The Bernays-Schönfinkel-Ramsey Class of Separation Logic on Arbitrary Domains

Mnacho Echenim[1], Radu Iosif[2] and Nicolas Peltier[1]

[1] Univ. Grenoble Alpes, CNRS, LIG, F-38000 Grenoble France
[2] Univ. Grenoble Alpes, CNRS, VERIMAG, F-38000 Grenoble France

Abstract. This paper investigates the satisfiability problem for Separation Logic with k record fields, with unrestricted nesting of separating conjunctions and implications, for prenex formulae with quantifier prefix $\exists^*\forall^*$. In analogy with first-order logic, we call this fragment Bernays-Schönfinkel-Ramsey Separation Logic [$\mathsf{BSR}(\mathsf{SL}^k)$]. In contrast to existing work in Separation Logic, in which the universe of possible locations is assumed to be infinite, both finite and infinite universes are considered. We show that, unlike in first-order logic, the (in)finite satisfiability problem is undecidable for $\mathsf{BSR}(\mathsf{SL}^k)$. Then we define two non-trivial subsets thereof, that are decidable for finite and infinite satisfiability respectively, by controlling the occurrences of universally quantified variables within the scope of separating implications, as well as the polarity of the occurrences of the latter. Beside the theoretical interest, our work has natural applications in program verification, for checking that constraints on the shape of a data-structure are preserved by a sequence of transformations.

1 Introduction

Separation Logic [10,14], or SL, is a logical framework used in program verification to describe properties of the dynamically allocated memory, such as topologies of data structures (lists, trees), (un)reachability between pointers, etc.

In a nutshell, given an integer $k \geq 1$, the logic SL^k is obtained from the first-order theory of a finite partial function $h : U \rightharpoonup U^k$ called a *heap*, by adding two substructural connectives: (i) the *separating conjunction* $\phi_1 * \phi_2$, that asserts a split of the heap into disjoint heaps satisfying ϕ_1 and ϕ_2 respectively, and (ii) the *separating implication* or *magic wand* $\phi_1 \mathrel{-\!\!*} \phi_2$, stating that each extension of the heap by a heap satisfying ϕ_1 must satisfy ϕ_2. Intuitively, U is the universe of possible of memory locations (cells) and k is the number of record fields in each memory cell.

The separating connectives $*$ and $\mathrel{-\!\!*}$ allow concise definitions of program semantics, via weakest precondition calculi [10] and easy-to-write specifications of recursive linked data structures (e.g. singly- and doubly-linked lists, trees with linked leaves and parent pointers, etc.), when higher-order inductive definitions are added [14]. Investigating the decidability and complexity of the satisfiability problem for fragments of SL is of theoretical and practical interest. In this paper, we consider prenex SL formulae with prefix $\exists^*\forall^*$. In analogy with first-order logic with equality and uninterpreted predicates [12], we call this fragment Bernays-Schönfinkel-Ramsey SL [$\mathsf{BSR}(\mathsf{SL}^k)$].

As far as we are aware, all existing work on SL assumes that the universe (set of available locations) is countably infinite. This assumption is not necessarily realistic in practice since the available memory is usually finite, although the bound depends on the hardware and is not known in advance. The finite universe hypothesis is especially useful when dealing with bounded memory issues, for instance checking that the execution of a program satisfies its postcondition, provided that there are sufficiently many available memory cells. In this paper we consider both the finite and infinite satisfiability problems. We show that both problems are undecidable for $\mathsf{BSR}(\mathsf{SL}^k)$ (unlike in first-order logic) and that they become PSPACE-complete under some additional restrictions, related to the occurrences of the magic wand and universal variables:

1. The infinite satisfiability problem is PSPACE-complete if the positive occurrences of $-\!\!*$ (i.e., the occurrences of $-\!\!*$ that are in the scope of an even number of negations) contain no universal variables.
2. The finite satisfiability problem is PSPACE-complete if there is no positive occurrence of $-\!\!*$ (i.e., $-\!\!*$ only occurs in the scope of an odd number of negations).

Reasoning on finite domains is more difficult than on infinite ones, due to possibility of asserting cardinality constraints on unallocated cells, which explains that the latter condition is more restrictive than the former one. Actually, the finite satisfiability problem is undecidable even if there is only one positive occurrence of a $-\!\!*$ with no variable within the scope of $-\!\!*$. These results establish sharp decidability frontiers within $\mathsf{BSR}(\mathsf{SL}^k)$.

Undecidability is shown by reduction from BSR first-order formulae with two monadic function symbols. To establish the decidability results, we first show that every quantifier-free SL formula can be transformed into an equivalent boolean combination of formulae of some specific patterns, called *test formulae*. This result is interesting in itself, since it provides a precise and intuitive characterization of the expressive power of SL: it shows that separating connectives can be confined to a small set of test formulae. Afterward, we show that such test formulae can be transformed into first-order formulae. If the above conditions (1) or (2) are satisfied, then the obtained first-order formulae are in the BSR class, which ensures decidability. The PSPACE upper-bound relies on a careful analysis of the maximal size of the test formulae. The analysis reveals that, although the boolean combination of test formulae is of exponential size, its components (e.g., the conjunctions in its dnf) are of polynomial size and can be enumerated in polynomial space. For space reasons, full details and proofs are given in a technical report [7].

Applications. Besides theoretical interest, our work has natural applications in program verification. Indeed, purely universal SL formulae are useful to express pre- or post-conditions asserting "local" constraints on the shape of the data structures manipulated by a program. Consider the atomic proposition $x \mapsto (y_1, \ldots, y_k)$ which states that the value of the heap at x is the tuple $(y_1, \ldots, y_k)$ and there is no value, other than x, in the domain of h. With this in mind, the following formula describes a well-formed doubly-linked list:

$$\forall x_1, x_2, x_3, x_4, x_5 \,.\, x_1 \mapsto (x_2, x_3) * x_2 \mapsto (x_4, x_5) * \top \Rightarrow x_5 \approx x_1 \wedge \neg x_3 \approx x_4 \qquad (1)$$

Such constraints could also be expressed by using inductively defined predicates, unfortunately checking satisfiability of SL formulae, even of very simple fragments with no

occurrence of $-\!\!*$ in the presence of user-defined inductive predicates is undecidable, unless some rather restrictive conditions are fulfilled [9]. In contrast, checking entailment between two universal formulae boils down to checking the satisfiability of a $\mathsf{BSR}(\mathsf{SL}^k)$ formula, which can be done thanks to the decidability results in our paper.

The separating implication (magic wand) seldom occurs in such shape constraints. However, it is useful to describe the dynamic transformations of the data structures, as in the following Hoare-style axiom, giving the weakest precondition of $\forall \mathbf{u} \,.\, \psi$ with respect to redirecting the i-th record field of x to z [10]:

$$\{\mathsf{x} \mapsto (y_1, \ldots, y_k) * [\mathsf{x} \mapsto (y_1, \ldots, y_{i-1}, \mathsf{z}, \ldots, y_k) -\!\!* \forall \mathbf{u} \,.\, \psi]\}\ \mathsf{x.i} := \mathsf{z}\ \{\forall \mathbf{u} \,.\, \psi\}$$

The precondition is equivalent to $\forall \mathbf{u} \,.\, \mathsf{x} \mapsto (y_1, \ldots, y_k) * [\mathsf{x} \mapsto (y_1, \ldots, y_{i-1}, \mathsf{z}, \ldots, y_k) -\!\!* \psi]$ because, although hoisting universal quantifiers outside of the separating conjunction is unsound in general, this is possible here due to the special form of the left-hand side $\mathsf{x} \mapsto (y_1, \ldots, y_{i-1}, \mathsf{z}, \ldots, y_k)$ which unambiguously defines a single heap cell. Therefore, checking that $\forall \mathbf{u} \,.\, \psi$ is an invariant of the program statement $\mathsf{x.i} := \mathsf{z}$ amounts to checking that the formula $\forall u \,.\, \psi \wedge \exists \mathbf{u} \,.\, \neg[\mathsf{x} \mapsto (y_1, \ldots, y_k) * (\mathsf{x} \mapsto (y_1, \ldots, y_{i-1}, \mathsf{z}, \ldots, y_k) -\!\!* \psi)]$ is unsatisfiable. Because the magic wand occurs negated, this formula falls into a decidable class defined in the present paper, for both finite and infinite satisfiability. The complete formalization of this deductive program verification technique and the characterization of the class of programs for which it is applicable is outside the scope of the paper and is left for future work.

Related Work. In contrast to first-order logic for which the decision problem has been thoroughly investigated [1], only a few results are known for SL. For instance, the problem is undecidable in general and PSPACE-complete for quantifier-free formulae [4]. For $k = 1$, the problem is also undecidable, but it is PSPACE-complete if in addition there is only one quantified variable [6] and decidable but nonelementary if there is no magic wand [2]. In particular, we have also studied the prenex form of SL^1 [8] and found out that it is decidable and nonelementary, whereas $\mathsf{BSR}(\mathsf{SL}^1)$ is PSPACE-complete. In contrast, in this paper we show that undecidability occurs for $\mathsf{BSR}(\mathsf{SL}^k)$, for $k \geq 2$.

Expressive completeness results exist for quantifier-free SL^1 [11,2] and for SL^1 with one and two quantified variables [5,6]. There, the existence of equivalent boolean combinations of test formulae is showed implicitly, using a finite enumeration of equivalence classes of models, instead of an effective transformation. Instead, here we present an explicit equivalence-preserving transformation of quantifier-free SL^k into boolean combinations of test formulae, and translate the latter into first-order logic. Further, we extend the expressive completeness result to finite universes, with additional test formulae asserting cardinality constraints on unallocated cells.

