Decidable Entailments in Separation Logic with Inductive Definitions: Beyond Establishment

Mnacho Echenim
Univ. Grenoble Alpes, CNRS, LIG, F-38000 Grenoble France

Radu Iosif
Univ. Grenoble Alpes, CNRS, VERIMAG, F-38000 Grenoble France

Nicolas Peltier
Univ. Grenoble Alpes, CNRS, LIG, F-38000 Grenoble France

Abstract

We define a class of Separation Logic [10, 16] formulæ, whose entailment problem given formulæ \( \phi, \psi_1, \ldots, \psi_n \), is every model of \( \phi \) a model of some \( \psi_i \)? is 2-EXPTIME-complete. The formulæ in this class are existentially quantified separating conjunctions involving predicate atoms, interpreted by the least sets of store-heap structures that satisfy a set of inductive rules, which is also part of the input to the entailment problem. Previous work [8, 12, 15] consider established sets of rules, meaning that every existentially quantified variable in a rule must eventually be bound to an allocated location, i.e. from the domain of the heap. In particular, this guarantees that each structure has treewidth bounded by the size of the largest rule in the set. In contrast, here we show that establishment, although sufficient for decidability (alongside two other natural conditions), is not necessary, by providing a condition, called equationally restrictedness, which applies syntactically to (dis-)equalities. The entailment problem is more general in this case, because equationally restricted rules define richer classes of structures, of unbounded treewidth. In this paper we show that (1) every established set of rules can be converted into an equationally restricted one and (2) the entailment problem is 2-EXPTIME-complete in the latter case, thus matching the complexity of entailments for established sets of rules [12, 15].

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Separation logic, Induction definitions, Inductive theorem proving, Entailments, Complexity

1 Introduction

Separation Logic (SL) [10, 16] is widely used to reason about programs manipulating recursively linked data structures, being at the core of several industrial-scale static program analysis techniques [3, 2, 5]. Given an integer \( R \geq 1 \), denoting the number of fields in a record datatype, and an infinite set \( L \) of memory locations (addresses), the assertions in this logic describe heaps, that are finite partial functions mapping locations to records, i.e., \( R \)-tuples of locations. A location \( \ell \) in the domain of the heap is said to be allocated and the points-to atom \( x \mapsto (y_1, \ldots, y_R) \) states that the location associated with \( x \) refers to the tuple of locations associated with \( (y_1, \ldots, y_R) \). The separating conjunction \( \phi \land \psi \) states that the formulæ \( \phi \) and \( \psi \) hold in non-overlapping parts of the heap, that have disjoint domains. This connective allows for modular program analyses, because the formulæ specifying the behaviour of a program statement refer only to the small (local) set of locations that are manipulated by that statement, with no concern for the rest of the program’s state.

Formulæ consisting of points-to atoms connected with separating conjunctions describe heaps of bounded size only. To reason about recursive data structures of unbounded sizes (lists, trees, etc.), the base logic is enriched by predicate symbols, with a semantics specified by user-defined inductive rules. For instance, the rules: \( \text{excls}(x, y) \iff \exists z \cdot x \mapsto (z, y) \land z \neq c \) and \( \text{excls}(x, y) \iff \exists z \forall v \cdot x \mapsto (z, v) \land \text{excls}(v, y) \land z \neq c \) describe a non-empty list segment, whose elements are records with two fields: the first is a data field, that keeps a list of locations, which excludes the location assigned to the
The treewidth of a graph is a parameter measuring how close the graph is to a tree, see [7, Ch. 11] for a definition.

An important problem in program verification, arising during construction of Hoare-style correctness proofs, is the discharge of verification conditions, that are entailments of the form \( \phi \models \psi_1, \ldots, \psi_n \), where \( \phi \) and \( \psi_i \), \( i = 1, \ldots, n \), are separating conjunctions of points-to predicates and (dis-)equalities, also known as symbolic heaps. The entailment problem then asks if every model of \( \phi \) is a model of some \( \psi_i \). In general, the entailment problem is undecidable and becomes decidable when the inductive rules used to interpret the predicates satisfy three restrictions [8]: (1) progress, stating that each rule allocates exactly one memory cell, (2) connectivity, ensuring that the allocated memory cells form a tree-shaped structure, and (3) establishment, stating that all existentially quantified variables introduced by an inductive rule must be assigned to some allocated memory cell, in every structure defined by that rule. For instance, the above rules are progressing and connected but not established, because the \( \exists z \) variables are not explicitly assigned an allocated location, unlike the \( \forall v \) variables, passed as first parameter of the \( \text{excls}(x, y) \) predicate, and thus always allocated by the points-to atoms \( x \mapsto (z, y) \) or \( x \mapsto (z, v) \), from the first and second rule defining \( \text{excls}(x, y) \), respectively.

The argument behind the decidability of a progressing, connected and established entailment problem is that every model of the left-hand side is encoded by a graph whose treewidth\(^1\) is bounded by the size of the largest symbolic heap that occurs in the problem [8]. Moreover, the progress and connectivity conditions ensure that the set of models of a symbolic heap can be represented by a Monadic Second Order (MSO) logic formula interpreted over graphs, that can be effectively built from the symbolic heap and the set of rules of the problem. The decidability of entailments follows then from the decidability of the satisfiability problem for MSO over graphs of bounded treewidth (Courcelle’s Theorem) [4]. Initially, no upper bound better than elementary recursive was known to exist. Recently, a 2-EXPTIME algorithm was proposed [12, 14] for sets of rules satisfying these three conditions, and, moreover, this bound was shown to be tight [6].

Several natural questions arise: are the progress, connectivity and establishment conditions really necessary for the decidability of entailments? How much can these restriction be relaxed, without jeopardizing the complexity of the problem? Can one decide entailments that involve sets of heaps of unbounded treewidth? In this paper, we answer these questions by showing that entailments are still 2-EXPTIME-complete when the establishment condition is replaced by a condition on the (dis-)equalities occurring in the symbolic heaps of the problem. Informally, such (dis-)equations must be of the form \( x = c \ (x \neq c) \), where \( c \) ranges over some finite and fixed set of globally visible constants (including special symbols such as \( \text{nil} \), that denotes a non_allocated address, but also any free variable occurring on the left-hand side of the entailment). We also relax slightly the progress and connectivity conditions, by allowing forest-like heap structures (instead of just trees), provided that every root is mapped to a constant symbol. These entailment problems are called equationally restricted (e-restricted, for short). For instance, the entailment problem \( \text{excls}(x, y) \ast \text{excls}(y, z) \ast \text{excls}(x, z) \), with the above rules, falls in this category.

We prove that the e-restricted condition loses no generality compared to establishment, because any established entailment problem can be transformed into an equivalent e-restricted entailment problem. E-restricted problems allow reasoning about structures that contain dangling pointers, which frequently occur in practice, especially in the context of modular program analysis. Moreover, the set of structures considered in an e-restricted entailment problem may contain infinite sequences of heaps of strictly increasing treewidths, that are out of the scope of established problems [8].

The decision procedure for e-restricted problems proposed in this paper is based on a similar idea as the one given, for established problems, in [14, 15]. We build a suitable abstraction of the set

---

\(^1\) The treewidth of a graph is a parameter measuring how close the graph is to a tree, see [7, Ch. 11] for a definition.
of structures satisfying the left-hand side of the entailment bottom-up, starting from points-to and predicate atoms, using abstract operators to compose disjoint structures, to add and remove variables, and to unfold the inductive rules associated with the predicates. The abstraction is precise enough to allow checking that all the models of the left-hand side fulfill the right-hand side of the entailment and also general enough to ensure termination of the entailment checking algorithm.

Although both procedures are similar, there are essential differences between our work and [14, 15]. First, we show that instead of using a specific language for describing those abstractions, the considered set of structures can themselves be defined in SL, by means of formulæ of some specific pattern called core formulæ. Second, the fact that the systems are not established makes the definition of the procedure much more difficult, due to the fact that the considered structures can have an unbounded treewidth. This is problematic because, informally, this boundedness property is essential to ensure that the abstractions can be described using a finite set of variables, denoting the frontier of the considered structures, namely the locations that can be shared with other structures. In particular, the fact that disjoint heaps may share unallocated (or “unnamed”) locations complexifies the definition of the composition operator. This problem is overcome by considering a specific class of structures, called normal structures, of bounded treewidth, and proving that the validity of an entailment can be decided by considering only normal structures.

In terms of complexity, we show that the running time of our algorithm is doubly exponential w.r.t. the maximal size among the symbolic heaps occurring in the input entailment problem (including those in the rules) and simply exponential w.r.t. the number of such symbolic heaps (hence w.r.t. the number of rules). This means that the 2-EXPTIME upper bound is preserved by any reduction increasing exponentially the number of rules, but increasing only polynomially the size of the rules. On the other hand, the 2-EXPTIME-hard lower bound is proved by a reduction from the membership problem for exponential-space bounded Alternating Turing Machines [6]. Due to space restrictions, most proofs are shifted to an appendix.

2 Separation Logic with Inductive Definitions

Let $\mathbb{N}$ denote the set of natural numbers. For a countable set $S$, we denote by $||S|| \in \mathbb{N} \cup \{\infty\}$ its cardinality. For a partial mapping $f : A \rightarrow B$, let $\text{dom}(f) \overset{\text{def}}{=} \{x \in A \mid f(x) \in B\}$ and $\text{rng}(f) \overset{\text{def}}{=} \{f(x) \mid x \in \text{dom}(f)\}$ be its domain and range, respectively. We say that $f$ is total if $\text{dom}(f) = A$, written $f : A \rightarrow B$ and finite, written $f : A \rightarrow^\text{fin} B$ if $||\text{dom}(f)|| < \infty$. Given integers $n$ and $m$, we denote by $[n \ldots m]$ the set \{n, n+1, ..., m\}, so that $[n \ldots n] = \emptyset$ if $n > m$. For a relation $\triangleleft \subseteq A \times A$, we denote by $\triangleleft^*$ its reflexive and transitive closure.

For an integer $n \geq 0$, let $A^n$ be the set of $n$-tuples with elements from $A$. Given a tuple $\mathbf{a} = (a_1, \ldots, a_n)$ and $i \in [1 \ldots n]$, we denote by $a_i$ the $i$-th element of $\mathbf{a}$ and by $|\mathbf{a}| \overset{\text{def}}{=} n$ its length. By $f(\mathbf{a})$ we denote the tuple obtained by the pointwise application of $f$ to the elements of $\mathbf{a}$. If multiplicity and order of the elements are not important, we blur the distinction between tuples and sets, using the set-theoretic notations $x \in \mathbf{a}, \mathbf{a} \cup \mathbf{b}, \mathbf{a} \cap \mathbf{b}$ and $\mathbf{a} \setminus \mathbf{b}$.

Let $\mathcal{V} = \{x, y, \ldots\}$ be an infinite countable set of logical first-order variables and $\mathcal{P} = \{p, q, \ldots\}$ be an infinite countable set (disjoint from $\mathcal{V}$) of relation symbols, called predicates, where each predicate $p$ has arity $\#p \geq 0$. We also consider a finite set $\mathcal{C}$ of constants, of known bounded cardinality, disjoint from both $\mathcal{V}$ and $\mathcal{P}$. Constants will play a special rôle in the upcoming developments and the fact that $\mathcal{C}$ is bounded is of a particular importance. A term is either a variable or a constant and we denote by $\mathcal{T} \overset{\text{def}}{=} \mathcal{V} \cup \mathcal{C}$ the set of terms.

Throughout this paper we consider an integer $R \geq 1$ that, intuitively, denotes the number of fields in a record datatype. Although we do not assume $R$ to be a constant in any of the algorithms presented in the following, considering that every datatype has exactly $R$ records simplifies the definition. The
logic $SL^R$ is the set of formulæ generated inductively by the syntax:

$$
\phi \ := \ \text{emp}\ | \ t_0 \mapsto (t_1, \ldots, t_k) \ | \ p(t_1, \ldots, t_{n_p}) \ | \ t_1 = t_2 \ | \ \phi_1 \ast \phi_2 \ | \ \phi_1 \land \phi_2 \ | \ \neg \phi_1 \ | \ \exists x \ . \ \phi_1
$$

where $p \in \mathbb{P}$, $t_i \in \mathbb{T}$ and $x \in \mathbb{V}$. Atomic propositions of the form $t_0 \mapsto (t_1, \ldots, t_k)$ are called points-to atoms and those of the form $p(t_1, \ldots, t_{n_p})$ are predicate atoms. If $R = 1$, we write $t_0 \mapsto t_1$ for $t_0 \mapsto (t_1)$.

The connective $\ast$ is called separating conjunction, in contrast with the classical conjunction $\land$. The size of a formula $\phi$, denoted by $\text{size}(\phi)$, is the number of occurrences of symbols in it. We write $\text{fv}(\phi)$ for the set of free variables in $\phi$ and $\text{trm}(\phi) \equiv \text{fv}(\phi) \cup \mathbb{C}$. A formula is predicate-free if it has no predicate atoms. As usual, $\phi_1 \lor \phi_2 \overset{\text{df}}{=} \neg (\neg \phi_1 \land \neg \phi_2)$ and $\forall x \ . \ \phi \overset{\text{df}}{=} \neg \exists x \ . \ \neg \phi$. For a set of variables $x = \{x_1, \ldots, x_n\}$ and a quantifier $Q \in \{\exists, \forall\}$, we write $Qx \ . \ \phi \overset{\text{df}}{=} Qx_1 \ldots Qx_n \ . \ \phi$. By writing $t_1 = t_2 (\phi_1 = \phi_2)$ we mean that the terms (formulæ) $t_1$ and $t_2$ ($\phi_1$ and $\phi_2$) are syntactically the same.

A substitution is a partial mapping $\sigma : \mathbb{V} \rightarrow \mathbb{T}$ that maps variables to terms. We denote by $[t_1/x_1, \ldots, t_n/x_n]$ the substitution that maps the variable $x_i$ to $t_i$, for each $i \in \{1 \ldots n\}$ and is undefined elsewhere. By $\sigma \phi$ we denote the formula obtained from $\phi$ by substituting each variable $x \in \text{fv}(\phi)$ by $\sigma(x)$ (we assume that bound variables are renamed to avoid collisions if needed). By abuse of notation, we sometimes write $\sigma(x)$ for $x$, when $x \notin \text{dom}(\sigma)$.

To interpret $SL^R$ formulæ, we consider an infinite countable set $\mathbb{L}$ of locations. The semantics of $SL^R$ formulæ is defined in terms of structures $(s, h)$, where:

$\mathbb{s} : \mathbb{T} \rightarrow \mathbb{L}$ is a partial mapping of terms into locations, called a store, that interprets at least all the constants, i.e. $\mathbb{C} \subseteq \text{dom}(s)$ for every store $s$ and

$\mathbb{h} : \mathbb{L} \rightarrow \text{fin} \mathbb{L}^R$ is a finite partial mapping of locations into $\mathbb{R}$-tuples of locations, called a heap.

Given a heap $\mathbb{h}$, let $\text{loc}(\mathbb{h}) \overset{\text{df}}{=} \{t_0, \ldots, t_{\ell} | t_0 \in \text{dom}(\mathbb{h}), b(t_0) = (t_1, \ldots, t_{\ell})\}$ be the set of locations that occur in the heap $\mathbb{h}$. Two heaps $\mathbb{h}_1$ and $\mathbb{h}_2$ are disjoint iff $\text{dom}(\mathbb{h}_1) \cap \text{dom}(\mathbb{h}_2) = \emptyset$, in which case their disjoint union is denoted by $\mathbb{h}_1 \uplus \mathbb{h}_2$, otherwise undefined. The frontier between $\mathbb{h}_1$ and $\mathbb{h}_2$ is the set of common locations $\text{Fr}(\mathbb{h}_1, \mathbb{h}_2) \overset{\text{df}}{=} \text{loc}(\mathbb{h}_1) \cap \text{loc}(\mathbb{h}_2)$. Note that disjoint heaps may have nonempty frontier.

The satisfaction relation $\models$ between structures $(s, h)$ and predicate-free $SL^R$ formulæ $\phi$ is defined recursively on the structure of formulæ:

$$(s, h) \models t_1 = t_2 \iff t_1, t_2 \in \text{dom}(s) \text{ and } s(t_1) = s(t_2)$$

$$(s, h) \models \text{emp} \iff h = \emptyset$$

$$(s, h) \models t_0 \mapsto (t_1, \ldots, t_k) \iff t_0, \ldots, t_k \in \text{dom}(s), \text{dom}(h) = \{s(t_0)\} \text{ and } h(s(t_0)) = (s(t_1), \ldots, s(t_k))$$

$$(s, h) \models \phi_1 \land \phi_2 \iff (s, h) \models \phi_1, i = 1,2$$

$$(s, h) \models \neg \phi_1 \iff (s, h) \not\models \phi_1$$

$$(s, h) \models \exists x . \phi \iff (s[x \leftarrow \ell], h) \models \phi, \text{ for some location } \ell \in \mathbb{L}$$

where $s[x \leftarrow \ell]$ is the store, with domain $\text{dom}(s) \cup \{x\}$, that maps $x$ to $\ell$ and behaves like $s$ over $\text{dom}(s) \setminus \{x\}$. For a tuple of variables $x = (x_1, \ldots, x_n)$ and locations $\ell = (\ell_1, \ldots, \ell_n)$, we call the store $s[x \leftarrow \ell] \overset{\text{df}}{=} s[x_1 \leftarrow \ell_1] \ldots s[x_n \leftarrow \ell_n]$ an $x$-associate of $s$. A structure $(s, h)$ such that $(s, h) \models \phi$, is called a model of $\phi$. Note that $(s, h) \models \phi$ only if $\text{fv}(\phi) \subseteq \text{dom}(s)$.

The fragment of symbolic heaps is obtained by confining the negation and conjunction to the formulæ $t_1 = t_2 \overset{\text{df}}{=} t_1 = t_2 \land \text{emp}$ and $t_1 \neq t_2 \overset{\text{df}}{=} \neg t_1 = t_2 \land \text{emp}$, called equational atoms, by abuse of language. We denote by $\text{SH}^R$ the set of symbolic heaps, formally defined below:

$$\phi \ := \ \text{emp}\ | \ t_0 \mapsto (t_1, \ldots, t_k) \ | \ p(t_1, \ldots, t_{n_p}) \ | \ t_1 = t_2 \ | \ t_1 \neq t_2 \ | \ \phi_1 \ast \phi_2 \ | \ \exists x . \ \phi_1 \ | \ \exists x \ . \ \phi_1 \ast \phi_2$$

Given quantifier-free symbolic heaps $\phi_1, \phi_2 \in \text{SH}^R$, it is not hard to check that $\exists x . \ \phi_1 \ast \exists y . \ \phi_2$ and $\exists x \ . \ \phi_1 \ast \phi_2$ have the same models. Consequently, each symbolic heap can be written in prenex form, as $\phi = \exists x_1 \ldots \exists x_n . \psi$, where $\psi$ is a quantifier-free separating conjunction of points-to atoms and (dis-)equalities. A variable $x \in \text{fv}(\phi)$ is allocated in $\phi$ iff there exists a (possibly empty) sequence of equalities $x = \ldots = t_0$ and a points-to atom $t_0 \mapsto (t_1, \ldots, t_k)$ in $\psi$. 

© Mnacho Echenim, Radu Iosif and Nicolas Peltier; licensed under Creative Commons License CC-BY LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
The predicates from \( \mathcal{P} \) are interpreted by a given set \( \mathcal{S} \) of rules \( p(x_1, \ldots, x_n) \models \rho \), where \( \rho \) is a symbolic heap, such that \( \text{fv}(\rho) \subseteq \{x_1, \ldots, x_n\} \). We say that \( p(x_1, \ldots, x_n) \) is the head and \( \rho \) is the body of the rule. For conciseness, we write \( p(x_1, \ldots, x_n) \models_S \rho \) instead of \( p(x_1, \ldots, x_n) \models \rho \in \mathcal{S} \). In the following, we shall often refer to a given set of rules \( \mathcal{S} \).

**Definition 1 (Unfolding).** A formula \( \psi \) is a step-unfolding of a formula \( \phi \in \mathcal{S}^R \), written \( \phi \Rightarrow_S \psi \), if \( \psi \) is obtained by replacing an occurrence of an atom \( p(t_1, \ldots, t_p) \) in \( \phi \) with \( \rho[t_1/x_1, \ldots, t_p/x_n] \), for a rule \( p(x_1, \ldots, x_n) \models_S \rho \). An unfolding of \( \phi \) is a formula \( \psi \) such that \( \phi \Rightarrow_S \psi \).

It is easily seen that any unfolding of a symbolic heap is again a symbolic heap. We implicitly assume that all bound variables are \( \alpha \)-renamed throughout an unfolding, to avoid name clashes. Unfolding extends the semantics from predicate-free to arbitrary \( \mathcal{S}^R \) formulae:

**Definition 2.** Given a structure \((s, h)\) and a formula \( \phi \in \mathcal{S}^R \), we write \((s, h) \vDash \phi \) iff there exists a predicate-free unfolding \( \phi \Rightarrow_S \psi \) such that \((s, h) \models \psi\). In this case, \((s, h)\) is an \( \mathcal{S} \)-model of \( \phi \). For two formulae \( \phi, \psi \in \mathcal{S}^R \), we write \( \phi \vDash \psi \) iff every \( \mathcal{S} \)-model of \( \phi \) is an \( \mathcal{S} \)-model of \( \psi \).

Note that, if \((s, h) \vDash \phi \), then \( \text{dom}(s) \) might have to contain constants that do not occur in \( \phi \). For instance if \( p(x) \vDash_S x \mapsto a \) is the only rule with head \( p(x) \), then any \( \mathcal{S} \)-model \((s, h)\) must map \( a \) to some location, which is taken care of by the assumption \( \subseteq \text{dom}(s) \), which applies to any store.

**Definition 3 (Entailment).** Given symbolic heaps \( \phi, \psi_1, \ldots, \psi_n \), such that \( \phi \) is quantifier-free and \( \text{fv}(\phi) = \text{fv}(\psi_1) = \ldots = \text{fv}(\psi_n) = \emptyset \), the sequent \( \phi \vdash \psi_1, \ldots, \psi_n \) is valid for \( \mathcal{S} \) iff \( \phi \vDash_S \bigvee_{1 \leq i \leq n} \psi_i \). An entailment problem \( \mathcal{P} = (\mathcal{S}, \Sigma) \) consists of a set of rules \( \mathcal{S} \) and a set \( \Sigma \) of sequents, asking whether each sequent in \( \Sigma \) is valid for \( \mathcal{S} \).

Note that we consider entailments between formulæ without free variables. This is not restrictive, since any free variable can be replaced by a constant from \( \mathcal{C} \), with no impact on the validity status or the computational complexity of the problem. We silently assume that \( \mathcal{C} \) contains enough constants to allow this replacement. For conciseness, we write \( \phi \vdash_p \psi_1, \ldots, \psi_n \) for \( \phi \vdash \psi_1, \ldots, \psi_n \in \Sigma \), where \( \Sigma \) is the set of sequents of \( \mathcal{P} \). The following example shows an entailment problem asking whether the concatenation of two acyclic lists is again an acyclic list:

**Example 4.** The entailment problem below consists of four rules, defining the predicates \( \text{ls}(x, y) \) and \( \text{sls}(x, y, z) \), respectively, and two sequents:

\[
\begin{align*}
\text{ls}(x, y) & \iff x \mapsto y \land x \neq y \land \exists v. x \mapsto v \land \text{ls}(v, y) \\
\text{sls}(x, y, z) & \iff x \mapsto y \land x \neq y \land \exists v. x \mapsto v \land \text{sls}(v, y, z) \\
\text{ls}(a, b) \land \text{ls}(b, c) & \vdash \exists x. a \mapsto x \land \text{ls}(x, c) \land a \neq c \land \text{sls}(a, b, c) \land \text{ls}(b, c) \land \exists x. a \mapsto x \land \text{ls}(x, c) \
\end{align*}
\]

Here \( \text{ls}(x, y) \) describes non-empty acyclic list segments with head and tail pointed to by \( x \) and \( y \), respectively. The first sequent is invalid, because \( c \) can be allocated within the list segment defined by \( \text{ls}(a, b) \), in which case the entire list has a cycle starting and ending with the location associated with \( c \).

To avoid the cycle, the left-hand side of the second sequent uses the predicate \( \text{sls}(x, y, z) \) describing an acyclic list segment from \( x \) to \( y \) that skips the location pointed to by \( z \). The second sequent is valid.

The complexity analysis of the decision procedure described in this paper relies on two parameters. First, the width of an entailment problem \( \mathcal{P} = (\mathcal{S}, \Sigma) \) is (roughly) the maximum among the sizes of the symbolic heaps occurring in \( \mathcal{P} \) and the number of constants in \( \mathcal{C} \). Second, the size of the entailment problem is (roughly) the number of symbols needed to represent it, namely:

\[
\begin{align*}
\text{width}(\mathcal{P}) & \equiv \max \{ \text{size}(\rho) + \#p \mid p(x_1, \ldots, x_n) \models_S \rho \} \\
\text{size}(\mathcal{P}) & \equiv \sum_{p(x_1, \ldots, x_n) \models_S \rho} \text{size}(\rho) + \#p + \sum_{\psi_0 \vdash_p \psi_1, \ldots, \psi_n} \sum_{i=1}^{n} \text{size}(\psi_i)
\end{align*}
\]

In the next section we give a transformation of an entailment problems with a time complexity that is bounded by the product of the size and a simple exponential of the width of the input, such that,
This assumption loses no generality, because one can enumerate all the equivalence relations on $C$ and test the entailments separately for each of these relations, by replacing all the constants in the same class by a unique representative\(^2\), while assuming that constants in distinct classes are mapped to distinct locations. The overall complexity of the procedure is still doubly exponential, since the number of such equivalence relations is bounded by the number of partitions of $C$, that is $2^O(|C|\log|C|) = 2^O(|\text{width}(P)|\log |\text{width}(P)|)$, for any entailment problem $P$. Thanks to Assumption 1, the considered symbolic heaps can be, moreover, safely assumed not to contain atoms $c \equiv d$, with $\equiv \in \{=, \neq\}$ and $c, d \in C$, since these atoms are either unsatisfiable or equivalent to $\text{emp}$.

