{h}_i$ then $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ but this is a consequence of Equivalence (D.2);
- let us check Property P2. By definition of H and Property (D.3), there are two possibilities for $(s(\mathtt{x}_i), w) \in H$:
  - either $(s(\mathtt{x}_i), w) = (s(\mathtt{x}_k), s(\mathtt{x}_j))$ with $\mathtt{x}_k \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. But from $s(\mathtt{x}_k) = s(\mathtt{x}_i)$ we deduce $\mathtt{x}_k = \mathtt{x}_i \in \mathsf{P}$ by Equivalence (D.1). As P is closed under the rule

$$\frac{\mathtt{x}_k = \mathtt{x}_i \quad \mathtt{x}_k \hookrightarrow \mathtt{x}_j}{\mathtt{x}_i \hookrightarrow \mathtt{x}_j}$$

  we deduce $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ and $v = s(\mathtt{x}_j)$;

– or $(s(\mathtt{x}_i), w) = (s(\mathtt{x}_k), \mathfrak{h}_j)$ with $\mathtt{conv}(\mathtt{x}_k, \mathtt{x}_j) \in \mathsf{P}$ and $\{\mathtt{x}_k \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_k \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. From $s(\mathtt{x}_i) = s(\mathtt{x}_k)$ we deduce $\{\mathtt{x}_k = \mathtt{x}_i, \mathtt{x}_i = \mathtt{x}_k\} \subseteq \mathsf{P}$ by Equivalence (D.1). But P is closed under the rule

$$\frac{\mathtt{x}_k = \mathtt{x}_i \quad \mathtt{conv}(\mathtt{x}_k, \mathtt{x}_j)}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)}{\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_i)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \quad \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_i)}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)} \qquad \frac{\mathtt{x}_i = \mathtt{x}_k \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_p}{\mathtt{x}_k \hookrightarrow \mathtt{x}_p}$$

hence we deduce $\{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j), \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)\} \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. We conclude $w = \mathfrak{h}_j = \mathfrak{h}_i$ using Equivalence (D.2);

– let us check Property P3. By definition of H and Property (D.3), there are three possibilities for $(\mathfrak{h}_i, w) \in H$:
  – either $(\mathfrak{h}_i, w) = (\mathfrak{h}_k, s(\mathtt{x}_j))$ with $\mathtt{btwn}(\mathtt{x}_k, \mathtt{x}_j) \in \mathsf{P}$ and $\{\mathtt{x}_k \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_k \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. We deduce $\mathfrak{h}_i = \mathfrak{h}_k$ and $w = s(\mathtt{x}_j)$. Thus $\{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k), \mathtt{conv}(\mathtt{x}_k, \mathtt{x}_i), \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)\} \subseteq \mathsf{P}$ by Equivalence (D.2). Since P is closed under the rules

$$\frac{\mathtt{conv}(\mathtt{x}_k, \mathtt{x}_i) \quad \mathtt{btwn}(\mathtt{x}_k, \mathtt{x}_j)}{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k) \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_p}{\mathtt{x}_k \hookrightarrow \mathtt{x}_p}$$

we deduce $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$;
  – $(\mathfrak{h}_i, w) = (\mathfrak{h}_k, \mathfrak{h}_j)$ with $\{\mathtt{conv}(\mathtt{x}_k, \mathtt{x}_j), \mathtt{toloop}(\mathtt{x}_k)\} \subseteq \mathsf{P}$ and $\{\mathtt{x}_k \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_k \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. We deduce $\mathfrak{h}_i = \mathfrak{h}_k$ and $w = \mathfrak{h}_j$. By Equivalence (D.2), we deduce the inclusion $\{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k), \mathtt{conv}(\mathtt{x}_k, \mathtt{x}_i), \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)\} \subseteq \mathsf{P}$. Since P is closed under the rules

$$\frac{\mathtt{conv}(\mathtt{x}_k, \mathtt{x}_i) \quad \mathtt{toloop}(\mathtt{x}_k)}{\mathtt{toloop}(\mathtt{x}_i)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k) \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_p}{\mathtt{x}_k \hookrightarrow \mathtt{x}_p}$$

we deduce $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$;
  – $(\mathfrak{h}_i, w) = (\mathfrak{h}_k, 0)$ with $\mathtt{toalloc}(\mathtt{x}_k) \in \mathsf{P}$ and $\{\mathtt{x}_k \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_k \hookrightarrow \mathtt{x}_q, \mathtt{btwn}(\mathtt{x}_k, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_k, \mathtt{x}_q), \mathtt{toloop}(\mathtt{x}_k)\} \cap \mathsf{P} = \varnothing$. We deduce $\mathfrak{h}_i = \mathfrak{h}_k$ and $w = 0$. We get $\{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k), \mathtt{conv}(\mathtt{x}_k, \mathtt{x}_i), \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)\} \subseteq \mathsf{P}$ using Equivalence (D.2). Since P is closed under the rules

$$\frac{\mathtt{conv}(\mathtt{x}_k, \mathtt{x}_i) \quad \mathtt{toalloc}(\mathtt{x}_k)}{\mathtt{toalloc}(\mathtt{x}_i)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k) \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_p}{\mathtt{x}_k \hookrightarrow \mathtt{x}_p}$$

$$\frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k) \quad \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_p)}{\mathtt{btwn}(\mathtt{x}_k, \mathtt{x}_p)} \qquad \frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_k) \quad \mathtt{toloop}(\mathtt{x}_i)}{\mathtt{toloop}(\mathtt{x}_k)}$$

we deduce $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q), \mathtt{toloop}(\mathtt{x}_i)\} \cap \mathsf{P} = \varnothing$;

– let us finally check Property P4. By definition of H and Property (D.3), there are three possibilities for $(0, w) \in H$:
  – $(0, w) = (0, s(\mathtt{x}_i))$ with $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$; Hence $w = s(\mathtt{x}_i)$ and all the other properties hold;
  – $(0, w) = (0, 0)$ with $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}, \mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. Hence $w = 0$ and all the other properties hold;
  – $(0, w) = (0, 2q + 1)$ with $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}, \mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q, \mathtt{u} \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$. Hence $w = 2q + 1$ and all the other properties hold.

We can now check that H is a functional graph. Assume that $\{(u, v), (u, w)\} \subseteq H$. Let us show $v = w$. We have three cases:

– either $u = s(\mathtt{x}_i)$
  – $v = s(\mathtt{x}_j)$ and $w = s(\mathtt{x}_k)$ with $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_j, \mathtt{x}_i \hookrightarrow \mathtt{x}_k\} \subseteq \mathsf{P}$. But P is closed under the rule

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_j \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_k}{\mathtt{x}_j = \mathtt{x}_k}$$

hence $\mathtt{x}_j = \mathtt{x}_k \in \mathsf{P}$ and thus $v = s(\mathtt{x}_j) = s(\mathtt{x}_k) = w$ by Equivalence (D.1);

  - $v = s(\mathtt{x}_j)$ and $w = \mathfrak{h}_i$ is impossible because $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$ contradicts $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$;
  - $v = \mathfrak{h}_i$ and $w = \mathfrak{h}_i$ imply $v = w$;
- or $u = \mathfrak{h}_i$ with $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$
  - $v = s(\mathtt{x}_j)$ and $w = s(\mathtt{x}_k)$ with $\{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j), \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_k)\} \subseteq \mathsf{P}$. But P is closed under the rule
$$\frac{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \quad \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_k)}{\mathtt{x}_j = \mathtt{x}_k}$$
    hence $\mathtt{x}_j = \mathtt{x}_k \in \mathsf{P}$ and thus $v = s(\mathtt{x}_j) = s(\mathtt{x}_k) = w$ by Equivalence (D.1);
  - $v = s(\mathtt{x}_j)$ and $w = \mathfrak{h}_i$ with $\{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j), \mathtt{toloop}(\mathtt{x}_i)\} \subseteq \mathsf{P}$. But P is closed under the rule
$$\frac{\mathtt{toloop}(\mathtt{x}_i) \quad \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)}{\mathtt{x}_i \hookrightarrow \mathtt{x}_j}$$
    hence we deduce $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$ which contradicts $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$;
  - $v = s(\mathtt{x}_j)$ and $w = 0$ is impossible because $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ contradicts $\{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q), \mathtt{toloop}(\mathtt{x}_i)\} \cap \mathsf{P} = \varnothing$;
  - $v = \mathfrak{h}_i$ and $w = \mathfrak{h}_i$ implies $v = w$;
  - $v = \mathfrak{h}_i$ and $w = 0$ is impossible because $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$ contradicts $\{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q), \mathtt{toloop}(\mathtt{x}_i)\} \cap \mathsf{P} = \varnothing$;
  - $v = 0$ and $w = 0$ implies $v = w$;
- or $u = 0$ with $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$
  - $v = s(\mathtt{x}_i)$ and $w = s(\mathtt{x}_j)$ with $\{\mathtt{u} \hookrightarrow \mathtt{x}_i, \mathtt{u} \hookrightarrow \mathtt{x}_j\} \subseteq \mathsf{P}$. But P is closed under the rule
$$\frac{\mathtt{u} \hookrightarrow \mathtt{x}_i \quad \mathtt{u} \hookrightarrow \mathtt{x}_j}{\mathtt{x}_i = \mathtt{x}_j}$$
    hence $\mathtt{x}_i = \mathtt{x}_j \in \mathsf{P}$ and thus $v = s(\mathtt{x}_i) = s(\mathtt{x}_j) = w$ by Equivalence (D.1);
  - $v = s(\mathtt{x}_i)$ and $w = 0$ is impossible because $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ contradicts $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$;
  - $v = s(\mathtt{x}_i)$ and $w = 2q + 1$ is impossible because $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ contradicts $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q, \mathtt{u} \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$;
  - $v = 0$ and $w = 0$ implies $v = w$;
  - $v = 0$ and $w = 2q + 1$ is impossible because $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$ contradicts $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q, \mathtt{u} \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$;
  - $v = 2q + 1$ and $w = 2q + 1$ implies $v = w$.