Another translation of quantifier-free SL^k into first-order logic with equality has been described in [3]. There, the small model property of quantifier-free SL^k [4] is used to bound the number of first-order variables to be considered and the separating connectives are interpreted as first-order quantifiers. The result is an equisatisfiable first-order formula. This translation scheme cannot be, however, directly applied to $\mathsf{BSR}(\mathsf{SL}^k)$, which does not have a small model property, being moreover undecidable. Theory-parameterized versions of $\mathsf{BSR}(\mathsf{SL}^k)$ have been shown to be undecidable, e.g. when integer linear arithmetic is used to reason about locations, and claimed to be PSPACE-complete for countably infinite and finite unbounded location sorts, with no relation

other than equality [13]. In the present paper, we show that this claim is wrong, and draw a precise chart of decidability for both infinite and finite satisfiability of $\mathsf{BSR}(\mathsf{SL}^k)$.

2 Preliminaries

Basic Definitions. Let $\mathbb{Z}_\infty = \mathbb{Z} \cup \{\infty\}$ and $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$, where for each $n \in \mathbb{Z}$ we have $n + \infty = \infty$ and $n < \infty$. For a countable set S we denote by $||S|| \in \mathbb{N}_\infty$ the cardinality of S. Let Var be a countable set of variables, denoted as x, y, z and U be a sort. Vectors of variables are denoted by $\mathbf{x}$, $\mathbf{y}$, etc. A *function symbol* f has $\#(f) \geq 0$ arguments of sort U and a sort $\sigma(f)$, which is either the boolean sort Bool or U. If $\#(f) = 0$, we call f a *constant*. We use $\bot$ and $\top$ for the boolean constants false and true, respectively. First-order (FO) terms t and formulae φ are defined by the following grammar:

$$t := x \mid f(t_1, \ldots, t_{\#(f)}) \qquad \varphi := \bot \mid \top \mid t \approx t \mid p(t_1, \ldots, t_{\#(p)}) \mid \varphi \wedge \varphi \mid \neg\varphi \mid \exists x \,.\, \varphi$$

where $x \in \mathsf{Var}$, f and p are function symbols, $\sigma(f) = U$ and $\sigma(p) = \mathsf{Bool}$. We write $\varphi_1 \vee \varphi_2$ for $\neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\varphi_1 \rightarrow \varphi_2$ for $\neg\varphi_1 \vee \varphi_2$, $\varphi_1 \leftrightarrow \varphi_2$ for $\varphi_1 \rightarrow \varphi_2 \wedge \varphi_2 \rightarrow \varphi_1$ and $\forall x \,.\, \varphi$ for $\neg\exists x \,.\, \neg\varphi$. The size of a formula φ, denoted as $\mathsf{size}(\varphi)$, is the number of symbols needed to write it down. Let $\mathsf{var}(\varphi)$ be the set of variables that occur free in φ, i.e. not in the scope of a quantifier. A *sentence* φ is a formula where $\mathsf{var}(\varphi) = \emptyset$.

First-order formulae are interpreted over FO-structures (called structures, when no confusion arises) $\mathcal{S} = (\mathfrak{U}, \mathfrak{s}, \mathfrak{i})$, where $\mathfrak{U}$ is a countable set, called the *universe*, the elements of which are called *locations*, $\mathfrak{s} : \mathsf{Var} \rightharpoonup \mathfrak{U}$ is a mapping of variables to locations, called a *store* and $\mathfrak{i}$ interprets each function symbol f by a function $f^{\mathfrak{i}} : \mathfrak{U}^{\#(f)} \rightarrow \mathfrak{U}$, if $\sigma(f) = U$ and $f^{\mathfrak{i}} : \mathfrak{U}^{\#(f)} \rightarrow \{\bot^{\mathfrak{i}}, \top^{\mathfrak{i}}\}$ if $\sigma(f) = \mathsf{Bool}$. A structure $(\mathfrak{U}, \mathfrak{s}, \mathfrak{i})$ is *finite* when $||\mathfrak{U}|| \in \mathbb{N}$ and *infinite* otherwise. We write $\mathcal{S} \models \varphi$ iff φ is true when interpreted in $\mathcal{S}$. This relation is defined recursively on the structure of φ, as usual. When $\mathcal{S} \models \varphi$, we say that $\mathcal{S}$ is a *model* of φ. A formula is [finitely] *satisfiable* when it has a [finite] model. We write $\varphi_1 \equiv \varphi_2$ when $(\mathfrak{U}, \mathfrak{s}, \mathfrak{i}) \models \varphi_1 \Leftrightarrow (\mathfrak{U}, \mathfrak{s}, \mathfrak{i}) \models \varphi_2$, for every structure $(\mathfrak{U}, \mathfrak{s}, \mathfrak{i})$.

The Bernays-Schönfinkel-Ramsey fragment of FO, denoted by $\mathsf{BSR}(\mathsf{FO})$, is the set of sentences $\exists x_1 \ldots \exists x_n \forall y_1 \ldots \forall y_m \,.\, \varphi$, where φ is a quantifier-free formula in which all function symbols f of arity $\#(f) > 0$ have sort $\sigma(f) = \mathsf{Bool}$.

Separation Logic. Let k be a strictly positive integer. The logic SL^k is the set of formulae generated by the grammar:

$$\varphi := \bot \mid \top \mid \mathsf{emp} \mid x \approx y \mid x \mapsto (y_1, \ldots, y_k) \mid \varphi \wedge \varphi \mid \neg\varphi \mid \varphi * \varphi \mid \varphi \mathrel{-\!\!*} \varphi \mid \exists x \,.\, \varphi$$

where $x, y, y_1, \ldots, y_k \in \mathsf{Var}$. The connectives $*$ and $\mathrel{-\!\!*}$ are respectively called the *separating conjunction* and *separating implication* (*magic wand*). We write $\varphi_1 \multimap \varphi_2$ for $\neg(\varphi_1 \mathrel{-\!\!*} \neg\varphi_2)$ ($\multimap$ is called *septraction*). The size and set of free variables of an SL^k formula φ are defined as for first-order formulae.

Given an SL^k formula ϕ and a subformula ψ of ϕ, we say that ψ *occurs at polarity* $p \in \{-1, 0, 1\}$ iff one of the following holds: (i) $\phi = \psi$ and $p = 1$, (ii) $\phi = \neg\phi_1$ and ψ occurs at polarity $-p$ in ϕ_1, (iii) $\phi = \phi_1 \wedge \phi_2$ or $\phi = \phi_1 * \phi_2$, and ψ occurs at polarity p in ϕ_i, for some $i = 1, 2$, or (iv) $\phi = \phi_1 \mathrel{-\!\!*} \phi_2$ and either ψ is a subformula of ϕ_1 and $p = 0$, or ψ occurs at polarity p in ϕ_2. A polarity of $1, 0$ or -1 is also referred to as

positive, neutral or negative, respectively. Note that our notion of polarity is slightly different than usual, because the antecedent of a separating implication is of neutral polarity while the antecedent of an implication is usually of negative polarity. This is meant to strengthen upcoming decidability results, see Remark 1.

SL^k formulae are interpreted over SL-*structures* $I = (\mathfrak{U}, \mathfrak{s}, \mathfrak{h})$, where $\mathfrak{U}$ and $\mathfrak{s}$ are as before and $\mathfrak{h} : \mathfrak{U} \rightharpoonup_{fin} \mathfrak{U}^k$ is a finite partial mapping of locations to k-tuples of locations, called a *heap*. As before, a structure $(\mathfrak{U}, \mathfrak{s}, \mathfrak{h})$ is finite when $||\mathfrak{U}|| \in \mathbb{N}$ and infinite otherwise. We denote by $\mathrm{dom}(\mathfrak{h})$ the domain of the heap $\mathfrak{h}$ and by $||\mathfrak{h}|| \in \mathbb{N}$ the cardinality of $\mathrm{dom}(\mathfrak{h})$. Two heaps $\mathfrak{h}_1$ and $\mathfrak{h}_2$ are *disjoint* iff $\mathrm{dom}(\mathfrak{h}_1) \cap \mathrm{dom}(\mathfrak{h}_2) = \emptyset$, in which case $\mathfrak{h}_1 \uplus \mathfrak{h}_2$ denotes their union. A heap $\mathfrak{h}'$ is an *extension* of $\mathfrak{h}$ iff $\mathfrak{h}' = \mathfrak{h} \uplus \mathfrak{h}''$, for some heap $\mathfrak{h}''$. The relation $(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models \varphi$ is defined inductively, as follows:

$$
\begin{array}{ll}
(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models \mathsf{emp} & \Leftrightarrow \mathfrak{h} = \emptyset \\
(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models x \mapsto (y_1, \ldots, y_k) & \Leftrightarrow \mathfrak{h} = \{\langle \mathfrak{s}(x), (\mathfrak{s}(y_1), \ldots, \mathfrak{s}(y_k)) \rangle\} \\
(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models \varphi_1 * \varphi_2 & \Leftrightarrow \text{there exist disjoint heaps } h_1, h_2 \text{ such that } h = h_1 \uplus h_2 \\
 & \quad \text{and } (\mathfrak{U}, \mathfrak{s}, \mathfrak{h}_i) \models \varphi_i, \text{ for } i = 1, 2 \\
(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models \varphi_1 \mathrel{-\!\!*} \varphi_2 & \Leftrightarrow \text{for all heaps } \mathfrak{h}' \text{ disjoint from } \mathfrak{h} \text{ such that } (\mathfrak{U}, \mathfrak{s}, \mathfrak{h}') \models \varphi_1, \\
 & \quad \text{we have } (\mathfrak{U}, \mathfrak{s}, \mathfrak{h} \uplus \mathfrak{h}') \models \varphi_2
\end{array}
$$

The semantics of equality, boolean and first-order connectives is the usual one. Satisfiability, entailment and equivalence are defined for SL^k as for FO formulae.