### 3 Decidable Classes of Entailments

In general, the entailment problem (Definition 3) is undecidable and we refer the reader to [9, 1] for two different proofs. A first attempt to define a naturally expressive class of formulæ with a decidable entailment problem was reported in [8]. The entailments considered in [8] involve sets of different proofs. A first attempt to define a naturally expressive class of formulæ with a decidable entailment problem was reported in [8]. The entailments considered in [8] involve sets of rules restricted by three conditions, recalled below, in a slightly generalized form.

First, the progress condition requires that each rule adds to the heap exactly one location, associated either to a constant or to a designated parameter. Formally, we consider a mapping $\text{root} : P \rightarrow \mathbb{N} \cup C$, such that $\text{root}(p) \in [1 \ldots \#p] \cup C$, for each $p \in P$. The term $\text{root}(p(t_1, \ldots, t_{\#p}))$ denotes either $t_i$, if $\text{root}(p) = i \in [1 \ldots \#p]$, or the constant $\text{root}(p)$ itself if $\text{root}(p) \in C$. The notation $\text{root}(\alpha)$ is extended to points-to atoms $\alpha$ as $\text{root}(\alpha) \ni \rightarrow \{ (t_1, \ldots, t_{\#t}) \} \overset{\ni}{\mapsto} \alpha$. Second, the connectivity condition requires that all locations added during an unfolding of a predicate atom form a set of connected trees (a forest) rooted in locations associated either with a parameter of the predicate or with a constant.

### Definition 5 (Progress & Connectivity). A set of rules $S$ is progressing if each rule in $S$ is of the form $p(x_1, \ldots, x_{\#p}) \iff \exists y_1 \ldots \exists y_{\#t}, \text{root}(p(x_1, \ldots, y_{\#p})) \ni \rightarrow \{ (t_1, \ldots, t_{\#t}) \} \ni \psi$ and $\psi$ contains no occurrences of points-to atoms. Moreover, $S$ is connected if $\text{root}(q(u_1, \ldots, u_{\#q})) \ni \{ (t_1, \ldots, t_{\#t}) \} \cup C$, for each predicate atom $q(u_1, \ldots, u_{\#q})$ occurring in $\psi$. An entailment problem $P = (S, \Sigma)$ is progressing (connected) if $S$ is progressing (connected).

The progress and connectivity conditions can be checked in polynomial time by a syntactic inspection of the rules in $S$, even if the $\text{root}(\cdot)$ function is not known a priori. Note that this definition of connectivity is less restrictive than the definition from [8], that asked for $\text{root}(q(\alpha_1, \ldots, \alpha_{\#q})) \ni \{ \alpha_1, \ldots, \alpha_{\#q} \}$. For instance, the set of rules $\{ p(x) \iff \exists y \mapsto \psi(p(y), p(x)) \ni x \rightarrow \text{nil} \}$, where $c \in C$ is progressing and connected (with $\text{root}(p) = 1$) in the sense of Definition 5, but not connected in the sense of [8], because $c \not\equiv (y)$. Note also that nullary predicate symbols are allowed, for instance $q() \ni c \mapsto \text{nil}$ is progressing and connected (with $\text{root}(q) = c$). Further, the entailment problem from Example 4 is both progressing and connected.

Third, the establishment condition is defined, slightly extended from its original statement [8]:

---

\(^2\) The replacement must be performed also within the inductive rules, not only in the considered formulæ.
An entailment problem $P = (S, \Sigma)$ is established if $S$ is established, and strongly established if, moreover, $\phi_i$ is $S$-established, for each sequent $\phi_0 \vdash \phi_1, \ldots, \phi_n$ and each $i \in \llbracket 0 \ldots n \rrbracket$.

For example, the entailment problem from Example 4 is strongly established.

In this paper, we replace establishment with a new condition that, as we show, preserves the decidability and computational complexity of progressing, connected and established entailment problems. The new condition can be checked in time linear in the size of the problem. This condition, called \textit{equational restrictedness} (e-restrictedness, for short), requires that each equational atom occurring in a formula involves at least one constant. We will show that the e-restrictedness condition is more general than establishment, in the sense that every established problem can be reduced to an equivalent e-restricted problem (Theorem 13). Moreover, the class of structures defined using e-restricted symbolic heaps is a strict superset of the one defined by established symbolic heaps.

A symbolic heap $\psi$ is e-restricted if, for every equational atom $t \equiv u$ from $\phi$, where $\equiv \in \{=, \neq\}$, we have $\{t, u\} \cap C \neq \emptyset$. A set of rules $S$ is e-restricted if the body $\rho$ of each rule $p(x_1, \ldots, x_{\#p})$ is e-restricted. An entailment problem $P = (S, \Sigma)$ is e-restricted if $S$ is e-restricted and $\phi_i$ is e-restricted, for each sequent $\phi_0 \vdash \phi_1, \ldots, \phi_n$ and each $i \in \llbracket 0 \ldots n \rrbracket$.

For instance, the entailment problem from Example 4 is not e-restricted, because several rule bodies have disequalities between parameters, e.g. $ls(x, y) \equiv x \Rightarrow y \neq x$. However, the set of rules $\{ls_C(x) \equiv x \Rightarrow c \neq x \neq c, ls_C(x) \equiv \exists y. x \Rightarrow y \neq ls_C(y) \neq y \neq c\}$, where $c \in \mathbb{C}$ and $ls_C$ is a new predicate symbol, denoting an acyclic list ending with $c$, is e-restricted. Note that any atom $ls(x, y)$ can be replaced by $ls_C(x)$, provided that $y$ occurs free in a sequent and can be viewed as a constant.

We show next that every established entailment problem (Definition 6) can be reduced to an e-restricted entailment problem (Definition 7). The transformation incurs an exponential blowup, however, as we show, the blowup is exponential only in the width and polynomial in the size of the input problem. This is to be expected, because checking e-restrictedness of a problem can be done in linear time, in contrast with checking establishment, which is at least co-NP-hard [11].

We begin by showing that each problem can be translated into an equivalent normalized problem:

A set of rules $S$ is normalized iff for each rule $p(x_1, \ldots, x_{\#p}) \subseteq \Sigma \rho$ the symbolic heap $\rho$ is normalized and, moreover:

1. A symbolic heap $\exists x. \psi \in \text{SH}^1$, where $\psi$ is quantifier-free, is normalized iff for every atom $\alpha$ in $\psi$:
   a. If $\alpha$ is an equational atom, then it is of the form $x \neq t$ (t $\neq x$), where $x \in X$.
   b. Every variable $x \in \text{lv}(\psi)$ occurs in a points-to or predicate atom of $\psi$.
   c. If $\alpha$ is a predicate atom $q(t_1, \ldots, t_{\#q})$, then $\{t_1, \ldots, t_{\#q}\} \cap C = \emptyset$ and $t_i \neq t_j$, for all $i \neq j \in \llbracket 1 \ldots \#q \rrbracket$.

2. A set of rules $S$ is normalized iff for each rule $p(x_1, \ldots, x_{\#p}) \subseteq \Sigma \rho$, the symbolic heap $\rho$ is normalized and, moreover:
   a. For every $i \in \llbracket 1 \ldots \#p \rrbracket$ and every predicate-free unfolding $p(x_1, \ldots, x_{\#p}) \Rightarrow_S^* \phi$, $\phi$ contains a points-to atom $t_i \mapsto (t_1, \ldots, t_{\#t_i})$, such that $x_i \in \{t_0, \ldots, t_{\#t_i}\}$.
   b. There exist sets $\text{palloc}_S(p) \subseteq \llbracket 1 \ldots \#p \rrbracket$ and $\text{calloc}_S(p) \subseteq \mathbb{C}$ such that, for each predicate-free unfolding $p(x_1, \ldots, x_{\#p}) \Rightarrow_S^* \phi$:
      a. $i \in \text{palloc}_S(p)$ iff $\phi$ contains an atom $x_i \mapsto (t_1, \ldots, t_{\#t_i})$, for every $i \in \llbracket 1 \ldots \#p \rrbracket$.
      b. $c \in \text{calloc}_S(p)$ iff $\phi$ contains an atom $c \mapsto (t_1, \ldots, t_{\#t_i})$, for every $c \in \mathbb{C}$.
   c. For every predicate-free unfolding $p(x_1, \ldots, x_{\#p}) \Rightarrow_S^* \phi$, if $\phi$ contains an atom $t_0 \mapsto (t_1, \ldots, t_{\#t_i})$ such that $t_0 \in \bigvee \bigwedge \{x_1, \ldots, x_{\#p}\}$, then $\phi$ also contains atoms $t_0 \neq c$, for every $c \in \mathbb{C}$.

3. An entailment problem $P = (S, \Sigma)$ is normalized if $S$ is normalized and, for each sequent $\phi_0 \vdash \phi_1, \ldots, \phi_n$ the symbolic heap $\phi_i$ is normalized, for each $i \in \llbracket 0 \ldots n \rrbracket$. 

---

© Mnacho Echenim, Radu Iosif and Nicolas Peltier; licensed under Creative Commons License CC-BY

Leibniz International Proceedings in Informatics

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
The intuition behind Condition (2a) is that no term can “disappear” while unfolding an inductive definition. Condition (2b) states that the set of terms eventually allocated by a predicate atom is the same in all unfoldings. This allows to define the set of symbols that occur freely in a symbolic heap \( \phi \) and are necessarily allocated in every unfolding of \( \phi \), provided that the set of rules is normalized:

**Definition 9.** Given a normalized set of rules \( S \) and a symbolic heap \( \phi \in SH^p \), the set \( alloc_S(\phi) \) is defined recursively on the structure of \( \phi \):

\[
alloc_S(\{t_0 \mapsto t_1, \ldots, t_n \}) \overset{\text{def}}{=} \{t_1\} \\
alloc_S(\{t_1 \mapsto t_2\}) \overset{\text{def}}{=} \emptyset, \overline{\Rightarrow \{z, \#\}} \\
alloc_S(\phi_1 \ast \phi_2) \overset{\text{def}}{=} alloc_S(\phi_1) \cup alloc_S(\phi_2) \\
alloc_S(\exists x. \phi) \overset{\text{def}}{=} \{\phi\} \setminus \{x\}
\]

**Example 10.** The rules \( p(x, y) \equiv \exists z. x \mapsto z \ast p(z, y) \ast x \neq y \) and \( p(x, y) \equiv \exists z. x \mapsto z \) are not normalized, because they contradict Conditions (1a) and (2a) of Definition 8, respectively. A set \( S \) containing the rules \( q(x, y) \equiv \exists z. x \mapsto y \ast q(y, z) \) and \( q(x, y) \equiv \exists z. x \mapsto y \) is not normalized, because it is not possible to find a set \( alloc_S(q) \) satisfying Condition (2b). Indeed, if \( 2 \notin alloc_S(q) \) then the required equivalence does not hold for the second rule (because it does not allocate \( y \)), and if \( 2 \notin alloc_S(q) \) then it fails for the first one (since the predicate \( q(y, z) \) allocates \( y \)). On the other hand, \( S' = \{p(x, y) \equiv \exists z. x \mapsto z \ast p(z, y) \ast x \neq y \}, q(x, y) \equiv \exists z. x \mapsto y \ast q(y, z) \} \) is normalized (assuming \( \emptyset = \{\#\} \)), with \( alloc_S(p) = alloc_S(q) = \{1\} \) and \( alloc_{S'}(\pi) = \emptyset \), for all \( \pi \in \{p, q, r\} \). Then \( alloc_{S'}(p(x_1, x_2)) \equiv q(x_1, x_2) \ast r(x_3) \equiv (x_1, x_2, x_3) \).

The following lemma states that every entailment problem can be transformed into a normalized entailment problem, by a transformation that preserves e-restricted-ness and (strongly) established.

**Lemma 11.** An entailment problem \( P \) can be translated to an equivalent normalized problem \( P_n \), such that width\((P_n) = O(\text{width}(P)^2) \) in time size\((P) \cdot 2^O(\text{width}(P)) \). Further, \( P_n \) is e-restricted and (strongly) established if \( P \) is e-restricted and (strongly) established.

**Example 12.** The entailment problem \( P = (S, \{p(a, b) \equiv \exists x, y. q(x, y)\}) \) with:

\[
S \overset{\text{def}}{=} \begin{cases} p(x, y) \equiv \exists z. x \mapsto z \ast p(z, y) \ast x \neq y \\ q(x, y) \equiv \exists z. x \mapsto y \ast q(y, z) \neq z \equiv a \ast z \neq b \end{cases}
\]

may be transformed into \( (S', \{p_1() \equiv \exists x, y. q_1(x, y) \}) \) with:

\[
S' \overset{\text{def}}{=} \begin{cases} p_1() \equiv \exists a. x \mapsto z \ast p_2(z) \ast z \equiv a \ast z \neq b \\ p_2(x) \equiv \exists x. a \mapsto x \ast z \\ p_3(x) \equiv \exists x, y. q_1(x, y) \neq z \equiv a \ast z \neq b \\ q_1(x, y) \equiv \exists x. x \mapsto y \ast q_2(y, z) \neq z \equiv a \ast z \neq b \\ q_2(x, y) \equiv x \mapsto y 
\end{cases}
\]

The predicate atoms \( p_1() \), \( p_2(x) \) and \( p_3() \) are equivalent to \( p(a, b) \), \( p(x, b) \) and \( p(b, b) \), respectively. \( q(x, y) \) is equivalent to \( q_1(x, y) \lor q_2(x, y) \). Note that \( p_2(x) \) is only used in a context where \( x \neq b \) holds, thus this atom may be omitted from the rules of \( p_2() \). Recall that \( a \) and \( b \) are mapped to distinct locations, by Assumption 1.

We show that every established problem \( P \) can be reduced to an e-restricted problem in time linear in the size and exponential in the width of the input, at the cost of a polynomial increase of its width:

**Theorem 13.** Every established entailment problem \( P = (S, \Sigma) \) can be reduced in time size\((P) \cdot 2^O(\text{width}(P)^2) \) to normalized an e-restricted problem \( P_r \), such that width\((P_r) = O(\text{width}(P)) \).

The class of e-restricted problems is more general than the class of established problems, in the following sense: for each established problem \( P = (S, \Sigma) \), the treewidth of each \( S \)-model of a \( S \)-established symbolic heap \( \phi \) is bounded by width\((P) \) [8], while e-restricted symbolic heaps may have infinite sequences of models with strictly increasing treewidth.
Example 14. Consider the set of rules \( \{\lll(x, y) \leftarrow x \mapsto (y, \text{nil}); \lll(x, y) \leftarrow \exists z. x \mapsto (z, v) \ast \lll(z, y)\}. \)

The existentially quantified variable \( v \) in the second rule in never allocated in any predicate-free unfolding of \( \lll(a, b) \), thus the set of rules is not established. However, it is trivially e-restricted, because no equational atoms occur within the rules. Among the models of \( \lll(a, b) \), there are all \( n \times n \)-square grid structures, known to have treewidth \( n \), for \( n > 1 \) [17].

4 Normal Structures

The decidability of e-restricted entailment problems relies on the fact that, to prove the validity of a sequent, it is sufficient to consider only a certain class of structures, called normal, that require the variables not mapped to the same location as a constant to be mapped to pairwise distinct locations:

Definition 15. A structure \( (s, b) \) is a normal \( S \)-model of a symbolic heap \( \phi \) iff there exists:

1. a predicate-free unfolding \( \phi \Rightarrow_S \exists x. \psi \), where \( \psi \) is quantifier-free, and
2. an \( x \)-associate \( \overline{s} \) of \( s \), such that \( (\overline{s}, b) \models_S \psi \) and \( \overline{s}(x) = \overline{s}(y) \land x \neq y \Rightarrow \overline{s}(x) \in s(\text{n}) \), for all \( x, y \in \text{fv}(\phi) \).

Example 16. Consider the formula \( \psi = \varphi(x_1) \ast p(x_2) \), with \( p(x_2) = S \exists x. \varphi \) and \( C = \{a\} \). Then the structures: \( (s, b) \) and \( (s, b') \) with \( s = \{(x_1, \ell_1), (x_2, \ell_2), (a, \ell_3)\} \) and \( b = \{\ell_1, \ell_2, \ell_3\} \) but it is not normal, because any associate of \( s \) will map the existentials from the predicate-free unfolding of \( p(x_1) \ast p(x_2) \) into the same location, different from \( s(a) \).

Since the left-hand side symbolic heap \( \phi \) of each sequent \( \phi \vdash \psi_1, \ldots, \psi_n \) is quantifier-free and has no free variables (Definition 3) and moreover, by Assumption 1, every constant is associated a distinct location, to check the validity of a sequent it is enough to consider only structures with injective stores. We say that a structure \( (s, b) \) is injective if the store \( s \) is injective. As a syntactic convention, by stacking a dot on the symbol denoting the store, we mean that the store is injective.

The key property of normal structures is that validity of e-restricted entailment problems can be checked considering only (injective) normal structures. The intuition is that, since the (dis-)equalities occurring in the considered formula involve a constant, it is sufficient to assume that all the existential and universal variables not equal to a constant are mapped to pairwise distinct locations, as all other structures can be obtained from such structures by applying a morphism that preserves the truth value of the considered formulae.

Lemma 17. Let \( \mathcal{P} = (S, \Sigma) \) be a normalized and e-restricted entailment problem and let \( \phi \vdash_p \psi_1, \ldots, \psi_n \) be a sequent. Then \( \phi \vdash_p \psi_1, \ldots, \psi_n \) is valid for \( S \) iff \( (\overline{s}, b) \models_S \bigwedge_{i=1}^n \psi_i \), for each normal injective \( S \)-model \( (\overline{s}, b) \) of \( \phi \).

Another important property of injective normal structures is that the frontier \( \text{Fr}(b_1, b_2) \) of a heap decomposition \( b = b_1 \cup b_2 \) such that \( (s, b) \models_S \phi_1 \ast \phi_2 \) and \( (s, b_i) \models_S \phi_i \), for each \( i = 1, 2 \), is contained in the image of the common free variables and constants via \( s \), i.e. \( \text{Fr}(b_1, b_2) \subseteq s(\text{fv}(\phi_1) \cap \text{fv}(\phi_2) \cup C) \).

5 Core Formulæ

Given an e-restricted entailment problem \( \mathcal{P} = (S, \Sigma) \), the idea of the entailment checking algorithm is to compute, for each symbolic heap \( \varphi \) that occurs as the left-hand side of a sequent \( \phi \vdash_p \psi_1, \ldots, \psi_n \), a finite set of sets of formulæ \( F(\varphi) = \{F_1, \ldots, F_m\} \), of some specific pattern, called core formulæ. The
set \( \mathcal{F}(\phi) \) defines an equivalence relation, of finite index, on the set of injective normal \( \mathcal{S} \)-models of \( \phi \), such that each set \( F \in \mathcal{F}(\phi) \) encodes an equivalence class. Because the validity of each sequent can be checked by testing whether every (injective) normal model of its left-hand side is a model of some symbolic heap on the right-hand side (Lemma 17), an equivalent check is that each set \( F \in \mathcal{F}(\phi) \) contains a core formula entailing some formula \( \psi_i \), for \( i = 1, \ldots, n \). To improve the presentation, we first formalize the notions of core formulæ and abstractions by sets of core formulæ, while deferring the effective construction of \( \mathcal{F}(\phi) \), for a symbolic heap \( \phi \), to the next section (§6). In the following, we refer to a given entailment problem \( \mathcal{P} = (\mathcal{S}, \Sigma) \).

First, we define core formulæ as a fragment of \( \text{SL}^\mathcal{R} \). Consider the formulæ \( \text{loc}(x) \equiv \exists y_0 \ldots \exists y_k \cdot y_0 \mapsto (y_1, \ldots, y_k) \ast \bigwedge_{i=0}^k x \approx y_i \). Note that a structure is a model of \( \text{loc}(x) \) iff the variable \( x \) is assigned to a location from the domain or the range of the heap. We define also the following bounded quantifiers:

\[
\begin{align*}
\exists x \cdot \phi &\iff \exists x \cdot (\int_{(\mathcal{N}(\phi)(\{x\})) \cup \mathcal{C}} \sim x \equiv t) \land \phi \\
\exists_{\phi} x \cdot \phi &\iff \exists x \cdot \neg \text{loc}(x) \land \phi \\
\forall_{\phi} x \cdot \phi &\iff \forall x \cdot \neg \text{loc}(x) \land \phi
\end{align*}
\]

In the following, we shall be extensively using the \( \exists_{\phi} x \cdot \phi \) and \( \forall_{\phi} x \cdot \phi \) quantifiers. The formulæ \( \exists x \cdot \phi \) states that there exists a location \( \ell \) which occurs in the domain or range of the heap and is distinct from the locations associated with the constants and free variables, such that \( \phi \) holds when \( x \) is associated with \( \ell \). Similarly, \( \forall_{\phi} x \cdot \phi \) states that \( \phi \) holds if \( x \) is associated with any location \( \ell \) that is outside of the heap and distinct from all the constants and free variables. The use of these special quantifiers will allow us to restrict ourselves to injective stores (since all variables and constants are mapped to distinct locations), which greatly simplifies the handling of equalities.

The main ingredient used to define core formulæ are context predicates. Given a tuple of predicate symbols \( (p, q_1, \ldots, q_n) \in \mathcal{P}^{n+1} \), where \( n \geq 0 \), we consider a context predicate symbol \( \Gamma_{p,q_1,\ldots,q_n} \) of arity \( #p + \sum_{i=1}^n #q_i \). The informal intuition of a context predicate atom \( \Gamma_{p,q_1,\ldots,q_n}(t, u_1, \ldots, u_n) \) is the following: a structure \( (s, h) \) is a model of this atom if there exist models \( (s, h) \) of \( q_i(u_i) \), \( i = 1 \ldots n \) respectively, with mutually disjoint heaps, an unfolding \( \psi \) of \( p(t) \) in which the atoms \( q_i(u_i) \) occur, and an associate \( \mathcal{S} \) of \( s \) such that \( (-\mathcal{S}, h) \cup \{s_i\}_{1=1}^n h_i \) is a model of \( \psi \).

For readability’s sake, we adopt a notation close in spirit to SL’s separating implication (known as the magic wand), and we write \( \mathcal{K}^n_{i=1} q_i(y_i) \mapsto p(x) \) for \( \Gamma_{p,q_1,\ldots,q_n}(x, y_1, \ldots, y_n) \) and \( \text{emp} \mapsto p(x) \), when \( n = 0 \). The set of rules defining the interpretation of context predicates is the least set defined by the inference rules below, denoted \( \mathcal{E}_S \):

\[
\begin{align*}
&\text{(I)} \\
&\text{(II)}
\end{align*}
\]

\[\text{Context predicates are similar to the strong magic wand introduced in [13]. A context predicate } \alpha \rightarrow \beta \text{ is also related to the usual separating implication } \alpha \rightarrow \beta \text{ of separation logic, but it is not equivalent. Intuitively, } \rightarrow \text{ represents a difference between two heaps, whereas } \rightarrow \text{ removes some atoms in an unfolding. For instance, if } p \text{ and } q \text{ are defined by the same inductive rules, up to a renaming of predicates, then } p(x) \rightarrow q(x) \text{ always holds in a structure with an empty heap, whereas } p(x) \rightarrow q(x) \text{ holds if, moreover, } p(x) \text{ and } q(x) \text{ are the same atom.}\]
Note that \( \mathcal{C}_S \) is not progressing, since the rule for \( p(x) \rightarrow p(y) \) does not allocate any location.

However, if \( S \) is progressing, then the set of rules obtained by applying (II) only is also progressing. Rule (I) says that each predicate atom \( p(t) \rightarrow p(u) \), such that \( t \) and \( u \) are mapped to the same tuple of locations, is satisfied by the empty heap. To understand rule (II), let \( (s, h) \) be an \( S \)-model of \( p(t) \) and assume there are a predicate-free unfolding \( \psi \) of \( p(t) \) and an associate \( s' \) of \( s \), such that \( q_1(u_1), \ldots, q_n(u_n) \) occur in \( \psi \) and \( (s', h) \models_S \psi \) (Fig. 1). If the first unfolding step is an instance of a rule \( p(x) \leftarrow \exists z \cdot \psi \ast \gamma \) then there exist a \( z \)-associate \( S \) of \( s \) and a split of \( h \) into disjoint heaps \( h_0, \ldots, h_n \) such that \( (S, h) \models_S q_1(u_1), \ldots, q_n(u_n) \). Assume, for simplicity, that \( u_1 \cup \ldots \cup u_n \subseteq \text{dom}(S) \) and let \( h_1, \ldots, h_n \) be disjoint heaps such that \( (S, h) \models_S q_i(u_i) \). Then there exists a partition \( \{i_j, \ldots, i_{jk_j}\} \cup \{1 \ldots m\} \) of \( \{1 \ldots n\} \), such that \( h_{i_1}, \ldots, h_{i_{jk_j}} \subseteq h_j \), for all \( j \in \{1 \ldots m\} \). Let \( y_j = \ast_{i=1}^{k_j} q_i(u_i) \), then \( (S, h) \wedge h_i_j \models_S y_j \rightarrow p_j(w_j)[t/x] \), for each \( j \in \{1 \ldots m\} \). This observation leads to the inductive definition of the semantics for \( \ast_{j=1}^m q_j(u_i) \rightarrow p(t) \), by the rule that occurs in the conclusion of (II), where the substitution \( \sigma : z \rightarrow S \wedge \bigcup_{j=1}^n y_j \) is used to instantiate\(^6\) some of the existentially quantified variables from the original rule \( p(x) \Leftarrow S \exists z \cdot \psi \ast \gamma \).

Example 18. Consider the set \( S = \{ p(x) \Leftarrow \exists z_1, z_2 \cdot x \mapsto (z_1, z_2) \ast q(z_1) \ast q(z_2) \ast q(x) \Leftarrow x \mapsto (x, x) \} \). We have \( (s, h) \models_S p(x) \) with \( s = ((x, \ell_1)) \) and \( h = \{(\ell_1, \ell_2, \ell_3), (\ell_2, \ell_2, \ell_2), (\ell_3, \ell_3, \ell_3)\} \). The atom \( q(x) \rightarrow p(x) \) is defined by the following non-progressing rules:

\[
q(y) \rightarrow p(x) \leftarrow \exists z_1, z_2 \cdot x \mapsto (z_1, z_2) \ast q(y) \ast q(z_1) \ast q(z_2) \ast q(x) \Leftarrow x \mapsto (x, x)
\]

The two rules for \( q(y) \rightarrow p(x) \) correspond to the two ways of distributing \( q(y) \) over \( q(z_1) \) and \( q(z_2) \).