Let us now show that L is a singleton set. For that, we first show that L contains no more than one location:

- if $s(\mathtt{x}_i) \in L$ and $s(\mathtt{x}_j) \in L$ then $\{\mathtt{x}_i = \mathtt{u}, \mathtt{x}_j = \mathtt{u}\} \subseteq \mathsf{P}$. But P is closed under the rules
$$\frac{\mathtt{x}_j = \mathtt{u}}{\mathtt{u} = \mathtt{x}_j} \qquad \frac{\mathtt{x}_i = \mathtt{u} \quad \mathtt{u} = \mathtt{x}_j}{\mathtt{x}_i = \mathtt{x}_j}$$
  thus $\mathtt{x}_i = \mathtt{x}_j \in \mathsf{P}$ and $s(\mathtt{x}_i) = s(\mathtt{x}_j)$;
- $s(\mathtt{x}_i) \in L$ and $\mathfrak{h}_j \in L$ is impossible because $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ contradicts $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}\} \cap \mathsf{P} = \varnothing$;
- the case when $s(\mathtt{x}_i) \in L$ and $0 \in L$ is impossible because $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ contradicts $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \ldots\} \cap \mathsf{P} = \varnothing$;
- if $\mathfrak{h}_i \in L$ and $\mathfrak{h}_j \in L$ then we have $\{\mathtt{x}_i \hookrightarrow \mathtt{u}, \mathtt{x}_j \hookrightarrow \mathtt{u}\} \in \mathsf{P}$. But P is closed under the rule
$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{u} \quad \mathtt{x}_j \hookrightarrow \mathtt{u}}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)}$$
  hence $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ and thus $\mathfrak{h}_i = \mathfrak{h}_j$;
- $\mathfrak{h}_i \in L$ and $0 \in L$ is impossible because $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$ contradicts $\{\ldots, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$.

Then we show that L is not empty. If there exists i such that $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ then $s(\mathtt{x}_i) \in L$. Otherwise we have $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}\} \cap \mathsf{P} = \varnothing$. If there exists j such that $\mathtt{x}_j \hookrightarrow \mathtt{u} \in \mathsf{P}$ then, because P is closed under the rule

$$\frac{\mathtt{x}_j \hookrightarrow \mathtt{u} \quad \mathtt{x}_j \hookrightarrow \mathtt{u}}{\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)}$$

we have $\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j) \in \mathsf{P}$ and thus $\mathfrak{h}_j$ is defined (see Equivalence (D.2)) and we deduce $\mathfrak{h}_j \in L$. Otherwise $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$ and in that case $0 \in L$.

We consider the memory state (s, h) and the location l such that H is the graph of the heap h and $L = \{l\}$. Let us show that the inclusion $\mathrm{dom}(h) \subseteq \heartsuit(s, h) \cup \{l\}$ holds. For this we show that the three following properties hold:

- $\mathrm{dom}(h) \subseteq \{s(\mathtt{x}_i) \mid i \in [1, q]\} \cup \{\mathfrak{h}_i \mid i \in [1, q] \text{ and } \mathfrak{h}_i \text{ is defined}\} \cup \{0\}$;
- $\{s(\mathtt{x}_i) \mid i \in [1, q]\} \cup \{\mathfrak{h}_i \mid i \in [1, q] \text{ and } \mathfrak{h}_i \text{ is defined}\} \subseteq \mathfrak{p}\heartsuit(s, h)$;
- if $0 \in \mathrm{dom}(h)$ then $l = 0$.

The first property is trivial by definition of H. For the second property, we first notice that $\{s(\mathtt{x}_i) \mid i \in [1, q]\} \subseteq \mathfrak{p}\heartsuit(s, h)$. Then if $\mathfrak{h}_i$ is defined then $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$ by characteristic Property P3 of H. Hence we have $(s(\mathtt{x}_i), \mathfrak{h}_i) \in H$ and we deduce $\mathfrak{h}_i \in h(s(\{\mathtt{x}_1, \ldots, \mathtt{x}_q\}))$. Finally, if $0 \in \mathrm{dom}(h)$ then by characteristic Property P4 of H we have $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$ and as a consequence we get $0 \in L$ by definition of L. Hence $l = 0$.

From the three previous properties we deduce $\mathrm{dom}(h) \subseteq \mathfrak{p}\heartsuit(s, h) \cup \{l\}$ and hence the inclusion $\mathrm{dom}(h) \subseteq \heartsuit(s, h) \cup \{l\}$ holds.

Let us finally show that for any basic formula $\mathcal{B} \in \mathsf{Basic}^{\mathtt{u}}$ we have $(s, h) \models_l \mathcal{B}$ iff $\mathcal{B} \in \mathsf{P}$. We proceed by case analysis on $\mathcal{B}$:

if $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{x}_j$. Then $(s, h) \models_l \mathtt{x}_i = \mathtt{x}_j$ iff $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ iff $\mathtt{x}_i = \mathtt{x}_j \in \mathsf{P}$ by Equivalence (D.1);

if $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$. Let us first assume $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and show $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. We have $h(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$ hence $(s(\mathtt{x}_i), s(\mathtt{x}_j)) \in H$. By the characteristic Property P2 of H and Property (D.3), the only possibility is that there exists k such that $s(\mathtt{x}_j) = s(\mathtt{x}_k)$ and $\mathtt{x}_i \hookrightarrow \mathtt{x}_k \in \mathsf{P}$. Hence by Equivalence (D.1), we have $\mathtt{x}_k = \mathtt{x}_j \in \mathsf{P}$. But P is closed under the rule

$$\frac{\mathtt{x}_k = \mathtt{x}_j \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_k}{\mathtt{x}_i \hookrightarrow \mathtt{x}_j}$$

hence we derive $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$.
Let us now assume $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. Then $(s(\mathtt{x}_i), s(\mathtt{x}_j)) \in H$ by definition of H and thus $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$;

if $\mathcal{B}$ is $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$. Let us first assume $(s, h) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$ and show $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$. We have $h(s(\mathtt{x}_i)) = h(s(\mathtt{x}_j)) = v$. Hence $\{(s(\mathtt{x}_i), v), (s(\mathtt{x}_j), v)\} \subseteq H$. By the characteristic Property P2 of H and Property (D.3), we have two cases:

- $v = s(\mathtt{x}_k)$ and $v = s(\mathtt{x}_r)$ with $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_k, \mathtt{x}_j \hookrightarrow \mathtt{x}_r\} \subseteq \mathsf{P}$. We deduce $s(\mathtt{x}_k) = s(\mathtt{x}_r)$ and thus $\mathtt{x}_k = \mathtt{x}_r \in \mathsf{P}$ by Equivalence (D.1). But P is closed under the rules

$$\frac{\mathtt{x}_k = \mathtt{x}_r \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_k}{\mathtt{x}_i \hookrightarrow \mathtt{x}_r} \qquad \frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_r \quad \mathtt{x}_j \hookrightarrow \mathtt{x}_r}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)}$$

  hence we get $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$;
- $v = \mathfrak{h}_i$ and $v = \mathfrak{h}_j$ with $\{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i), \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)\} \subseteq \mathsf{P}$. From $\mathfrak{h}_i = \mathfrak{h}_j$, we deduce $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ by Equivalence (D.2);

Now let us assume $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ and let us show $(s, h) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$. We have two cases:

- if $\mathtt{x}_i \hookrightarrow \mathtt{x}_k \in \mathsf{P}$ holds for some $k \in [1, q]$ then as P is closed under the rule

$$\frac{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_k}{\mathtt{x}_j \hookrightarrow \mathtt{x}_k}$$

  then $\mathtt{x}_j \hookrightarrow \mathtt{x}_k \in \mathsf{P}$ and $\{(s(\mathtt{x}_i), s(\mathtt{x}_k)), (s(\mathtt{x}_j), s(\mathtt{x}_k))\} \subseteq H$ by definition of H. Hence $(s, h) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$;

- otherwise $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. From $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ we deduce $\mathfrak{h}_i = \mathfrak{h}_j$ and $\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j) \in \mathsf{P}$ by Equivalence (D.2). Hence by definition of H we have $\{(s(\mathtt{x}_i), \mathfrak{h}_j), (s(\mathtt{x}_j), \mathfrak{h}_j)\} \subseteq H$ and we conclude $(s, h) \models_l \mathtt{conv}(\mathtt{x}_i, \mathtt{x}_j)$;

if $\mathcal{B}$ is $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$. Let us first assume $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$ and show $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$. We have $\{(s(\mathtt{x}_i), v), (v, s(\mathtt{x}_j))\} \subseteq H$ for some v. By characteristic Property P2 of H, we have two cases:

- $v = s(\mathtt{x}_k)$ with $\mathtt{x}_i \hookrightarrow \mathtt{x}_k \in \mathsf{P}$. From $(s(\mathtt{x}_k), s(\mathtt{x}_j)) \in H$ we deduce $(s, h) \models_l \mathtt{x}_k \hookrightarrow \mathtt{x}_j$ and thus $\mathtt{x}_k \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ (from the earlier case $\mathcal{B} = \mathtt{x}_k \hookrightarrow \mathtt{x}_j$). Since P is closed under the rule
$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_k \quad \mathtt{x}_k \hookrightarrow \mathtt{x}_j}{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)}$$
we deduce $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$;
- $v = \mathfrak{h}_i$ with $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. By characteristic Property P3 of H and Property (D.3), there is only one possible case for $(\mathfrak{h}_i, s(\mathtt{x}_j)) \in H$: there must exist k such that $s(\mathtt{x}_j) = s(\mathtt{x}_k)$ and $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_k) \in \mathsf{P}$. By Equivalence (D.1), we deduce $\mathtt{x}_k = \mathtt{x}_j \in \mathsf{P}$. Since P is closed under the rule
$$\frac{\mathtt{x}_k = \mathtt{x}_j \quad \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_k)}{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)}$$
we deduce $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$;