The Bernays-Schönfinkel-Ramsey fragment of SL^k, denoted by $\mathsf{BSR}(\mathsf{SL}^k)$, is the set of sentences $\exists x_1 \ldots \exists x_n \forall y_1 \ldots \forall y_m \ . \ \phi$, where ϕ is a quantifier-free SL^k formula. Since there is no function symbol of arity greater than zero in SL^k, there is no restriction, other than the form of the quantifier prefix defining $\mathsf{BSR}(\mathsf{SL}^k)$.

3 Test Formulae for SL^k

We define a small set of SL^k patterns of formulae, possibly parameterized by a positive integer, called *test formulae*. These patterns capture properties related to allocation, points-to relations in the heap and cardinality constraints.

Definition 1. *The following patterns, along with atom $x \approx y$, are called* test formulae*:*

$$
\begin{array}{rl}
x \hookrightarrow \mathbf{y} \stackrel{\mathsf{def}}{=} x \mapsto \mathbf{y} * \top & |U| \geq n \stackrel{\mathsf{def}}{=} \top \mathrel{-\!\!\circ} |h| \geq n, \ n \in \mathbb{N} \\
\mathsf{alloc}(x) \stackrel{\mathsf{def}}{=} x \mapsto \underbrace{(x, \ldots, x)}_{k \text{ times}} \mathrel{-\!\!*} \bot & |h| \geq |U| - n \stackrel{\mathsf{def}}{=} |h| \geq n+1 \mathrel{-\!\!*} \bot, n \in \mathbb{N}
\end{array}
$$

$$
|h| \geq n \stackrel{\mathsf{def}}{=} \begin{cases} |h| \geq n-1 * \neg\mathsf{emp}, & \text{if } n > 0 \\ \top, & \text{if } n = 0 \\ \bot, & \text{if } n = \infty \end{cases}
$$

where $x, y \in \mathsf{Var}$, $\mathbf{y} \in \mathsf{Var}^k$ and $n \in \mathbb{N}_\infty$ is a positive integer or ∞.

The semantics of test formulae is very natural: $x \hookrightarrow \mathbf{y}$ means that x points to vector $\mathbf{y}$, $\mathsf{alloc}(x)$ means that x is allocated, and the arithmetic expressions are interpreted as usual, where $|h|$ and $|U|$ respectively denote the number of allocated cells and the number of locations (possibly ∞). Formally:

Proposition 1. *Given an* SL*-structure* $(\mathfrak{U},\mathfrak{s},\mathfrak{h})$*, the following equivalences hold, for all variables* $x,y_1,\ldots,y_k \in \mathsf{Var}$ *and integers* $n \in \mathbb{N}$*:*

$$(\mathfrak{U},\mathfrak{s},\mathfrak{h}) \models x \hookrightarrow \mathbf{y} \Leftrightarrow \mathfrak{h}(\mathfrak{s}(x)) = \mathfrak{s}(\mathbf{y}) \qquad (\mathfrak{U},\mathfrak{s},\mathfrak{h}) \models |h| \geq |U| - n \Leftrightarrow ||\mathfrak{h}|| \geq ||\mathfrak{U}|| - n$$
$$(\mathfrak{U},\mathfrak{s},\mathfrak{h}) \models |U| \geq n \Leftrightarrow ||\mathfrak{U}|| \geq n \qquad (\mathfrak{U},\mathfrak{s},\mathfrak{h}) \models |h| \geq n \Leftrightarrow ||\mathfrak{h}|| \geq n$$
$$(\mathfrak{U},\mathfrak{s},\mathfrak{h}) \models \mathsf{alloc}(x) \Leftrightarrow \mathfrak{s}(x) \in \mathrm{dom}(\mathfrak{h})$$

Not all atoms of SL^k are test formulae, for instance $x \mapsto \mathbf{y}$ and emp are not test formulae. However, by Proposition 1, we have the equivalences $x \mapsto \mathbf{y} \equiv x \hookrightarrow \mathbf{y} \wedge \neg|h| \geq 2$ and $\mathsf{emp} \equiv \neg|h| \geq 1$. Note that, for any $n \in \mathbb{N}$, the test formulae $|U| \geq n$ and $|h| \geq |U| - n$ are trivially true and false respectively, if the universe is infinite. We write $t < u$ for $\neg(t \geq u)$ and $t \approx u$ for $t \geq u \wedge t < u+1$, where $t,u \in \{n, |h|, |U|, |U| - n \mid n \in \mathbb{N}\}$.

We need to introduce a few notations useful to describe upcoming transformations in a concise and precise way. A *literal* is a test formula or its negation. Unless stated otherwise, we view a conjunction T of literals as a set[3] and we use the same symbol to denote both a set and the formula obtained by conjoining the elements of the set. The equivalence relation $x \approx_T y$ is defined as $T \models x \approx y$ and we write $x \not\approx_T y$ for $T \models \neg x \approx y$. Observe that $x \not\approx_T y$ is not the complement of $x \approx_T y$. For a set X of variables, $|X|_T$ is the number of equivalence classes of $\approx_T$ in X.

Definition 2. *A variable* x *is* allocated *in an* SL*-structure* I *iff* $I \models \mathsf{alloc}(x)$*. For a set of variables* $X \subseteq \mathsf{Var}$*, let* $\mathsf{alloc}(X) \stackrel{\text{def}}{=} \bigwedge_{x \in X} \mathsf{alloc}(x)$ *and* $\mathsf{nalloc}(X) \stackrel{\text{def}}{=} \bigwedge_{x \in X} \neg\mathsf{alloc}(x)$*. For a set* T *of literals, let:*

$$\begin{array}{rcl} \mathsf{av}(T) & \stackrel{\text{def}}{=} & \left\{x \in \mathsf{Var} \mid x \approx_T x',\ T \cap \{\mathsf{alloc}(x'), x' \hookrightarrow \mathbf{y} \mid \mathbf{y} \in \mathsf{Var}^k\} \neq \emptyset\right\} \\ \mathsf{nv}(T) & \stackrel{\text{def}}{=} & \{x \in \mathsf{Var} \mid x \approx_T x',\ \neg\mathsf{alloc}(x') \in T\} \\ \mathsf{fp}_X(T) & \stackrel{\text{def}}{=} & T \cap \{\mathsf{alloc}(x), \neg\mathsf{alloc}(x), x \hookrightarrow \mathbf{y}, \neg x \hookrightarrow \mathbf{y} \mid x \in X, \mathbf{y} \in \mathsf{Var}^k\} \end{array}$$

We let $\#_a(T) \stackrel{\text{def}}{=} |\mathsf{av}(T)|_T$ *be the number of equivalence classes of* $\approx_T$ *containing variables allocated in every model of* T *and* $\#_n(X,T) \stackrel{\text{def}}{=} |X \cap \mathsf{nv}(T)|_T$ *be the number of equivalence classes of* $\approx_T$ *containing variables from* X *that are not allocated in any model of* T*. We also let* $\mathsf{fp}_a(T) \stackrel{\text{def}}{=} \mathsf{fp}_{\mathsf{av}(T)}(T)$*.*

Intuitively, $\mathsf{av}(T)$ [$\mathsf{nv}(T)$] is the set of variables that must be [are never] allocated in every [any] model of T, and $\mathsf{fp}_X(T)$ is the *footprint* of T relative to the set $X \subseteq \mathsf{Var}$, i.e. the set of formulae describing allocation and points-to relations over variables from X. For example, if $T = \{x \approx z, \mathsf{alloc}(x), \neg\mathsf{alloc}(y), \neg z \hookrightarrow \mathbf{y}\}$, then $\mathsf{av}(T) = \{x,z\}$, $\mathsf{nv}(T) = \{y\}$, $\mathsf{fp}_a(T) = \{\mathsf{alloc}(x), \neg z \hookrightarrow \mathbf{y}\}$ and $\mathsf{fp}_{\mathsf{nv}(T)}(T) = \{\neg\mathsf{alloc}(y)\}$.

3.1 From Test Formulae to FO

The introduction of test formulae (Definition 1) is motivated by the reduction of the (in)finite satisfiability problem for quantified boolean combinations thereof to the same problem for FO. The reduction is devised in such a way that the obtained formula is in

[3] The empty set is thus considered to be true.