We have \( h = h_1 \cup h_2 \), with \( h_1 = \{(\ell_1, \ell_2, \ell_3), (\ell_2, \ell_2, \ell_2)\} \) and \( h_2 = \{(\ell_3, \ell_3, \ell_3)\} \). It is easy to check that \( (s[y \leftarrow \ell_3]) \models_S \exists z_1, z_2 \cdot q(y) \rightarrow p(x) \), and \( (s[y \leftarrow \ell_3]) \models_S \exists z_1, z_2 \cdot q(y) \). Note that we also have \( (s[y \leftarrow \ell_2]) \models_S \exists z_1, z_2 \cdot q(y) \rightarrow p(x) \), with \( h_1 = \{(\ell_1, \ell_2, \ell_3), (\ell_3, \ell_3, \ell_3)\} \).

Having introduced context predicates, the pattern of core formulæ is defined below:

Definition 19. A core formula \( \varphi \) is an instance of the pattern:

\[
\exists h x \forall y \ast \gamma \cdot \ast_{i=1}^{\ast} \left( \ast_{j=1}^{k_j} q_j(u_j) \rightarrow p_j(t_j) \right) \ast_{i=1}^{m} t_i \leftarrow (t'_i, \ldots, t'_R)
\]

such that:

(i) each variable occurring in \( y \) also occurs in an atom in \( \varphi \);

(ii) for every variable \( x \in X \), either \( x \in t_1 \cup \bigcup_{j=1}^{k_j} u_j \) for some \( i \in \{1 \ldots n\} \), or \( x = t'_i \) for some \( i \in \{n+1 \ldots m\} \) and some \( j \in \{0 \ldots R\} \);

(iii) each term \( t \) occurs at most once as \( t = \text{root}(\alpha) \), where \( \alpha \) is an atom of \( \varphi \).

We define moreover the set of terms roots(\( \varphi \)) \( \sqsubseteq \text{roots}(\varphi) \cup \text{roots}(\varphi) \), where \( \text{roots}(\varphi) = \{ \text{root}(\varphi)^{\ast} q_j(u_j)^{\ast}\} \)

\( i \in \{1 \ldots n\} \) and \( \text{roots}(\varphi) = \{ \text{root}(p(t_i)) \mid i \in \{1 \ldots n\} \cup \{t'_i \mid i \in \{n+1 \ldots m\} \} \}

Note that an unfolding of a core formula using the rules in \( \mathcal{C}_S \) is not necessarily a core formula, because of the unbounded existentially quantifiers and equational atoms that occur in the rules from \( \mathcal{C}_S \). Note also that a core formula cannot contain an occurrence of a predicate of the form \( p(t) \rightarrow p(t) \) because otherwise, Condition (iii) of Definition 19 would be violated.

Lemma 20 shows that any symbolic heap is equivalent to an effectively computable finite disjunction of core formulæ, when the interpretation of formulæ is restricted to injective structures. For a

\( ^6 \) Note that this instantiation is, in principle, redundant (i.e. the same rules are obtained if \( \text{dom}(\sigma) = \emptyset \) by choosing appropriate \( z \)-associates) but we keep it to simplify the related proofs.
symbolic heap $\phi \in \text{SH}^h$, we define the set $\mathcal{T}(\phi)$, recursively on the structure of $\phi$, implicitly assuming w.l.o.g. that $\text{emp} \ast \phi = \phi \ast \text{emp} = \phi$:

$$
\begin{align*}
\mathcal{T}(\text{emp}) &\equiv \{\text{emp}\} \\
\mathcal{T}(p(t)) &\equiv \{\text{emp} \leftarrow p(t)\} \\
\mathcal{T}(t_1 \leftarrow t_2) &\equiv \begin{cases} 
\text{emp} & \text{if } t_1 = t_2 \\
0 & \text{if } t_1 \neq t_2 
\end{cases}
\end{align*}
$$

For instance, if $\phi = \exists x . p(x, y) \ast x \neq y$ and $\mathbb{C} = \{c\}$, then $\mathcal{T}(\phi) = \{\exists x . \text{emp} \leftarrow p(x, y), \text{emp} \leftarrow p(c, y)\}$.

Next, we give an equivalent condition for the satisfaction of a context predicate atom, that relies on an unfolding of a symbolic heap into a core formula:

**Definition 21.** A formula $\varphi$ is a core unfolding of a predicate atom $\exists^m x_1 q_1(u_1) \rightarrow p(t)$, written $\exists^m x_1 q_1(u_1) \rightarrow p(t) \leadsto_{\mathcal{E}_S} \varphi$, if there exists:

1. a rule $\exists^m x_1 q_1(u_1) \rightarrow p(x) \in \mathcal{E}_S \exists x . \phi$, where $\phi$ is quantifier free, and
2. a substitution $\sigma = [t/X, u_1/y_1, \ldots, u_n/y_n] \cup \zeta$, $\zeta \subseteq \{z, t\} \setminus \{x, t\}$, $t \in T \cup \bigcup_{i=1}^n u_i$, such that $\varphi \in \mathcal{T}(\phi)$. 

A core unfolding of a predicate atom is always a quantifier-free formula, obtained from the translation (into a disjunctive set of core formulæ) of the quantifier-free matrix of the body of a rule, in which some of the existentially quantified variables in the rule occur instantiated by the substitution $\sigma$. For instance, the rule $\text{emp} \rightarrow p(x) \in \mathcal{E}_S \exists y . x \rightarrow y$ induces the core unfoldings $\text{emp} \rightarrow p(a) \leadsto_{\mathcal{E}_S} a \rightarrow a$ and $\text{emp} \rightarrow p(a) \leadsto_{\mathcal{E}_S} a \rightarrow u$, via the substitutions $[a/x, a/y]$ and $[a/x, a/y]$, respectively.

We now define a equivalence relation, of finite index, on the set of injective structures. Intuitively, an equivalence class is defined by the set of core formulæ that are satisfied by all structures in the class (with some additional conditions). First, we introduce the overall set of core formulæ, over which these equivalence classes are defined:

**Definition 22.** Let $\mathcal{V}_P \equiv \mathcal{V}_P^1 \cup \mathcal{V}_P^2$, such that $\mathcal{V}_P^1 \cap \mathcal{V}_P^2 = \emptyset$ and $||\mathcal{V}_P^i|| = \text{width}(P)$, for $i = 1, 2$ and denote by $\text{Core}(P)$ the set of core formulæ $\varphi$ such that $\text{roots}(\varphi) \cap \text{fv}(\varphi) \subseteq \mathcal{V}_P^1$, $\text{roots}(\varphi) \setminus \text{fv}(\varphi) \subseteq \mathcal{V}_P^2 \cup \mathbb{C}$ and no variable in $\mathcal{V}_P^1$ is bound in $\varphi$.

Note that $\text{Core}(P)$ is a finite set, because both $\mathcal{V}_P$ and $\mathbb{C}$ are finite. Intuitively, $\mathcal{V}_P^1$ will denote “local” variables introduced by unfolding the definitions on the left-hand sides of the entailments, whereas $\mathcal{V}_P^2$ will denote existential variables occurring on the right-hand sides. Second, we characterize an injective structure by the set of core formulæ it satisfies:

**Definition 23.** For a core formula $\varphi = \exists h x \forall \neg y . \psi$, we denote by $\mathcal{W}_S(h, b, \varphi)$ the set of stores $\delta$ that are injective $(X \cup Y)$-associates of $h$, and such that: (1) $(\delta, b) \in \mathcal{S}_x \psi$, (2) $\bar{\delta}(x) \subseteq \text{loc}(b)$, and (3) $\bar{\delta}(y) \cap \text{loc}(b) = \emptyset$. The elements of this set are called witnesses for $(h, b)$ and $\varphi$.

The core abstraction of an injective structure $(h, b)$ is the set $\mathcal{C}(h, b, \varphi) \in \text{Core}(P)$ for which there exists a witness $\delta \in \mathcal{W}_S(h, b, \varphi)$ such that $\bar{\delta}(\text{roots}_{\text{sh}}(\varphi)) \cap \text{dom}(b) = \emptyset$.

An injective structure $(h, b)$ satisfies each core formula $\varphi \in \mathcal{C}(h, b, \varphi)$, fact that is witnessed by an extension of the store assigning the universally quantified variables random locations outside of the heap. Further, any core formula $\varphi$ such that $(h, b) \models \varphi$ and $\text{roots}_{\text{sh}}(\varphi) = \emptyset$ occurs in $\mathcal{C}(h, b)$.
Our entailment checking algorithm relies on the definition of the profile of a symbolic heap. Since
each symbolic heap is equivalent to a finite disjunction of existential core formulæ, when interpreted
over injective normal structures, it is sufficient to consider only profiles of core formulæ:

**Definition 24.** A profile for an entailment problem \( P = (S, \Sigma) \) is a relation \( \mathcal{F} \subseteq \text{Core}(P) \times \text{Core}(P) \)
such that, for any core formula \( \phi \in \text{Core}(P) \) and any set of core formulæ \( F \in \text{Core}(P) \), we have
\[(\phi, F) \in \mathcal{F} \text{ iff } F = \mathcal{C}(\delta, \beta), \text{ for some injective normal } \Sigma_S\text{-model } (\delta, \beta) \text{ of } \phi, \text{ with } \text{dom}(\delta) = \text{fv}(\phi) \cup C.\]

Assuming the existence of a profile, the effective construction of which will be given in Section 6, the
following problem provides an algorithm that decides the validity of \( P \):

**Lemma 25.** Let \( P = (S, \Sigma) \) be a normalized e-restricted entailment problem and \( \mathcal{F} \subseteq \text{Core}(P) \times \text{Core}(P) \)
be a profile for \( P \). Then \( \mathcal{F} \) is valid iff, for each sequent \( \phi \vdash \psi_1, \psi_2, \ldots, \psi_n \), each core formula \( \varphi \in \mathcal{T}(\phi) \) and each pair \((\varphi, F) \in \mathcal{F}\), we have \( F \cap \mathcal{T}(\psi_i) \neq \emptyset \), for some \( i \in \{1 \ldots n\} \).

The proof relies on Lemma 17, according to which entailments can be tested by considering only
normal models. As one expects, Lemma 20 is used in this proof to ensure that the translation \( \mathcal{T}(\cdot) \) of
symbolic heaps into core formulæ preserves the injective models.

### 6 Construction of the Profile Function

For a given normalized entailment problem \( P = (S, \Sigma) \), describe the construction of a profile \( \mathcal{F}_P \subseteq \text{Core}(P) \times \text{Core}(P) \), recursively on the structure of core formulæ. We assume that the set of rules \( S \)
is progressing, connected and e-restricted. The relation \( \mathcal{F}_P \) is the least set satisfying the recursive
constraints (1), (2), (3) and (4), given in this section. Since these recursive definitions are monotonic,
the least fixed point exists and is unique. We shall prove later (Theorem 32) that the least fixed point
can, moreover, be attained in a finite number of steps by a standard Kleene iteration.

**Points-to Atoms** For a points-to atom \( t_0 \mapsto (t_1, \ldots, t_k) \), such that \( t_0, \ldots, t_k \in \mathcal{V}_P \cup C \), we have:
\[(t_0 \mapsto (t_1, \ldots, t_k), F) \in \mathcal{F}_P \text{ iff } F \text{ is the set containing } t_0 \mapsto (t_1, \ldots, t_k) \text{ and all core formulæ of the form } \exists \varphi(x). \bigwedge_{i=1}^{n} q_i(u_i) \Rightarrow p(t) \in \text{Core}(P), \text{ where } z = (t \cup u_1 \cup \ldots \cup u_n) \setminus (t_0, \ldots, t_k) \cup C \text{ such that } \text{emp} \Rightarrow p(t) \Rightarrow q_i(u_i) \Rightarrow q(t) \Rightarrow q(t) \Rightarrow p(t).\]
\[(1)\]

For instance, if \( S = \{p(x) \iff \exists y z . x \Rightarrow y \Rightarrow q(y, z), q(x, y) \iff x \Rightarrow y \text{ with } \mathcal{V}_P^1 = \{u, v\} \text{ and } \mathcal{V}_P^2 = \{z\} \}
then \( \mathcal{F}_P \) contains the pair \((u \mapsto v, F) \text{ with } F = [u \mapsto v, \text{emp} \Rightarrow q(u, v), v \mapsto q(v, z) \Rightarrow p(t)].\]

**Predicate Atoms** Since profiles involve only the core formulæ obtained by the syntactic translation
of a symbolic heap, the only predicate atoms that occur in the argument of a profile are of the form
\( \text{emp} \Rightarrow p(t) \). We consider the constraint:
\[(\text{emp} \Rightarrow p(t), F) \in \mathcal{F}_P \text{ if } (\exists \psi \cdot \varphi, F) \in \mathcal{F}_P, \text{emp} \Rightarrow p(t) \Rightarrow q(t) \Rightarrow \psi \in \text{Core}(P) \text{ and } y = \text{fv}(\psi) \setminus t \]
\[(2)\]

**Separating Conjunctions** Computing the profile of a separating conjunction is the most technical
point of the construction. To ease the presentation, we assume the existence of a binary operation
called composition:

**Definition 26.** Given a set \( D \subseteq \mathcal{V}_P^1 \cup C \), a binary operator \( \oplus_D : \text{Core}(P) \times \text{Core}(P) \rightarrow \text{Core}(P) \)
is a composition if \( \mathcal{C}(\delta, \beta_1) \oplus_D \mathcal{C}(\delta, \beta_2) = \mathcal{C}(\delta, \beta) \), for any injective structure \((\delta, \beta)\), such that
\[(i) \text{ dom}(\delta) \subseteq \mathcal{V}_P^1, (ii) \beta_1 |\beta_2, (iii) \text{Fr}(\beta_1, \beta_2) \subseteq \delta(D) \cup \{\delta(D) \subseteq \text{dom}(\delta) \}.\]

We recall that \( \text{Fr}(\beta_1, \beta_2) = \text{loc}(\beta_1) \cap \text{loc}(\beta_2) \). If \( S \) is a normalized set of rules, then for any core formulæ \( \phi \) whose only occurrences of predicate atoms are of the form \( \text{emp} \Rightarrow p(t) \), we define \( \text{alloc}_{\Sigma}(F) \) as
the homomorphic extension of \( \text{alloc}_{\Sigma}(\text{emp} \Rightarrow p(t)) \) to \( \phi \) (see Definition 9). Assuming

© Mnacho Echenim, Radu Iosif and Nicolas Pelerin;
Licensed under Creative Commons License CC-BY
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
We prove that
We show that
\[ \exists (\exists h) . \phi(x' / x), \text{rem}(x, F) \in F_P, \text{if } x \in \text{fv}(\phi), \ x' \not\in \text{dom}(\phi) \rightarrow \phi, (\phi, F) \in F_P. \] (4)
\[
\text{rem}(x, F) \equiv [\exists h. \psi[x / x] | \psi \in F, x \not\in \text{fv}(\psi)] \rightarrow \psi \not\in \text{fv}(\psi)]
\]
\[
\text{rem}(x_1, \ldots, x_n), F) \equiv \text{rem}(x_1, \ldots, \text{rem}(x_n, F), \ldots)
\]
Note that \( \tilde{x} \) is a fresh variable, which is not bound or free in \( \psi \). In particular, if \( x \in \text{roots}(\psi) \), then we must have \( \tilde{x} \in \text{Core}(F) \), so that \( \exists h . \phi[\tilde{x} / x] \in \text{Core}(F) \). Similarly the variable \( x \) is replaced by a fresh variable \( x' \in \text{Core}(F) \) in \( \exists h . \phi[x' / x] \) to ensure that \( \exists h . \phi[x' / x] \) is a core formula.

The **Profile Function** Let \( F_P \) be the least relation that satisfies the constraints (1), (2), (3) and (4).

We prove that \( F_P \) is a valid profile for \( P \), in the sense of Definition 24:

**Lemma 27.** Given a progressing and normalentailed nonempty problem \( P = (S, \Sigma) \), a symbolic heap \( \varphi \in \mathcal{H}^S \) with \( \text{fv}(\varphi) \subseteq \mathcal{V}^P_\varphi \), a core formula \( \phi \in \mathcal{T}(\varphi) \) and a set of core formulae \( F \subseteq \text{Core}(P) \), we have

\[ (\phi, F) \in F_P \iff F = C_P(\bar{s}, b), \text{for some injective normal } \mathbb{C}_S\text{-model } (\bar{s}, b) \text{ of } \phi, \text{with } \text{dom}(\bar{s}) = \text{fv}(\phi) \cup C. \]

The composition operation \( @_D \) works symbolically on core formulæ, by saturating the separating conjunction of two core formulæ via a **modus ponens**-style consequence operator.

**Definition 28.** Given formulæ \( \phi, \psi \), we write \( \phi \vdash^* \psi \) if \( \phi \vdash [\alpha \rightarrow \text{p}(t)] \rightarrow \text{q}(u) \) and \( \psi = \varphi \vdash [(\alpha \ast \beta) \rightarrow \text{q}(u)] \) (up to the commutativity of * and the neutrality of emp) for some formulæ \( \varphi, \psi \), predicate atoms \( p(t) \) and \( q(u) \) and conjunctions of predicate atoms \( \alpha \text{ and } \beta \).

**Example 29.** Consider the structure \((s, b)\) and the rules of Example 18. We have \( h = h_1 \bowtie h_2 \), with

\[ (s[y \leftarrow f_1, b_1] \models \exists x, g (y) \rightarrow p(x) \text{ and } (s[y \leftarrow f_1, b_2] \models \exists x, g (y), \text{i.e., } (s[y \leftarrow f_1, b_2] \models \exists x, g (y), \text{ thus} (s[y \leftarrow f_1, b_1] \models \exists x, g (y) \rightarrow p(x) \text{ and } \exists x, g (y) \rightarrow p(x) \text{ and } \text{emp} \rightarrow q(y) \text{ and } \text{emp} \rightarrow p(x)). \]

We define a relation on the set of core formulæ \( \text{Core}(P) \), parameterized by a set \( D \subseteq \mathcal{V}^P_\varphi \cup C \):

\[ \exists h. x \models \varphi[x_1] \models \psi, \models \exists h. x \models \varphi[x_2] \models \psi \] (5)

\[ \text{if } \psi_1 \vdash \psi_2 \vdash^* \psi, x_1 \cap x_2 = \emptyset, x = (x_1 \cup x_2) \cap \text{fv}(\psi), \psi = ((y_1 \cup y_2) \cap \text{fv}(\psi)) \setminus x, \text{roots}_{\text{h}(\psi)}(\psi) \cap D = \emptyset. \]

The composition operator is defined by lifting the \( \vdash^* \) relation to sets of core formulæ:

\[ F_1 \vdash)=='@_D F_2 \equiv [\psi | \phi_1 \models F_1, \phi_2 \models F_2, \phi_1, \phi_2 \vdash^* \psi] \]

We show that \( @_D \) is indeed a composition, in the sense of Definition 26:
We presented a class of SL formulæ built from a set of inductively defined predicates, used to describe pointer-linked recursive data structures, whose entailment problem is 2-EXPTIME-complete. This fragment, consisting of so-called e-restricted formulæ, is a strict generalization of previous work defining three sufficient conditions for the decidability of entailments between SL formulæ, namely progress, connectivity and establishment [8, 12, 14]. On one hand, every progressing, connected and established entailment problem can be translated into an e-restricted problem. On the other hand, the models of e-restricted formulæ form a strict superset of the models of established formulæ. The proof for the 2-EXPTIME upper bound for e-restricted entailments leverages from a novel technique used to prove the upper bound of established entailments [12, 14]. A natural question is whether the e-restrictedness condition can be dropped. We conjecture that this is not the case, and that entailment is undecidable for progressing, connected and non-e-restricted sets. Another issue is whether the generalization of symbolic heaps to use guarded negation, magic wand and septraction from [15] is possible for e-restricted entailment problems. The proof of these conjectures is on-going work.
References


A Proof of Lemma 11 (Section 3)

Let $P = (S, \Sigma)$ be an input entailment problem. We transform $P$ in order to meet points (1a), (1c), (2a), (2b) and (2c) of Definition 8, as follows.

(1a) First, we apply exhaustively, to each symbolic heap occurring in $P$, the following transformations, for each term $t \in \mathbb{T}$:

$$\exists x \cdot x = t \Rightarrow \phi[t/x] \quad (7)$$

$$t = t \Rightarrow \phi \quad (8)$$

Note that, at this point, there are no equality atoms involving an existentially quantified variable (recall that equalities between constants can be dismissed since they are either trivially false or equivalent to $\text{emp}$). We apply the following transformations, that introduce disequalities between the remaining existential variables and the rest of the terms.

$$p(x) \Leftrightarrow \exists x. \rho \Rightarrow \begin{cases} p(x) \Leftrightarrow \rho[t/x] \\ p(x) \Leftrightarrow \exists x. \rho \wedge x \neq t \end{cases} \quad (9)$$

for all $t \in (fv(\rho) \setminus \{x\}) \cup \mathbb{C}$, where $x \neq t$ does not occur in $\rho$

$$\phi \vdash \psi_1, \ldots, \exists x. \psi_i, \ldots, \psi_n \Rightarrow \phi \vdash \psi_1, \ldots, \psi_{i-1}, \psi_i[t/x], \exists x \cdot x \neq t \Rightarrow \psi_i, \ldots, \psi_n \quad (10)$$

for all $t \in \mathbb{C}$, such that $x \neq t$ does not occur in $\psi_i$

Let $P_1 = (S_1, \Sigma_1)$ be the result of applying the transformations (7-10) exhaustively. Because every transformation preserves the equivalence of rules and sequents, $P_1$ is valid iff $P$ is valid. Note that, by Definition 3, there are no free variables occurring in a sequent from $\Sigma$. Then the only remaining equality atoms $t = u$ occurring in $P_1$ must occur in a rule $p(x_1, \ldots, x_{#p}) \Leftrightarrow S_1 \rho$ and neither $t$ nor $u$ can be an existentially quantified variable, hence $t, u \in \{x_1, \ldots, x_{#p}\} \cup \mathbb{C}$. Before proceeding further with Condition (1a), we make sure that Condition (1c) is satisfied.

(1c) Let $q(t_1, \ldots, t_{#q})$ be a predicate atom occurring in a rule or a sequent from $P_1$, where $t_1, \ldots, t_{#q} \in \mathbb{T}$, and let $(t_1, \ldots, t_{#m})$ be the subsequence obtained by removing the terms from the set $\{t_i \mid i \in \{1 \ldots #q\}, \exists j < i, t_i = t_j \} \cup \mathbb{C}$ from $(t_1, \ldots, t_{#q})$. We consider a fresh predicate symbol $q_{1 \ldots #m}$ of arity $m$, with the new rules $q_{1 \ldots #m}(x_1, \ldots, x_m) \Leftrightarrow \rho \sigma$, for each rule $q(x_1, \ldots, x_{#q}) \Leftrightarrow S \rho$, where the substitution $\sigma$ is defined such that, for all $j \in \{1 \ldots #q\}$:

$$\sigma(x_j) \overset{\text{def}}{=} x_{i_j} \text{ if } j = i_j, \text{ for some } \ell \in \{1 \ldots m\},$$

$$\sigma(x_j) \overset{\text{def}}{=} x_j \text{ if } t_j \in \mathbb{C}, \text{ and}$$

$$\sigma(x_j) \overset{\text{def}}{=} x_j \text{ otherwise}.$$ 

Note that the definition of the sequence $(t_1, \ldots, t_{#m})$ guarantees that such a substitution exists and it is unique. If the rule body obtained by applying the substitution $\sigma$ contains a disequality $t \neq t$, for some $t \in \mathbb{T}$, we eliminate the rule. Otherwise, we apply transformation (8) to the newly obtained rule to eliminate trivial equalities. Finally, we replace each occurrence of $q(t_1, \ldots, t_{#q})$ in $P_1$ with $q_{1 \ldots #m}(t_1, \ldots, t_{#m})$. Because $q(t_1, \ldots, t_{#m})$ and $q_{1 \ldots #m}(t_1, \ldots, t_{#m})$ have the same step unfoldings, they have the same predicate-free unfoldings and this transformation preserves equivalence, yielding a problem that satisfies condition (1c). Let $P_2 = (S_2, \Sigma_2)$ be the outcome of this transformation, where $S_2$ is the set of newly introduced rules and $\Sigma_2$ is obtained from $\Sigma_1$ by the replacement of each predicate atom $q(t_1, \ldots, t_{#q})$ with $q_{1 \ldots #m}(t_1, \ldots, t_{#m})$. It is easy to check that $P_2$ and $P_1$ have the same validity status, which is that of $P$.

(1a) We will now finish the proof of Condition (1a). Since the transformation (7) removes equalities involving an existentially quantified variable and the equalities between constants can be eliminated as explained above, the only equalities that occur in the body of a rule $p(x_1, \ldots, x_{#p}) \Leftrightarrow S \rho$ are of the form $x_i = t$, where $i \in \{1 \ldots #p\}$ and $t \in \{x_j \mid j \in \{1 \ldots #p\}, j \neq i\} \cup \mathbb{C}$. We show that if such an equality holds, then...
occurs in the body of a rule, then this rule can safely be removed because any unfolding involving it generates an unsatisfiable symbolic heap. Let \( p(u_1, \ldots, u_\#p) \) be a predicate atom that occurs in a some unfolding of a symbolic heap from \( \mathcal{P} \) and assume a step-unfolding that substitutes \( p(u_1, \ldots, u_\#p) \) with \( p[u_1/1, \ldots, u_\#p/x_\#p] \). We distinguish two cases:

(i) \( t = x_i \), for some \( j \in [1 \ldots \#p] \) \( \ni \) the point \( (1c) \), \( u_i \) and \( u_j \) must be distinct terms. If \( u_i, u_j \in \mathbb{C} \), then \( u_i \neq u_j \) necessarily holds, by Assumption 1, thus the equality \( x_i = t \) is false when \( x_i, x_j \) are instantiated by \( u_i, u_j \). Otherwise, if \( u_i \not\in \mathbb{V} \) (the case \( u_j \not\in \mathbb{V} \) is symmetric) then \( u_i \) and \( u_j \) were necessarily introduced by existential quantifiers, in which case the disequality \( u_i \neq u_j \) has been asserted by transformations (9) or (10), thus \( x_i = t \) is false when \( x_j \) is replaced by \( u_i \).