Now let us assume $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ and let us show $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$. We have two cases:

- either $\mathtt{x}_i \hookrightarrow \mathtt{x}_k \in \mathsf{P}$ holds for some $k \in [1, q]$. As P is closed under the rule
$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_k \quad \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)}{\mathtt{x}_k \hookrightarrow \mathtt{x}_j}$$
we deduce $\mathtt{x}_k \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ and thus we have $\{(s(\mathtt{x}_i), s(\mathtt{x}_k)), (s(\mathtt{x}_k), s(\mathtt{x}_j))\} \subseteq H$ by definition of H. As a consequence, we get $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$;
- or $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. Since P is closed under the rule
$$\frac{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)}$$
we deduce $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and thus $\{(s(\mathtt{x}_i), \mathfrak{h}_i), (\mathfrak{h}_i, s(\mathtt{x}_j))\} \subseteq H$ by definition of H. As a consequence we derive $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$;

if $\mathcal{B}$ is $\mathtt{toloop}(\mathtt{x}_i)$. Let us first assume $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ and show $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$. We have $\{(s(\mathtt{x}_i), v), (v, v)\} \subseteq H$ for some $v \in \mathbb{N}$. By characteristic Property P2 of H, we have two cases for $(s(\mathtt{x}_i), v) \in H$:

- $v = s(\mathtt{x}_j)$ with $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. From $(s(\mathtt{x}_j), s(\mathtt{x}_j)) \in H$ we deduce $(s, h) \models_l \mathtt{x}_j \hookrightarrow \mathtt{x}_j$ and thus $\mathtt{x}_j \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ (from the earlier case $\mathcal{B} = \mathtt{x}_j \hookrightarrow \mathtt{x}_j$). Since P is closed under the rule
$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_j \quad \mathtt{x}_j \hookrightarrow \mathtt{x}_j}{\mathtt{toloop}(\mathtt{x}_i)}$$
we deduce $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$;
- $v = \mathfrak{h}_i$ with $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. By characteristic Property P3 of H and Property (D.3), from $(\mathfrak{h}_i, \mathfrak{h}_i) \in H$ we deduce $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$;

Now let us assume $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$ and let us show $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$. We have two cases:

- either $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ holds for some for some $j \in [1, q]$. As P is closed under the rule
$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_j \quad \mathtt{toloop}(\mathtt{x}_i)}{\mathtt{x}_j \hookrightarrow \mathtt{x}_j}$$
we deduce $\mathtt{x}_j \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ and thus we have both $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and $(s, h) \models_l \mathtt{x}_j \hookrightarrow \mathtt{x}_j$ (from the earlier cases $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and $\mathcal{B} = \mathtt{x}_j \hookrightarrow \mathtt{x}_j$). Hence we derive $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$;

– or $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. As P is closed under the rules

$$\frac{\mathtt{toloop}(\mathtt{x}_i)}{\mathtt{toalloc}(\mathtt{x}_i)} \qquad \frac{\mathtt{toalloc}(\mathtt{x}_i)}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)}$$

we also get $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and thus $\{(s(\mathtt{x}_i), \mathfrak{h}_i), (\mathfrak{h}_i, \mathfrak{h}_i)\} \subseteq H$ by definition of H. Hence $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$;

if $\mathcal{B}$ is $\mathtt{toalloc}(\mathtt{x}_i)$. Let us first assume $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ and show $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$. We have $\{(s(\mathtt{x}_i), v), (v, w)\} \subseteq H$ for some $v, w \in \mathbb{N}$. By characteristic Property P2 of H, we have two cases for $(s(\mathtt{x}_i), v) \in H$:

– $v = s(\mathtt{x}_j)$ with $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. From $(s(\mathtt{x}_j), w) \in H$ we deduce $(s, h) \models_l \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)$ and thus $\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j) \in \mathsf{P}$ (from the earlier case $\mathcal{B} = \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)$). Since P is closed under the rule

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_j \quad \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)}{\mathtt{toalloc}(\mathtt{x}_i)}$$

we deduce $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$;

– $v = \mathfrak{h}_i$ with $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. By characteristic Property P3 of H, we have three cases for $(\mathfrak{h}_i, w) \in H$.
  – $w = s(\mathtt{x}_j)$ with $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$. But P is closed under the rule

$$\frac{\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)}{\mathtt{toalloc}(\mathtt{x}_i)}$$

  hence $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$;
  – $w = \mathfrak{h}_i$ with $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$. But P is closed under the rule

$$\frac{\mathtt{toloop}(\mathtt{x}_i)}{\mathtt{toalloc}(\mathtt{x}_i)}$$

  hence $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$;
  – $w = 0$ and in this case $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$;

Now let us assume $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$ and let us show $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$. We have four cases:

– either $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ holds for some $j \in [1, q]$. As P is closed under the rule

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_j \quad \mathtt{toalloc}(\mathtt{x}_i)}{\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)}$$

we deduce $\mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j) \in \mathsf{P}$ and thus we have both $(s, h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and $(s, h) \models_l \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)$ (from the earlier cases $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and $\mathcal{B} = \mathtt{conv}(\mathtt{x}_j, \mathtt{x}_j)$). Hence we derive $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$;

– or $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$ and $\mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j) \in \mathsf{P}$ for some $j \in [1, q]$. Then we have $(s, h) \models_l \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$ (from the earlier case $\mathcal{B} = \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_j)$). We deduce $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$;

– or $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q)\} \cap \mathsf{P} = \varnothing$ and $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$. Then we have $(s, h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ (from the earlier case $\mathcal{B} = \mathtt{toloop}(\mathtt{x}_i)$). We deduce $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$;

– or $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_1), \ldots, \mathtt{btwn}(\mathtt{x}_i, \mathtt{x}_q), \mathtt{toloop}(\mathtt{x}_i)\} \cap \mathsf{P} = \varnothing$. Since P is closed under the rule

$$\frac{\mathtt{toalloc}(\mathtt{x}_i)}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)}$$

we deduce $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and thus $\{(s(\mathtt{x}_i), \mathfrak{h}_i), (\mathfrak{h}_i, 0)\} \subseteq H$ hence we conclude $(s, h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$;

if $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$. Let us first assume $(s,h) \models_l \mathtt{x}_i = \mathtt{u}$ and show $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$. We have $l = s(\mathtt{x}_i)$. According to the definition of L and Property (D.3), we must have $l = s(\mathtt{x}_j)$ with $\mathtt{x}_j = \mathtt{u} \in \mathsf{P}$. But then we have $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ hence $\mathtt{x}_i = \mathtt{x}_j \in \mathsf{P}$ by Equivalence (D.1). As P is closed under the rule

$$\frac{\mathtt{x}_i = \mathtt{x}_j \quad \mathtt{x}_j = \mathtt{u}}{\mathtt{x}_i = \mathtt{u}}$$

we get $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$.
Conversely, if we assume $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ then by definition of L we have $s(\mathtt{x}_i) \in L$ and thus $l = s(\mathtt{x}_i)$. As a consequence, we have $(s,h) \models_l \mathtt{x}_i = \mathtt{u}$;

if $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$. Let us first assume $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ and show $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$. We have $(s(\mathtt{x}_i), l) \in H$. By the characteristic Property P2 of H, we have two cases:
- either $l = s(\mathtt{x}_j)$ with $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$ for some $j \in [1,q]$. We derive $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ from the earlier case $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ and thus we get $(s,h) \models_l \mathtt{x}_j = \mathtt{u}$. Hence we have $\mathtt{x}_j = \mathtt{u} \in \mathsf{P}$ (from the earlier case $\mathcal{B} = \mathtt{x}_j = \mathtt{u}$). As P is closed under the rule

$$\frac{\mathtt{x}_j = \mathtt{u} \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_j}{\mathtt{x}_i \hookrightarrow \mathtt{u}}$$

  we get $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$;
- or $l = \mathfrak{h}_i$. But in that case, according to the definition of L and Property (D.3), we must have $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$;

Now let us assume $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$ and let us show $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$. We have two cases:
- either $\mathtt{x}_j = \mathtt{u} \in \mathsf{P}$ for some $j \in [1,q]$. As P is closed under the rules

$$\frac{\mathtt{x}_j = \mathtt{u}}{\mathtt{u} = \mathtt{x}_j} \qquad \frac{\mathtt{u} = \mathtt{x}_j \quad \mathtt{x}_i \hookrightarrow \mathtt{u}}{\mathtt{x}_i \hookrightarrow \mathtt{x}_j}$$

  we get $\mathtt{x}_i \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. Then we have $(s,h) \models_l \mathtt{x}_j = \mathtt{u}$ and $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ (from the earlier cases $\mathcal{B} = \mathtt{x}_j = \mathtt{u}$ and $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{x}_j$). Hence we deduce $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$;
- or $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}\} \cap \mathsf{P} = \varnothing$ and in that case $l = \mathfrak{h}_i$. But P is closed under the rules