the BSR class, if possible. Given a quantified boolean combination of test formulae ϕ, the FO formula $\tau(\phi)$ is defined by induction on the structure of ϕ:

$$\begin{array}{rlrl}
\tau(|h| \geq n) & \stackrel{\text{def}}{=} \mathfrak{a}_n & \tau(|U| \geq n) & \stackrel{\text{def}}{=} \mathfrak{b}_n \\
\tau(|h| \geq |U| - n) & \stackrel{\text{def}}{=} \neg \mathfrak{c}_{n+1} & \tau(\neg\phi_1) & \stackrel{\text{def}}{=} \neg\tau(\phi_1) \\
\tau(x \hookrightarrow \mathbf{y}) & \stackrel{\text{def}}{=} \mathfrak{p}(x, y_1, \ldots, y_k) & \tau(\mathsf{alloc}(x)) & \stackrel{\text{def}}{=} \exists y_1 \ldots \exists y_k \,.\, \mathfrak{p}(x, y_1, \ldots, y_k) \\
\tau(\phi_1 \wedge \phi_2) & \stackrel{\text{def}}{=} \tau(\phi_1) \wedge \tau(\phi_2) & \tau(\exists x \,.\, \phi_1) & \stackrel{\text{def}}{=} \exists x \,.\, \tau(\phi_1)
\end{array}$$

where $\mathfrak{p}$ is a $(k+1)$-ary function symbol of sort Bool and $\mathfrak{a}_n, \mathfrak{b}_n$ and $\mathfrak{c}_n$ are constants of sort Bool, for all $n \in \mathbb{N}$. These function symbols are related by the following axioms, where $\mathfrak{u}_n, \mathfrak{v}_n$ and $\mathfrak{w}_n$ are constants of sort U, for all $n > 0$:

$$\begin{array}{ll}
 & P : \forall x \forall \mathbf{y} \forall \mathbf{y}' \,.\, \mathfrak{p}(x, \mathbf{y}) \wedge \mathfrak{p}(x, \mathbf{y}') \rightarrow \bigwedge_{i=1}^{k} y_i \approx y_i' \\
A_0 : \mathfrak{a}_0 & A_n : \left\{ \begin{array}{l} \exists \mathbf{y} \,.\, \mathfrak{a}_n \rightarrow \mathfrak{a}_{n-1} \wedge \mathfrak{p}(\mathfrak{u}_n, \mathbf{y}) \wedge \bigwedge_{i=1}^{n-1} \neg \mathfrak{u}_i \approx \mathfrak{u}_n \\ \wedge\ \forall x \forall \mathbf{y} \,.\, \neg \mathfrak{a}_n \wedge \mathfrak{p}(x, \mathbf{y}) \rightarrow \bigvee_{i=1}^{n-1} x \approx \mathfrak{u}_i \end{array} \right\} \\
B_0 : \mathfrak{b}_0 & B_n : \left\{ \begin{array}{l} \mathfrak{b}_n \rightarrow \mathfrak{b}_{n-1} \wedge \bigwedge_{i=1}^{n-1} \neg \mathfrak{v}_i \approx \mathfrak{v}_n \\ \wedge\ \forall x \,.\, \neg \mathfrak{b}_n \rightarrow \bigvee_{i=1}^{n-1} x \approx \mathfrak{v}_i \end{array} \right\} \\
C_0 : \mathfrak{c}_0 & C_n : \forall \mathbf{y} \,.\, \mathfrak{c}_n \rightarrow \mathfrak{c}_{n-1} \wedge \neg \mathfrak{p}(\mathfrak{w}_n, \mathbf{y}) \wedge \bigwedge_{i=1}^{n-1} \neg \mathfrak{w}_n \approx \mathfrak{w}_i
\end{array}$$

Intuitively, $\mathfrak{p}$ encodes the heap and $\mathfrak{a}_n$ (resp. $\mathfrak{b}_n$) is true iff there are at least n cells in the domain of the heap (resp. in the universe), namely $\mathfrak{u}_1, \ldots, \mathfrak{u}_n$ (resp. $\mathfrak{v}_1, \ldots, \mathfrak{v}_n$). If $\mathfrak{c}_n$ is true, then there are at least n locations $\mathfrak{w}_1, \ldots, \mathfrak{w}_n$ outside of the domain of the heap (free), but the converse does not hold. The C_n axioms do not state the equivalence of $\mathfrak{c}_n$ with the existence of at least n free locations, because such an equivalence cannot be expressed in BSR(FO)[4]. As a consequence, the transformation preserves sat-equivalence only if the formulae $|h| \geq |U| - n$ occur only at negative polarity (see Lemma 1, Point 2). If the domain is infinite then this problem does not arise since the formulae $|h| \geq |U| - n$ are always false.

Definition 3. *For a quantified boolean combination of test formulae ϕ, we let $\mathcal{N}(\phi)$ be the maximum integer n occurring in a test formula θ of the form $|h| \geq n$, $|U| \geq n$, or $|h| \geq |U| - n$ from ϕ and define $\mathcal{A}(\phi) \stackrel{\text{def}}{=} \{P\} \cup \{A_i\}_{i=0}^{\mathcal{N}(\phi)} \cup \{B_i\}_{i=0}^{\mathcal{N}(\phi)} \cup \{C_i\}_{i=0}^{\mathcal{N}(\phi)+1}$ as the set of axioms related to ϕ.*

The relationship between ϕ and $\tau(\phi)$ is stated below.

Lemma 1. *Let ϕ be a quantified boolean combination of test formulae. The following hold, for any universe $\mathfrak{U}$ and any store $\mathfrak{s}$:*

1. *if $(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models \phi$, for a heap $\mathfrak{h}$, then $(\mathfrak{U}, \mathfrak{s}, \mathfrak{i}) \models \tau(\phi) \wedge \mathcal{A}(\phi)$, for an interpretation $\mathfrak{i}$;*
2. *if each test formula $|h| \geq |U| - n$ in ϕ occurs at a negative polarity and $(\mathfrak{U}, \mathfrak{s}, \mathfrak{i}) \models \tau(\phi) \wedge \mathcal{A}(\phi)$ for an interpretation $\mathfrak{i}$ such that $||\mathfrak{p}^{\mathfrak{i}}|| \in \mathbb{N}$, then $(\mathfrak{U}, \mathfrak{s}, \mathfrak{h}) \models \phi$, for a heap $\mathfrak{h}$.*

The translation of $\mathsf{alloc}(x)$ introduces existential quantifiers depending on x. For instance, $\forall x \,.\, \mathsf{alloc}(x)$ is translated as $\forall x \exists y_1 \ldots \exists y_k \,.\, \mathfrak{p}(x, y_1, \ldots, y_k)$, which lies outside

[4] The converse of C_n: $\forall x \,.\, (\neg \mathfrak{c}_n \wedge \forall \mathbf{y} \,.\, \neg \mathfrak{p}(x, \mathbf{y})) \rightarrow \bigvee_{i=1}^{n-1} x \approx \mathfrak{w}_i$ is not in BSR(FO).

of the $\mathsf{BSR}(\mathsf{FO})$ fragment. Because upcoming decidability results (Thm. 2) require that $\tau(\phi)$ be in $\mathsf{BSR}(\mathsf{FO})$, we end this section by delimiting a fragment of SL^k whose translation falls into $\mathsf{BSR}(\mathsf{FO})$.

Lemma 2. *Given an SL^k formula $\varphi = \forall z_1 \ldots \forall z_m \,.\, \phi$, where ϕ is a boolean combination of test formulae containing no positive occurrence of $\mathsf{alloc}(z_i)$ for any $i \in [1,m]$, $\tau(\varphi)$ is equivalent (up to transformation into prenex form) to a $\mathsf{BSR}(\mathsf{FO})$ formula with the same constants and free variables as $\tau(\varphi)$.*

Intuitively, if a formula $\mathsf{alloc}(x)$ occurs negatively then the quantifiers $\exists y_1 \ldots \exists y_k$ added when translating $\mathsf{alloc}(x)$ can be transformed into universal ones by transformation into nnf, and if x is not universal then they may be shifted at the root of the formula since $y_1, \ldots, y_k$ depend only on x. In both cases, the quantifier prefix $\exists^*\forall^*$ is preserved.

4 From Quantifier-Free SL^k to Test Formulae

This section states the expressive completeness result of the paper, namely that any quantifier-free SL^k formula is equivalent, on both finite and infinite models, to a boolean combination of test formulae. Starting from a quantifier-free SL^k formula φ, we define a set $\mu(\varphi)$ of conjunctions of test formulae and their negations, called *minterms*, such that $\varphi \equiv \bigvee_{M \in \mu(\varphi)} M$. Although the number of minterms in $\mu(\varphi)$ is exponential in the size of φ, checking the membership of a given minterm M in $\mu(\varphi)$ can be done in PSPACE. Together with the translation of minterms into FO (§3.1), this fact is used to prove PSPACE membership of the two decidable fragments of $\mathsf{BSR}(\mathsf{SL}^k)$, defined next (§5.2).

4.1 Minterms

A *minterm* M is a set (conjunction) of literals containing: exactly one literal $|h| \geq \min_M$ and one literal $|h| < \max_M$, where $\min_M \in \mathbb{N} \cup \{|U| - n \mid n \in \mathbb{N}\}$ and $\max_M \in \mathbb{N}_\infty \cup \{|U| - n \mid n \in \mathbb{N}\}$, and at most one literal of the form $|U| \geq n$, respectively $|U| < n$.

A minterm may be viewed as an abstract description of a heap. The conditions are for technical convenience only and are not restrictive. For instance, tautological test formulae of the form $|h| \geq 0$ and/or $|h| < \infty$ may be added if needed so that the first condition holds. If M contains two literals $t \geq n_1$ and $t \geq n_2$ with $n_1 < n_2$ and $t \in \{|h|, |U|\}$ then $t \geq n_1$ is redundant and can be removed – and similarly if M contains literals $|h| \geq |U| - n_1$ and $|h| \geq |U| - n_2$. Heterogeneous constraints are merged by performing a case split on the value of $|U|$. For example, if M contains both $|h| \geq |U| - 4$ and $|h| \geq 1$, then the first condition prevails if $|U| \geq 5$ yielding the equivalence disjunction: $|h| \geq 1 \wedge |U| < 5 \vee |h| \geq |U| - 4 \wedge |U| \geq 5$. Thus, in the following, we assume that any conjunction of literals can be transformed into a disjunction of minterms [7].