(ii) \( t \in \mathbb{C} \): by a similar argument we show that all the relevant instances of the equality \( x_i = t \) are unsatisfiable.

Consequently, if an equality occurs in a rule, then this the rule can safely be removed.

(1b) To ensure that all variables occur within a points-to or predicate atom, we apply exhaustively the following transformation to each symbolic heap in the problem:

\[
\exists x. \, x^n_{i=1} x \neq t_i \implies \psi \iff \psi, \text{ if } x \notin \mathcal{V}(\psi)
\]  

(11)

Let \( \mathcal{P}_3 = (S_3, \Sigma_2) \) be the outcome of this transformation. Because \( \mathbb{L} \) is infinite, any formula \( \exists x. \, x^n_{i=1} x \neq t_i \) is equivalent to \( \text{emp} \). Consequently, \( \mathcal{P}_3 \) and \( \mathcal{P}_2 \) have the same validity status as \( \mathcal{P} \) and \( \mathcal{P}_3 \) satisfies conditions (1a), (1b) and (1c).

(2a+2b) For each predicate symbol \( p \) that occurs in \( S_3 \), we consider the predicate symbols \( p_{X,Y,Z,A,B,C} \) of arities \( \#p \) each, where \( (X,Y,Z) \) is a partition of \( [1 \ldots \#p] \) and \( (A,B,C) \) is a partition of \( \mathbb{C} \), along with the following rules: \( p_{X,Y,Z,A,B,C}(x_1, \ldots, x_\#p) \equiv \rho' \) if and only if \( p(x_1, \ldots, x_\#p) \equiv S_3 \rho \) and \( \rho' \) is obtained from \( \rho \) by replacing each predicate atom \( q_t(t_1, \ldots, t_\#p) \) by a predicate atom \( q_{X',Y',Z',A',B',C'}(t_1, \ldots, t_\#p) \), for some partition \( (X',Y',Z') \) of \( [1 \ldots \#q] \) and some partition \( (A',B',C') \) of \( \mathbb{C} \), such that the following holds. For each \( i \in [1 \ldots \#p] \):

\[ i \in X \iff \text{either a points-to atom } x_i \mapsto (t_1, \ldots, t_\#p) \text{ occurs in } \rho, \text{ or } \rho \text{ contains a predicate atom } \]

\[ r_{X^n,Y^n,Z^n,A^n,B^n,C^n}(t_1, \ldots, t_\#p) \text{ such that } x_i = t_j \text{ and } j \in X^n, \]

\[ i \in Y \iff \text{either } x_i \in \{t_1, \ldots, t_\#p\} \text{ for a points-to atom } t_i \mapsto (t_1, \ldots, t_\#p) \text{ occurring in } \rho, \text{ or } \rho \text{ contains a predicate atom } \]

\[ r_{X^n,Y^n,Z^n,A^n,B^n,C^n}(t_1, \ldots, t_\#p) \text{ such that } x_i = t_j \text{ and } j \in Y^n. \]

Further, for each constant \( c \in \mathbb{C} \):

\[ c \in A \iff \text{a points-to atom } c \mapsto (t_1, \ldots, t_\#p) \text{ occurs in } \rho, \text{ or } \rho \text{ contains a predicate atom } r_{X^n,Y^n,Z^n,A^n,B^n,C^n}(t_1, \ldots, t_\#p) \text{ such that } c \in A^n, \]

\[ c \in B \iff \text{either } c \in \{t_1, \ldots, t_\#p\}, \text{ for a points-to atom } t_0 \mapsto (t_1, \ldots, t_\#p) \text{ occurring in } \rho, \text{ or } \rho \text{ contains a predicate atom } \]

\[ r_{X^n,Y^n,Z^n,A^n,B^n,C^n}(t_1, \ldots, t_\#p) \text{ such that } c \in B^n, \]

Let \( S_4 \) (resp. \( S_3 \)) be the set of sequents (resp. rules) obtained by replacing each predicate atom \( p_t(t_1, \ldots, t_\#p) \) with \( p_{X,Y,Z,A,B,C}(t_1, \ldots, t_\#p) \), for some partition \( (X,Y,Z) \) of \( [1 \ldots \#p] \) and some partition \( (A,B,C) \) of \( \mathbb{C} \). For each predicate symbol \( p_{X,Y,Z,A,B,C} \) we consider a fresh predicate symbol \( \overline{p}_{X,Y,Z,A,B,C} \) of arity \( \#p \in \#p \) and \( \mathcal{P}_3 \) contains a predicate atom \( p_{X,Y,Z,A,B,C}(t_1, \ldots, t_\#p) \) occurring in either \( S_3 \) or \( S_4 \) is replaced by \( \overline{p}_{X,Y,Z,A,B,C}(t_1, \ldots, t_\#p) \), where \( t_1, \ldots, t_\#p \) is the subsequence of \( t_1 \ldots, t_\#p \) obtained by removing the terms from \( t_i \mid i \in X \) and each atom involving these terms is removed from \( S_4 \) and \( S_3 \). Let the result of this transformation be denoted by \( \mathcal{P}_5 = (S_5, \Sigma_5) \), with \( \text{palloc}_{S_5}(\overline{p}_{X,Y,Z,A,B,C}) \equiv X \) and \( \text{calloc}_{S_5}(\overline{p}_{X,Y,Z,A,B,C}) \equiv A \). Properties 2a and 2b follow from the definition of the rules of \( \overline{p}_{X,Y,Z,A,B,C} \) by an easy induction on the length of the unfolding. The equivalence between the validity of \( \mathcal{P}_5 \) and the validity of \( \mathcal{P}_3 \) is based on the following:

\[ \textbf{Fact 1.} \text{ Let } \phi \text{ be a symbolic heap occurring in a sequent from } \Sigma_4, \phi \vdash_{S_4} \psi \text{ be a predicate-free unfolding of } \phi \text{ and } p_{X,Y,Z,A,B,C}(t_1, \ldots, t_\#p) \text{ be a predicate atom that occurs at some intermediate step of this predicate-free unfolding. Then each variable } t_i \in \mathcal{V}(\psi), \text{ such that } i \in Z, \text{ occurs existentially quantified in a subformula } \exists y_{i=1} t_i \neq u \text{ of } \psi \text{ and nowhere else.} \]
Proof: Since \( \text{fv}(\phi) = \emptyset \), it must be the case that \( x_i \) has been introduced as an existentially quantified variable by an intermediate unfolding step. We show, by induction on the length of the unfolding from the point where the variable was introduced that \( t_i \) cannot occur in a points-to atom.

Since \( L \) is infinite, any formula \( \exists x. \Phi \) is trivially satisfied in any structure \((s, h)\), such that \( \{u_1, \ldots, u_\ell\} \in \text{dom}(s) \). By Fact 1, it follows that eliminating the terms \{\( t_i \mid i \in I \}\) from each predicate atom \( p_XYZABC(t_1, \ldots, t_\ell) \) preserves equivalence.

(2c) The exhaustive application of rules (9) and (10), that add all possible disequalities between existentially quantified variables and constants, ensures that Condition (2c) is satisfied. Consequently, \( P_n \) is normalized.

Assume now that \( P \) is e-restricted, namely that each equational atom \( t \bowtie u \) occurring in \( P \) is such that \( \{t, u\} \cap \emptyset \neq \emptyset \). Note that the transformations (9) and (10) may introduce disequalities \( x \neq t' \), where \( x \) is an existentially quantified variable. In the case where \( P \) is e-restricted, we apply these rules only for \( t \in C \). Suppose that, after applying rules (7-8) exhaustively, there exist some equality \( t = u \) in a rule, such that neither \( t \) nor \( u \) is an existentially quantified variable. But since \( P \) is e-restricted, \( \{t, u\} \cap \emptyset \neq \emptyset \) and this rule will be eliminated by the disequalities introduced by the modified versions of the transformations (9) and (10). Finally, if \( P \) is (strongly) established then \( P_n \) is (strongly) established, because the transformation does not introduce new existential quantifiers and preserves equivalence.

Let us now compute the time complexity of the normalization procedure and the width of the output entailment problem. Observe that transformations (7–10) either instantiate existentially quantified variables, add or remove equalities, thus they can be applied \( O(\text{size}(P)) \) times, increasing the width of the problem by at most \( O(\text{size}(P)) \). After the exhaustive application of transformations (7–10), the number of rules in \( S \) and the number of sequents in \( \Sigma \) has increased by a factor of \( 2^{\text{width}(P)} \) and the width of the problem by a linear factor. Then \( \text{size}(P_1) = O(\text{size}(P) \cdot 2^\text{width}(P)) \) and \( \text{width}(P_1) = O(\text{width}(P)) \). The transformation of step (1c) increases the number of rules in \( S_1 \) by a factor of \( 2^\alpha = 2^{O(\text{width}(P)))} = 2^\Omega(\text{width}(P)^3) \), where \( \alpha = \max\{\#p \mid p(x_1, \ldots, x_\ell) \sqsubseteq S_1, \rho \} \leq \text{width}(P) \) and does not change the width of the problem, i.e. \( \text{size}(P_2) = \text{size}(P) \cdot 2^\Omega(\text{width}(P)^3) \) and \( \text{width}(P_2) = O(\text{width}(P)^3) \).

Next, going from \( P_2 \) to \( P_3 \), we do not increase the bounds on the size or width of the problem and we trivially obtain \( \text{size}(P_3) = \text{size}(P) \cdot 2^\Omega(\text{width}(P)^3) \) and \( \text{width}(P_3) = O(\text{width}(P)^3) \). Finally, going from \( P_3 \) to \( P_4 \), we increase the size of the problem by a factor of \( 2^{\text{size}(P)} \cdot 2^{\text{size}(\Sigma)} \) and, because \( |\Sigma| \leq \text{width}(P) \), by the definition of \( \text{width}(P) \), we obtain \( \text{size}(P_4) = \text{size}(P) \cdot 2^\Omega(\text{width}(P)^3) \) and \( \text{width}(P_4) = O(\text{width}(P)^3) \). Finally, the entire procedure has to be repeated for each partition \( \emptyset \) of the set of constants \( C \). Since the number of partitions is \( 2^{|\Sigma| \cdot \log_2 |\Sigma|} = 2^{\Omega(\text{width}(P) \cdot \log_2 \text{width}(P))} \), we obtain that the size of the result is \( \text{size}(P) \cdot 2^{\Omega(\text{width}(P)^3)} \). Since the increase in the size of the output problem is mirrored by the time required to obtain it, the execution of the procedure takes time \( \text{size}(P) \cdot 2^{\Omega(\text{width}(P)^3)} \).

B Proof of Theorem 13 (Section 3)

\boldsymbol{\triangleright} \textbf{Lemma 33.} Every established entailment problem \( P = (S, \Sigma) \) can be reduced in time \( 2^{\Omega(\text{width}(P)^3)} \) to a normalized and strongly established entailment problem \( P_e \), such that \( \text{width}(P_e) = O(\text{width}(P)^3) \).

\textbf{Proof:} First, we use Lemma 11 to reduce \( P \) to an established normalized problem \( P_n = (S_n, \Sigma_n) \) in time \( \text{size}(P) \cdot 2^{\Omega(\text{width}(P)^3)} \), such that \( \text{size}(P_n) = \text{size}(P) \cdot 2^{\Omega(\text{width}(P)^3)} \) and \( \text{width}(P_n) = O(\text{width}(P)^3) \).

Second, given a symbolic heap \( \phi \) and a variable \( x \), we define the set of symbolic heaps \( \mathcal{H}(\phi, x) \).
recursively on the structure of $\phi$, as follows:

\[
\begin{align*}
\mathcal{A}(t_1 \leftarrow t_2, x) & \equiv \emptyset \\
\mathcal{A}(t_0 \rightarrow (t_1, \ldots, t_k), x) & \equiv \{ t_0 \rightarrow (t_1, \ldots, t_k) \mid x = t_0 \} \\
\mathcal{A}(p(t_1, \ldots, t_n), x) & \equiv \mathcal{A}(p(x, t_1, \ldots, t_n)) \\
\mathcal{A}(\phi_1 \land \phi_2, x) & \equiv \bigcup_{i=1,2} \{ \phi_1 \land \phi_2 \mid \phi_i \in \mathcal{A}(\phi_{3-i}, x) \}
\end{align*}
\]

where $P$ is a fresh predicate symbol not occurring in $\mathcal{P}$, of arity $\#P = \#p + 1$ and the set of inductive rules is updated by replacing each rule $p(x_1, \ldots, x_p) \leftarrow S_x \mathcal{P}$ by the set of rules $\{ p(x, t_0, \ldots, t_n) \mid \phi \in \mathcal{A}(\rho, x_0) \}$. It is straightforward to show by induction that if $(s, h)$ is a structure such that $(s, h) \models \psi$ for some $\psi \in \mathcal{A}(\phi, x)$, then we have $s(x) \in \text{dom}(h)$. Observe that $||A(\phi, x)|| \leq 2^{\text{size}(\phi)}$ and $\text{size}(\phi) = O(\text{size}(\phi))$, for each $\psi \in \mathcal{A}(\phi, x)$.

Let $\phi_0 \land \phi_1, \phi_1, \ldots, \phi_n$ be a sequent from $\mathcal{P}_n$ and $(s, h)$ be a structure such that $(s, h) \models \phi_0$. By Definition 33, $\phi_0$ is quantifier-free. Assume that $\phi_1 = \exists x \cdot \psi_1$ (the argument is repeated for all existential quantifiers occurring in $\phi_1, \ldots, \phi_n$). Note that, since $\mathcal{P}_n$ is normalized, $x$ occurs in a points-to or a predicate atom in $\phi_1$. This implies that $x$ necessarily occurs in a points-to atom in each symbolic heap $\psi_1$ obtained by a predicate-free unfolding $\phi_1 \Rightarrow S_n \phi_1$, by point (2a) of Definition 8. Thus, $s'(x) \in \text{loc}(h)$, for each $s'$-associate $s'$ of $s$ such that $(s', h) \models \psi_1$. Since $S_n$ is established, each location from $\text{loc}(h)$ belongs to $s(C) \cup \text{dom}(h)$, thus $s'(x) \in s'(C) \cup \text{dom}(h)$. Hence $\phi_1$ can safely be replaced by the set of symbolic heaps $\{ \psi_1[t/x] \mid t \in C \cup [\exists x \cdot \psi \mid \psi \in \mathcal{A}(\phi_1, x)] \}$. Applying this transformation to each existentially quantified variable occurring in a sequent from $\mathcal{P}_n$ yields a strongly established problem $\mathcal{P}'$. Moreover, the reduction of $\mathcal{P}_n$ to $\mathcal{P}'$ requires $\text{size}(\mathcal{P}_n) \cdot 2^{O(\text{width}(\mathcal{P}_n))} = \text{size}(\mathcal{P}) \cdot 2^{O(\text{width}(\mathcal{P}))}$ time and the width of the outcome is $\text{width}(\mathcal{P}') = O(\text{width}(\mathcal{P}_n)) = O(\text{width}(\mathcal{P}))$.

### C Proof of Theorem 13 (Section 3)

Lemma 33, we can reduce $\mathcal{P}$ to a normalized strongly established entailment problem $\mathcal{P}_e = (S_e, \Sigma_e)$ in time $\text{size}(\mathcal{P}) \cdot 2^{O(\text{width}(\mathcal{P}))}$, such that width($\mathcal{P}_e$) = $O(\text{width}(\mathcal{P}))^2$. Let $\phi \Rightarrow S_n \phi$ be an arbitrary predicate-free unfolding of a symbolic heap $\phi$ on the right-hand side of a sequent in $\Sigma_e$, where $\phi = \exists x_1 \ldots \exists x_n \cdot \psi$ and $\psi$ is quantifier-free. Because $\mathcal{P}_e$ is normalized, there are no equalities in $\psi$. Let $x \neq y$ be a disequality from $\psi$, where $\{x, y\} \cap C = \emptyset$. By Definition 33, all variables from $\mathcal{P}_e$ are existentially quantified, thus it must be the case that $x, y \in \{x_1, \ldots, x_n\}$. Because $\mathcal{P}_e$ is strongly established, $\phi$ is $S_e$-established, thus both $x$ and $y$ are allocated in $\psi$. Moreover, since there are no equalities in $\psi$, there must exist two distinct points-to atoms $x \rightarrow (t_1, \ldots, t_0)$ and $y \rightarrow (u_1, \ldots, u_0)$ in $\psi$ such that, $(s, h) \models S_e \phi$ implies $(s', h') \models S_e \phi \Rightarrow S_n (x \rightarrow (t_1, \ldots, t_0) \land y \rightarrow (u_1, \ldots, u_0))$, for any structure $(s, h)$, for some heap $h' \subseteq h$ and $s'$ is a $(x_1, \ldots, x_n)$-associate of $s$. But then $(s', h') \models S_e \phi \Rightarrow S_n \phi$ and, since the choice of the structure $(s, h)$ was arbitrary, we can remove any disequality $x \neq y$ such that $\{x, y\} \cap C = \emptyset$ from $\mathcal{P}_e$. This transformation takes time $O(\text{size}(\mathcal{P}_e)) = \text{size}(\mathcal{P}) \cdot 2^{O(\text{width}(\mathcal{P}))}$ and does not increase the width of the problem. The outcome is of an e-restricted entailment problem.

### D Additional Material for Normal Structures (Section 4)

- **Definition 34.** Given symbolic heaps $\phi_1, \phi_2 \in \text{SH}^s$, a pair of structures $((s_1, h_1), (s_2, h_2))$ is a normal $S$-companion for $(\phi_1, \phi_2)$ iff $(s_i, h_i)$ is a normal $S$-model of $\phi_i$, for $i = 1, 2$ and:
  1. $\bar{s}_i(t) = \bar{s}_i(t)$, for each term $t \in \text{fv}(\phi_1) \cup \text{fv}(\phi_2)$.
  2. $\bar{s}_i(x) \cap \bar{s}_i(-) = \bar{s}(x)$, for $i = 1, 2$.

where $\phi_i \Rightarrow S_n \bar{s}_i \cdot \bar{u}_i$ are the predicate-free unfoldings and $\bar{s}_i$ is the $x_i$-associate of $s_i$ satisfying conditions (1) and (2) of Definition 15, for $i = 1, 2$, respectively. The normal $S$-companion $((s_1, h_1), (s_2, h_2))$ is, moreover, injective iff $s_1$ and $s_2$ are injective and $s_1((\text{fv}(\phi_1) \setminus \text{fv}(\phi_2)) \cup s_2((\text{fv}(\phi_2) \setminus \text{fv}(\phi_1))) = \emptyset$. 

© Mnacho Echenim, Radu Iosif and Nicolas Peltier; licensed under Creative Commons License CC-BY

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
Lemma 35. Given symbolic heaps $\phi_1, \phi_2 \in \text{SH}^d$, a structure $(s, h)$ is a (injective) normal $S$-model of $\phi_1 \ast \phi_2$ if there exists a (injective) normal $S$-companion $\langle (s_1, h_1), (s_2, h_2) \rangle$ for $\langle \phi_1, \phi_2 \rangle$, such that $h = h_1 \cup h_2$.

Proof: “⇒” Let $(s, h)$ be a normal $S$-model of $\phi_1 \ast \phi_2$. Then there exists a predicate-free unfolding $\phi_1 \ast \phi_2 \Rightarrow \exists \chi_1 . \psi_1 \ast \exists \chi_2 . \psi_2$ such that $\psi_1$ and $\psi_2$ are quantifier-free and $(s, h) \models \exists \chi_1 . \psi_1 \ast \exists \chi_2 . \psi_2$. By $\alpha$-renaming if necessary, we can assume that $x_i \cap \text{fv}(\psi_{3-i}) = \emptyset$, for $i = 1, 2$, thus $(s, h) \models \exists \chi_1 \exists \chi_2 . \psi_1 \ast \psi_2$.

Hence there exist an $(\chi_1 \cup \chi_2)$-associate $\bar{s}$ of $s$ and two disjoint heaps $h_1$ and $h_2$, such that $h = h_1 \cup h_2$ and $(\bar{s}, h_1) \models \psi_1$, for $i = 1, 2$. Let $\tilde{s}_i \overset{\text{def}}{=} s$, for $i = 1, 2$, so that $s = s_1 \cup s_2$. By considering the $x_i$-associate of $\bar{s}$ defined as the restriction of $\bar{s}$ to $x_i \cup \text{dom}(s)$ and using the fact that $(s, h)$ is a normal $S$-model of $\phi_1 \ast \phi_2$, it is easy to check that $\langle (s_1, h_1), (s_2, h_2) \rangle$ is a normal $S$-model of $\phi_1$. Further, points (1) and (2) of Definition 34 are easy checks. Finally, if $s$ is injective then trivially $s_1$ and $s_2$ are injective and $s_1(\text{trm}(\phi_1) \setminus \text{trm}(\phi_2)) \cap s_2(\text{trm}(\phi_2) \setminus \text{trm}(\phi_1)) = s(\text{trm}(\phi_1) \setminus \text{trm}(\phi_2)) \cap s(\text{trm}(\phi_2) \setminus \text{trm}(\phi_1)) = \emptyset$.

“⇐” If $\langle (s_1, h_1), (s_2, h_2) \rangle$ is a normal $S$-model of $\phi_1$, then there exist predicate-free unfoldings $\phi_1 \Rightarrow \exists \chi_1 . \psi_1 \ast \exists \chi_2 . \psi_2$ and $\langle (s_1, h_1), (s_2, h_2) \rangle$ is a predicate-free unfolding. Let $\tilde{s}_1$ and $\tilde{s}_2$ be the restrictions of $s_1$ and $s_2$ to $\text{trm}(\psi_1) \cup \text{trm}(\psi_2)$ for $i = 1, 2$, respectively. By point (1) of Definition 34, $\bar{s} \overset{\text{def}}{=} \tilde{s}_1 \cup \tilde{s}_2$ is a well-defined store and, since $x_1 \cap x_2 = \emptyset$, we obtain that $\bar{s} \overset{\text{def}}{=} \tilde{s}_1 \cup \tilde{s}_2$ is a well-defined $(x_1 \cup x_2)$-associate of $s$. To show that $(s, h) \models \phi_2$ is a normal $S$-model of $\phi_1 \ast \phi_2$, let $t_1, t_2 \in \text{trm}(\phi_1) \setminus \text{trm}(\phi_2)$ be distinct terms such that $\bar{s}(t_1) = \bar{s}(t_2)$ and suppose, for a contradiction, that $\bar{s}(t_1) \notin \bar{s}(C)$. Since $\langle s_1, h_1 \rangle$ is a normal $S$-model of $\phi_1$, for $i = 1, 2$, the only interesting cases are $t_i \in \text{trm}(\psi_1) \setminus \text{trm}(\psi_{3-i})$ and $t_i \in \text{trm}(\psi_{3-i}) \setminus \text{trm}(\psi_i)$. Assume $t_i \in \text{trm}(\psi_i)$ for $i = 1, 2$, the other case is symmetric. Since $t_i \notin \text{ctv}(\phi_1 \ast \phi_2)$, it must be the case that $t_i \in x_i$, for $i = 1, 2$. Then $\bar{s}(t_1) = \tilde{s}(t_1) = \tilde{s}(t_2) = \bar{s}(t_2)$, which contradicts point (2) of Definition 34. Finally, it is easy to check that $s = s_1 \cup s_2$ is injective, provided that $s_1$ and $s_2$ are injective and that $s_1(\text{trm}(\phi_1) \setminus \text{trm}(\phi_2)) \cap s_2(\text{trm}(\phi_2) \setminus \text{trm}(\phi_1)) = s_1(\text{trm}(\phi_1) \setminus \text{trm}(\phi_2)) \cap s_2(\text{trm}(\phi_2) \setminus \text{trm}(\phi_1)) = \emptyset$.

The following lemma states an important property of normal $S$-models, that will be used to build abstract composition operators, needed to define a finite-range abstraction of an infinite set normal structures.

Lemma 36. Given symbolic heaps $\phi_1, \phi_2 \in \text{SH}^d$ and $\langle (\bar{s}, h_1), (\bar{s}, h_2) \rangle$ an injective normal $S$-companion for $\langle \phi_1, \phi_2 \rangle$, we have $\text{Fr}(h_1, h_2) \subseteq \bar{s}(\text{fv}(\phi_1) \cap \text{fv}(\phi_2) \cup C)$.

Proof: Let $\ell \in \text{Fr}(h_1, h_2) = \text{loc}(h_1) \cap \text{loc}(h_2)$ be a location, $\phi_i \Rightarrow \exists \chi_i . \psi_i$ be predicate-free unfoldings and $\bar{s}_i$ be the $x_i$-associates of $\bar{s}$ that satisfy points (1) and (2) of Definition 34, such that $\bar{s}_i(h_i) = \psi_i$, for $i = 1, 2$. By $\alpha$-renaming, if necessary, we assume w.l.o.g. that $x_i \cap \text{fv}(\psi_{3-i}) = \emptyset$, for $i = 1, 2$. Because $\ell \in \text{loc}(h_1)$, there exist points-to atoms $t_0 : \rightarrow (t'_1, \ldots, t'_k) \in \text{psi}$, such that $\ell = \bar{s}_1(t'_1) = \bar{s}_2(t'_2)$, for some $i_1, i_2 \in \{0, \ldots, k\}$ and $i = 1, 2$. We distinguish two cases:

- if $t'_1 \in \text{trm}(\phi_1)$ and $t'_2 \in \text{trm}(\phi_2)$, since $\bar{s}_i$ is a $x_i$-associate of $\bar{s}$, $\bar{s}_i$ and $\bar{s}$ agree over $\text{trm}(\phi_i)$, for $i = 1, 2$, we obtain $\bar{s}(t'_1) = \bar{s}_1(t'_1) = \bar{s}_2(t'_2) = \bar{s}(t'_2)$, thus $t'_1 = t'_2$, because $\bar{s}$ is injective, hence $\ell \in \bar{s}(\text{trm}(\phi_1) \cup \text{trm}(\phi_2)) \subseteq \bar{s}(\text{fv}(\phi_1) \cup \text{fv}(\phi_2) \cup C)$.