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{u} \quad \mathtt{x}_i \hookrightarrow \mathtt{u}}{\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i)} \qquad \frac{\mathtt{x}_i \hookrightarrow \mathtt{x}_p \quad \mathtt{x}_i \hookrightarrow \mathtt{u}}{\mathtt{x}_p = \mathtt{u}}$$

  hence $\mathtt{conv}(\mathtt{x}_i, \mathtt{x}_i) \in \mathsf{P}$ and and $\{\mathtt{x}_i \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_i \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. Thus $(s(\mathtt{x}_i), \mathfrak{h}_i = l) \in H$ by definition of H. We conclude $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$;

if $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_i$. Let us first assume $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$ and show $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$. According to the definition of L, for $l \in L$ we have three cases:
- either $l = s(\mathtt{x}_j)$ with $\mathtt{x}_j = \mathtt{u} \in \mathsf{P}$ for some $j \in [1,q]$. From $(s(\mathtt{x}_j), s(\mathtt{x}_i)) \in H$, using characteristic Property P2 of H and Property (D.3), we deduce $\mathtt{x}_j \hookrightarrow \mathtt{x}_i \in \mathsf{P}$. But P is closed under the rule

$$\frac{\mathtt{x}_j = \mathtt{u} \quad \mathtt{x}_j \hookrightarrow \mathtt{x}_i}{\mathtt{u} \hookrightarrow \mathtt{x}_i}$$

  hence we get $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$;
- or $l = \mathfrak{h}_j$ with $\mathtt{x}_j \hookrightarrow \mathtt{u} \in \mathsf{P}$ for some $j \in [1,q]$. From $(\mathfrak{h}_j, s(\mathtt{x}_i)) \in H$, using characteristic Property P3 of H and Property (D.3), we deduce $\mathtt{btwn}(\mathtt{x}_j, \mathtt{x}_i) \in \mathsf{P}$. Since P is closed under the rule

$$\frac{\mathtt{x}_j \hookrightarrow \mathtt{u} \quad \mathtt{btwn}(\mathtt{x}_j, \mathtt{x}_i)}{\mathtt{u} \hookrightarrow \mathtt{x}_i}$$

  we get $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$;
- or $l = 0$. From $(0, s(\mathtt{x}_i)) \in H$, using characteristic Property P4 of H and Property (D.3), we deduce $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ and $\mathtt{u} \hookrightarrow \mathtt{x}_j \in \mathsf{P}$. From Equivalence (D.1) we get $\mathtt{x}_j = \mathtt{x}_i \in \mathsf{P}$ and as P is closed under the rule

$$\frac{\mathtt{x}_j = \mathtt{x}_i \quad \mathtt{u} \hookrightarrow \mathtt{x}_j}{\mathtt{u} \hookrightarrow \mathtt{x}_i}$$

  we conclude $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$;

Now let us assume $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ and let us show $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$. We have three cases for $l \in L$:

- either $l = s(\mathtt{x}_j)$ with $\mathtt{x}_j = \mathtt{u} \in \mathsf{P}$ for some $j \in [1,q]$. As P is closed under the rules

$$\frac{\mathtt{x}_j = \mathtt{u}}{\mathtt{u} = \mathtt{x}_j} \qquad \frac{\mathtt{u} = \mathtt{x}_j \quad \mathtt{u} \hookrightarrow \mathtt{x}_i}{\mathtt{x}_j \hookrightarrow \mathtt{x}_i}$$

  we get $\mathtt{x}_j \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ and thus $(s,h) \models_l \mathtt{x}_j \hookrightarrow \mathtt{x}_i$ from the earlier case $\mathcal{B} = \mathtt{x}_j \hookrightarrow \mathtt{x}_i$. We deduce $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$;
- or $l = \mathfrak{h}_j$ with $\mathtt{x}_j \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}\} \cap \mathsf{P} = \varnothing$. As P is closed under the rules

$$\frac{\mathtt{x}_j \hookrightarrow \mathtt{u} \quad \mathtt{u} \hookrightarrow \mathtt{x}_i}{\mathtt{btwn}(\mathtt{x}_j, \mathtt{x}_i)} \qquad \frac{\mathtt{x}_j \hookrightarrow \mathtt{x}_p \quad \mathtt{x}_j \hookrightarrow \mathtt{u}}{\mathtt{x}_p = \mathtt{u}}$$

  we get $\mathtt{btwn}(\mathtt{x}_j, \mathtt{x}_i) \in \mathsf{P}$ and $\{\mathtt{x}_j \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{x}_j \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. Hence by definition of H we get $(\mathfrak{h}_j, s(\mathtt{x}_i)) \in H$. We deduce $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$;
- or $l = 0$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$. Then $(0, s(\mathtt{x}_i)) \in H$ by definition of H and we deduce $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$;

if $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$. Let us first assume $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ and show $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$. According to the definition of L, for $l \in L$ we have three cases:

- either $l = s(\mathtt{x}_i)$ with $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ for some $i \in [1,q]$. From the earlier case $\mathcal{B} = \mathtt{x}_i = \mathtt{u}$ we deduce $(s,h) \models_l \mathtt{x}_i = \mathtt{u}$. Hence we get $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_i$ and as a consequence $\mathtt{x}_i \hookrightarrow \mathtt{x}_i \in \mathsf{P}$. But P is closed under the rules

$$\frac{\mathtt{x}_i = \mathtt{u} \quad \mathtt{x}_i \hookrightarrow \mathtt{x}_i}{\mathtt{u} \hookrightarrow \mathtt{x}_i} \qquad \frac{\mathtt{x}_i = \mathtt{u} \quad \mathtt{u} \hookrightarrow \mathtt{x}_i}{\mathtt{u} \hookrightarrow \mathtt{u}}$$

  hence $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$;
- or $l = \mathfrak{h}_i$ with $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots \mathtt{x}_q = \mathtt{u}\} \cap \mathsf{P} = \varnothing$ for some $i \in [1,q]$. We deduce $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ from the earlier case $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{u}$. Hence we derive $(s,h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ and thus $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$ from the earlier case $\mathcal{B} = \mathtt{toloop}(\mathtt{x}_i)$. But P is closed under the rule

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{u} \quad \mathtt{toloop}(\mathtt{x}_i)}{\mathtt{u} \hookrightarrow \mathtt{u}}$$

  hence $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$;
- or $l = 0$. Then $(0,0) \in H$ and by characteristic Property P4 of H and Property (D.3), we must have $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$;

Now let us assume $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$ and let us show $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$. We have three cases for $l \in L$ we have three cases:

- either $l = s(\mathtt{x}_i)$ with $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ for some $i \in [1,q]$. As P is closed under the rules

$$\frac{\mathtt{x}_i = \mathtt{u}}{\mathtt{u} = \mathtt{x}_i} \qquad \frac{\mathtt{u} = \mathtt{x}_i \quad \mathtt{u} \hookrightarrow \mathtt{u}}{\mathtt{x}_i \hookrightarrow \mathtt{u}} \qquad \frac{\mathtt{u} = \mathtt{x}_i \quad \mathtt{x}_i \hookrightarrow \mathtt{u}}{\mathtt{x}_i \hookrightarrow \mathtt{x}_i}$$

  we get $\mathtt{x}_i \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ hence $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_i$ (from the earlier case $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{x}_i$). Since $l = s(\mathtt{x}_i)$ we deduce $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$;
- or $l = \mathfrak{h}_i$ with $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$. As P is closed under the rule

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{u} \quad \mathtt{u} \hookrightarrow \mathtt{u}}{\mathtt{toloop}(\mathtt{x}_i)}$$

  we get $\mathtt{toloop}(\mathtt{x}_i) \in \mathsf{P}$. From the earlier cases $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{u}$ and $\mathcal{B} = \mathtt{toloop}(\mathtt{x}_i)$ we deduce $(s,h) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ and $(s,h) \models_l \mathtt{toloop}(\mathtt{x}_i)$ hence $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$;
- or $l = 0$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$. As P is closed under the rule

$$\frac{\mathtt{u} \hookrightarrow \mathtt{x}_p \quad \mathtt{u} \hookrightarrow \mathtt{u}}{\mathtt{x}_p = \mathtt{u}}$$

  we deduce $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$ and then $(0,0) \in H$ by the definition of H and we conclude $(s,h) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$;

if $\mathcal{B}$ is $\mathtt{alloc}(\mathtt{u})$. Let us first assume $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$ and show $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$. According to the definition of L, for $l \in L$ we have three cases:

- either $l = s(\mathtt{x}_i)$ with $\mathtt{x}_i{=}\mathtt{u} \in \mathsf{P}$ for some $i \in [1,q]$. We deduce $s(\mathtt{x}_i) \in \mathrm{dom}(h)$ and thus $(s,h) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)$. Using the earlier case $\mathcal{B} = \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)$, we get $\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i) \in \mathsf{P}$. But P is closed under the rule

$$\frac{\mathtt{x}_i = \mathtt{u} \quad \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)}{\mathtt{alloc}(\mathtt{u})}$$

  hence $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$;
- or $l = \mathfrak{h}_i$ with $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$ for some $i \in [1,q]$. Using the earlier case $\mathcal{B} = \mathtt{x}_i \hookrightarrow \mathtt{u}$, we deduce $(s,h) \models_l \mathtt{x}_i{\hookrightarrow}\mathtt{u}$ and then $(s,h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$. Hence we get $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$ (from the earlier case $\mathcal{B} = \mathtt{toalloc}(\mathtt{x}_i)$). As P is closed under the rule

$$\frac{\mathtt{toalloc}(\mathtt{x}_i) \quad \mathtt{x}_i \hookrightarrow \mathtt{u}}{\mathtt{alloc}(\mathtt{u})}$$

  we get $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$;
- or $l = 0$. Then $(0,v) \in H$ for some $v \in \mathbb{N}$. Using characteristic Property P4 we deduce $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$ and:
  - either $v = s(\mathtt{x}_i)$ and $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ for some $i \in [1,q]$. As P is closed under the rule

$$\frac{\mathtt{u} \hookrightarrow \mathtt{x}_i}{\mathtt{alloc}(\mathtt{u})}$$

    we get $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$;
  - or $v = 0$ and $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$. As P is closed under the rule

$$\frac{\mathtt{u} \hookrightarrow \mathtt{u}}{\mathtt{alloc}(\mathtt{u})}$$

    we get $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$;
  - or $v = 2q+1$ and $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$ and $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q, \mathtt{u} \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$;