Definition 4. *Given a minterm M, we define the sets:*

$$M^e \stackrel{\text{def}}{=} M \cap \{x \approx y, \neg x \approx y \mid x, y \in \mathsf{Var}\} \quad M^a \stackrel{\text{def}}{=} M \cap \{\mathsf{alloc}(x), \neg\mathsf{alloc}(x) \mid x \in \mathsf{Var}\}$$
$$M^u \stackrel{\text{def}}{=} M \cap \{|U| \geq n, |U| < n \mid n \in \mathbb{N}\} \quad M^p \stackrel{\text{def}}{=} M \cap \{x \hookrightarrow \mathbf{y}, \neg x \hookrightarrow \mathbf{y} \mid x \in \mathsf{Var}, \mathbf{y} \in \mathsf{Var}^k\}$$

Thus, $M = M^e \cup M^u \cup M^a \cup M^p \cup \{|h| \geq \min_M, |h| < \max_M\}$, for each minterm M. Given a set of variables $X \subseteq \mathsf{Var}$, a minterm M is (1) *E-complete* for X iff for all $x, y \in X$ exactly one of $x \approx y \in M$, $\neg x \approx y \in M$ holds, and (2) *A-complete* for X iff for each $x \in X$ exactly one of $\mathsf{alloc}(x) \in M$, $\neg\mathsf{alloc}(x) \in M$ holds.

For a literal ℓ, we denote by $\overline{\ell}$ its complement, i.e. $\overline{\theta} \stackrel{\mathsf{def}}{=} \neg\theta$ and $\overline{\neg\theta} \stackrel{\mathsf{def}}{=} \theta$, where θ is a test formula. Let $\overline{M}$ be the minterm obtained from M by replacing each literal with its complement. The *complement closure* of M is $\mathsf{cc}(M) \stackrel{\mathsf{def}}{=} M \cup \overline{M}$. Two tuples $\mathbf{y}, \mathbf{y}' \in \mathsf{Var}^k$ are *M-distinct* if $y_i \not\approx_M y'_i$, for some $i \in [1,k]$. Given a minterm M that is E-complete for $\mathsf{var}(M)$, its *points-to closure* is $\mathsf{pc}(M) \stackrel{\mathsf{def}}{=} \bot$ if there exist literals $x \hookrightarrow \mathbf{y}, x' \hookrightarrow \mathbf{y}' \in M$ such that $x \approx_M x'$ and $\mathbf{y}$, $\mathbf{y}'$ are M-distinct, and $\mathsf{pc}(M) \stackrel{\mathsf{def}}{=} M$, otherwise. Intuitively, $\mathsf{pc}(M)$ is $\bot$ iff M contradicts the fact that the heap is a partial function[5]. The *domain closure* of M is $\mathsf{dc}(M) \stackrel{\mathsf{def}}{=} \bot$ if either $\min_M = n_1$ and $\max_M = n_2$ for some $n_1, n_2 \in \mathbb{Z}$ such that $n_1 \geq n_2$, or $\min_M = |U| - n_1$ and $\max_M = |U| - n_2$, where $n_2 \geq n_1$; and otherwise:

$$\begin{array}{rl} \mathsf{dc}(M) \stackrel{\mathsf{def}}{=} M \cup \left\{ |U| \geq \left\lceil \sqrt[k]{\max_{x \in \mathsf{av}(M)}(\delta_x(M) + 1)} \right\rceil \right\} \\ \quad \cup \{|U| \geq n_1 + n_2 + 1 \mid \min_M = n_1, \max_M = |U| - n_2, n_1, n_2 \in \mathbb{N}\} \\ \quad \cup \{|U| < n_1 + n_2 \mid \min_M = |U| - n_1, \max_M = n_2, n_1, n_2 \in \mathbb{N}\} \end{array}$$

where $\delta_x(M)$ is the number of pairwise M-distinct tuples $\mathbf{y}$ for which there exists $\neg x' \hookrightarrow \mathbf{y} \in M$ such that $x \approx_M x'$. Intuitively, $\mathsf{dc}(M)$ asserts that $\min_M < \max_M$ and that the domain contains enough elements to allocate all cells. Essentially, given a structure $(\mathfrak{U}, \mathfrak{s}, \mathfrak{h})$, if $\mathfrak{h}(x)$ is known to be defined and distinct from n pairwise distinct vectors of locations $\mathbf{v}_1, \ldots, \mathbf{v}_n$, then necessarily at least $n+1$ vectors must exist. Since there are $||\mathfrak{U}||^k$ vectors of length k, we must have $||\mathfrak{U}||^k \geq n+1$, hence $||\mathfrak{U}|| \geq \sqrt[k]{n+1}$. For instance, if $M = \{\neg x \hookrightarrow y_i, \mathsf{alloc}(x), y_i \not\approx y_j \mid i, j \in [1,n], i \neq j\}$, then it is clear that M is unsatisfiable if there are less than n locations, since x cannot be allocated in this case.

Definition 5. *A minterm M is* footprint-consistent[6] *if for all $x, x' \in \mathsf{Var}$ and $\mathbf{y}, \mathbf{y}' \in \mathsf{Var}^k$, such that $x \approx_M x'$ and $y_i \approx_M y'_i$ for all $i \in [1,k]$, we have (1) if $\mathsf{alloc}(x) \in M$ then $\neg\mathsf{alloc}(x') \notin M$, and (2) if $x \hookrightarrow \mathbf{y} \in M$ then $\neg\mathsf{alloc}(x'), \neg x' \hookrightarrow \mathbf{y}' \notin M$.*

We are now ready to define a boolean combination of test formulae that is equivalent to $M_1 * M_2$, where M_1 and M_2 are minterms satisfying a number of additional conditions. Let $\mathsf{npto}(M_1, M_2) \stackrel{\mathsf{def}}{=} (M_1 \cap M_2) \cap \{\neg x \hookrightarrow \mathbf{y} \mid x \notin \mathsf{av}(M_1 \cup M_2), \mathbf{y} \in \mathsf{Var}^k\}$ be the set of negative points-to literals common to M_1 and M_2, involving left-hand side variables not allocated in either M_1 or M_2.

---

[5] Note that we do not assert the equality $\mathbf{y} \approx \mathbf{y}'$, instead we only check that it is not falsified. This is sufficient for our purpose because in the following we always assume that the considered minterms are E-complete.

[6] Footprint-consistency is a necessary, yet not sufficient, condition for satisfiability of minterms. For example, the minterm $M = \{x \hookrightarrow y, x' \hookrightarrow y', \neg y \approx y', |h| < 2\}$ is at the same time footprint-consistent and unsatisfiable.

Lemma 3. *Let M_1 and M_2 be two minterms that are footprint-consistent and E-complete for $\mathsf{var}(M_1 \cup M_2)$, with $\mathsf{cc}(M_1^p) = \mathsf{cc}(M_2^p)$. Then $M_1 * M_2 \equiv \mathsf{elim}_*(M_1, M_2)$, where*

$$\mathsf{elim}_*(M_1,M_2) \stackrel{\mathsf{def}}{=} M_1^e \wedge M_2^e \wedge \mathsf{dc}(M_1)^u \wedge \mathsf{dc}(M_2)^u \wedge \tag{2}$$

$$\bigwedge_{x \in \mathsf{av}(M_1),\ y \in \mathsf{av}(M_2)} \neg x \approx y \wedge \mathsf{fp}_a(M_1) \wedge \mathsf{fp}_a(M_2) \wedge \tag{3}$$

$$\mathsf{nalloc}(\mathsf{nv}(M_1) \cap \mathsf{nv}(M_2)) \wedge \mathsf{npto}(M_1,M_2) \wedge \tag{4}$$

$$|h| \geq \min_{M_1} + \min_{M_2} \ \wedge\ |h| < \max_{M_1} + \max_{M_2} - 1 \tag{5}$$

$$\wedge\ \eta_{12} \wedge \eta_{21} \tag{6}$$

and $\eta_{ij} \stackrel{\mathsf{def}}{=} \bigwedge_{Y \subseteq \mathsf{nv}(M_j) \setminus \mathsf{av}(M_i)} \mathsf{alloc}(Y) \rightarrow \begin{pmatrix} |h| \geq \#_a(M_i) + |Y|_{M_i} + \min_{M_j} \\ \wedge\ \#_a(M_i) + |Y|_{M_i} < \max_{M_i} \end{pmatrix}$.

Intuitively, if M_1 and M_2 hold separately, then all heap-independent literals from $M_1 \cup M_2$ must be satisfied (2), the variables allocated in M_1 and M_2 must be pairwise distinct and their footprints, relative to the allocated variables, jointly asserted (3). Moreover, unallocated variables on both sides must not be allocated and common negative points-to literals must be asserted (4). Since the heap satisfying $\mathsf{elim}_*(M_1,M_2)$ is the disjoint union of the heaps for M_1 and M_2, its bounds are the sum of the bounds on both sides (5) and, moreover, the variables that M_2 never allocates $[\mathsf{nv}(M_2)]$ may occur allocated in the heap of M_1 and viceversa, thus the constraints η_{12} and η_{21}, respectively (6).

Next, we show a similar result for the separating implication. For technical convenience, we translate the septraction $M_1 \mathrel{-\!\!\circ} M_2$, instead of $M_1 \mathrel{-\!\!*} M_2$, as an equivalent boolean combination of test formulae. This is without loss of generality, because $M_1 \mathrel{-\!\!*} M_2 \equiv \neg(M_1 \mathrel{-\!\!\circ} \neg M_2)$. Unlike with the case of the separating conjuction (Lemma 3), here the definition of the boolean combination of test formulae depends on whether the universe is finite or infinite.

If the complement of some literal $\ell \in \mathsf{fp}_a(M_1)$ belongs to M_2 then no extension by a heap that satisfies ℓ may satisfy $\overline{\ell}$. Therefore, as an additional simplifying assumption, we suppose that $\mathsf{fp}_a(M_1) \cap \overline{M_2} = \emptyset$, so that $M_1 \mathrel{-\!\!\circ} M_2$ is not trivially unsatisfiable. We write $\phi \equiv^{fin} \psi$ [$\phi \equiv^{inf} \psi$] if ϕ has the same truth value as ψ in all finite [infinite] structures.