- else $t'_1 \in \text{trm}(\phi_1) \setminus \text{trm}(\phi_2)$ (the case $t'_2 \in \text{trm}(\phi_2) \setminus \text{trm}(\phi_2)$ is symmetric). If $t'_1 \in C$, we obtain $\ell = \bar{s}_1(t'_1) = \bar{s}(t'_1) \in C$, because $C \subseteq \text{dom}(\bar{s})$ and $\bar{s}$ agrees with $\bar{s}$ over $C$. Else $t'_1 \in x_1$ and we distinguish two cases:

- if $t'_2 \in \text{ctv}(\phi_2)$, we obtain $\ell = \bar{s}_2(t'_2) = \bar{s}(t'_2) \in C$, by the above argument.

- else $t'_2 \in \text{fv}(\phi_2)$ and $\bar{s}_1(t'_1) = \bar{s}_2(t'_2) \in \bar{s}(C)$ by point (2) of Definition 34.

Example 37. Consider the structures defined in Example 16. The structure $(s, h)$ is a normal model of $p(x_1) \ast p(x_2)$: we have $(s, h) \models p(x_1)$ with $h = (\ell \mapsto \ell_3)$ (for $i = 1, 2$), $h = h_1 \cup h_2$ and $\text{Fr}(h_1, h_2) = \{\ell_3\} \subseteq C$. Similarly, $(s, h')$ is a normal model of $p(x_1) \ast p(x_2)$, $(s, h') \models p(x_i)$ with
\(b'_i = (\ell_i \mapsto \ell_{i+1})\) (for \(i = 1, 2\)), \(b'_2 = b'_1 \uplus b'_2\) and \(Fr(b'_1, b'_2) = \emptyset\). On the other hand, \((s, b'')\) is not normal: we have \((s, b'') \models p(x_i)\) with \(b''_i = (\ell_i \mapsto \ell_4)\) (for \(i = 1, 2\)), \(b'' = b''_1 \uplus b''_2\) and \(Fr(b''_1, b''_2) = \{\ell_4\} \not\subseteq s(\fv(p(x_1)) \cup \fv(p(x_2)) \cup \emptyset) = \{\ell_3\}\. \\

The proof of this result (Lemma 17) relies on the following definition and lemmas.

▶ **Definition 38.** A total function \(\gamma : L \rightarrow L\) is compatible with a structure \((s, b)\) if and only if, for all \(\ell_1, \ell_2 \in L\) such that either \(\ell_1, \ell_2 \in \dom(b)\) or \(\ell_1 \in s(\emptyset)\), if \(\gamma(\ell_1) = \gamma(\ell_2)\) then \(\ell_1 = \ell_2\). We define 

\[
\gamma(b) \equiv ((\gamma(\ell), (\gamma(\ell_1), \ldots, \gamma(\ell_n))) \mid (b(\ell) = (\ell_1, \ldots, \ell_n)), \text{ whenever } \gamma \text{ is compatible with } (s, b)\. \\

▶ **Lemma 39.** Let \(S\) be an e-restricted (resp. normalized) set of rules and \(\phi\) be an e-restricted formula. Then, each unfolding \(\psi\) of \(\phi\) is e-restricted (resp. normalized).

**Proof:** The proof is by induction on the length of the unfolding sequence \(\phi \Rightarrow^*_S \psi\). ▷

▶ **Lemma 40.** If \(S\) is an e-restricted set of rules, \(\phi\) is an e-restricted formula and \((s, b)\) is an \(S\)-model of \(\phi\), then for any total function \(\gamma\) compatible with \((s, b)\), the following hold: (1) \(\gamma(b)\) is a heap, (2) \((\gamma \circ s, \gamma(b)) \models_S \phi\).

**Proof:** (1) The set \(\{\gamma(\ell) \mid \ell \in \dom(b)\}\) is finite, because \(\dom(b)\) is finite. Consider two tuples 

\[
(\gamma(\ell_1), (\gamma(\ell_1), \ldots, \gamma(\ell_n))) \text{ and } (\gamma(\ell'), (\gamma(\ell'), \ldots, \gamma(\ell_n'))) \in \gamma(b)\text{ and assume that } \gamma(\ell) = \gamma(\ell')\. \\

Then since \(\gamma\) is compatible with \((s, b)\), necessarily \(\ell = \ell'\). Since \(b\) is a partial function, we have \((\ell_1, \ldots, \ell_n) = (\ell'_1, \ldots, \ell'_n)\), so that \(\gamma(b)\) is also a finite partial function.

(2) If \((s, b) \models_S \phi\) then there exists a predicate-free unfolding \(\phi \Rightarrow^*_s \psi = \exists x. \ast x^{n}_{i=1} t_i = u_i * \ast x^{m}_{i=1} t'_{i} \neq u'_{i} * \ast x^{k}_{i=1} x_i \mapsto (t'_{1}, \ldots, t'_{k})\), such that \((\bar{s}, b) \models \psi\), for an \(x\)-associate \(\bar{s}\) of \(s\). Note that \(\gamma \circ s\) is an \(x\)-associate of \(\gamma \circ \bar{s}\), because \(\gamma\) is total. Moreover, because \(\phi\) and \(S\) are both e-restricted, by Lemma 39, \(\psi\) is e-restricted, thus we can assume that \(t_i \in C\), for all \(i \in [1 \ldots n]\) and that \(t'_{i} \in C\), for all \(i \in [1 \ldots m]\). We consider the three types of atoms from \(\psi\) below:

\[
\text{For any } i \in [1 \ldots n], \text{ since } (\bar{s}, 0) \models t_i = u_i, \text{ we have } \bar{s}(t_i) = \bar{s}(u_i), \text{ thus } \gamma(\bar{s}(t_i)) = \gamma(\bar{s}(u_i)), \text{ leading to } \\
(\gamma \circ \bar{s}, 0) \models t_i = u_i. \\
\text{For any } i \in [1 \ldots m], \text{ since } (\bar{s}, 0) \models t'_{i} \neq u'_{i}, \text{ we have } \bar{s}(t'_{i}) \neq \bar{s}(u'_{i}). \text{ Because } t'_{i} \in C \text{ and } (s, b) \models \phi, \text{ we have } t'_{i} \in \dom(s) \text{ and } \bar{s}(t'_{i}) = s(t'_{i}) \in s(\emptyset). \text{ By Definition 38, we obtain } \gamma(\bar{s}(t'_{i})) = \gamma(\bar{s}(u'_{i})), \text{ thus } \\
(\gamma \circ \bar{s}, 0) \models t'_{i} = u'_{i}. \\
\text{If } (\bar{s}, 0) \models \ast x^{k}_{i=1} x_i \mapsto (t'_{1}, \ldots, t'_{k}) \text{ then } \bar{s}(x_1), \ldots, \bar{s}(x_k) \text{ are pairwise distinct and } \dom(b) = (\bar{s}(x_1), \ldots, \bar{s}(x_k)). \text{ Since } \bar{s}(x_1), \ldots, \bar{s}(x_k) \in \dom(b), \text{ by Definition 38, we obtain that } \gamma(\bar{s}(x_1)), \ldots, \gamma(\bar{s}(x_k)) \text{ are pairwise distinct and } \dom(\gamma(b)) = (\gamma(\bar{s}(x_1)), \ldots, \gamma(\bar{s}(x_k))). \text{ We have } b(\bar{s}(x_1)) = (\bar{s}(x_1), \ldots, \bar{s}(x_k)), \text{ thus } \\
\gamma(b)(\bar{s}(x_1)) = (\gamma(\bar{s}(x_1)), \ldots, \gamma(\bar{s}(x_k))), \text{ for each } i \in [1 \ldots k], \text{ by Definition 38 and } (\gamma \circ \bar{s}, \gamma(b)) \models \\
\ast x^{k}_{i=1} x_i \mapsto (t'_{1}, \ldots, t'_{k}). \text{ ▷}

**E **Proof of Lemma 17 (Section 4)

This direction is trivial. “⇐” Let \((s, b)\) be an injective \(S\)-model of \(\phi\). Then by Lemma 39, there exists a predicate-free unfolding \(\phi \Rightarrow^*_s \exists x. \varphi\), where \(\varphi = \ast x^{m}_{i=1} t_i = u_i * \ast x^{k}_{i=1} x_i \mapsto (t'_{1}, \ldots, t'_{k})\) is e-restricted and normalized, and an \(x\)-associate \(\bar{s}\) of \(s\) such that 

\[
(\bar{s}, 0) \models \varphi. \text{ Note that } \varphi \text{ contains no equalities since it is normal and, since it is e-restricted, we can assume that } t_i \in C, \text{ for all } i \in [1 \ldots m]. \text{ We consider a store } s' : \dom(\bar{s}) \rightarrow L \text{ that satisfies the following hypothesis: } \\
(a) \ s'(t) = \bar{s}(t), \text{ for each } t \in \dom(\bar{s}) \text{ such that } \bar{s}(t) \in \bar{s}(C), \\
(b) \ s'(t) \neq s'(u), \text{ for all terms } t \neq u \in \dom(\bar{s}) \text{ such that } \bar{s}(t) \notin \bar{s}(C) \text{ or } \bar{s}(u) \notin \bar{s}(C). \\
\text{Note that such a store exists because } L \text{ is infinite, thus all terms that are not already mapped by } \bar{s} \text{ into locations from } \bar{s}(C) \text{ can be mapped to pairwise distinct locations, not occurring in } \bar{s}(C). \text{ Then we define }
The formal semantics of the bounded quantifiers is stated below:

Let \( \gamma \) be a \( \mathcal{S} \)-model of \( \phi \), according to Definition 15 (simply let \( s \)' be its x-associate). Because \( s'' \) is injective, by the assumption of the Lemma, we obtain \( (s'', b') \models_{\mathcal{S}} \psi_i \), for some \( i \in \{1 \ldots n\} \), and we are left with proving the sufficient condition \((s, b) \models_{\mathcal{S}} \psi_i \). To this end, consider the function \( \gamma : L \rightarrow \mathbb{L} \), defined as:

\[
\gamma(x) = \frac{\bar{x}}{\mathcal{S}}, \quad \text{for all } x \in \text{dom}(s''),
\]

\[
\gamma(\ell) = \frac{\ell}{1}, \quad \text{for all } \ell \in L \setminus \text{rng}(s'').
\]

Observe that \( \gamma \) is well-defined, since by definition of \( s', s'(x) = s'(x') \Rightarrow \overline{s}(x) = \overline{s}(x') \). Below we check that \( \gamma \) is compatible with \((s'', b')\). Let \( \ell_1, \ell_2 \in L \) be two locations such that \( \gamma(\ell_1) = \gamma(\ell_2) \):

- if \( \ell_1, \ell_2 \in \text{dom}(b) \) then \( \ell_1 = s''(x_i) \) and \( \ell_2 = s''(x_j) \), for some \( i, j \in \{1 \ldots k\} \), by definition of \( b' \). Suppose, for a contradiction, that \( i \neq j \). Then \( \overline{s}(x_i) = \gamma(s''(x_i)) = \gamma(s'(x_j)) = \overline{s}(x_j) \), which contradicts the fact that \( (s, b) \models_{\mathcal{S}} \psi_i \Rightarrow (t'_1, \ldots, t'_k) \). Hence \( i = j \), leading to \( \ell_1 = \ell_2 \).
- if \( \ell_1 \in \text{dom}(c) \) then let \( c \in \mathbb{C} \) be a constant such that \( \ell_1 = c''(c) \), so that \( \gamma(\ell_1) = c(\mathcal{C}) \). Suppose, for a contradiction, that \( \ell_2 \notin \text{rng}(s'') \). Then \( \gamma(\ell_2) = \overline{\ell_2}(c) \), hence \( \ell_2 \notin \mathbb{C} \). But since \( \bar{c} \) and \( c'' \) agree over \( \mathcal{C} \), we have \( \overline{\ell_2}(c) = c''(c) \), which contradicts with \( \ell_2 \notin \text{rng}(s'') \). Thus \( \ell_2 \in \text{rng}(s'') \) and let \( \ell_2 = s''(t) \), for some term \( t \). We have \( \gamma(s''(t)) = \overline{\mathcal{C}}(t) \), thus \( \overline{\ell_2} = \gamma(\ell_2) = \gamma(\ell_1) = \overline{\ell_1} \). By point (a), we obtain \( \ell_2 = s''(t) = \overline{\mathcal{C}}(t) = s''(c) = \ell_1 \). Moreover, it is easy to check that \( (s, b) = (\gamma \circ s'', \gamma(b')) \). Since \( \bar{s} \) is the restriction of \( \bar{t} \) to \( \text{trm}(\phi) \), by Lemma 40, we obtain \((s, b) \models \psi_i \). \( \square \)

**F** Additional Material on Core Formulae (Section 5)

The formal semantics of the bounded quantifiers is stated below:

**Lemma 41.** Given a \( \mathcal{S}\text{-}\mathcal{L}^k \) formula \( \phi \) and \( x \in \text{fv}(\phi) \), the following hold, for any structure \((s, b)\):

1. \((s, b) \models_{\mathcal{S}} \exists x. \phi \iff (s(x \leftarrow \ell, b) \models_{\mathcal{S}} \phi, \text{ for some } \ell \in \text{loc}(b) \setminus \text{fv}(\phi) \setminus \{x\} \cup \mathcal{C}) \)
2. \((s, b) \models_{\mathcal{S}} \forall x. \phi \iff (s(x \leftarrow \ell, b) \models_{\mathcal{S}} \phi, \text{ for all } \ell \in L \setminus \text{loc}(b) \setminus \text{fv}(\phi) \setminus \{x\} \cup \mathcal{C}) \)

**Proof:** First, for any structure \((s, b)\), we have \((s, b) \models \text{loc}(x) = s(x) \in \text{loc}(b)\).

(1) By definition, \( \exists x. \phi \) is equivalent to \( \exists x. (\text{loc}(\phi) \text{)} \cup \mathcal{C}) \Rightarrow x = \gamma \land \text{loc}(x) \land \phi \).

(2) By definition, \( \forall x. \phi \) is equivalent to \( \forall x. (\text{loc}(\phi) \text{)} \cup \mathcal{C}) \Rightarrow x = \gamma \land \text{loc}(x) \land \phi \).

Below we prove the equivalence between the atoms \( p(t) \) and \( \text{emp} \rightarrow p(t) \).

**Lemma 42.** A structure \((s, b)\) is an \( \mathcal{S} \)-model of \( p(t) \) if and only if \((s, b)\) is a \( \mathcal{S}_{\mathcal{C}} \)-model of \( \text{emp} \rightarrow p(t) \).

**Proof:** “⇒” For each rule \( p(x) \Rightarrow_{\mathcal{S}} \exists x. \psi \Rightarrow_{\mathcal{S}} p(y) \), there exists a rule \( \text{emp} \rightarrow p(x) \Rightarrow_{\mathcal{S}} \exists x. \phi \Rightarrow_{\mathcal{S}} \text{emp} \rightarrow q_1(y) \), corresponding to the case where the substitution \( \sigma \) is empty. The proof follows by a simple induction on the length of the predicate-free unfolding of \( p(t) \). “⇐” We prove the other direction by induction on the length of the predicate-free unfolding of \( \text{emp} \rightarrow p(t) \). Assume \((s, b)\) is a \( \mathcal{S}_{\mathcal{C}} \)-model of \( \text{emp} \rightarrow p(t) \). Then there exist a rule \( \text{emp} \rightarrow p(x) \Rightarrow_{\mathcal{S}} \exists x. \psi \Rightarrow_{\mathcal{S}} \text{emp} \rightarrow p_1(\sigma(w_j)) \)
in $\mathbb{C}_S$ and a $v$-associate $s'$ of $s$ such that $(s', b) \models \psi \sigma \theta * \xi^m_{j=1} (\text{emp} \to p_j(\theta \circ \sigma(w_j)))$. By definition of $\mathbb{C}_S$, this entails that $p(t)$ can be unfolded into $\exists z \cdot \psi \theta * \xi^m_{j=1} p_j(\theta(w_j))$ using the rules in $S$. The heap $h$ can be decomposed into $h_0 \uplus \cdots \uplus h_m$, where $(s', b_j) \models \text{emp} \to p_j(\theta \circ \sigma(w_j))$, for $j \in \{1 \ldots m\}$.

By the induction hypothesis, $(s', b_j)$ is an $S$-model of $p_j(\theta \circ \sigma(w_j))$, and we deduce that $(s, b)$ is an $S$-model of $\exists z \cdot \psi \theta * \xi^m_{j=1} p_j(\theta(w_j))$.

Another property of context predicate atoms is stated by the lemma below:

**Lemma 43.** If $S$ is progressing, then for each store (resp. injective store) $s$, we have $(s, \emptyset) \models_{S} \psi \to (p(t))$ if and only if $n = 1$, $p = q_1$ and $s(t) = s(u_1)$ (resp. $t = u_1$).

**Proof:** “⇒” If $(s, \emptyset) \models_{S} \psi \to (p(t))$, then there exists a rule $\psi^m_{i=1} q_i(u_i) \to p(t)$ such that $(s, 0) \models_{S} \phi \sigma$ and a substitution $\sigma$ such that $(s, 0) \models_{S} \phi \sigma$, where $\sigma = [t/x, u_1/y_1, \ldots, u_n/y_n]$. If the rule is an instance of (I) then $n = 1$, $p = q_1$ and $(s, \emptyset) \models t = u_1$, leading to $s(t) = s(u_1)$. If, moreover, $s$ is injective, we get $t = u_1$. Otherwise, if the rule is an instance of (II), then since $S$ is progressing, $\phi \sigma$ may contain exactly one points-to atom, hence $(s, \emptyset) \models \phi \sigma$ cannot be the case. “⇐” This is a simple application of rule (I).

The following lemma states a technical result about core formulæ, which will be used in the proof of Lemma 30:

**Lemma 44.** For each quantifier-free core formula $\varphi$, each injective $\mathbb{C}_S$-model $(s, b)$ of $\varphi$, such that $||b|| = 1$, and each term $t \in \text{roots}_{SB}(\varphi)$, we have $(\hat{s}(t)) \in \text{loc}(b) \cup \hat{s}(C)$.

**Proof:** Let $\varphi$ be a quantifier-free core formula of the following form (cf. Definition 19):

$$\xi^m_{i=1} (\xi^k_{j=1} q_i(u_i) \to p_1(t_1)) * \xi^m_{i=m+1} v \equiv (t_1', \ldots, t_q')$$ (12)

The proof goes by induction on $||b||$. In the base case, $||b|| = 1$, we prove first that the formula contains exactly one points-to or predicate atom. Suppose, for a contradiction, that it contains two or more points-to atoms, i.e. $\varphi = \alpha_1 \ast \ldots \ast \alpha_m$, for $m \geq 2$. If $\alpha_1$ and $\alpha_2$ are points-to atoms, it cannot be the case that $(s, b)$ is a $\mathbb{C}_S$-model of $\varphi$, so we distinguish two cases:

- If $\alpha_1 = \xi^k_{i=1} q_i(u_i) \to p(t_1)$ and $\alpha_2$ is a points-to atom then, since $||b|| = 1$, we must have $(s, \emptyset) \models_{S} \alpha_1$ and $(s, b) \models \alpha_2$. By Lemma 43, we obtain $k = 1$ and $q_1(u_1) = p(t_1)$, which violates the condition on the uniqueness of roots in $q_1(u_1) \to p(t_1)$, in Definition 19.

- Otherwise, $\alpha_1$ and $\alpha_2$ are both predicate atoms; we assume that $(s, \emptyset) \models_{S} \alpha_1$ (the case $(s, \emptyset) \models_{S} \alpha_2$ is identical). We obtain a contradiction by the argument used at the previous point.

If $\varphi$ consists of a single points-to atom, then $\text{roots}_{SB}(\varphi) = \emptyset$ and there is nothing to prove. Otherwise, $\varphi$ is of the form $\alpha_1 = \xi^k_{i=1} q_i(u_i) \to p(t)$. By Lemma 43, since $S$ is progressing and $(s, b) \models_{S} \alpha_1$, $\xi^k_{i=1} q_i(u_i) \to p(t)$, either $k > 1$ or $k = 1$ and $q_1(u_1) \neq p(t)$. By Condition (II), there exists:

- a rule $p(x) \equiv_{S} \exists z \cdot \psi \ast \xi^m_{i=1} p_j(w_j)$,
- (b) separating conjunctions of predicate atoms $y_1, \ldots, y_m$, such that $\xi^m_{j=1} y_j = \xi^k_{i=1} q_i(y_i)$,
- (c) a substitution $\tau : z \to x \cup \{y_i \mid i \in [\ell_1, \ell_2]\}$

that induce the rule:

$$\xi^k_{i=1} q_i(y_i) \to p(x) \equiv_{S} \exists v \cdot \psi \tau * \xi^m_{j=1} y_j \to p_j(\tau(w_j)),$$

where $v = z \setminus \text{dom}(\tau)$. Assume w.l.o.g. that $(s, b) \models_{S} \xi^k_{i=1} q_i(u_i) \to p(t)$ is the consequence of the above rule, meaning that:

$$(s, b) \models_{S} (\exists v \cdot \psi \tau * \xi^m_{j=1} (y_j \to p_j(\tau(w_j)))) \sigma,$$ where $\sigma = [t/x, u_1/y_1, \ldots, u_n/y_n]$.

Let $\exists$ be the $v$-associate of $s$ such that $(\exists, b) \models_{S} \psi \sigma \ast \xi^m_{i=1} (y_j \to p_j(\tau(w_j))))$. Since $S$ is progressing, $\psi$ contains a points-to atom $t_0 \mapsto (t_1, \ldots, t_q)$, such that $(\exists, b) \models_{S} t_0 \mapsto (t_1, \ldots, t_q) \tau \sigma$ and
Lemma 45. Let $\phi = \bigwedge_{j=1}^k q_j(u_j) \rightarrow p(t)$ be a core formula and let $(s, h)$ be an injective structure. If $\phi \in \mathcal{T}(\phi)$ then $\phi|^h \in \mathcal{T}(\phi|h)$. The following lemmas relate a symbolic heap $\psi$ with the core formulæ $\psi \in \mathcal{T}(\phi)$. By considering separately the cases where $\phi$ is quantifier-free, or existentially quantified. In the latter case, we require moreover that the set of rules providing the interpretation of predicates be normalized.

Lemma 47. Given a quantifier-free symbolic heap $\psi \in \mathcal{SH}^s$, containing only predicate atoms that are contexts, an injective structure $(s, h)$ is a $E_S$-model of $\phi$ if $(\hat{s}, h) \models E_S \psi$, for some $\psi \in \mathcal{T}(\phi)$. The proof follows from the above lemmas, by considering separately the cases where $\phi$ is quantifier-free, or existentially quantified. In the latter case, we require moreover that the set of rules providing the interpretation of predicates be normalized.

Proof: “$\Rightarrow$” By induction on the structure of $\phi$. We consider the following cases:

- $\phi = \text{emp}$, $\phi = \text{true}$, and $\phi = \bigwedge_{j=1}^k q_j(u_j) \rightarrow p(t)$: in these cases, the only element in $\mathcal{T}(\phi)$ is $\phi$ itself and we have the result.
- $\phi = t_1 = t_2$: since $(\hat{s}, h) \models t_1 = t_2$, we have $(\hat{s}(t_1) = \hat{s}(t_2)$ and $h = \emptyset$. Since $\hat{s}$ is injective, we obtain $t_1 = t_2$.
- $\phi = t_1 \neq t_2$: since $(\hat{s}, h) \models t_1 \neq t_2$, we have $(\hat{s}(t_1) \neq \hat{s}(t_2)$ and $h = \emptyset$, therefore $t_1 \neq t_2$, $\mathcal{T}(t_1 \neq t_2) = \{\text{emp}\}$ and $(\hat{s}, h) \models \text{emp}$, because $h = \emptyset$.
- $\phi = \phi_1 \ast \phi_2$: since $(\hat{s}, h) \models \phi_1 \ast \phi_2$, there exist heaps $h_1$ and $h_2$, such that $h = h_1 \cup h_2$ and $(\hat{s}, h) \models \mathcal{S}\phi_1$, for $i = 1, 2$. By the inductive hypothesis, there exists $\psi_i \in \mathcal{T}(\phi_i)$ such that $(\hat{s}, h) \models E_S \psi_i$, for $i = 1, 2$. Then $(\hat{s}, h) \models E_S \psi_1 \ast \psi_2$, where $\psi_1 \ast \psi_2 \in \mathcal{T}(\phi_1 \ast \phi_2)$.

“$\Leftarrow$” By induction on the structure of $\phi$, we consider only the equational atoms below, the proofs in the remaining cases are straightforward:

- $\phi = t_1 = t_2$: since there exists $\psi \in \mathcal{T}(\phi)$ such that $(\hat{s}, h) \models \text{em} \psi$ and $\phi = \bigwedge{t_1 \neq t_2}$, we have $\psi(t_1) \neq \psi(t_2)$, for some $\psi \in \mathcal{T}(\phi)$, which implies $t_1 \neq t_2$. Since $(\hat{s}, h) \models \text{em} \psi$, $h = \emptyset$ and $(\hat{s}, h) \models t_1 = t_2$.
- $\phi = t_1 \neq t_2$: since there exists $\psi \in \mathcal{T}(\phi)$ such that $(\hat{s}, h) \models \text{em} \psi$ and $\phi = \bigwedge{t_1 \neq t_2}$, which implies $t_1 \neq t_2$. Since $(\hat{s}, h) \models \text{em} \psi$, $h = \emptyset$ and $(\hat{s}, h) \models t_1 = t_2$.
Proof of Lemma 20 (Section 5)

"⇒" By induction on size(φ). We consider the following cases:

1. φ = emp, φ = t₁ → (t₁, ..., tₖ), φ = t₁ = t₂, φ = t₁ ≠ t₂ and φ = φ₁ ∗ φ₂: the proof is the same as the one in Lemma 47.

2. φ = p(t): in this case T(φ) = [emp → p(t)] and the conclusion follows application of Lemma 42.