Now let us assume $\mathtt{alloc}(\mathtt{u}) \in \mathsf{P}$ and let us show $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$. We have three cases for $l \in L$:

- either $l = s(\mathtt{x}_i)$ with $\mathtt{x}_i = \mathtt{u} \in \mathsf{P}$ for some $i \in [1,q]$. As P is closed under the rules

$$\frac{\mathtt{x}_i = \mathtt{u}}{\mathtt{u} = \mathtt{x}_i} \qquad \frac{\mathtt{u} = \mathtt{x}_i \quad \mathtt{alloc}(\mathtt{u})}{\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)}$$

  we get $\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i) \in \mathsf{P}$ and thus $(s,h) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)$ from the earlier case $\mathcal{B} = \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_i)$. Hence $l = s(\mathtt{x}_i) \in \mathrm{dom}(h)$ and we deduce $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$;
- or $l = \mathfrak{h}_i$ with $\mathtt{x}_i \hookrightarrow \mathtt{u} \in \mathsf{P}$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}\} \cap \mathsf{P} = \varnothing$. As P is closed under the rules

$$\frac{\mathtt{x}_i \hookrightarrow \mathtt{u} \quad \mathtt{alloc}(\mathtt{u})}{\mathtt{toalloc}(\mathtt{x}_i)}$$

  we get $\mathtt{toalloc}(\mathtt{x}_i) \in \mathsf{P}$. We derive $(s,h) \models_l \mathtt{x}_i{\hookrightarrow}\mathtt{u}$ (from the earlier case $\mathcal{B} = \mathtt{x}_i{\hookrightarrow}\mathtt{u}$) and $(s,h) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ (from the earlier case $\mathcal{B} = \mathtt{toalloc}(\mathtt{x}_i)$). Hence we get $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$;
- or $l = 0$ and $\{\mathtt{x}_1 = \mathtt{u}, \ldots, \mathtt{x}_q = \mathtt{u}, \mathtt{x}_1 \hookrightarrow \mathtt{u}, \ldots, \mathtt{x}_q \hookrightarrow \mathtt{u}\} \cap \mathsf{P} = \varnothing$. We consider three cases:
  - either $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathsf{P}$ holds for some $i \in [1,q]$. In this case, $(0, s(\mathtt{x}_i)) \in H$ by definition of H and we get $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$;
  - or $\{\mathtt{u} \hookrightarrow \mathtt{x}_1, \ldots, \mathtt{u} \hookrightarrow \mathtt{x}_q\} \cap \mathsf{P} = \varnothing$ and $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathsf{P}$. In this case, $(0,0) \in H$ by definition of H and we get $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$;
  - or $\{\mathtt{u}{\hookrightarrow}\mathtt{x}_1, \ldots, \mathtt{u}{\hookrightarrow}\mathtt{x}_q, \mathtt{u}{\hookrightarrow}\mathtt{u}\} \cap \mathsf{P} = \varnothing$. In that case $(0, 2q+1) \in H$ by definition of H and we get $(s,h) \models_l \mathtt{alloc}(\mathtt{u})$. □

The next proposition is exclusively used in the proof of upcoming Proposition 5.5.

Proposition D.1 *Let $q \geqslant 1$. Let s be a store, h_1 and h_2 be two heaps and l be a location. We assume that $\heartsuit(s,h_1) \cup \{l\} = \heartsuit(s,h_2) \cup \{l\}$ and that h_1 and h_2 are identical maps on that subset of locations. Then $(s,h_1,l) \simeq_b (s,h_2,l)$.*

Proof Let us denote $D = \heartsuit(s,h_1) \cup \{l\} = \heartsuit(s,h_2) \cup \{l\}$. We show that $(s,h_1) \models_l \mathcal{B}$ implies $(s,h_2) \models_l \mathcal{B}$ by case analysis on $\mathcal{B}$:

- $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{x}_j$: if $(s,h_1) \models_l \mathtt{x}_i = \mathtt{x}_j$ then $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ and thus $(s,h_2) \models_l \mathtt{x}_i = \mathtt{x}_j$;
- $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{x}_j$: if $(s,h_1) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$ then $h_1(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$. Hence $s(\mathtt{x}_i) \in \mathrm{ref}(s,h_1) \subseteq D$ and we deduce $h_2(s(\mathtt{x}_i)) = h_1(s(\mathtt{x}_i)) = s(\mathtt{x}_j)$. We conclude $(s,h_2) \models_l \mathtt{x}_i \hookrightarrow \mathtt{x}_j$;
- $\mathcal{B}$ is $\mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$: if $(s,h_1) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$ then $h_1(s(\mathtt{x}_i)) = h_1(s(\mathtt{x}_j))$. Hence we have the inclusion $\{s(\mathtt{x}_i), s(\mathtt{x}_j)\} \subseteq D$ and we deduce $h_2(s(\mathtt{x}_i)) = h_1(s(\mathtt{x}_i)) = h_1(s(\mathtt{x}_j)) = h_2(s(\mathtt{x}_j))$. We conclude $(s,h_2) \models_l \mathtt{conv}(\mathtt{x}_i,\mathtt{x}_j)$;
- $\mathcal{B}$ is $\mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$: if $(s,h_1) \models_l \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$ then $h_1(h_1(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$. Hence we have $\{s(\mathtt{x}_i), h_1(s(\mathtt{x}_i))\} \subseteq D$ and we deduce $h_2(h_2(s(\mathtt{x}_i))) = h_2(h_1(s(\mathtt{x}_i))) = h_1(h_1(s(\mathtt{x}_i))) = s(\mathtt{x}_j)$. We conclude $(s,h_2) \models_l \mathtt{btwn}(\mathtt{x}_i,\mathtt{x}_j)$;
- $\mathcal{B}$ is $\mathtt{toalloc}(\mathtt{x}_i)$: if $(s,h_1) \models_l \mathtt{toalloc}(\mathtt{x}_i)$ then $h_1(s(\mathtt{x}_i)) \in \mathrm{dom}(h_1)$. Hence we get the inclusion $\{s(\mathtt{x}_i), h_1(s(\mathtt{x}_i))\} \subseteq D$. Then h_1 and h_2 have the same value at $s(\mathtt{x}_i)$ hence $h_2(s(\mathtt{x}_i)) = h_1(s(\mathtt{x}_i)) = u$. But h_1 and h_2 must also have the same value on $u \in D$, hence $h_2(u)$ must be defined (and equal to $h_1(u)$) and we deduce $h_2(s(\mathtt{x}_i)) = u \in \mathrm{dom}(h_2)$. We conclude $(s,h_2) \models_l \mathtt{toalloc}(\mathtt{x}_i)$;
- $\mathcal{B}$ is $\mathtt{toloop}(\mathtt{x}_i)$: if $(s,h_1) \models_l \mathtt{toloop}(\mathtt{x}_i)$ then $h_1(h_1(s(\mathtt{x}_i))) = h_1(s(\mathtt{x}_i))$. Hence we get $\{s(\mathtt{x}_i), h_1(s(\mathtt{x}_i))\} \subseteq D$. We deduce $h_2(h_2(s(\mathtt{x}_i))) = h_2(h_1(s(\mathtt{x}_i))) = h_1(h_1(s(\mathtt{x}_i))) = h_1(s(\mathtt{x}_i)) = h_2(s(\mathtt{x}_i))$. We conclude $(s,h_2) \models_l \mathtt{toloop}(\mathtt{x}_i)$;
- $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{u}$: if $(s,h_1) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$ then $h_1(l) = l$. As $l \in D$, we deduce $h_2(l) = h_1(l) = l$ and we conclude $(s,h_2) \models_l \mathtt{u} \hookrightarrow \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{alloc}(\mathtt{u})$: if $(s,h_1) \models_l \mathtt{alloc}(\mathtt{u})$ then $l \in \mathrm{dom}(h_1)$. As $l \in D$, we deduce $l \in \mathrm{dom}(h_2)$ and we conclude $(s,h_2) \models_l \mathtt{alloc}(\mathtt{u})$;
- $\mathcal{B}$ is $\mathtt{x}_i = \mathtt{u}$: if $(s,h_1) \models_l \mathtt{x}_i = \mathtt{u}$ then $s(\mathtt{x}_i) = l$ and we conclude $(s,h_2) \models_l \mathtt{x}_i = \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{x}_i \hookrightarrow \mathtt{u}$: if $(s,h_1) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$ then $h_1(s(\mathtt{x}_i)) = l$. Then $s(\mathtt{x}_i) \in D$ and we deduce $h_2(s(\mathtt{x}_i)) = h_1(s(\mathtt{x}_i)) = l$. We conclude $(s,h_2) \models_l \mathtt{x}_i \hookrightarrow \mathtt{u}$;
- $\mathcal{B}$ is $\mathtt{u} \hookrightarrow \mathtt{x}_i$: if $(s,h_1) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$ then $h_1(l) = s(\mathtt{x}_i)$. As $l \in D$, we deduce $h_2(l) = h_1(l) = s(\mathtt{x}_i)$ and we conclude $(s,h_2) \models_l \mathtt{u} \hookrightarrow \mathtt{x}_i$. □