Lemma 4. *Let M_1 and M_2 be footprint-consistent minterms that are E-complete for $\mathsf{var}(M_1 \cup M_2)$, such that: M_1 is A-complete for $\mathsf{var}(M_1 \cup M_2)$, $M_2^a \cup M_2^p \subseteq \mathsf{cc}(M_1^a \cup M_1^p)$ and $\mathsf{fp}_a(M_1) \cap \overline{M_2} = \emptyset$.*

Then, $M_1 \mathrel{-\!\!\circ} M_2 \equiv^{fin} \mathsf{elim}^{fin}_{-\!\!\circ}(M_1,M_2)$ and $M_1 \mathrel{-\!\!\circ} M_2 \equiv^{inf} \mathsf{elim}^{inf}_{-\!\!\circ}(M_1,M_2)$, where:

$$\mathsf{elim}^{\dagger}_{-\!\!\circ}(M_1,M_2) \stackrel{\mathsf{def}}{=} \mathsf{pc}(M_1)^e \wedge M_2^e \wedge \mathsf{dc}(M_1)^u \wedge \mathsf{dc}(M_2)^u \wedge \tag{7}$$

$$\mathsf{nalloc}(\mathsf{av}(M_1)) \wedge \mathsf{fp}_{\mathsf{nv}(M_1)}(M_2) \wedge \tag{8}$$

$$|h| \geq \min_{M_2} - \max_{M_1} + 1 \wedge |h| < \max_{M_2} - \min_{M_1} \tag{9}$$

$$\wedge\ \lambda^{\dagger} \tag{10}$$

for $\lambda^{fin} \stackrel{\mathsf{def}}{=} \bigwedge_{Y \subseteq \mathsf{var}(M_1 \cup M_2)} \mathsf{nalloc}(Y) \rightarrow \begin{pmatrix} |h| < |U| - \min_{M_1} - \#_n(Y,M_1) + 1 \\ \wedge\ |U| \geq \min_{M_2} + \#_n(Y,M_1) \end{pmatrix}$, *and* $\lambda^{inf} \stackrel{\mathsf{def}}{=} \top$.

A heap satisfies $M_1 \mathrel{-\!\!\circ} M_2$ iff it has an extension, by a disjoint heap satisfying M_1, that satisfies M_2. Thus, $\mathsf{elim}^{\dagger}_{\mathrel{-\!\!\circ}}(M_1, M_2)$ must entail the heap-independent literals of both M_1 and M_2 (7). Next, no variable allocated by M_1 must be allocated by $\mathsf{elim}^{\dagger}_{\mathrel{-\!\!\circ}}(M_1, M_2)$, otherwise no extension with a heap satisfying M_1 is possible and, moreover, the footprint of M_2 relative to the unallocated variables of M_1 must be asserted (8). The heap's cardinality constraints depend on the bounds of M_1 and M_2 (9) and, if Y is a set of variables not allocated in the heap, these variables can be allocated in the extension (10). Actually, this is where the finite universe assumption first comes into play. If the universe is infinite, then there are enough locations outside the heap to be assigned to Y. However, if the universe is finite, then it is necessary to ensure that there are at least $\#_n(Y, M_1)$ free locations to be assigned to Y (10).

4.2 Translating Quantifier-free SL^k into Minterms

We prove next that each quantifier-free SL^k formula is equivalent to a finite disjunction of minterms:

Lemma 5. *Given a quantifier-free SL^k formula ϕ, there exist two sets of minterms $\mu^{fin}(\phi)$ and $\mu^{inf}(\phi)$ such that the following equivalences hold: (1) $\phi \equiv^{fin} \bigvee_{M \in \mu^{fin}(\phi)} M$, and (2) $\phi \equiv^{inf} \bigvee_{M \in \mu^{inf}(\phi)} M$.*

The formal definition of $\mu^{fin}(\phi)$ and $\mu^{inf}(\phi)$ is given in [7] and omitted for the sake of conciseness and readability. Intuitively, these sets are defined by induction on the structure of the formula. For base cases, the following equivalences are used:

$$x \mapsto \mathbf{y} \equiv x \hookrightarrow \mathbf{y} \wedge |h| \approx 1 \qquad \mathsf{emp} \equiv |h| \approx 0 \qquad x \approx y \equiv x \approx y \wedge |h| \geq 0 \wedge |h| < \infty$$

For formulae $\neg\psi_1$ or $\psi_1 \wedge \psi_2$, the transformation is first applied recursively on ψ_1 and ψ_2, then the obtained formula is transformed into dnf. For formulae $\psi_1 * \psi_2$ or $\psi_1 \mathrel{-\!\!\circ} \psi_2$, the transformation is applied on ψ_1 and ψ_2, then the following equivalences are used to shift $*$ and $\mathrel{-\!\!\circ}$ innermost in the formula:

$$(\phi_1 \vee \phi_2) * \phi \equiv (\phi_1 * \phi) \vee (\phi_2 * \phi) \qquad (\phi_1 \vee \phi_2) \mathrel{-\!\!\circ} \phi \equiv (\phi_1 \mathrel{-\!\!\circ} \phi) \vee (\phi_2 \mathrel{-\!\!\circ} \phi)$$
$$\phi * (\phi_1 \vee \phi_2) \equiv (\phi * \phi_1) \vee (\phi * \phi_2) \qquad \phi \mathrel{-\!\!\circ} (\phi_1 \vee \phi_2) \equiv (\phi \mathrel{-\!\!\circ} \phi_1) \vee (\phi \mathrel{-\!\!\circ} \phi_2)$$

Afterwards, the operands of $*$ and $\mathrel{-\!\!\circ}$ are minterms, and the result is obtained using the equivalences in Lemmas 3 and 4, respectively (up to a transformation into dnf). The only difficulty is that these lemmas impose some additional conditions on the minterms (e.g., being E-complete, or A-complete). However, the conditions are easy to enforce by case splitting, as illustrated by the following example:

Example 1. Consider the formula $x \mapsto x \mathrel{-\!\!\circ} y \mapsto y$. It is easy to check that $\mu^{\dagger}(x \mapsto x) = \{M_1\}$, for $\dagger \in \{fin, inf\}$, where $M_1 = x \hookrightarrow x \wedge |h| \geq 1 \wedge |h| < 2$ and $\mu^{\dagger}(y \mapsto y) = \{M_2\}$, where $M_2 = y \hookrightarrow y \wedge |h| \geq 1 \wedge |h| < 2$. To apply Lemma 4, we need to ensure that M_1 and M_2 are E-complete, which may be done by adding either $x \approx y$ or $x \not\approx y$ to each minterm. We also have to ensure that M_1 is A-complete, thus for $z \in \{x, y\}$, we add either $\mathsf{alloc}(z)$ or $\neg\mathsf{alloc}(z)$ to M_1. Finally, we must have $M_2^a \cup M_2^p \subseteq \mathsf{cc}(M_1^a \cup M_1^p)$, thus we add either $y \hookrightarrow y$ or $\neg y \hookrightarrow y$ to M_1. After removing redundancies, we get (among others) the minterms: $M_1' = x \hookrightarrow x \wedge |h| \geq 1 \wedge |h| < 2 \wedge x \approx y$ and $M_2' = y \hookrightarrow y \wedge |h| \geq 1 \wedge |h| < 2 \wedge x \approx y$. Afterwards we compute $\mathsf{elim}^{fin}_{\mathrel{-\!\!\circ}}(M_1', M_2') = x \approx y \wedge \neg\mathsf{alloc}(x) \wedge |h| \geq 0 \wedge |h| < 1$. ■

As explained in Section 3.1, boolean combinations of minterms can only be transformed into sat-equivalent $\mathsf{BSR}(\mathsf{FO})$ formulae if there is no positive occurrence of test-formulae $|h| \geq |U| - n$ or $\mathsf{alloc}(x)$ (see the conditions in Lemmas 1 (2) and 2). Consequently, we relate the polarity of these formulae in some minterm $M \in \mu^{fin}(\phi) \cup \mu^{inf}(\phi)$ with that of a separating implication within ϕ. The analysis depends on whether the universe is finite or infinite.

Lemma 6. *For any quantifier-free SL^k formula ϕ, the following properties hold:*

1. *For all $M \in \mu^{inf}(\phi)$, we have $M \cap \{|h| \geq |U| - n, |h| < |U| - n \mid n \in \mathbb{N}\} = \emptyset$.*
2. *If $|h| \geq |U| - n \in M$ $[|h| < |U| - n \in M]$ for some minterm $M \in \mu^{fin}(\phi)$, then a formula $\psi_1 \mathrel{-\!\!*} \psi_2$ occurs at a positive [negative] polarity in ϕ.*
3. *If $\mathsf{alloc}(x) \in M$ $[\neg\mathsf{alloc}(x) \in M]$ for some minterm $M \in \mu^{inf}(\phi)$, then a formula $\psi_1 \mathrel{-\!\!*} \psi_2$, such that $x \in \mathsf{var}(\psi_1) \cup \mathsf{var}(\psi_2)$, occurs at a positive [negative] polarity in ϕ.*
4. *If $M \cap \{\mathsf{alloc}(x), \neg\mathsf{alloc}(x) \mid x \in \mathsf{Var}\} \neq \emptyset$ for some minterm $M \in \mu^{fin}(\phi)$, then a formula $\psi_1 \mathrel{-\!\!*} \psi_2$, such that $x \in \mathsf{var}(\psi_1) \cup \mathsf{var}(\psi_2)$, occurs in ϕ at some polarity $p \in \{-1, 1\}$. Moreover, $\mathsf{alloc}(x)$ occurs at a polarity $-p$, only if $\mathsf{alloc}(x)$ is in the scope of a λ^{fin} subformula (10) of a formula $\mathsf{elim}^{fin}_{-\circ}(M_1, M_2)$ used to compute $\bigvee_{M \in \mu^{fin}(\phi)} M$.*