3. φ = ∃x . φ₁: since (s, b) |=₆ ∃x . φ₁, there exists ℓ ∈ L such that (s[x ← ℓ], b) |=₆ φ₁ and we distinguish the following cases.

   a. If ℓ ∉ dom(φ₁) ∩ C, since φ is normalized, by Definition 8 (1a) x occurs in a points-to or in a predicate atom of φ₁. Since S is normalized, by Definition 8 (2a), we have ℓ ∈ loc(b).

   b. Since dom(φ₁) = tv(φ₁) ∩ C, the store (s[x ← ℓ]) is necessarily injective, hence (s[x ← ℓ], b) |=₆ φ₁, for some φ₁ ∈ T(φ₁), by the inductive hypothesis and (s, b) |=₆ ∃x . φ₁, by Lemma 41.

   c. Otherwise, ℓ ∈ dom(φ₁) ∩ C and let t ∈ tv(φ₁) ∩ C be a term such that ℓ = ℓ(t). Then (s, b) |=₆ φ₁[1/t] and (s, b) |=₆ ∃x . φ₁, for some φ₁ ∈ T(φ₁[1/x]), by the inductive hypothesis.

"⇐" By induction on size(φ), considering the following cases:

1. φ = emp, φ = t₁ → (t₁, ..., tₖ), φ = t₁ = t₂, φ = t₁ ≠ t₂ and φ = φ₁ ∗ φ₂: the proof is the same as the one in Lemma 47.

2. φ = p(t): in this case ψ = emp → p(t) is the only possibility and the conclusion follows by an application of Lemma 42.

3. φ = ∃x . φ₁: by the definition of T(φ), we distinguish the following cases:

   a. If (s, b) |=₆ ∃x . φ₁, for some φ₁ ∈ T(φ₁), then (s[x ← ℓ], b) |=₆ φ₁, for some ℓ ∈ loc(b) \ dom(φ₁) \ ∪ C. By the definition of T(φ₁), we have tv(φ₁) ⊆ tv(φ₁) and suppose, for a contradiction, that there exists a variable y ∈ tv(φ₁) \ tv(φ₁). Then y can only occur either in an equality atom y = y or in some disequality y ≠ t, for some term t ≠ y, and nowhere else.

   b. Both cases are impossible, because φ is normalized, thus by Condition (1b) of Definition 8, y necessarily occurs in a points-to or predicate atom. Hence, tv(φ₁) = tv(φ₁) and consequently, we obtain ℓ ∈ loc(b) \ dom(φ₁) \ ∪ C. Since dom(φ₁) = tv(φ₁) \ ∪ C, by the hypothesis of the Lemma, s[x ← ℓ] is injective and, by the inductive hypothesis, we obtain (s[x ← ℓ], b) |=₆ φ₁, thus (s, b) |=₆ φ₁.

   c. Otherwise, (s, b) |=₆ ∃x . φ₁, for some φ₁ ∈ T(φ₁[1/x]) and some t ∈ tv(φ₁) ∩ C. By the induction hypothesis, we have (s, b) |=₆ φ₁[1/t], thus (s, b) |=₆ ∃x . φ₁. □

Additional Material for Core Formulae (Section 5)

Lemma 48. Given an injective structure (s, b) and a context predicate atom *n₁=q₁(u₁) → p(t), we have (s, b) |=₆ *n₁=q₁(u₁) → p(t) iff (s, b) |=₆ φ, for some core unfolding *n₁=q₁(u₁) → p(t) ≈₁ₑ φ and some injective extension ˘s of s.

Proof: We assume w.l.o.g. a total well-founded order ≤ on the set of terms T and, for a set T ⊆ T, we denote by min₆ T the minimal term from T with respect to this order. In the following, let θ = t₁/x₁, y₁/₁, ..., uₙ/yₙ.

⇒ If (s, b) |=₆ *n₁=q₁(u₁) → p(t) then there exists a rule *n₁=q₁(y₁) → p(x) |=₆ ∃z . φ, where φ is quantifier-free, such that (s, b) |=₆ ∃z . φ. Let ˘s be a (not necessarily injective) z-associate of s such that (˘s, b) |=₆ φ. We define a substitution τ, such that dom(τ) = trm(φ) ≤ dom(˘s) and for each x ∈ dom(τ):

   a. if x ∈ dom(˘s) then τ(x) = x,

   b. else, if x ∉ dom(˘s) and ˘s(x) = ˘s(y), for some y ∈ dom(˘s), then τ(x) = min₆ z ∈ dom(˘s) \ ˘s(z) = ˘s(y),

   c. otherwise, if x ∉ dom(˘s) and ˘s(x) ≠ ˘s(y), for all y ∈ dom(˘s), then τ(x) = min₆ y ∈ dom(˘s) \ ˘s(y) = ˘s(x).

© Mnacho Echenim, Radu Iosif and Nicolas Peltier; licensed under Creative Commons License CC-BY
Let $E \equiv \{ \{ y \in \text{dom}(\tilde{h}) \mid \exists x (x = \tilde{h}(y)) \} \mid x \in \text{dom}(\tilde{h}) \}$; by construction, the sets in $E$ are pairwise disjoint.

Let $\tilde{h}$ be the restriction of $\tilde{h}$ to the set $\text{dom}(\tilde{h}) \cup \{ \text{min}, \text{max} \}$ such that $K \cap \text{dom}(\tilde{h}) = \emptyset$. Because $\tilde{h}$ is injective, $\tilde{h}$ is easily shown to be injective, thus it is an injective extension of $\tilde{h}$. Moreover, because $(\tilde{h}, h) \models E \phi$ and $\tilde{h}$ agrees with $\tilde{h} \circ \tau$ on $\text{dom}(\tilde{h})$, we deduce that $(\tilde{h}, h) \models E \phi (\tau \circ \theta)$. We conclude by noticing that $(\tilde{h}, h) \models \phi \tau$ for some $\phi \in T (\phi (\tau \circ \theta))$, by an application of Lemma 47, because $\phi (\tau \circ \theta)$ is quantifier-free.

Given a injective structure $(\tilde{h}, h)$ and a context predicate atom $\phi_{\tau} \models C(u_i)$, we have $(\tilde{h}, h) \models \phi \tau$ if and only if $(\tilde{h}, h) \models \phi \tau$ for some core unfolding $\phi_{\tau} \models C(u_i)$.

Proof: “$\Rightarrow$” Let $\tilde{h}'$ be the restriction of $\tilde{h}$ to $t \cup \bigcup_{i=1}^{n} u_i$. Clearly, we have $(\tilde{h}', h) \models \phi_{\tau} \models C(u_i)$, and we prove the other direction.

By Lemma 48, there exists a core unfolding $\phi_{\tau} \models C(u_i)$ and an injective extension $\tilde{h}'$ of $\tilde{h}$, such that $(\tilde{h}, h) \models \phi \tau$ for some $\phi \in T (\phi (\tau \circ \theta))$. Let $\tau$ be the substitution defined by $\tau(x) = u$ if and only if $\tilde{h}'(x) = \tilde{h}(u)$, for all $x \in \text{dom}(\tilde{h})$. Since $\tilde{h}$ is injective, for each $x \in \text{dom}(\tilde{h})$, there exists a unique $u \in T$, such that $\tilde{h}'(x) = \tilde{h}(u)$. Since $\tilde{h}$ is injective, $\tilde{h}$ is also injective. We have $(\tilde{h}, h) \models \phi \tau$ and we are left with proving that $\phi_{\tau} \models C(u_i)$, and the implication holds.

Let $\phi \tau \models C(u_i)$, hence the result. “$\Leftarrow$” This is a consequence of Lemma 48, using the fact that $\tilde{h}$ is an injective extension of itself.

The following property of core formulae leads to a necessary and sufficient condition for their satisfiability (Lemma 52). The idea is that the particular identity of locations outside of the heap, assigned by the $\forall \cdot \phi$ quantifier, is not important when considering a model of a core formula.

For a set of locations $L \subseteq \mathbb{L}$, we define $\tilde{h} = L \leq \tilde{h}'$ and only if $\text{dom}(\tilde{h}) = \text{dom}(\tilde{h}')$, and for each term $t \in \text{dom}(\tilde{h})$, if $\{t(\tilde{h}), t'(\tilde{h})\} \cap L \neq \emptyset$ then $t(\tilde{h}) = t'(\tilde{h})$.

It is easy to check that $\equiv_L$ is an equivalence relation, for each set $L \subseteq \mathbb{L}$.

Let $\tilde{h}$ and $\tilde{h}'$ be two injective stores and $\tilde{h} \models \phi \tau$, then for every core formula $\phi$, we have $(\tilde{h}, h) \models \phi$ if and only if $(\tilde{h}', h) \models \phi$.

Proof: We assume that $(\tilde{h}, h) \models \phi \tau$ and show that $(\tilde{h}', h) \models \phi$; the proof in the other direction is identical since $\equiv_{\text{loc}(\tilde{h})}$ is symmetric. The proof is carried out by nested induction on $|\phi|$ and $|\phi|$. We assume, w.l.o.g., that $\text{dom}(\tilde{h}) = \text{dom}(\tilde{h}') = \text{fv}(\phi) \cup C$. This is without loss of generality since the truth value of $\phi$ in $(\tilde{h}, h)$ and $(\tilde{h}', h)$ depends only on the restriction of $\tilde{h}$ (resp. $\tilde{h}'$) to $\text{fv}(\phi) \cup C$.

For the base case assume that $|\phi| = 0$. By hypothesis, $\varphi = \exists_{\phi} \forall_{\phi} \phi$, $(\tilde{h}, h) \models \phi \tau$ and show that $(\tilde{h}', h) \models \phi$. The proof in the other direction is identical since $\equiv_{\text{loc}(\tilde{h})}$ is symmetric. The proof is carried out by nested induction on $|\phi|$ and $|\phi|$. We assume, w.l.o.g., that $\text{dom}(\tilde{h}) = \text{dom}(\tilde{h}') = \text{fv}(\phi) \cup C$. This is without loss of generality since the truth value of $\phi$ in $(\tilde{h}, h)$ and $(\tilde{h}', h)$ depends only on the restriction of $\tilde{h}$ (resp. $\tilde{h}'$) to $\text{fv}(\phi) \cup C$.

For the base case assume that $|\phi| = 0$. By hypothesis, $\varphi = \exists_{\phi} \forall_{\phi} \phi$, $(\tilde{h}, h) \models \phi \tau$ and show that $(\tilde{h}', h) \models \phi$. The proof in the other direction is identical since $\equiv_{\text{loc}(\tilde{h})}$ is symmetric. The proof is carried out by nested induction on $|\phi|$ and $|\phi|$. We assume, w.l.o.g., that $\text{dom}(\tilde{h}) = \text{dom}(\tilde{h}') = \text{fv}(\phi) \cup C$. This is without loss of generality since the truth value of $\phi$ in $(\tilde{h}, h)$ and $(\tilde{h}', h)$ depends only on the restriction of $\tilde{h}$ (resp. $\tilde{h}'$) to $\text{fv}(\phi) \cup C$.
\[ \varphi = \emptyset \mapsto (t_1, \ldots, t_n) : \] in this case \( b = (\hat{s}(t_0), \hat{s}(t_1), \ldots, \hat{s}(t_n)) \) and since \( \hat{s}(t_0), \hat{s}(t_1), \ldots, \hat{s}(t_n) \in \text{loc}(b) \)
\[ \text{and } \hat{s} \approx_{\text{loc}(b)} \hat{s}' \text{, we also have } b = (\hat{s}'(t_0), \hat{s}'(t_1), \ldots, \hat{s}'(t_n)) \), thus \( (\hat{s}', \hat{s}) \mapsto t_0 \mapsto (t_1, \ldots, t_n) \).

\[ \varphi = \emptyset \mapsto q_i(u) \mapsto p(t) : \] since \( ||b|| > 0 \), \( \varphi \) cannot be \( p(t) \mapsto p(t) \). Thus the first unfolding step is an instance of a rule obtained from \( l \). By Lemma 48, there exists an injective extension \( \hat{s} \) of \( s \) such that \( (\hat{s}, b) \models \varphi \psi \) where \( \varphi \varphi \models_{\text{SS}} \psi \), and because \( S \) is progressing, \( \psi \) is of the form \( t_0 \mapsto (t_1, \ldots, t_n) \ast \psi' \). Since the truth value of \( \psi \) in \( (\hat{s}, b) \) depends only on the restriction of \( \hat{s} \) to \( \text{fv}(\varphi) \cup C \), we assume, w.l.o.g., that \( \text{dom}(\hat{s}) \) is finite. The heap \( h \) can thus be decomposed into \( b_0 \cup b' \), where \( (\hat{s}, b_0) \models_{\text{SS}} t_0 \mapsto (t_1, \ldots, t_n) \) and \( (\hat{s}, b') \models_{\text{SS}} (\psi \ast) \). Consider the store \( s_1 \overset{=}{=} \{ (x_1, \hat{s}(x_1)) \mid x_1 \in \text{dom}(\hat{s}) \setminus \text{dom}(\hat{s}) \wedge \hat{s}(x_1) \in \text{loc}(b) \} \) and let \( s_1 \overset{=}{=} \hat{s} \ast s_1 \). Since \( \text{dom}(\hat{s}) \ast \text{dom}(\hat{s}) \ast \hat{s} \) by hypothesis, \( s_1 \) is well-defined. It is also injective because \( \hat{s}' \) and \( \hat{s} \) are both injective, and if \( s_1(x) = s_1(y), \) then \( x = y \).

Now let \( s_2 \) be an injection from \( \text{dom}(\hat{s}) \setminus \text{dom}(s_1) \) onto \( \mathbb{L} \setminus (|\text{rng}(\hat{s}) \cup \text{rng}(\hat{s}') \cup \text{loc}(b)) \).

Note that such an extension necessarily exists since \( \text{dom}(\hat{s}) \), \( \text{dom}(\hat{s}') \) and \( \text{loc}(b) \) are all finite whereas \( \mathbb{L} \) is infinite. Let \( \hat{s} \overset{=}{=} s_1 \cup s_2 \), and it is straightforward to verify that \( \hat{s} \) is injective and that \( (\hat{s}, b') \) is injective. By the inductive hypothesis we have \( (\hat{s}, b_0) \models_{\text{SS}} t_0 \mapsto (t_1, \ldots, t_n) \) and \( (\hat{s}, b') \overset{-n}{\models} \), and by Lemma 48 we deduce that \( (\hat{s}, b') \models_{\text{SS}} \hat{s} \mapsto q_i(u) \mapsto p(t) \).

\[ \varphi = \exists x . \psi : \] by Lemma 41, there exists an \( x \)-associate \( \hat{s} \) of \( s \), such that \( \hat{s}(x) \in \text{loc}(b) \setminus (\text{fv}(\varphi) \cup C) \) and \( (\hat{s}, b) \models_{\text{SS}} \psi \). We distinguish two cases.

\[ \text{If } \hat{s}(x) = \hat{s}(y) \text{ for some } y \in \text{dom}(\hat{s}) \text{ then } (\hat{s}, b) \models_{\text{SS}} \psi[y/x] \text{ and, by the induction hypothesis, we have } (\hat{s}', b) \models_{\text{SS}} \psi[y/x] . \text{ Since } \hat{s} \overset{=}{=} (\hat{s}'(x), \hat{s}'(y)) \in \text{loc}(b) \text{, we have } \hat{s}(x) = \hat{s}'(y) \text{. Furthermore, since } \hat{s}(x) \not\in (\text{fv}(\varphi) \cup C) \text{, necessarily } y \not\in (\text{fv}(\varphi) \cup C) \text{ and, because } \hat{s}' \text{ is injective, } \hat{s}'(x) \not\in \hat{s}'(\text{fv}(\varphi) \cup C) \text{. Hence } \hat{s}'(x) = \hat{s}'(y) \text{ if and only if } (\hat{s}', b) \models_{\text{SS}} \psi[y/x] \text{, we deduce that } (\hat{s}', b) \models_{\text{SS}} \exists x . \psi \text{.} \]

\[ \text{Otherwise we have } \hat{s}(x) \neq \hat{s}(y) \text{ for all } y \in \text{dom}(\hat{s}) \text{ and } \hat{s} \text{ is therefore injective. Let } \hat{s}' \overset{=}{=} \{ x \mapsto \hat{s}'(x) \} . \] Suppose that \( \hat{s}(x) = \hat{s}'(y) \), for some \( y \in \text{dom}(\hat{s}) \text{.} \) Since \( \hat{s} \overset{=}{=} (\hat{s}', s) \) we have \( \hat{s}(s) \ast \hat{s}(x) \ast (\hat{s}', b) \) and, since \( \hat{s}(y) \in \text{loc}(b) \text{, we obtain } \hat{s}(x) = \hat{s}'(x) = \hat{s}(x) \text{, in contradiction with the assumption of this case.} \) Thus \( \hat{s}' \) is injective and, using the fact that \( \hat{s} \overset{=}{=} \hat{s}' \ast \hat{s} \) we deduce that \( (\hat{s}', b) \models_{\text{SS}} \psi \) by the induction hypothesis. Since \( \hat{s}(x) = \hat{s}(x) \not\in \text{dom}(\hat{s}) \ast \hat{s}(s) \) we have \( \hat{s}(x) \not\in \hat{s}'(\text{fv}(\varphi) \cup C) \) and, moreover, \( \hat{s}(x) = \hat{s}(x) \in \text{loc}(b) \text{, thus } (\hat{s}', b) \models_{\text{SS}} \exists x . \psi \) by definition of \( \forall_{\lambda} \text{.} \)

The following lemma gives an alternative condition for the satisfiability of core formulæ. Intu- isitively, it is sufficient to instantiate the bounded universal quantifiers with arbitrary locations that are not in the image of the store, nor in the range of the heap.

\[ \text{Lemma 52. Given a core formulæ } \varphi = \exists b \forall_{\lambda} \forall_{\lambda} \psi , \text{ where } \psi \text{ is quantifier-free, and an injective structure } (s, b) , \text{ such that } \text{dom}(s) = \text{fv}(\varphi) \cup C, \text{ we have } (s, b) \models_{\text{SS}} \varphi \text{ if and only if } (s, b) \models_{\text{SS}} \psi , \text{ for some injective } (x \cup y)\text{-associate } \hat{s} \text{ of } s, \text{ such that } \hat{s}(x) \subseteq \text{loc}(b) \text{ and } \hat{s}(y) \cap \text{loc}(b) = \emptyset . \]

\[ \text{Proof: "⇒" Since } \mathbb{L} \text{ is infinite and } \text{dom}(s) \cup \text{loc}(b) \text{ is finite, there exists an injective } (x \cup y)\text{-associate } \hat{s} \text{ of } s, \text{ such that } \hat{s}(x) \subseteq \text{loc}(b) \text{, } \hat{s}(y) \cap \text{loc}(b) = \emptyset \text{ and } (s, b) \models_{\text{SS}} \psi \text{, by the semantics of the bounded quantifiers } \exists b \text{ and } \forall_{\lambda} \text{ (see Lemma 41).} \]

"⇐" Since \( \mathbb{L} \) is infinite and \( \text{dom}(s) \cup \text{loc}(b) \) is finite, there exists an injective \((x \cup y)\)-associate \( \hat{s} \) of \( s \), such that \( \hat{s}(x) \subseteq \text{loc}(b) \), \( \hat{s}(y) \cap \text{loc}(b) = \emptyset \) and \( (s, b) \models_{\text{SS}} \psi \), by the semantics of the bounded quantifiers \( \exists b \) and \( \forall_{\lambda} \) (see Lemma 41).
Proof of Lemma 25 (Section 5)

"⇒" Let \( \phi \vdash \psi_1, \ldots, \psi_n \) be a sequent and \( \varphi \in T(\phi) \) be a core formula. Since \( \phi \) is quantifier-free and \( \forall \varphi(\phi) = \emptyset \) (Definition 3), we deduce that \( \varphi \) is quantifier-free and \( \text{roots}(\varphi) \subseteq \text{trm}(\varphi) \subseteq C \), hence \( \varphi \in \text{Core}(P) \), by Definition 22. If there is no set of core formulae \( F \in 2^{\text{Core}(P)} \) such that \( \varphi, F \in T \), then there is nothing to prove. Otherwise, let \( F \in 2^{\text{Core}(P)} \) be a set of core formulae, such that \( (\varphi, F) \in T \). By Definition 24, there exists an injective normal \( \mathcal{S}_S \)-model \((\bar{s}, h)\) of \( \varphi \), such that \( F = \mathcal{C}_P(\bar{s}, h) \). Since \( P \) is valid, \( \varphi \models_S \bigvee_{i=1}^n \psi_i \), hence there exists \( i \in \llbracket 1 .. n \rrbracket \), such that \( (\bar{s}, h) \models_S \psi_i \).

Since \( \text{dom}(\bar{s}) = \mathcal{C} = \text{fv}(\psi_i) \cup C \), by Lemma 20, we obtain \( (\bar{s}, h) \models_S \bar{\zeta} \), for some \( \bar{\zeta} \in T(\psi_i) \). Since \( \text{fv}(\bar{\zeta}) \subseteq \text{fv}(\psi_i) \neq \emptyset \), we also have that \( (\bar{s}, h) \models_S \bar{\zeta} \). We show that \( \bar{\zeta} \in \text{Core}(P) \). First, all predicate atoms in \( \bar{\zeta} \) are of the form \( \text{emp} \to p(t) \), and if \( \bar{\zeta} \) contains two distinct occurrences of atoms \( \text{emp} \to p(t) \) and \( \text{emp} \to q(s) \) with roots \( p(t) = q(s) \) then \( \bar{\zeta} \) cannot be satisfiable, because the same location cannot be allocated in two disjoint parts of the heap. Second, since \( P \) is normalized, all existential variables must occur in a predicate or points-to atom. Thus all the conditions of Definition 19 are satisfied. Finally, since \( | \mathcal{V}_P^2 | = \text{size}(\mathcal{P}) \geq \text{size}(\psi_i) \), we may assume up to \( \alpha \)-renaming that all the bound variables in \( \bar{\zeta} \) are in \( \mathcal{V}_P^2 \), hence the same holds for \( \bar{\zeta} \). Since any predicate atom that occurs in a core formula in \( T(\psi_i) \) is of the form \( \text{emp} \to p(t) \), we have \( \text{roots}_{\text{emp}}(\psi_i \sigma) = \emptyset \). By Definition 23, we have \( \bar{\zeta} \in \text{Core}(P) \), thus \( F \cap T(\psi_i) \neq \emptyset \).

"⇐" Let \( \phi \vdash \psi_1, \ldots, \psi_n \) be a sequent. Let \( (\bar{s}, h) \) be an \( \mathcal{S} \)-model of \( \phi \). Since \( \forall \varphi(\phi) = \emptyset \) (Definition 3), we may assume, w.l.o.g., that \( \text{dom}(\bar{s}) = C \), and that \( \bar{s} \) is injective (by Assumption 1 all constants are mapped to pairwise distinct locations). It is sufficient to prove that \( (\bar{s}, h) \models_S \psi_i \), for some \( i \in \llbracket 1 .. n \rrbracket \), because in this case, we also have \( (\bar{s}, h) \models_S \psi_i \). By Lemma 17, it is sufficient to show that any injective normal \( \mathcal{S} \)-model of \( \phi \) is an \( \mathcal{S} \)-model of \( \psi_i \), for some \( i \in \llbracket 1 .. n \rrbracket \). By Definition 24, we have \( (\varphi, \mathcal{C}_P(\bar{s}, h)) \in T \), hence \( \mathcal{C}_P(\bar{s}, h) \cap T(\psi_i) \neq \emptyset \), for some \( i \in \llbracket 1 .. n \rrbracket \). Then there exists a core formula \( \bar{\zeta} \in T(\psi_i) \), such that \( (\bar{s}, h) \models_S \bar{\zeta} \), by Definition 23 and, since \( \text{dom}(\bar{s}) = C = \text{fv}(\psi_i) \cup C \), by Lemma 20, we obtain \( (\bar{s}, h) \models_S \psi_i \). Since the choice of \( (\bar{s}, h) \) is arbitrary, each injective normal \( \mathcal{S} \)-model of \( \phi \sigma \) is a model of \( \psi_i \sigma \), for some \( i \in \llbracket 1 .. n \rrbracket \).

Additional Material for the Construction of Profiles (Section 6)

Lemma 53. If \( S \) is progressing, then for all terms \( t_0, \ldots, t_n \in \mathcal{V}_P^2 \cup C \) and all sets of core formulae \( F \in 2^{\text{Core}(P)} \), we have \( (t_0 \mapsto (t_1, \ldots, t_n), F) \in T_F \) if and only if \( F = \mathcal{C}_P(\bar{s}, h) \), for some injective \( \mathcal{S} \)-model \((\bar{s}, h) \mapsto (t_1, \ldots, t_n), \text{dom}(\bar{s}) = \{t_0, \ldots, t_n\} \cup C \).

Proof: Let \( (\bar{s}, h) \) be an arbitrary injective model of \( t_0 \mapsto (t_1, \ldots, t_n) \) where \( \text{dom}(\bar{s}) = \{t_0, \ldots, t_k\} \cup C \) and \( h = ((\bar{s}(t_0), (\bar{s}(t_1), \ldots, \bar{s}(t_n))) \). We show \( F = \mathcal{C}_P(\bar{s}, h) \), below, where \( F \) is defined by (1):

"⇒" Let \( \phi \in F \) and consider the following cases:

- If \( \phi = t_0 \mapsto (t_1, \ldots, t_k) \) then \( (\bar{s}, h) \models t_0 \mapsto (t_1, \ldots, t_k) \) and \( \text{roots}_{\text{emp}}(\phi) = \emptyset \), thus \( \phi \in \mathcal{C}_P(\bar{s}, h) \) (see Definition 23).
- Otherwise, \( \phi = \mathcal{V}_P \cdot z \cdot (\bigcup_{i=1}^n q_i(u_i) \mapsto p(t)) \), where \( z = \left( \bigcup_{i=1}^n u_i \cup t \right) \setminus \{t_0, \ldots, t_k\} \cup C \) and \( \text{emp} \mapsto p(t) \mapsto_{\mathcal{C}_P(\bar{s}, h)} (t_1, \ldots, t_n) \). Note that by the progressivity condition, we have
$t_0 = \text{root}(p(t))$. By Definition 21, there exists a rule:

$$\text{emp} \to p(x) \iff \exists q \forall . \psi * \text{emp} \to q_i(y_i) \quad (\dagger)$$

such that $t_0 \to (t_1, \ldots, t_n) \in T(\psi \sigma)$ and $\sigma$ is an extension of $[t/x, u_i/y_i, \ldots, y_n/u_n]$ with pairs $(z, i)$, where $z \in v$ and $i \in u \cup \bigcup_{i=1}^{n} u_i$. By (II), the rule (\dagger) occurs because of the existence of a rule

$$p(x) \iff S \exists w . \psi * \text{emp} \to q_i(z_i) \quad (\dagger\dagger)$$

and a substitution $\tau : w \to x$, such that $\psi = \psi \tau$, $v = w \setminus \text{dom}(\tau)$ and $y_i = \tau(z_i)$, for all $i \in \llbracket 1 \ldots n \rrbracket$. Applying $\tau$ to (\dagger\dagger), by (II), we obtain the rule:

$$\text{emp} \to p(x) \iff \exists q \forall . \psi * \text{emp} \to q_i(y_i) \quad (\ddagger)$$

Let $\exists v$ be an injective $v$-associate of $h$. Such an associate necessarily exists, for instance if $\exists v$ maps $v$ into pairwise distinct locations, that are further distinct from $\text{rng}(\exists h)$; since $L$ is infinite and dom($h$) is assumed to be finite, such locations always exist. By $\alpha$-renaming if necessary, we can assume that $v \cap \llbracket t_0, \ldots, t_n \rrbracket = \emptyset$, thus $\exists v$ and $\exists h$ agree on $\llbracket t_0, \ldots, t_n \rrbracket$ and we obtain $\exists (h, b) = t_0 \to (t_1, \ldots, t_n)$. Since $t_0 \to (t_1, \ldots, t_n) \in T(\psi \sigma)$, by Lemma 47, we have $\exists (h, b) = \psi \sigma$. By Lemma 43, we have $\exists (h, b) = \exists (\text{emp} \to q_i(y_i))$ and, by rule (\ddagger) we obtain $\exists (h, b) = \exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I).