Proposition 5.5 *Let $q \geqslant 1$. Let $s : \mathcal{V} \to \mathbb{N}$ be a store, $h : \mathbb{N} \rightharpoonup \mathbb{N}$ be a heap and $l \in \mathbb{N}$ be a location. Let $(\mathsf{p}_1, \ldots, \mathsf{p}_q, \mathsf{l}, \mathsf{r})$ be a cardinality assignment such that:*

1. *$s(\mathtt{x}_i) = s(\mathtt{x}_j)$ implies $\mathsf{p}_i = \mathsf{p}_j$ for all $i,j \in [1,q]$;*
2. *$\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \leqslant \mathsf{p}_i$ for any $i \in [1,q]$;*
3. *$\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \leqslant \mathsf{l}$;*
4. *$\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \leqslant \mathsf{r}$.*

There exists a heap h' such that:

- *$(s,h,l) \simeq_b (s,h',l)$;*
- *$\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h',i)) = \mathsf{p}_i$ for any $i \in [1,q]$;*
- *$\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h')) = \mathsf{l}$;*
- *$\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h')) = \mathsf{r}$.*

Proof Let us define $\underline{i} = \min\{j \in [1,q] \mid s(\mathtt{x}_i) = s(\mathtt{x}_j)\}$ for every $i \in [1,q]$, $m = \mathrm{maxval}(s,h,l)$, $n = \max\{\mathsf{p}_1, \ldots, \mathsf{p}_q, \mathsf{l}, \mathsf{r}\}$, $\mathsf{p}'_i = \mathsf{p}_i - \mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i))$ for every $i \in [1,q]$, $\mathsf{l}' = \mathsf{l} - \mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h))$ and $\mathsf{r}' = \mathsf{r} - \mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h))$. We define h' by the following rules:

- $h'(u) = v$ when $u \leqslant m$ and $h(u) = v$;
- $h'(u) = v$ when $u = m + 2 + \underline{i}.n + d$, $v = s(\mathtt{x}_i)$, $i \in [1,q]$, $1 \leqslant d \leqslant \mathsf{p}'_i$;
- $h'(u) = v$ when $u = m + 2 + (q+1).n + d$, $v = u$ and $1 \leqslant d \leqslant \mathsf{l}'$;
- $h'(u) = v$ when $u = m + 2 + (q+2).n + d$, $v = m+1$ and $1 \leqslant d \leqslant \mathsf{r}'$.

Then it is easy to check that h' is a heap that satisfies the following properties:

- $\heartsuit(s,h') = \heartsuit(s,h)$ and thus $\heartsuit(s,h') \cup \{l\} = \heartsuit(s,h) \cup \{l\}$;

– the restrictions of h and h' to $\heartsuit(s,h) \cup \{l\}$ are identical maps;
– $\mathrm{pred}_{\overline{\heartsuit}}(s,h',i) = \mathrm{pred}_{\overline{\heartsuit}}(s,h,i) \uplus [m+2+\underline{i}.n+1, m+2+\underline{i}.n+\mathsf{p}'_i]$;
– $\mathrm{loop}_{\overline{\heartsuit}}(s,h') = \mathrm{loop}_{\overline{\heartsuit}}(s,h) \uplus [m+2+(q+2).n+1, m+2+(q+2).n+\mathsf{l}']$;
– $\mathrm{rem}_{\overline{\heartsuit}}(s,h') = \mathrm{rem}_{\overline{\heartsuit}}(s,h) \uplus [m+2+(q+2).n+1, m+2+(q+2).n+\mathsf{r}']$.

Hence the cardinality identities $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h',i)) = \mathsf{p}_i$ for $i \in [1,q]$, $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h')) = \mathsf{l}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h')) = \mathsf{r}$ are obvious. The basic equivalence $(s,h,l) \simeq_b (s,h',l)$ comes from Proposition D.1 (page 78 in Appendix D). □

Proposition 5.7 *If the conjunction of the formulæ in* $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ *is satisfiable then the triple* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *is* n*-consistent for any* $n \in \{1,2,3\}$.

Proof Let us first prove the result for 1-consistency. Let us fix a triple $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ and consider a memory state (s,h) and a location l such all the formulæ in $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ are satisfied in (s,h,l). Let us show that Conditions C1.1 to C1.4 hold:

C1.1 no formula in B^- is satisfied in (s,h,l) and by Proposition 5.1, all the formulæ of $\mathrm{cl}(\mathsf{B}^+)$ are satisfied in (s,h,l). Hence we deduce $\mathsf{B}^- \cap \mathrm{cl}(\mathsf{B}^+) = \varnothing$;
C1.2 if $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$ and $\{\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_i) \geqslant a, \neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant b\} \subseteq \mathsf{S}$ then we deduce $s(\mathtt{x}_i) = s(\mathtt{x}_j)$, $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,i)) \geqslant a$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)) < b$. But $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ implies $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) = \mathrm{pred}_{\overline{\heartsuit}}(s,h,j)$ hence $a < b$;
C1.3 if $\{\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a, \neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant b\} \subseteq \mathsf{S}$ then the relations $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) \geqslant a$ and $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) < b$ hold hence $a < b$;
C1.4 if $\{\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant a, \neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant b\} \subseteq \mathsf{S}$ then the relations $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) \geqslant a$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h)) < b$ hold hence $a < b$.

Let us prove the result for 2-consistency. Let us fix a triple $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$. Let us consider a memory state (s,h) and a location l such that all the formulæ in $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ are satisfied in (s,h,l). We have already established that $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 1-consistent in this case. Let us show that Conditions C2.1 and C2.2 hold.

C2.1 if $\{\mathtt{x}_i = \mathtt{x}_j, \mathtt{u} \hookrightarrow \mathtt{x}_i\} \subseteq \mathrm{cl}(\mathsf{B}^+)$ and $\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant 1 \in \mathsf{S}$ then we deduce $s(\mathtt{x}_i) = s(\mathtt{x}_j)$, $h(l) = s(\mathtt{x}_i)$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h,j)) < 1$. But then $h(l) = s(\mathtt{x}_j)$ hence $l \in \mathrm{pred}(s,h,j)$. From $\mathrm{pred}_{\overline{\heartsuit}}(s,h,j) = \varnothing$ we derive $l \in \mathrm{pred}_{\heartsuit}(s,h,j)$ hence $l \in \heartsuit(s,h) \subseteq s(\mathcal{V}) \cup h(s(\mathcal{V}))$. Hence (s,h,l) satisfies at least one formula $\mathcal{B}$ of $\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}$. We deduce that all the formulæ of $\mathsf{B}^+ \cup \{\mathcal{B}\} \cup \neg\mathsf{B}^- \cup \mathsf{S}$ are satisfied in (s,h,l) hence $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 1-consistent;
C2.2 if $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathrm{cl}(\mathsf{B}^+)$ and $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1 \in \mathsf{S}$ then we have $h(l) = l$ and $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h)) < 1$. From $\mathrm{loop}_{\overline{\heartsuit}}(s,h) = \varnothing$ and $l \in \mathrm{loop}_{\heartsuit}(s,h)$ we deduce $l \in \heartsuit(s,h) \subseteq s(\mathcal{V}) \cup h(s(\mathcal{V}))$. Hence (s,h,l) satisfies at least one formula $\mathcal{B}$ of $\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}$. We deduce that all the formulæ of $\mathsf{B}^+ \cup \{\mathcal{B}\} \cup \neg\mathsf{B}^- \cup \mathsf{S}$ are satisfied in (s,h,l) hence $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 1-consistent;

Let us prove the result for 3-consistency. Let us fix a triple $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ and consider a memory state (s,h) and a location l such all the formulæ in $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ are satisfied in (s,h,l). We have already established that $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 2-consistent in this case. Let us show that Condition C3.1 holds:

C3.1 if $\mathtt{alloc}(\mathtt{u}) \in \mathrm{cl}(\mathsf{P})$ and $\neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant 1 \in \mathsf{S}$ then $l \in \mathrm{dom}(h)$ and $\mathrm{rem}_{\overline{\heartsuit}}(s,h) = \varnothing$. Hence by Lemma 2.6, either $l \in \heartsuit(s,h)$ or $l \in \mathrm{pred}_{\overline{\heartsuit}}(s,h,i)$ for some $i \in [1,q]$ or $l \in \mathrm{loop}_{\overline{\heartsuit}}(s,h)$. As a consequence, (s,h,l) satisfies at least one formula $\mathcal{B}$ of $\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}$. We deduce that all the formulæ of $\mathsf{B}^+ \cup \{\mathcal{B}\} \cup \neg\mathsf{B}^- \cup \mathsf{S}$ are satisfied in (s,h,l) hence $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 2-consistent. □

Proposition 5.8 *If the triple* $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ *is 3-consistent then the conjunction of the formulæ in* $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ *is satisfiable.*

Proof Let us first consider the case where $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 1-consistent (which is the weakest of the assumptions of 1-, 2- or 3-consistency). We define a cardinality assignment $(\mathsf{p}_1, \ldots, \mathsf{p}_q, \mathsf{l}, \mathsf{r})$ by:

$$\begin{array}{rl} \mathsf{p}_i & = \max\{a \mid \exists k \in [1,q],\ \mathtt{x}_i = \mathtt{x}_k \in \mathrm{cl}(\mathsf{B}^+) \wedge \#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_k) \geqslant a \in \mathsf{S}\} \quad \text{for } i \in [1,q] \\ \mathsf{l} & = \max\{a \mid \#\,\mathtt{loop}_{\heartsuit} \geqslant a \in \mathsf{S}\} \\ \mathsf{r} & = \max\{a \mid \#\,\mathtt{rem}_{\heartsuit} \geqslant a \in \mathsf{S}\} \end{array}$$

where we assume $\max(\varnothing) = 0$. Since $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 1-consistent, we check that the following properties hold for any $a \in \mathbb{N}$ and all $i, j \in [1, q]$:

(P0) if $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$ then $\mathsf{p}_i = \mathsf{p}_j$;
(P1) if $\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant a \in \mathsf{S}$ then $\mathsf{p}_i \geqslant a$; (P2) if $\neg\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant a \in \mathsf{S}$ then $\mathsf{p}_i < a$;
(P3) if $\#\,\mathtt{loop}_{\heartsuit} \geqslant a \in \mathsf{S}$ then $\mathsf{l} \geqslant a$; (P4) if $\neg\#\,\mathtt{loop}_{\heartsuit} \geqslant a \in \mathsf{S}$ then $\mathsf{l} < a$;
(P5) if $\#\,\mathtt{rem}_{\heartsuit} \geqslant a \in \mathsf{S}$ then $\mathsf{r} \geqslant a$; (P6) if $\neg\#\,\mathtt{rem}_{\heartsuit} \geqslant a \in \mathsf{S}$ then $\mathsf{r} < a$.