Given a quantifier-free SL^k formula ϕ, the number of minterms occurring in $\mu^{fin}(\phi)$ $[\mu^{inf}(\phi)]$ is exponential in the size of ϕ, in the worst case. Therefore, an optimal decision procedure cannot generate and store these sets explicitly, but rather must enumerate minterms lazily. We show that (i) the size of the minterms in $\mu^{fin}(\phi) \cup \mu^{inf}(\phi)$ is bounded by a polynomial in the size of ϕ, and that (ii) the problem "*given a minterm M, does M occur in $\mu^{fin}(\phi)$ [resp. in $\mu^{inf}(\phi)$]?*" is in PSPACE. To this aim, we define a measure on a quantifier-free formula ϕ, which bounds the size of the minterms in the sets $\mu^{fin}(\phi)$ and $\mu^{inf}(\phi)$, inductively on the structure of the formulae:

$$\begin{array}{rlrl}
\mathcal{M}(x \approx y) & \stackrel{\text{def}}{=} 0 & \mathcal{M}(\bot) & \stackrel{\text{def}}{=} 0 \\
\mathcal{M}(\mathsf{emp}) & \stackrel{\text{def}}{=} 1 & \mathcal{M}(x \mapsto \mathbf{y}) & \stackrel{\text{def}}{=} 2 \\
\mathcal{M}(\neg\phi_1) & \stackrel{\text{def}}{=} \mathcal{M}(\phi_1) & \mathcal{M}(\phi_1 \wedge \phi_2) & \stackrel{\text{def}}{=} \max(\mathcal{M}(\phi_1), \mathcal{M}(\phi_2)) \\
\mathcal{M}(\phi_1 * \phi_2) & \stackrel{\text{def}}{=} \sum_{i=1}^{2} (\mathcal{M}(\phi_i) + ||\mathsf{var}(\phi_i)||) & \mathcal{M}(\phi_1 \mathrel{-\!\!*} \phi_2) & \stackrel{\text{def}}{=} \sum_{i=1}^{2} (\mathcal{M}(\phi_i) + ||\mathsf{var}(\phi_i)||)
\end{array}$$

Definition 6. *A minterm M is $\mathcal{M}$-*bounded *by a formula ϕ, if for each literal $\ell \in M$, the following hold: (i) $\mathcal{M}(\ell) \leq \mathcal{M}(\phi)$ if $\ell \in \{|h| \geq \min_{M_i}, |h| < \max_{M_i}\}$ (ii) $\mathcal{M}(\ell) \leq 2\mathcal{M}(\phi) + 1$, if $\ell \in \{|U| \geq n, |U| < n \mid n \in \mathbb{N}\}$.*

The following lemma provides the desired result:

Lemma 7. *Given a quantifier-free SL^k formula ϕ, each minterm $M \in \mu^{fin}(\phi) \cup \mu^{inf}(\phi)$ is $\mathcal{M}$-bounded by ϕ.*

The proof goes by a careful analysis of the test formulae introduced in Lemmas 3 and 4 or created by minterm transformations (see [7] for details). Since $\mathcal{M}(\phi)$ is polynomially bounded by $\mathsf{size}(\phi)$, this entails that it is possible to check whether $M \in \mu^{fin}(\phi)$ [resp. $\mu^{inf}(\phi)$] using space bounded also by a polynomial in $\mathsf{size}(\phi)$.

Lemma 8. *Given a minterm M and an SL^k formula ϕ, the problems of checking whether $M \in \mu^{fin}(\phi)$ and $M \in \mu^{inf}(\phi)$ are in PSPACE.*

5 Bernays-Schönfinkel-Ramsey SL^k

This section gives the main results of the paper concerning the (un)decidability of the (in)finite satisfiability problems within the $\mathsf{BSR}(\mathsf{SL}^k)$ fragment. We can assume w.l.o.g. that $\mathsf{BSR}(\mathsf{SL}^k)$ is the set of sentences $\forall y_1 \ldots \forall y_m \,.\, \phi$, where ϕ is a quantifier-free SL^k formula, with $\mathsf{var}(\phi) = \{x_1, \ldots, x_n, y_1, \ldots, y_m\}$, where the existentially quantified variables $x_1, \ldots, x_n$ are left free. First, we show that, contrary to $\mathsf{BSR}(\mathsf{FO})$, the satisfiability of $\mathsf{BSR}(\mathsf{SL}^k)$ is undecidable for $k \geq 2$. Second, we carve two nontrivial fragments of $\mathsf{BSR}(\mathsf{SL}^k)$, for which the infinite and finite satisfiability problems are both PSPACE-complete. Technically, these fragments are defined based on restrictions of (i) polarities of the occurrences of the separating implication, and (ii) occurrences of universally quantified variables in the scope of separating implications. These results draw a rather precise chart of decidability within the $\mathsf{BSR}(\mathsf{SL}^k)$ fragment. For $k = 1$, the satisfiability problem of $\mathsf{BSR}(\mathsf{SL}^1)$ is in PSPACE [8] (it is undecidable for arbitrary SL^1 formulae [2] and decidable but nonelementary for *prenex* SL^1 formulae [8]).

5.1 Undecidability of $\mathsf{BSR}(\mathsf{SL}^k)$

Theorem 1. *The finite and infinite satisfiability problems are undecidable for* $\mathsf{BSR}(\mathsf{SL}^k)$.

We provide a brief sketch of the proof, see [7] for details. We consider the finite satisfiability problem of the $[\forall, (0), (2)]_=$ fragment of FO, which consists of sentences of the form $\forall x \,.\, \phi(x)$, where ϕ is a quantifier-free boolean combination of atomic propositions $t_1 \approx t_2$, and t_1, t_2 are terms built using two function symbols f and g, of arity one, the variable x and constant c. It is known (see e.g. [1, Theorem 4.1.8]) that finite satisfiability is undecidable for $[\forall, (0), (2)]_=$. We reduce this problem to $\mathsf{BSR}(\mathsf{SL}^k)$ satisfiability. The idea is to encode the value of f and g into the heap, in such a way that every element x points to $(f(x), g(x))$. Given a sentence $\varphi = \forall x \,.\, \phi(x)$ in $[\forall, (0), (2)]_=$, we proceed by first *flattening* each term in ϕ consisting of nested applications of f and g. The result is an equivalent sentence $\varphi_{flat} = \forall x_1 \ldots \forall x_n \,.\, \phi_{flat}$, in which the only terms are x_i, c, $f(x_i)$, $g(x_i)$, $f(c)$ and $g(c)$, for $i \in [1, n]$. For example, the formula $\forall x \,.\, f(g(x)) \approx c$ is flattened into $\forall x_1 \forall x_2 \,.\, g(x_1) \not\approx x_2 \vee f(x_2) \approx c$. We define the following $\mathsf{BSR}(\mathsf{SL}^2)$ sentences $\varphi^{\dagger}_{\mathsf{sl}}$, for $\dagger \in \{fin, inf\}$:

$$\alpha^{\dagger} \wedge x_c \hookrightarrow (y_c, z_c) \wedge \forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_n \forall z_1 \ldots \forall z_n \,.\, \bigwedge_{i=1}^{n} (x_i \hookrightarrow (y_i, z_i) \rightarrow \phi_{\mathsf{sl}}) \quad (11)$$

where[7] $\alpha^{fin} \stackrel{\text{def}}{=} \forall x \,.\, \mathsf{alloc}(x)$ or $\alpha^{fin} \stackrel{\text{def}}{=} |h| \geq |U| - 0$, $\alpha^{inf} \stackrel{\text{def}}{=} \forall x \forall y \forall z \,.\, x \hookrightarrow (y, z) \rightarrow \mathsf{alloc}(y) \wedge \mathsf{alloc}(z)$ and ϕ_{sl} is obtained from ϕ_{flat} by replacing each occurrence of c by x_c, each term $f(c)$ $[g(c)]$ by y_c $[z_c]$ and each term $f(x_i)$ $[g(x_i)]$ by y_i $[z_i]$. Intuitively, α^{fin} asserts that the heap is a total function, and α^{inf} states that every referenced cell is allocated. It is easy to check that φ and φ_{sl} are equisatisfiable. Note that the undecidability result still holds for finite satisfiability if a single occurrence of $-\!\!*$ is allowed, in a (ground) formula $|h| \geq |U| - 0$ (see the definition of α^{fin} above).

[7] Note that the two definitions of α^{fin} are equivalent. The formula α^{fin} is unsatisfiable on infinite universes, which explains why the definitions of α^{fin} and α^{inf} differ.

5.2 Two Decidable Fragments of $\mathsf{BSR}(\mathsf{SL}^k)$

The reductions (11) use either positive occurences of $\mathsf{alloc}(x)$, where x is universally quantified, or test formulae $|h| \geq |U| - n$. We obtain decidable subsets of $\mathsf{BSR}(\mathsf{SL}^k)$ by eliminating the positive occurrences of both (i) $\mathsf{alloc}(x)$, with x universally quantified, and (ii) $|h| \geq |U| - n$, from $\mu^{\dagger}(\phi)$, where $\dagger \in \{fin, inf\}$ and $\forall y_1 \ldots \forall y_m \,.\, \phi$ is any $\mathsf{BSR}(\mathsf{SL}^k)$ formula. Note that $\mu^{inf}(\phi)$ does not contain formulae of the form $|h| \geq |U| - n$ anyway, which explains why slightly less restrictive conditions are needed for infinite structures.