By Definition 23, namely that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Hence, by Lemma 43, we have $\exists (h, b) = \exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I).

By Definition 23, namely that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Hence, by Lemma 43, we have $\exists (h, b) = \exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Hence, by Lemma 43, we have $\exists (h, b) = \exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Thus, we obtain the proposition $\exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Thus, we obtain the proposition $\exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Thus, we obtain the proposition $\exists q_i(y_i) \to q(t)$. There remains to show that $\exists h \in \mathcal{W}_s(\exists h, \phi)$. Since there are no existentially quantified variables in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, since no existential quantifier variable in $\phi$, it suffices to prove that $\exists h \in \mathcal{W}_s(\exists h, \phi)$, the conclusion follows, by (I). Thus, we obtain the proposition $\exists q_i(y_i) \to q(t)$.
and each substitution \( \tau : w \rightarrow x \cup \bigcup_{i=1}^{n} y_i \), there exists a rule

\[
\ast^{n}_{i=1} q_i(y_i) \rightarrow p(x) \in \mathcal{E}_s \quad \forall v \cdot \psi \tau \ast^{n}_{i=1} y_j \rightarrow p_j(\tau(z_j)) \quad (\dagger)
\]

where \( \ast^{n}_{i=1} j = \ast^{n}_{i=1} q_i(y_i) \) and \( v = w \setminus \text{dom}(\tau) \). Assume w.l.o.g. that \( (\mathfrak{h}, b) \in \mathcal{E}_s \), \( \ast^{n}_{i=1} q_i(u_i) \rightarrow p(t) \) is a consequence of the above rule, i.e., that there exists a \( \mathcal{V} \)-associate \( \mathcal{S}' \) of \( \mathcal{S} \) such that \( (s', b) \in \mathcal{E}_s \)

\[
\psi \tau \sigma \ast^{n}_{i=1} y_j \rightarrow p_j(\sigma(\tau(z_j))),
\]

where \( \mathcal{S}' \in \mathcal{S} \setminus \{ t/x, u_1/y_1, \ldots, u_n/y_n \} \). Since \( \mathcal{S} \) is progressing, there is exactly one points-to atom in \( \psi \) and, because \( ||b|| = 1 \), it must be the case that \( (s', b) \models \psi \sigma \) and 

\[
(s', \emptyset) \models \mathcal{E}_s \gamma \sigma \rightarrow p_j(\sigma(\tau(z_j))), \quad \text{for each } j \in [1 \ldots m].
\]

To prove that \( \phi \models F \), it is sufficient to show the existence of a core unfolding \( \text{emp} \rightarrow p(t) \leadsto_{\mathcal{E}_s} (t_1, \ldots, t_k) \ast^{n}_{i=1} \text{emp} \rightarrow q_i(u_i) \). To this end, we first prove the two points of Definition 21:

(1) Since \( (s', \emptyset) \models \mathcal{E}_s \gamma \sigma \rightarrow p_j(\sigma(\tau(z_j))) \), for each \( j \in [1 \ldots m] \), by Lemma 43, we obtain \( \gamma \sigma = p_j(\mathcal{w}_j) \), for a tuple of variables \( \mathcal{w}_j \in \text{dom}(s') \), such that \( s'(\mathcal{w}_j) = s'\sigma(\tau(z_j)) \). Since, moreover 

\[
\ast^{n}_{i=1} y_j = \ast^{n}_{i=1} q_i(u_i),
\]

we deduce that \( n = m \) and, for each \( i \in [1 \ldots n] \), we have \( q_i = p_j \), for some \( j_i \in [1 \ldots m] \). Then, by applying (H) to the rule \((\dagger \dagger)\), using the substitution \( \tau \), we obtain the rule:

\[
\text{emp} \rightarrow p(x) \in \mathcal{E}_s \quad \forall v \cdot \psi \tau \ast^{n}_{i=1} \text{emp} \rightarrow q_i(\tau(z_j)) \quad (\dagger \dagger)
\]

(2) Let \( \mu \) be the extension of \( \sigma \) with the pairs \( (z, u) \) such that \( z \in v \) and one of the following holds:

- if \( s'(z) = \tilde{s}(i_1) \), for some \( i \in \emptyset \ldots \emptyset \), then \( u = i_1 \),
- if \( s'(z) = \tilde{s}(u_i) \), for some \( i \in [1 \ldots n] \) and \( \ell \in [1 \ldots \#q_i] \), then \( u = u_i \ell \),
- otherwise, \( u = \min_{v \in v} \{ v \in v | s'(v) = s'(z) \} \), where \( z \) is a total order on \( V \).

Note that, since \( \tilde{s} \) is injective, for each \( z \in v \) there exist at most one pair \( (z, u) \in \mu \) which is well-defined. Moreover, we have \( \mu(\tau(z_j)) = u_i \), because \( s'(\tau(\tau(z_j))) = s'(u_i) = \tilde{s}(u_i) \), for all \( i \in [1 \ldots n] \).

We now prove that

\[
t_0 \mapsto (t_1, \ldots, t_k) \ast^{n}_{i=1} \text{emp} \rightarrow q_i(u_i) \in T(\psi \mu \ast^{n}_{i=1} \text{emp} \rightarrow q_i(\mu(\tau(z_j))))
\]

or, equivalently, that \( t_0 \mapsto (t_1, \ldots, t_k) \in T(\psi \mu) \). By a case split on the form of the atom \( \alpha \) in \( \psi \tau \), using the fact that \( (s', b) \models \psi \tau \):}

\[
\begin{align*}
\alpha &= u_1 = u_2: \quad \text{we have } s'(\sigma(u_1)) = s'(\sigma(u_2)), \quad \text{hence } \mu(u_1) = \mu(u_2), \quad \text{by definition of } \mu \quad \text{and} \quad \mathcal{T}(\alpha) = \{ \text{emp} \}; \\
\alpha &= u_1 \neq u_2: \quad \text{we have } s'(\sigma(u_1)) \neq s'(\sigma(u_2)), \quad \text{hence } \mu(u_1) \neq \mu(u_2), \quad \text{by definition of } \mu \quad \text{and} \quad \mathcal{T}(\alpha) = \{ \text{emp} \}; \\
\alpha &= u_0 \mapsto (u_1, \ldots, u_n): \quad \text{since } \mathcal{S} \text{ is progressing, } \alpha \text{ is the only points-to atom in } \psi \text{ and } \text{dom}(b) = \{ (\tilde{s}(t_0), b, (\tilde{s}(t_0), \ldots, \tilde{s}(t_k)) = (s'(\sigma(u_1)), \ldots, s'(\sigma(u_n))) \}. \quad \text{Then we obtain } (s(t_0)) = s'(\sigma(u_i)), \quad \text{hence } \mu(u_i) = t_i, \quad \text{for all } i \in [0 \ldots n] \quad \text{and} \quad \mathcal{T}(\alpha) = (t_0 \mapsto (t_1, \ldots, t_k)); \\
\end{align*}
\]

We obtain the core unfolding \( \text{emp} \rightarrow p(t) \ast^{n}_{i=1} \text{emp} \rightarrow q_i(u_i) \) and we are left with proving that \( t_0 \notin \{ \text{root}(q_i(u_i)) | i \in [1 \ldots n] \} \). By the definition of \( \mu \), there exists a points-to atom \( u_0 \mapsto (u_1, \ldots, u_n) \) in \( \psi \tau \). Since \( \mathcal{S} \) is progressing, it must be the case that \( t_0 = \text{root}(p(x)) \), hence \( t_0 = \text{root}(p(t)) \), by the definition of \( \mu \). Since \( \phi \) is a core formula, by Definition 19, we obtain \( \text{root}(p(t)) \notin \{ \text{root}(q_i(u_i)) | i \in [1 \ldots n] \} \) and we conclude that \( \phi \models \forall \mathfrak{a} \exists \mathcal{X} \cdot \mathcal{S} \models \phi \).

\[\]
unfolding and \( \tilde{S}_i \) be the \( x_i \)-associate of \( \tilde{s} \) that satisfy points (1) and (2) of Definition 34. Assume that 
\( \ell \in \text{dom}(h_1) \) (the case \( \ell \notin \text{dom}(h_2) \) is symmetric). Because \( (\tilde{s}_1, h_1) \models \varphi_1 \), there exists a points-to atom 
\( t_0 \mapsto (t_1, \ldots, t_n) \in \psi_1 \), such that \( \tilde{S}_1(t_0) = \ell \). Since \( S \) is normalized, by Definition 9, the set alloc\(_{S}(\varphi_1) \)

is well-defined and we distinguish two cases.

- If \( t_0 \in \text{alloc}_{S}(\varphi_1) \), then \( \ell \in \hat{s}(\text{alloc}_{S}(\varphi_1)) \), because \( \tilde{s}_1 \) and \( \hat{s} \) agree over \( \text{alloc}_{S}(\varphi_1) \).

- Otherwise, we must have \( t_0 \notin x_1 \). Since \( \ell \in \text{Fr}(h_1, h_2) \), we have \( \ell \in \text{loc}(h_2) \), thus there exists 
a points-to atom \( t_0 \mapsto (u_1, \ldots, u_k) \) in \( \varphi_2 \) such that \( \ell = \tilde{S}_2(u_i) \), for some \( i \in \{1, \ldots, k\} \). Note that 
\( \ell = \tilde{S}_2(u_0) \) is impossible, because \( \ell \notin \text{dom}(h_2) \). Suppose, for a contradiction, that \( \ell \in \tilde{s}(C) \). Then 
u_1 \in \text{fv}(\psi_2) \) must be the case, which contradicts the condition \( \tilde{s}_1(x_1) \cap \tilde{S}_2(\text{fv}(\psi_2)) \subseteq \tilde{s}(C) \), required at point (2) of Definition 34. Hence \( \ell \in \tilde{s}(C) \) must be the case. Since \( \ell = \tilde{S}_1(t_0) \) either \( t_0 \in C \) or 
\( t_0 \) is an existentially allocated variable. The second case cannot occur, because of the Condition 
(2c) of Definition 8. Then we have \( t_0 \in C \) and, moreover, we have \( t_0 \in \text{alloc}_{S}(\varphi_1) \), by Definition 9, 
thus \( t_0 \in \text{alloc}_{S}(\varphi_1) \cap C \).

In each case we obtain \( \ell \in \tilde{s}_1(\text{alloc}_{S}(\varphi_1)) \cup \tilde{S}_2(\text{alloc}_{S}(\varphi_2)) \subseteq \hat{s}(\text{alloc}_{S}(\varphi_1 \ast \varphi_2)) \), because \( \tilde{s}_1 \) agrees with 
\( \hat{s} \) over \( \text{alloc}_{S}(\varphi_1) \), for \( i = 1, 2 \). We obtain:

\[
\ell \in \hat{s}(\text{alloc}_{S}(\varphi_1 \ast \varphi_2)) \cap \hat{s}(\text{fv}(\varphi_1) \cap \text{fv}(\varphi_2) \cup C)
\]

Because \( \hat{s} \) is injective.

The second inclusion follows trivially from the fact that \( \hat{s}(\text{alloc}_{S}(\varphi_1)) \subseteq \text{dom}(h_i) \), for \( i = 1, 2 \), which is 
an easy consequence of Definition 9.

Lemma 55. Given an injective structure \( (\hat{s}, h) \), a variable \( x \notin \text{dom}(\hat{s}) \) and a location \( \ell \notin \text{loc}(h) \cup \text{rng}(\hat{s}) \), we have \( C_{P}(\hat{s}[x \leftarrow \ell], h) = \text{add}(x, C_{P}(\hat{s}, h)) \).

Proof: “\( \subseteq \)” Let \( \varphi = \exists_s x \forall_a y \cdot \phi \in C_{P}(\hat{s}[x \leftarrow \ell], h, s) \). By Definition 23, there exists a witness \( \tilde{S} \in W_s(\hat{s}[x \leftarrow \ell], h, \psi) \), such that \( \tilde{S}(\text{roots}_{h}^{s}(\varphi)) \cap \text{dom}(h) = \emptyset \). Let \( \hat{\tilde{S}} \) be the store identical to \( \tilde{S} \), except that \( x \notin \text{dom}(\hat{\tilde{S}}) \) and \( \hat{\tilde{S}}(\hat{x}) = \hat{\tilde{S}}(x) \), for some variable \( \hat{x} \notin V_{P} \). Since \( \ell \notin \text{loc}(h) \), we have \( \hat{\tilde{S}} \in W_{s}(\hat{s}, h, \exists_s x \forall_a y \forall_{-} h \cdot \phi[\hat{x}/x]) \), because \( \hat{x} \notin V_{P} \), we have 
\( \exists_s x \forall_a y \forall_{-} h \cdot \phi[\hat{x}/x] \in \text{Core}(P) \), hence \( \exists_s x \forall_a y \forall_{-} h \cdot \phi[\hat{x}/x] \in C_{P}(\hat{s}, h) \), from which we deduce that 
\( \phi \in \text{add}(x, C_{P}(\hat{s}, h)) \).

“\( \supseteq \)” Let \( \varphi = \exists_s x \forall_a y \cdot \phi \in \text{add}(x, C_{P}(\hat{s}, h)) \), where \( \phi \) is quantifier-free, and let \( \psi = \exists_s x \forall_a y \forall_{-} h \cdot \phi[\hat{x}/x] \). By (3), we have \( \psi \in C_{P}(\hat{s}, h) \). By Definition 23, there exists a witness \( \hat{\tilde{S}} \in W_{s}(\hat{s}, h, \psi) \), such that 
\( \tilde{S}(\text{roots}_{h}^{s}(\psi)) \cap \text{dom}(h) = \emptyset \). W.l.o.g., by Lemma 51, we can assume that \( \hat{\tilde{S}} \) is such that \( \ell = \hat{\tilde{S}}(y) \), for all \( y \in \text{dom}(\hat{\tilde{S}}) \) such that \( \hat{\tilde{S}}(y) \notin \text{loc}(h) \). With this assumption, \( \hat{\tilde{S}}[x \leftarrow \ell] \) is injective. We prove that 
\( \hat{\tilde{S}}[x \leftarrow \ell] \in W_{s}(\hat{s}[x \leftarrow \ell], h, \psi) \).

Let \( \hat{\tilde{S}} \) be the store identical to \( \hat{\tilde{S}}[x \leftarrow \ell] \) except that the images of \( x \) and \( \hat{x} \) are switched. Since 
\( (\hat{s}, h) \models_{\tilde{S}_{2}} \phi[\hat{x}/x] \) we have \( (\hat{s}', h) \models_{\tilde{S}_{2}} \phi[\hat{x}/x] \). Since \( \hat{\tilde{S}}(x), \ell \notin \text{loc}(h) \) (as \( \tilde{S}(x) = \hat{\tilde{S}}(x) \)), by definition, and 
\( \tilde{S}(\hat{x}) \notin \text{loc}(h) \), by Condition (3) of Definition 23, we have \( \hat{\tilde{S}} = \text{loc}(h) \hat{\tilde{S}}[x \leftarrow \ell] \) thus \( \hat{\tilde{S}}[x \leftarrow \ell] \in W_{s}(\hat{s}[x \leftarrow \ell], h, \psi) \), 
by Lemma 51.

Since \( x \notin \text{dom}(\tilde{S}) \), we have \( \hat{\tilde{S}}[x \leftarrow \ell](x) = \hat{\tilde{S}}(x) \subseteq \text{loc}(h) \).

Since \( \ell \notin \text{loc}(h) \), we have \( \hat{\tilde{S}}[x \leftarrow \ell](y) \cap \text{loc}(h) = \emptyset \).

Since \( \text{roots}_{h}(\exists_s x \forall_a y \forall_{-} h \cdot \phi[\hat{x}/x]) = \text{roots}_{h}(\varphi) \), we have \( \tilde{S}(\text{roots}_{h}(\varphi)) \cap \text{dom}(h) = \emptyset \), thus \( \hat{\tilde{S}}[x \leftarrow \ell] \in W_{s}(\hat{s}[x \leftarrow \ell], h, \psi) \), which implies \( \varphi \in C_{P}(\hat{s}[x \leftarrow \ell], h) \).

Lemma 56. Given an injective structure \( (\hat{s}, h) \) and a variable \( x \in \text{dom}(\hat{s}) \cap V_{P} \), such that \( \hat{s}(x) \in \text{loc}(h) \), we have \( C_{P}(\hat{s}', h) = \text{rem}(x, C_{P}(\hat{s}, h)) \), where \( \hat{s}' \) is the restriction of \( \hat{s} \) to \( \text{dom}(\hat{s}) \setminus \{x\} \).

Proof: First note that because \( \hat{s} \) is injective, \( \hat{s}' \) is necessarily injective, thus \( C_{P}(\hat{s}', h) \) is well defined.

We prove both inclusions.
“⇐” Let $\varphi = \exists x \forall y . \phi \in C_P(\delta', b)$ be a core formula, where $\phi$ is quantifier-free. By Definition 23, there exists a witness $\delta' \in W_S(\delta', b, \varphi)$, such that $\delta'(\text{roots}_{\text{Sh}}(\varphi)) \cap \text{dom}(b) = \emptyset$. Since $x \notin \text{dom}(\delta')$ and $(\delta', b) \models \varphi$, we have $x \notin \text{fv}(\varphi)$. By $\alpha$-renaming if necessary, we can assume w.l.o.g. that $x \notin x \cup y$ (†). This is possible since $x \in V_P$, hence if $x \notin x \cup y$ then by definition of $\text{Core}(P)$ it cannot occur in roots($\varphi$); it can therefore be renamed by a variable not occurring in $V_P'$. We distinguish the following cases.

If $\delta(x) \neq \delta'(x')$ for all $x' \in x$, then $\delta'[x \leftarrow \delta(x)]$ is an injective associate of $\delta'$: indeed, by hypothesis, $\delta(x) \in \text{loc}(b)$ and $\delta'(y) \cap \text{loc}(b) = \emptyset$, thus $\delta(x) \notin \delta'(y)$. Since $\phi$ is quantifier-free and $\delta'$ agrees with $\delta'[x \leftarrow \delta(x)]$ on $\text{fv}(\phi)$, we obtain $\delta'(x \leftarrow \delta(x))$ is of the form $\delta(x) \in W_S(\delta', b, \varphi)$, which suffices to show $\varphi \in C_P(\delta, b)$, by the definition of the latter set:

- $\delta'[x \leftarrow \delta(x)](x) \subseteq \delta'(x) \cup \{\delta(x)\} \subseteq \text{loc}(b)$, because $\delta' \in W_S(\delta', b, \varphi)$ and $\delta(x) \subseteq \text{loc}(b)$ by hypothesis.
- $\delta'[x \leftarrow \delta(x)](\text{roots}_{\text{Sh}}(\varphi)) = \delta'(\text{roots}_{\text{Sh}}(\varphi))$ because $\delta \in \text{fv}(\phi)$, and $\delta'(\text{roots}_{\text{Sh}}(\varphi)) \cap \text{dom}(b) = \emptyset$ because $\delta' \in W_S(\delta', b, \varphi)$.

Consequently we obtain $\varphi \in C_P(\delta, b)$, and since $x \notin \text{fv}(\varphi)$, we have $C_P(\delta, b) \subseteq \text{rem}(x, C_P(\delta, b))$, hence the result.

Otherwise, $\delta(x) = \delta'(x')$ for some $x' \in x$, hence $\varphi$ is of the form $\exists x' \exists x' \forall y . \phi$, where $x' \equiv x \setminus \{x'\}$. Clearly, the variable $x'$ must be unique, otherwise $\delta'$ would not be injective. Let $\delta$ be the injective store obtained from $\delta'[x \leftarrow \delta(x)]$ by removing the pair $(x', \delta(x))$ from it. We prove that $\delta \in W_S(\delta, b, \exists x' \exists x' \forall y . \phi[x/x'])$:

- $(\delta, b) \models \varphi[x/x']$, because $\delta$ agrees with $\delta'[x \leftarrow \delta(x)]$ on $\text{fv}(\phi[x/x'])$.
- $\delta(x') = \delta'(x') \subseteq \text{loc}(b)$, because $\delta' \in W_S(\delta', b, \varphi)$.
- $(\delta, y) = \delta'(y)$, because $x \notin y$ (†) and $\delta'(y) \cap \text{loc}(b) = \emptyset$, because $\delta' \in W_S(\delta', b, \varphi)$.

Further, we have $\delta[x \leftarrow \delta(x)](\text{roots}_{\text{Sh}}(\varphi)) = \delta'(\text{roots}_{\text{Sh}}(\varphi))$ because $x \notin \text{fv}(\phi)$, hence $x \notin \text{roots}_{\text{Sh}}(\varphi)$. Thus $\delta'(\text{roots}_{\text{Sh}}(\varphi)) \subseteq \delta'(\text{roots}_{\text{Sh}}(\varphi))$ and, since $\delta'(\text{roots}_{\text{Sh}}(\varphi)) \cap \text{dom}(b) = \emptyset$ by Definition 23, we deduce that $\delta'(\text{roots}_{\text{Sh}}(\varphi)) \cap \text{dom}(b) = \emptyset$. Still by Definition 23, we obtain that $\exists x' \exists x' \forall y . \phi[x/x'] \in C_P(\delta, b)$ and thus $\exists x' \exists x' \forall y . \phi \in \text{rem}(x, C_P(\delta))(s)$ (with $\Delta \equiv x'$).

“⇒” Let $\varphi = \exists x \forall y . \phi \in \text{rem}(x, C_P(\delta, b))$, for some quantifier-free formula $\phi$. We distinguish the following cases.

If $\varphi \in C_P(\delta, b)$ and $x \notin \text{fv}(\varphi)$, then for any injective structure $(\delta, b)$ meeting the conditions of Definition 23, the structure $(\delta', b)$ is injective and trivially meets the conditions of Definition 23, hence $\varphi \in C_P(\delta', b)$.

Otherwise, $\varphi = \exists x \exists x' \forall y . \phi[\delta/x], \phi[\delta'/x']$, $x \in \text{fv}(\exists x \exists x' \forall y . \phi)$ and $\exists x' \forall y . \phi \in C_P(\delta, b)$, where $x' \equiv x \setminus \{x\}$. Let $\delta$ be an injective $(x' \cup y)$-associate of $\delta$ meeting the conditions from Definition 23. It is easy to check that $(\delta \setminus \{(x, \delta(x))\}) \cup \{(x, \delta(x))\}) \in \text{W}^S(\delta, b, \varphi)$, thus $\varphi \in C_P(\delta', b)$.

**K. Proof of Lemma 27 (Section 6)**

By induction on the structure of $\mathcal{T}_P$, defined as the least set satisfying the constraints (1), (2), (3) and (4), we prove that $(\delta, b)$ is an injective normal $\xi_S$-model of $\phi$ if and only if $(\delta, C_P(\delta, b)) \in \mathcal{T}_P$. Based on the structure of the core formula $\phi \in \mathcal{T}(\varphi)$, for some symbolic heap $\varphi \in \text{SH}^4$, we distinguish the following cases:

- $\phi = t_0 \mapsto (t_1, \ldots, t_k)$: because $S$ is progressing, by Lemma 53, we obtain that $(\delta, F) \in \mathcal{T}_P$ if and only if $F \subseteq C_P(\delta, b)$, for some injective $S$-model $(\delta, b)$, such that $\text{dom}(\delta) = \{t_0, \ldots, t_k\} \cup C$. Since any injective $S$-model $(\delta, b)$, $(\delta, b)$ is also normal, we conclude this case.

- $\phi = \text{emp} \mapsto p(t)$: “⇒” Since $\mathcal{T}_P$ is the least relation satisfying (2), $(\text{emp} \mapsto p(t), F) \in \mathcal{T}_P$ if and
only if $(∃y . ψ, F) ∈ \mathcal{FP}$, for some core unfolding $emp \hookrightarrow p(t) ∨ ∃y. ψ$, where $y = fv(φ) \setminus t$. By
the induction hypothesis, there exists an injective normal $C_S$-model $(h, b)$ of $∃y . ψ$ such that
$F = C_P(s, b)$ and dom$(s) = fv(∃y . ψ) ⊆ C$. Since $P$ is normalized, by Condition 1b in Definition
8 we have $fv(∃y . ψ) = fv(φ)$. By Lemma 48, $(h, b)$ is an injective $C_S$-model of $emp \hookrightarrow p(t)$.
Because $φ$ is quantifier-free, $(h, b)$ is also an injective normal $C_S$-model of $φ$. \\
Let $(h, b)$ be an injective normal $C_S$-model of $emp \hookrightarrow p(t)$. By Lemma 48, there exists a core unfolding
$emp \hookrightarrow p(t) ∨ ∃y. ψ$ and an injective extension $h$ of $b$, such that $(h, b)$ is an injective $C_S$-model
of $ψ$. Let $y \equiv fv(φ) \setminus t$. Then every variable $x \in y$ occurs in a points-to or a predicate atom, by
Definition 21. Since $S$ is normalized, we obtain that $\overline{x}(x) ∈ loc(h)$, by point (2a) of Definition 8,
and therefore $(h, b)$ is an injective $C_S$-model of $∃y . ψ$. Since $ψ$ is satisfiable, it cannot contain two
atoms with the same root. We have $fv(φ) = fv(φ) \subseteq V^P_P$. Furthermore, since $||V^P_P|| = width(P)$ and
size$(φ) ≤ width(P)$, we can assume w.l.o.g. that $y ⊆ V^P_P$, hence $∃y . ψ$ is a core formula. By
the induction hypothesis, we obtain that $(∃y . ψ, C_P(s, b)) ∈ \mathcal{FP}$, thus $(emp \hookrightarrow p(t), C_P(s, b)) ∈ \mathcal{FP}$
follows, by (2).