Property (P0) let us assume $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$ and let us show $\mathsf{p}_i \leqslant \mathsf{p}_j$. Let $a \in \mathbb{N}$ and $k \in [1,q]$ be such that $\mathtt{x}_i = \mathtt{x}_k \in \mathrm{cl}(\mathsf{B}^+)$ and $\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_k) \geqslant a \in \mathsf{S}$. Let us show $a \leqslant \mathsf{p}_j$. Since $\mathrm{cl}(\mathsf{B}^+)$ is closed under rules

$$\frac{\mathtt{x}_i = \mathtt{x}_j}{\mathtt{x}_j = \mathtt{x}_i} \qquad \frac{\mathtt{x}_j = \mathtt{x}_i \quad \mathtt{x}_i = \mathtt{x}_k}{\mathtt{x}_j = \mathtt{x}_k}$$

we deduce $\mathtt{x}_j = \mathtt{x}_k \in \mathrm{cl}(\mathsf{B}^+)$. Hence by definition of p_j (max), we get $a \leqslant \mathsf{p}_j$. We conclude $\mathsf{p}_i \leqslant \mathsf{p}_j$. The relation $\mathsf{p}_j \leqslant \mathsf{p}_i$ is derived directly because $\mathtt{x}_j = \mathtt{x}_i \in \mathrm{cl}(\mathsf{B}^+)$ holds as well;

Property (P1) if $\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant a \in \mathsf{S}$ then, as $\mathrm{cl}(\mathsf{B}^+)$ is closed under rule

$$\frac{}{\mathtt{x}_i = \mathtt{x}_i}$$

we deduce $\mathtt{x}_i = \mathtt{x}_i \in \mathrm{cl}(\mathsf{B}^+)$ and thus $a \leqslant \mathsf{p}_i$ by definition of p_i;

Property (P2) let us assume $\neg\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant a \in \mathsf{S}$ and let us show $\mathsf{p}_i < a$. Hence, let $b \in \mathbb{N}$ and $k \in [1,q]$ be such that $\mathtt{x}_i = \mathtt{x}_k \in \mathrm{cl}(\mathsf{B}^+)$ and $\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_k) \geqslant b \in \mathsf{S}$ and let us show $b < a$. From $\mathtt{x}_i = \mathtt{x}_k \in \mathrm{cl}(\mathsf{B}^+)$ we deduce $\mathtt{x}_k = \mathtt{x}_i \in \mathrm{cl}(\mathsf{B}^+)$. As we also have $\{\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_k) \geqslant b, \neg\#\,\mathtt{pred}_{\heartsuit}(\mathtt{x}_i) \geqslant a\} \subseteq \mathsf{S}$, by Property C1.2 (which holds for 1-consistency) we deduce $b < a$. We conclude $\mathsf{p}_i < a$;

Property (P3) if $\#\,\mathtt{loop}_{\heartsuit} \geqslant a \in \mathsf{S}$ then by definition of l we have $a \leqslant \mathsf{l}$;

Property (P4) let us assume $\neg\#\,\mathtt{loop}_{\heartsuit} \geqslant a \in \mathsf{S}$ and let us show $\mathsf{l} < a$. Hence, let $b \in \mathbb{N}$ be s.t. $\#\,\mathtt{loop}_{\heartsuit} \geqslant b \in \mathsf{S}$ and let us show $b < a$. We have $\{\#\,\mathtt{loop}_{\heartsuit} \geqslant b, \neg\#\,\mathtt{loop}_{\heartsuit} \geqslant a\} \subseteq \mathsf{S}$ hence by Property C1.3 we deduce $b < a$. We conclude $\mathsf{l} < a$;

Property (P5) if $\#\,\mathtt{rem}_{\heartsuit} \geqslant a \in \mathsf{S}$ then by definition of r we have $a \leqslant \mathsf{r}$;

Property (P6) let us assume $\neg\#\,\mathtt{rem}_{\heartsuit} \geqslant a \in \mathsf{S}$ and let us show $\mathsf{r} < a$. Hence, let $b \in \mathbb{N}$ be such that $\#\,\mathtt{rem}_{\heartsuit} \geqslant b \in \mathsf{S}$ and let us show $b < a$. We have $\{\#\,\mathtt{rem}_{\heartsuit} \geqslant b, \neg\#\,\mathtt{rem}_{\heartsuit} \geqslant a\} \subseteq \mathsf{S}$ hence by Property C1.4 we deduce $b < a$. We conclude $\mathsf{r} < a$.

From Property (P0), we deduce that in the pre-canonical model (s, h, l) of $\mathrm{cl}(\mathsf{B}^+)$, if $s(\mathtt{x}_i) = s(\mathtt{x}_j)$ then $\mathsf{p}_i = \mathsf{p}_j$ by Proposition 5.3.

Now we show that the conjunction of the formulæ in $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable if one of following properties hold:

(S1) $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 1-consistent and either $\mathtt{alloc}(\mathtt{u}) \notin \mathrm{cl}(\mathsf{B}^+)$ or $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}) \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$;
(S2) $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 2-consistent and $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}) \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$;
(S3) $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 3-consistent.

Let us show (S1). We assume that $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 1-consistent and either $\mathtt{alloc}(\mathtt{u}) \notin \mathrm{cl}(\mathsf{B}^+)$ or $(\mathsf{Eq}_\mathtt{u} \cup \mathsf{To}_\mathtt{u}) \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$ hold, and we show that $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. We consider the canonical pre-model (s, h, l) of $\mathrm{cl}(\mathsf{B}^+)$; see Proposition 5.3. If $\mathtt{alloc}(\mathtt{u}) \notin \mathrm{cl}(\mathsf{B}^+)$ holds then $l \notin \mathrm{dom}(h)$; and if $(\mathsf{Eq}_\mathtt{u} \cup \mathsf{To}_\mathtt{u}) \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$ holds then $l \in \mathfrak{p}\heartsuit(s, h)$. As $\mathrm{dom}(h) \subseteq \heartsuit(s, h) \cup \{l\}$, under any of the two hypothesis $\mathtt{alloc}(\mathtt{u}) \notin \mathrm{cl}(\mathsf{B}^+)$ or $(\mathsf{Eq}_\mathtt{u} \cup \mathsf{To}_\mathtt{u}) \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$ we have $\mathrm{dom}(h) \subseteq \heartsuit(s, h)$. Hence $\mathrm{pred}_{\overline{\heartsuit}}(s, h, i) = \mathrm{loop}_{\overline{\heartsuit}}(s, h) = \mathrm{rem}_{\overline{\heartsuit}}(s, h) = \varnothing$ for any $i \in [1, q]$. Using Proposition 5.5 with Property (P0), there exists a heap h' such that $(s, h, l) \simeq_b (s, h', l)$ and $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', i)) = \mathsf{p}_i$, $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h')) = \mathsf{l}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h')) = \mathsf{r}$. By Properties (P1–6), we derive that (s, h', l) satisfies all the formulæ of S. For instance, if $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a \in \mathsf{S}$ then by (P4) we have $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h')) = \mathsf{l} < a$ and thus $(s, h') \models_l \neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a$. From $(s, h, l) \simeq_b (s, h', l)$, $\mathsf{B}^- \cap \mathrm{cl}(\mathsf{B}^+) = \varnothing$ and Proposition 5.3, we deduce that (s, h', l) satisfies all the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^-$. Hence the conjunction of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable.