Definition 7. *Given an integer $k \geq 1$, we define:*

1. $\mathsf{BSR}^{inf}(\mathsf{SL}^k)$ *as the set of sentences $\forall y_1 \ldots \forall y_m \,.\, \phi$ such that for all $i \in [1,m]$ and all formulae $\psi_1 \mathrel{-\!\!*} \psi_2$ occurring at polarity 1 in ϕ, we have $y_i \notin \mathsf{var}(\psi_1) \cup \mathsf{var}(\psi_2)$,*
2. $\mathsf{BSR}^{fin}(\mathsf{SL}^k)$ *as the set of sentences $\forall y_1 \ldots \forall y_m \,.\, \phi$ such that no formula $\psi_1 \mathrel{-\!\!*} \psi_2$ occurs at polarity 1 in ϕ.*

Note that $\mathsf{BSR}^{fin}(\mathsf{SL}^k) \subsetneq \mathsf{BSR}^{inf}(\mathsf{SL}^k) \subsetneq \mathsf{BSR}(\mathsf{SL}^k)$, for any $k \geq 1$.

Remark 1. Observe that, because the polarity of the antecedent of a $\mathrel{-\!\!*}$ is neutral, the conditions of Definition 7 impose no constraint on the occurrences of separating implications at the *left* of a separating implication[8].

The decidability result of this paper is stated below:

Theorem 2. *For any integer $k \geq 1$ not depending on the input, the infinite satisfiability problem for $\mathsf{BSR}^{inf}(\mathsf{SL}^k)$ and the finite satisfiability problem for $\mathsf{BSR}^{fin}(\mathsf{SL}^k)$ are both PSPACE-complete.*

We provide a brief sketch of the proof (all details are available in [7]). In both cases, PSPACE-hardness is an immediate consequence of the fact that the quantifier-free fragment of SL^k, without the separating implication, but with the separating conjunction and negation, is PSPACE-hard [4, Proposition 5]. For PSPACE-membership, consider a formula φ in $\mathsf{BSR}^{inf}(\mathsf{SL}^k)$, and its equivalent disjunction of minterms φ' (note that φ' cannot be computed explicitly since it is of exponential size). Lemma 8 gives us an upper bound on the size of test formulae in φ', hence on the number of constant symbols occurring in $\tau(\varphi')$. This, in turns, gives a bound on the cardinality of the model of $\tau(\varphi')$. Indeed, it is known that any satisfiable $\mathsf{BSR}(\mathsf{FO})$ sentence has a finite model with at most $\max(1,n)$ locations, where n is the length of the existential quantifier prefix (see, e.g., [1, Proposition 6.2.17]). We may thus guess such an interpretation, and check that it is indeed a model of $\tau(\varphi')$ by enumerating all the minterms in φ' (this is feasible in polynomial space thanks to Lemma 8) and translating them on-the-fly into first-order formulae. The only subtle point is that the model obtained in this way is finite, whereas our aim is to test that the obtained formula has a *infinite* model. This difficulty can be overcome by adding an axiom ensuring that the domain contains more *unallocated* elements than the total number of constant symbols and variables in the formula. This is

[8] The idea is that if a formula $\mathsf{alloc}(x)$ or $|h| \geq |U| - n$ occurs in the antecedent of a $\mathrel{-\!\!*}$, then it will be eliminated by the transformation in Lemma 4. In contrast, such test formulae will not be eliminated if they occur in the subsequent of a $\mathrel{-\!\!*}$.

sufficient to prove that the obtained model – although finite – can be extended into an infinite model, obtained by creating infinitely many copies of these elements.

The proof for $\mathsf{BSR}^{fin}(\mathsf{SL}^k)$ is similar, but far more involved. The problem is that, if the universe is finite, then $\mathsf{alloc}(x)$ test formulae may occur at a positive polarity, even if every $\phi_1 \mathrel{-\!\!*} \phi_2$ subformula occurs at a negative polarity, due to the positive occurrences of $\mathsf{alloc}(x)$ within λ^{fin} (10) in the definition of $\mathsf{elim}^{fin}_{-\!\!*}(M_1, M_2)$, used for the elimination of separating implications. As previously discussed, positive occurrences of $\mathsf{alloc}(x)$ hinder the translation into $\mathsf{BSR}(\mathsf{FO})$, because of the existential quantifiers that may occur in the scope of a universal quantifier. The solution is to distinguish a class of finite structures $(\mathfrak{U}, \mathfrak{s}, \mathfrak{h})$, the so-called α-*controlled structures*, for some $\alpha \in \mathbb{N}$, for which there exists a set of locations $\ell_1, \ldots, \ell_\alpha$, such that every location $\ell \in \mathfrak{U}$ is either ℓ_i or points to a tuple from the set $\{\ell_1, \ldots, \ell_\alpha, \ell\}$. For such structures, the formulae $\mathsf{alloc}(x)$ where x is universal can be eliminated in a straightforward way because they are equivalent to $\bigwedge_{i=1}^{\alpha}(x \approx \ell_i \rightarrow \mathsf{alloc}(\ell_i))$ (indeed, cells different from $\ell_1, \ldots, \ell_\alpha$ are always allocated, by definition of α-controlled structures). If the structure is not α-controlled, then we can show that there exist sufficiently many unallocated cells, so that all the cardinality constraints of the form $|h| \leq |U| - n$ or $|U| \geq n$ are always satisfied. This ensures that the truth value of the positive occurrences of $\mathsf{alloc}(x)$ are irrelevant, because they only occur in formulae λ^{fin} that are always true if all test formulae $|h| \leq |U| - n$ or $|U| \geq n$ are true (see the definition of λ^{fin} in Lemma 4).

6 Conclusions and Future Work

We have studied the decidability problem for the class of SL formulae with quantifier prefix in the language $\exists^*\forall^*$, denoted as $\mathsf{BSR}(\mathsf{SL}^k)$. Although the fragment was found to be undecidable, we identified two non-trivial subfragments for which the infinite and finite satisfiability are PSPACE-complete. These fragments are defined by restricting the use of universally quantified variables within the scope of separating implications that occur at positive polarity. In practice, the universal quantifiers and separating conjunctions are useful to express local constraints on the shape of the data-structure, whereas the separating implications allow one to express dynamic transformations of these data-structures. As a consequence, separating implications usually occur negatively in the formulae tested for satisfiability, and the decidable classes found in this work are of great practical interest. Future work involves formalizing and implementing an invariant checking algorithm based on the above ideas, and using the techniques for proving decidability (namely the translation of quantifier-free $\mathsf{SL}(k)$ formulae into boolean combinations of test formulae) to solve other logical problems, such as frame inference, abduction and possibly interpolation.

Acknowledgments

The authors wish to acknowledge the contributions of Stéphane Demri and Étienne Lozes to the insightful discussions during the early stages of this work.

References

1. Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Perspectives in Mathematical Logic. Springer, 1997.
2. Rémi Brochenin, Stéphane Demri, and Etienne Lozes. On the almighty wand. *Information and Computation*, 211:106 – 137, 2012.
3. Cristiano Calcagno, Philippa Gardner, and Matthew Hague. From separation logic to first-order logic. In *Foundations of Software Science and Computational Structures*, pages 395–409, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
4. Cristiano Calcagno, Hongseok Yang, and Peter W. O'Hearn. Computability and complexity results for a spatial assertion language for data structures. In *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science*, pages 108–119, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.
5. Stéphane Demri and Morgan Deters. Expressive completeness of separation logic with two variables and no separating conjunction. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 37:1–37:10. ACM, 2014. URL: http://doi.acm.org/10.1145/2603088.2603142, doi:10.1145/2603088.2603142.
6. Stéphane Demri, Didier Galmiche, Dominique Larchey-Wendling, and Daniel Méry. Separation logic with one quantified variable. *Theory Comput. Syst.*, 61(2):371–461, 2017. URL: https://doi.org/10.1007/s00224-016-9713-1, doi:10.1007/s00224-016-9713-1.
7. M. Echenim, R. Iosif, and N. Peltier. On the Expressive Completeness of Bernays-Sch\"onfinkel-Ramsey Separation Logic. *ArXiv e-prints*, 2018. arXiv:1802.00195.
8. Mnacho Echenim, Radu Iosif, and Nicolas Peltier. The complexity of prenex separation logic with one selector. *CoRR*, abs/1804.03556, 2018. URL: http://arxiv.org/abs/1804.03556, arXiv:1804.03556.
9. Radu Iosif, Adam Rogalewicz, and Jiri Simacek. The tree width of separation logic with recursive definitions. In *Proc. of CADE-24*, volume 7898 of *LNCS*, 2013.
10. Samin S Ishtiaq and Peter W O'Hearn. Bi as an assertion language for mutable data structures. In *ACM SIGPLAN Notices*, volume 36, pages 14–26, 2001.
11. Étienne Lozes. *Expressivité des logiques spatiales*. Thèse de doctorat, Laboratoire de l'Informatique du Parallélisme, ENS Lyon, France, November 2004. URL: http://www.lsv.ens-cachan.fr/Publis/PAPERS/PS/PhD-lozes.ps.
12. F. P. Ramsey. On a problem of formal logic. *Classic Papers in Combinatorics*, pages 1–24, 1987.
13. Andrew Reynolds, Radu Iosif, and Cristina Serban. Reasoning in the bernays-schönfinkel-ramsey fragment of separation logic. In Ahmed Bouajjani and David Monniaux, editors, *Verification, Model Checking, and Abstract Interpretation*, pages 462–482, Cham, 2017. Springer International Publishing.
14. John C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science*, LICS '02, pages 55–74. IEEE Computer Society, 2002.