Let us show (S2). We assume that $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 2-consistent and $(\mathsf{Eq}_\mathtt{u} \cup \mathsf{To}_\mathtt{u} \cup \mathsf{Fm}_\mathtt{u} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}) \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$ and we show that $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. We can further assume that $\mathtt{alloc}(\mathtt{u}) \in \mathrm{cl}(\mathsf{B}^+)$ and $(\mathsf{Eq}_\mathtt{u} \cup \mathsf{To}_\mathtt{u}) \cap \mathrm{cl}(\mathsf{B}^+) = \varnothing$ because otherwise, as $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 1-consistent, by Property (S1) we already have that $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. Hence have either $\mathsf{Fm}_\mathtt{u} \cap \mathrm{cl}(\mathsf{B}^+) \neq \varnothing$ or $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathrm{cl}(\mathsf{B}^+)$:

- if $\mathtt{u} \hookrightarrow \mathtt{x}_i \in \mathrm{cl}(\mathsf{B}^+)$ for some $i \in [1, q]$. In the canonical pre-model (s, h, l) of $\mathrm{cl}(\mathsf{B}^+)$, we have $l \in \mathrm{pred}_{\overline{\heartsuit}}(s, h, i)$. But since $\mathrm{dom}(h) \subseteq \heartsuit(s, h) \cup \{l\}$, we deduce $\mathrm{pred}_{\overline{\heartsuit}}(s, h, j) = \{l\}$ if $s(\mathtt{x}_i) = s(\mathtt{x}_j)$, $\mathrm{pred}_{\overline{\heartsuit}}(s, h, j) = \varnothing$ if $s(\mathtt{x}_i) \neq s(\mathtt{x}_j)$ and $\mathrm{loop}_{\overline{\heartsuit}}(s, h) = \mathrm{rem}_{\overline{\heartsuit}}(s, h) = \varnothing$. We consider two sub-cases depending on $\{\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant 1 \mid \mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)\} \cap \mathsf{S}$:
  - if $\{\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant 1 \mid \mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)\} \cap \mathsf{S} = \varnothing$. Then let us define a new cardinality assignment $(\mathsf{p}'_1, \ldots, \mathsf{p}'_q, \mathsf{l}, \mathsf{r})$ by $\mathsf{p}'_j = \max(1, \mathsf{p}_j)$ if $\mathtt{x}_i{=}\mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$, $\mathsf{p}'_j = \mathsf{p}_j$ if $\mathtt{x}_i = \mathtt{x}_j \notin \mathrm{cl}(\mathsf{B}^+)$. Let us show that $(\mathsf{p}'_1, \ldots, \mathsf{p}'_q, \mathsf{l}, \mathsf{r})$ satisfies the requirements of Proposition 5.5 for the canonical pre-model (s, h, l) of $\mathrm{cl}(\mathsf{B}^+)$: $s(\mathtt{x}_j) = s(\mathtt{x}_k)$ implies $\mathtt{x}_j = \mathtt{x}_k \in \mathrm{cl}(\mathsf{B}^+)$ implies $\mathsf{p}_j = \mathsf{p}_k$ implies $\mathsf{p}'_j = \mathsf{p}'_k$ for any $j, k \in [1, q]$; if $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$ then $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h, j)) = 1 \leqslant \max(1, \mathsf{p}_j) = \mathsf{p}'_j$; if $\mathtt{x}_i = \mathtt{x}_j \notin \mathrm{cl}(\mathsf{B}^+)$ then $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h, j)) = 0 \leqslant \mathsf{p}'_j$; $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h)) = 0 \leqslant \mathsf{l}$; $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h)) = 0 \leqslant \mathsf{r}$. Using the cardinality assignment $(\mathsf{p}'_1, \ldots, \mathsf{p}'_q, \mathsf{l}, \mathsf{r})$, we extend the canonical pre-model (s, h, l) of $\mathrm{cl}(\mathsf{B}^+)$ using Proposition 5.5 and we get a heap h' such that $(s, h, l) \simeq_b (s, h', l)$, $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', j)) = \max(1, \mathsf{p}_j)$ if $s(\mathtt{x}_i) = s(\mathtt{x}_j)$, $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', j)) = \mathsf{p}_j$ if $s(\mathtt{x}_i) \neq s(\mathtt{x}_j)$, $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h')) = \mathsf{l}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h')) = \mathsf{r}$. From the equivalence $(s, h, l) \simeq_b (s, h', l)$ we deduce that (s, h', l) satisfies all the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^-$. Let us check that (s, h', l) satisfies the formulæ of S:
    - if $\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant a \in \mathsf{S}$ then by Property (P1) we have $a \leqslant \mathsf{p}_j \leqslant \mathsf{p}'_j = \mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', j))$, hence $(s, h') \models_l \#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant a$;
    - if $\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant a \in \mathsf{S}$ then either $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$ in which case $a > 1$ and thus $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', j)) = \max(1, \mathsf{p}_j) < a$ by Property (P2), or $\mathtt{x}_i = \mathtt{x}_j \notin \mathrm{cl}(\mathsf{B}^+)$ in which case $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s, h', j)) = \mathsf{p}_j < a$ by Property (P2). In any case we have $(s, h') \models_l \neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant a$;
    - if $\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a \in \mathsf{S}$ then by Property (P3) we get $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h')) = \mathsf{l} \geqslant a$, hence $(s, h') \models_l \#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a$;
    - if $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a \in \mathsf{S}$ then by Property (P4) we get $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s, h')) = \mathsf{l} < a$, hence $(s, h') \models_l \neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant a$;
    - if $\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant a \in \mathsf{S}$ then by Property (P5) we get $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h')) = \mathsf{r} \geqslant a$, hence $(s, h') \models_l \#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant a$;
    - if $\neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant a \in \mathsf{S}$ then by Property (P6) we get $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s, h')) = \mathsf{r} < a$, hence $(s, h') \models_l \neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant a$;

    We deduce that (s, h', l) satisfies the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$;
  - if $\{\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant 1 \mid \mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)\} \cap \mathsf{S} \neq \varnothing$. Then there exists some $j \in [1, q]$ such that $\neg\#\,\mathtt{pred}_{\overline{\heartsuit}}(\mathtt{x}_j) \geqslant 1 \in \mathsf{S}$ and $\mathtt{x}_i = \mathtt{x}_j \in \mathrm{cl}(\mathsf{B}^+)$. Then by Condition C2.1,

$(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 1-consistent for some $\mathcal{B} \in \mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}$. By Property (S1), we deduce that the conjunction of the formulæ of $\mathsf{B}^+ \cup \{\mathcal{B}\} \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. Hence the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable as well;

- if $\mathtt{u} \hookrightarrow \mathtt{u} \in \mathrm{cl}(\mathsf{B}^+)$. In the canonical pre-model of $\mathrm{cl}(\mathsf{B}^+)$, we have $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) = \mathrm{rem}_{\overline{\heartsuit}}(s,h) = \varnothing$ for any $i \in [1,q]$ and $\mathrm{loop}_{\overline{\heartsuit}}(s,h) = \{l\}$. We consider two sub-cases:
  - either $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1 \notin \mathsf{S}$. As earlier, we extend the canonical pre-model (s,h,l) under the cardinality assignment $(\mathsf{p}_1, \ldots, \mathsf{p}_q, \max(1,\mathsf{l}), \mathsf{r})$ using Proposition 5.5 and we get a heap h' such that $(s,h,l) \simeq_b (s,h',l)$, $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h',i)) = \mathsf{p}_j$ for any $i \in [1,q]$, $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h')) = \max(1,\mathsf{l})$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h')) = \mathsf{r}$. We can then show that (s,h',l) satisfies the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$;
  - or $\neg\#\,\mathtt{loop}_{\overline{\heartsuit}} \geqslant 1 \in \mathsf{S}$. Then by Condition C2.2, $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 1-consistent for some $\mathcal{B} \in \mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}}$. By Property (S1), we deduce that the conjunction of the formulæ of $\mathsf{B}^+ \cup \{\mathcal{B}\} \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. Hence the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable as well.

Let us finally show (S3). We assume that $(\mathsf{B}^+, \mathsf{B}^-, \mathsf{S})$ is 3-consistent and we prove that the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. We further assume that $\mathtt{alloc}(\mathtt{u}) \in \mathrm{cl}(\mathsf{B}^+)$ and $(\mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}) \cap \mathrm{cl}(\mathsf{B}^+) = \varnothing$ because otherwise we can either apply Property (S1) or Property (S2). Hence in the canonical pre-model (s,h,l) of $\mathrm{cl}(\mathsf{B}^+)$, we have $l \in \mathrm{rem}_{\overline{\heartsuit}}(s,h)$. But since $\mathrm{dom}(h) \subseteq \heartsuit(s,h) \cup \{l\}$, we deduce $\mathrm{pred}_{\overline{\heartsuit}}(s,h,i) = \varnothing$ for any $i \in [1,q]$, $\mathrm{loop}_{\overline{\heartsuit}}(s,h) = \varnothing$ and $\mathrm{rem}_{\overline{\heartsuit}}(s,h) = \{l\}$. We consider two cases:

- either $\neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant 1 \notin \mathsf{S}$. As earlier, we extend the canonical model (s,h,l) under the cardinal assignment $(\mathsf{p}_1, \ldots, \mathsf{p}_q, \mathsf{l}, \max(1,\mathsf{r}))$ using Proposition 5.5 and we get a heap h' such that $(s,h,l) \simeq_b (s,h',l)$, $\mathrm{card}(\mathrm{pred}_{\overline{\heartsuit}}(s,h',i)) = \mathsf{p}_i$ for any $i \in [1,q]$, $\mathrm{card}(\mathrm{loop}_{\overline{\heartsuit}}(s,h')) = \mathsf{l}$ and $\mathrm{card}(\mathrm{rem}_{\overline{\heartsuit}}(s,h')) = \max(1,\mathsf{r})$. We deduce that (s,h',l) satisfies the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$;
- or $\neg\#\,\mathtt{rem}_{\overline{\heartsuit}} \geqslant 1 \in \mathsf{S}$. Then by Condition C3.1, $(\mathsf{B}^+ \cup \{\mathcal{B}\}, \mathsf{B}^-, \mathsf{S})$ is 2-consistent for some $\mathcal{B} \in \mathsf{Eq}_{\mathtt{u}} \cup \mathsf{To}_{\mathtt{u}} \cup \mathsf{Fm}_{\mathtt{u}} \cup \{\mathtt{u} \hookrightarrow \mathtt{u}\}$. By Property (S2), we deduce that the conjunction of the formulæ of $\mathsf{B}^+ \cup \{\mathcal{B}\} \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable. Hence the conjunction of the formulæ of $\mathsf{B}^+ \cup \neg\mathsf{B}^- \cup \mathsf{S}$ is satisfiable as well. □