Let $φ = φ_1 ∨ φ_2$; ‘⇒’ Since $\mathcal{FP}$ is the least set satisfying (3), $(φ_1 + φ_2, F) ∈ \mathcal{FP}$ if and only if $(φ_1, F_1) ∈ \mathcal{FP}$
and $F = add(X_1, F_1) ⊕ add(X_2, F_2)$, where $X_i = fv(φ_i) \setminus fv(φ_{i-1})$, for $i = 1, 2$, allocate$(φ_1) \cap$
allocate$(φ_2) = \emptyset$ and $D = alloc(S(φ_1) ∪ φ_1) \cap alloc(S(φ_2) ∪ φ_2) \cap alloc(φ_1) \cup alloc(φ_2)$. Since $fv(φ_i) \subseteq V^P_P$, by
the inductive hypothesis, there exist injective normal $C_S$-models $(h, b_i)$ of $φ_i$, such that $F_i = C_P(h_i, b_i)$,
for $i = 1, 2$. By renaming locations if necessary, we assume w.l.o.g. that $b_1$ and $b_2$ agree over
trm$(φ_1) \cap trm(φ_2)$ and that $b_i fv(φ_{i-1}) \subseteq bv(φ_i) \cup bv(φ_{i-1}) \cap loc(b_{i-1}) = \emptyset$, for $i = 1, 2$
(1>). This is feasible since the truth value of formulæ does not depend on the name of the locations.
Let $s_i = b_1 ∪ b_2$. It is easy to check that $(s_1, h_1), (s_2, h_2))$ is an injective normal $C_S$-companion
for $(φ_1, φ_2)$, by Definition 34. Moreover, by Lemma 55, we have $C_P(s, h) = add(X_1, F_1)$, for
$i = 1, 2$. Next, we prove that $b_1$ and $b_2$ are disjoint heaps. Suppose, for a contradiction, that
$dom(b_1) \cap dom(b_2) ≠ \emptyset$. By assumption (1>), there exists a variable $v ∈ fv(φ_1) \cap fv(φ_2)$, such that
$\overline{x}(v) ∈ dom(b_1) \cap dom(b_2)$. Since $P$ is normalized, by Conditions (2b) and (2c) in Definition
8, the only variables that can be allocated by a model of a core formulæ $φ_i$ are allocate$(φ_i)$, we
must have $x ∈ allocate(φ_1) \cap allocate(φ_2)$, which contradicts with the condition that allocate$(φ_1) \cap$
allocate$(φ_2) = \emptyset$. We conclude that $b_1$ and $b_2$ are disjoint and let $h = b_1 \cup b_2$. By Lemmas 36 and
54, we respectively have $Fr(b_1, b_2) ⊆ ℹ(fv(φ_1) \cap fv(φ_2) \cup C) \subseteq ℹ(V^P_P \cup C)$ and $Fr(b_1, b_2) \cap dom(h) ⊆ ℹ(D) \subseteq dom(h)$. Thus $(h, b)$ is an injective normal $C_S$-model of $φ_1 + φ_2$ and, by Definition 26, we
have $C_P(s, h) = C_P(s, b_1) ⊕ add(X_2, F_2)$.

Let $(h, b)$ be an injective normal $C_S$-model of $φ_1 + φ_2$. Note that since $φ_1 + φ_2$ is satisfiable
we must have $allocate(φ_1) \cap allocate(φ_2) = \emptyset$. By Lemma 35, there exists an injective $C_S$-normal
companion $((s_1, b_1), (s_2, b_2))$ for $(φ_1, φ_2)$, such that $h = b_1 \cup b_2$. Since $(s_1, b_1)$ is an injective normal
$C_S$-model of $φ_1$, we have $⟨φ_1, C_P(s_1, b_1)⟩ ∈ \mathcal{FP}$, by the inductive hypothesis, for $i = 1, 2$. We prove
that $s_1 lobc(b_1) = 0$, where $X_1 \equiv ℹ(fv(φ_1)) \cap ℹ(φ_{i-1})$, for $i = 1, 2$. Let $i = 1$, the case $i = 2$ being
symmetric, and suppose, for a contradiction, that $\overline{x}(v) ∈ loc(b_1)$, for some $v ∈ X_1$. Because $S$
is normalized, by point (2a) of Definition 8, we have $\overline{x}(v) ∈ loc(b_2)$, thus $\overline{x}(v) ∈ Fr(b_1, b_2)$. By Lemma
36, $\overline{x}(v) ⊆ ℹ(fv(φ_1) \cap fv(φ_2) \cup C)$ and, since $\overline{x}$ is injective, we deduce that $x ∈ ℹ(fv(φ_1) \cap ℹ(fv(φ_2) \cup C)$,
which contradicts the hypothesis that $x ∈ X_1$. Hence $s_1 lobc(b_1) = 0$ and, by Lemma 55, we
obtain $C_P(s, b_1) = add(X_1, C_P(s_1, b_1))$, for $i = 1, 2$. Moreover, by Lemmas 36 and 54, we respectively
have $Fr(b_1, b_2) ⊆ ℹ(V^P_P \cup C)$ and $Fr(b_1, b_2) \cap dom(h) ⊆ ℹ(D) \subseteq dom(h)$. By Definition 26, we have
$C_P(s, h) = C_P(s, b_1) ⊕ add(X_2, C_P(s_2, b_2))$, thus $φ_1 + φ_2, C_P(s, h)) ∈ \mathcal{FP}$, by (3).

$∃y x . φ_1$: By $α$-renaming if necessary, we assume that $x' ∈ V^P_P$. Note that this is possible because
$||V^P_P|| ≥ size(φ_1)$. Furthermore, since we also have $||V^P_P|| ≥ size(φ_1)$, we may assume that there
exists a variable $x ∈ ℹ(V^P_P \setminus ℹ(φ_1))$. It is clear that $φ_1 = φ_1[x / x']$ is a core formulæ. “⇒” Since $\mathcal{FP}$
is the least relation satisfying (4), we have $∃y x . φ_1, F) ∈ \mathcal{FP}$ only if there exists a set of core

© Mnacho Echenim, Radu Iosif and Nicolas Peltier; licensed under Creative Commons License CC-BY

LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
formule \( F_1 \subseteq \text{Core}(\mathcal{P}) \), such that \( F = \text{rem}(x, F_1) \) and \((\phi_1, F_1) \in \mathcal{T}_P\). By the inductive hypothesis, there exists an injective normal \( \mathcal{C}_{s}(\phi_1, b) \) of \( \phi_1 \) such that \( F_1 = \mathcal{C}_{s}(\phi_1, b) \). By Lemma 56, we obtain \( F = \mathcal{C}_{s}(\phi_1, b)\), where \( b \) is the restriction of \( \phi_1 \) to \( \text{dom}(\phi_1) \setminus \{x\} \). Since \( \mathcal{C} \) is normalized and the only occurrences of predicate atoms in \( \phi_1 \) are of the form \( \text{emp} \rightarrow q(t) \), we have \( \phi_1(x) \in \text{loc}(b) \). Thus we conclude by noticing that \((\phi_1, b)\) is an injective normal \( \mathcal{C}_{s}(\phi_1, b) \) of \( \phi_1 \). Since \( s \subseteq I \) is an injective extension of \( \phi_1 \) and \( \ell \in \text{loc}(b) \), by Lemma 56, \( \mathcal{C}_{s}(\phi_1, b) = \text{rem}(x, \mathcal{C}_{s}(\phi_1, b)) \) and \((\mathcal{C}_{s}(\phi_1, b), F) \in \mathcal{T}_P \), follows by the inductive hypothesis.

We prove below that \( \tau \) is a logical consequence relation:

\[ \text{Lemma 57.} \quad \text{If } \phi \vdash \psi \text{ then } \phi \models_{\mathcal{C}_S} \psi. \]  

\[ \text{Proof:} \quad \text{The proof is by induction on the length } n \geq 0 \text{ of the derivation sequence from } \phi \text{ to } \psi. \text{ If } n = 0 \text{ then } \phi = \psi \text{ and there is nothing to prove. Assume } n = 1, \text{ the case } n = 1 \text{ follows immediately by the inductive hypothesis. We assume that } \phi = [\alpha \rightarrow p(t)] \ast [\beta \ast p(t)] \rightarrow q(u) \text{ and } \psi = (\alpha \ast \beta) \rightarrow q(u), \text{ for some predicate atoms } p(t) \text{ and } q(u) \text{ and some possibly empty conjunctions of predicate atoms } \alpha \text{ and } \beta. \text{ Then there exist two disjoint heaps } h_1 \text{ and } h_2, \text{ such that } b = h_1 \uplus h_2, (s, h_1) \models_{\mathcal{C}_S} \alpha \rightarrow p(t) \text{ and } (s, h_2) \models_{\mathcal{C}_S} \beta \ast p(t) \rightarrow q(u). \text{ We prove that } (s, h) \models \psi \text{ by induction on } |b|. \text{ If } |b| = 0 \text{ then } \beta = \text{emp} \text{ and, by Lemma 43, we obtain } p = q \text{ and } s(t) = s(u). \text{ Thus } b = h_1 \text{ and } (s, h) \models (\alpha \ast \beta) \rightarrow q(u) \text{ follows trivially. If } |b| > 0, \text{ then there exists a rule } \]  

\[ (\delta \ast p(x)) \rightarrow q(y) \models_{\mathcal{C}_S} \rho \]  

and a substitution \( \tau \) such that \( [\delta \ast p(x)] \rightarrow q(y)] \tau = (\beta \ast p(t)) \rightarrow q(u) \) and \( (s', h') \models \rho \tau \), where \( s' \) is an associate of \( s \). Since \( |b| > 0 \), by definition of \( \mathcal{C}_{s} \), rule (13) must be an instance of (ll). Thus \( \rho \) is of the form \( \mathcal{C}_S \cdot \mathcal{V} \cdot \psi' \ast \ast \prod_{j=1}^{m}(\gamma_j \tau \rightarrow p_j(\sigma(w_j))) \) for some substitution \( \sigma \), where \( \gamma_1, \ldots, \gamma_m \) are separating conjunctions of predicate atoms such that \( \delta \ast p(x) = \ast_{j=1}^{m} \gamma_j \). Still because (13) is an instance of (ll), there exists a rule

\[ q(y) \models_{\mathcal{C}_S} \exists z. \psi' \ast \ast \prod_{j=1}^{m} p_j(w_j) \]  

and we have \( \mathcal{V} = z \setminus \text{dom}(\sigma) \). Since \( (s, h_2) \models_{\mathcal{C}_S} \exists z. \psi' \ast \ast \prod_{j=1}^{m}(\gamma_j \tau \rightarrow p_j(\sigma(w_j))) \), there exists a \( \mathcal{V} \)-associate \( \bar{s} \) of \( s \) such that \( (\bar{s}, h_2) \models_{\mathcal{C}_S} \psi' \ast \ast \prod_{j=1}^{m}(\gamma_j \tau \rightarrow p_j(\sigma(w_j))) \). Hence, there exist two disjoint heaps \( b'_1 \) and \( b'_2 \) such that \( b_2 = b'_1 \uplus b'_2 \), \( (\bar{s}, h'_2) \models \psi' \ast \ast \prod_{j=1}^{m}(\gamma_j \tau \rightarrow p_j(\sigma(w_j))) \). We deduce that

\[ (\bar{s}, h'_1 \uplus b'_2) \models_{\mathcal{C}_S} [\alpha \rightarrow p(t)] \ast [\ast_{j=1}^{m} \gamma_j \tau \rightarrow p_j(\sigma(w_j))]. \]  

Since \( \delta \ast p(x) = \ast_{j=1}^{m} \gamma_j \), we can assume w.l.o.g. that \( \gamma_1 \) is of the form \( p(t) \ast \ast \gamma' \), so that \( \gamma_1 \tau = p(t) \ast \ast \gamma' \tau \) and

\[ (\bar{s}, b_3) \models_{\mathcal{C}_S} [\alpha \rightarrow p(t)] \ast [\ast_{j=1}^{m} \gamma_j \tau \rightarrow p_j(\sigma(w_j))]. \]  

There therefore exist two disjoint heaps \( b_3 \) and \( b_4 \) such that \( b_1 \uplus b'_2 = b_3 \uplus b_4 \) and the following hold: \( (\bar{s}, b_3) \models_{\mathcal{C}_S} [\alpha \rightarrow p(t)] \ast [\ast_{j=1}^{m} \gamma_j \tau \rightarrow p_j(\sigma(w_j))]. \) Because \( \mathcal{S} \) is assumed to be progressing, \( \psi' \) contains exactly one points-to atom, thus \( |b'_2| = 1 \) and \( |b'||b_3| \leq |b_3| \leq |b_3| + |b_2| = |b| \). By the inductive hypothesis, we deduce that \( (\bar{s}, b_3) \models_{\mathcal{C}_S} \alpha \ast \ast \gamma' \tau \rightarrow p_j(\sigma(w_j)) \). Putting it all together, we obtain

\[ (\bar{s}, h_1) \models_{\mathcal{C}_S} \psi' \ast \ast \prod_{j=1}^{m} \gamma_j \tau \rightarrow p_j(\sigma(w_j)) \]  

and \( (s, b) \models \mathcal{C}_S \cdot \mathcal{V} \cdot \psi' \ast \ast \prod_{j=1}^{m} \gamma_j \tau \rightarrow p_j(\sigma(w_j)) \).
Since \( \delta = \delta' \ast \bigast_{j=2}^m \gamma_j \), rule (14) implies the existence of the following rule that is an instance of (II):
\[
(\eta \ast \delta) \rightarrow q(y) \iff_{S} \exists v. \psi' \sigma \ast [\eta \ast \delta' \rightarrow p_1(\sigma(w_1))] \ast [\bigast_{j=2}^m \gamma_j \rightarrow p_j(\sigma(w_j))],
\]
where \( \eta \) is a separating conjunction of predicate atoms, such that \( \eta \sigma = \alpha \). Thus we obtain \( (s, b) \models_{S} \alpha \ast \beta \rightarrow q(u) \) follows.

### L Proof of Lemma 30 (Section 6)

\[\begin{align*}
\text{"\leq"} & \text{ Let } \psi \in C_P(s, b) \text{ be a core formula. By equation (6), it is sufficient to show the existence of core formula } \psi_i \in C_P(s, b_i), \text{ for } i = 1, 2, \text{ such that } \psi_1 \ast \psi_2 \equiv_D \psi. \\
\text{(A) First, we proceed under the following assumptions:} \\
1. & \text{ } \psi \text{ is quantifier-free thus, by Definition 19, it is of the form:} \\
& \psi = \bigast_{i=1}^n \left( \bigast_{j=1}^k q_j(u_j) \rightarrow p_i(t_i) \right) \ast \bigast_{j=m+1}^m (t_1, \ldots, t_{m+1}) \text{, for some } 0 \leq n \leq m, \\
2. & \text{ } s \text{ is bijective, i.e. } \text{rng}(s) = \mathbb{L}; \\
3. & \text{ } (s, b) \models_{S} \psi \text{ and } s(\text{roots}_{S_{\phi}}(\psi)) \cap \text{dom}(s) = \emptyset \text{ (\( \dagger \))}. \\
\text{We show the existence of two quantifier-free core formulae } \psi_1, \psi_2 \text{ with } \psi_1, \psi_2 \equiv_D \psi, \text{ s } \epsilon \mathcal{W}_{S}(s, b, \psi_1) \text{ and roots}(\psi_1) \subseteq \mathcal{V}_{\alpha} \cup \mathcal{V}_{\beta} \cup \text{roots}(\psi), \text{ for } i = 1, 2. \text{ By definition, there exist } m \text{ disjoint heaps } b_1', \ldots, b_m', \text{ such that } \psi = b_1 \psi b_2 = (\bigcup_{i=1}^m b_i') \text{ and } (s, b) \models_{S} \lambda_i \text{, for all } i \in [1 \ldots m]. \text{ First, we prove that:} \\
& \text{roots}_{S_{\phi}}(\psi) \cap D = \emptyset. \text{ (\( \dagger\dagger \))} \\
\text{Suppose, for a contradiction, that there exists a variable } x \in \text{roots}_{S_{\phi}}(\psi) \cap D. \text{ Then } s(x) \in s(\text{roots}_{S_{\phi}}(\psi)), \text{ leading to } s(x) \notin \text{dom}(b), \text{ by (\( \dagger \))}. \text{ But we also have } s(x) \in s(D), \text{ hence } s(D) \notin \text{dom}(b), \text{ which contradicts the hypothesis } s(D) \subseteq s(\text{dom}(b)) \text{ of the statement of the Lemma. Second, we build } \psi_1 \text{ and } \psi_2, \text{ distinguishing the following cases:} \\
(A.1) & \text{ If for all } i \in [1 \ldots m], \text{ either } b_i' \subseteq b_1 \text{ or } b_i' \subseteq b_2, \text{ then we let } \psi_i \eqdef \bigast_{j=1}^k q_j(u_j) \rightarrow p_i(t_i) \text{ instead of } \lambda_i = \bigast_{j=1}^k q_j(u_j) \rightarrow p_i(t_i). \text{ Since } \text{roots}_{S_{\phi}}(\psi) \cap D = \emptyset \text{ by (\( \dagger\dagger \))}, \text{ we deduce that } \psi_1, \psi_2 \equiv_D \psi \text{ (5) trivially, since } \psi = \psi_1 \ast \psi_2. \\
(A.2) & \text{ Otherwise, there exists } i \in [1 \ldots m] \text{ such that } b_i' \not\subseteq b_1 \text{ and } b_i' \not\subseteq b_2. \text{ Thus, necessarily, } ||b_i'|| > 1. \text{ Furthermore, since } ||\mathcal{V}_{\psi_i}(t)|| = 1 \text{ for all } j \in [n+1 \ldots m], \text{ we must have } i \in [1 \ldots n]. \text{ For the sake of readability we drop all references to } i \text{ and write } \lambda_i = \bigast_{j=1}^k q_j(u_j) \rightarrow p_i(t_i) \text{ instead of } \lambda_i = \bigast_{j=1}^k q_j(u_j) \rightarrow p_i(t_i). \text{ Since } s \text{ is bijective by assumption, by Lemma 49, there exists a core unfolding } \lambda_i \approx_{S_{\phi}} \varphi_i, \text{ such that } (s, b) \models_{S} \varphi_i. \text{ Because } ||\mathcal{V}_{\psi_i}(t)|| > 1, \text{ entails that } \varphi_i \not\equiv_{S} \text{ the rule used to obtain this core unfolding (see Definition 21) must have been generated by inference rule (II). Since } S \text{ is progressing, we deduce that } \varphi_i \text{ is of the form } t_0 \mapsto (t_1, \ldots, t_k) \ast \bigast_{j=1}^k (y_j \mapsto p'(t_j)), \text{ for some separating conjunctions of predicate atoms } y_1, \ldots, y_k \text{ such that } \bigast_{j=1}^k y_j = \bigast_{j=1}^k q_j(u_j), \text{ and that } t_0 = \text{root}(p(t)). \text{ Then } s(t_0) \in \text{dom}(b_i') \subseteq \text{dom}(b) \text{ and assume that } s(t_0) \in \text{dom}(b_i) \text{ (the case } s(t_0) \in \text{dom}(b_2) \text{ is symmetric). We construct a sequence of formulae by applying the same process to each occurrence of a subformula of the form } \alpha \mapsto p'(t') \text{ such that } \text{if } (r)(r'(y_j)) \in \text{dom}(b), \text{ leading to } \bigast_{j=1}^k q_j(u_j) \rightarrow p(t) \approx_{S} \bigast_{j=1}^k q_j(u_j), \text{ where:} \\
& \text{\( \alpha \) is a separating conjunction of points-to atoms,} \\
& \alpha_1, \ldots, \alpha_n \text{ are separating conjunctions of predicate atoms, such that } \bigast_{j=1}^k \delta_j = \bigast_{j=1}^k q_j(u_j), \\
& (s, b_i') \models_{S} \alpha \ast \bigast_{j=1}^k \delta_j \rightarrow r_j(v_j), \\
& \lambda(s(\text{root}(r_j(v_j)))) \in \text{dom}(b_2), \text{ for all } j \in [1 \ldots h].}
\end{align*}\]
Let $\lambda_{i,1}^1 \equiv \bigwedge_{j=1}^k \delta_j \rightarrow r_j(x)$. By definition $b_i^1 = b_i^{1,1} \cup b_i^{1,2}$, with $(b_i^{1,1}) \models \alpha$ and $(b_i^{1,2}) \models \beta$. 

Since construction $b_i^{1,1} \subseteq b_i^1$ (but we do not necessarily have $b_i^{1,2} \subseteq b_i$).

Furthermore, it is easy to check that $\alpha \models_{\epsilon_S} \lambda_{i,1}^1$ (indeed, by construction, $\alpha$ is obtained by starting from $p(t)$ and repeatedly unfolding all atoms not occurring in $\bigwedge_{j=1}^k \delta_j \rightarrow r_j(x)$), hence $(\hat{s}, b_i^{1,1}) \models_{\epsilon_S} \lambda_{i,1}^1$. By Definition 28, we have $\lambda_{i,1}^1 \rightarrow \bigwedge_{j=1}^k \delta_j \rightarrow r_j(x)$. Thus, we now prove that:

$$\text{root}(r_j(x)) \in V_{p_i'}^j \cup C,$$

for each $j \in \{1 \ldots h\}$.

Since $(\hat{s}, b_i^{1,1}) \models_{\epsilon_S} \lambda_{i,1}^1$, by Lemma 44, we have $\hat{s}(\text{root}(r_j(x))) \in \text{loc}(b_i^{1,1}) \cup \hat{s}(C)$. If $\hat{s}(\text{root}(r_j(x))) \in \hat{s}(C)$, then by injectivity of $\hat{s}$, we obtain $\text{root}(r_j(x)) \in C$ (by injectivity of $\hat{s}$). Otherwise, we obtain a sequence of formulae $\psi(x)^{+ \ldots +} \psi(0) = \psi$ where $\psi(s)$ satisfies Condition (A.1), and $(\hat{s}, b_i) \models \psi(s)$, for all $i = 1, 2, \ldots, s$. By Point (A.1), we therefore obtain formulae $\psi_j$ such that $(\hat{s}, b_j) \models_{\epsilon_S} \psi_j$, for $j = 1, 2$ and $\psi_1 \models \psi_2$, which, by $(\ddagger \ddagger)$, leads to $\psi_1, \psi_2 \nvdash_{p_d} \psi (5)$.

We prove that $(\hat{s}(\text{root}(\psi_j))) \cap \text{dom}(b_i) = \emptyset$, for $i = 1, 2$. Let $i = 1$ and $x \in \text{roots}_{p_i}(\psi_1)$ (the proof is identical for the case $i = 2$). If $x \in \text{roots}_{p_i}(\psi_1)$ then $\hat{s}(x) \not\in \text{dom}(b_i)$, by $(\ddagger \ddagger)$. Otherwise, $x \not\in \text{roots}_{p_i}(\psi_1)$ was introduced during the unfolding, hence $\hat{s}(x) \not\in \text{dom}(b_i)$, by the construction of $\psi_1$. In both cases, we have $\hat{s}(x) \not\in \text{dom}(b_i)$. Since $(\hat{s}, b_i) \models_{\epsilon_S} \psi_1$ and $\psi_1$ is quantifier-free, by construction, we have $\hat{s} \in W_S(\hat{s}, b_1, h_1)$, thus $\psi_1 \in C_p(\hat{s}, b_1, h_1)$, as required.

Next, we show that for $i = 1, 2$, each root in $\psi_i$ is contained in $V_{p_i}^j \cup C \cup \text{roots}(\psi_i)$, and that it occurs with multiplicity one. We give the proof when $i = 1$, the proof for $i = 2$ is symmetric. First, each $x \in \text{roots}(\psi_1)$ is either a root of $\psi$ or it is introduced by the unfoldings described above. In the second case we have $x \in V_{p_i}^j \cup C$ (by $(\ddagger \ddagger)$). Second, we show that all variables from $\text{roots}(\psi_i)$ occur with multiplicity one. Suppose, for a contradiction, that $x$ occurs twice as a root in $\psi_i$. If both occurrences of $x$ are in points-to atoms $x \rightarrow (t_1, \ldots, t_k)$ or in a predicate atom $\delta \rightarrow p(t)$ with $x = \text{root}(p(t))$, then since all atoms are conjoined by separating conjunctions, $\phi_1$ is unsatisfiable, which contradicts the fact that $(\hat{s}, b_1) \models_{\epsilon_S} \psi_1$. If one occurrence of $x$ occurs in $\text{roots}_{p_i}(\psi_1)$ then we have shown that $\hat{s}(x) \not\in \text{dom}(b_1)$, thus the other occurrence of $x$ cannot occur in $\text{roots}_{p_i}(\psi_1)$, which entails that it also occurs in $\text{roots}_{p_i}(\psi_1)$. Finally, assume that both occurrences of $x$ occur in $\text{roots}_{p_i}(\psi_1)$. Because $\psi \in C_p(\hat{s}, b_1, h_1)$, it must be the case that at least one occurrence of $x$ was introduced during the unfolding.

This entails that $\hat{s}(x) \in \text{dom}(b_1)$ but $x$ cannot occur in $\text{roots}_{p_i}(\psi_1)$, because $\psi \in C_p(\hat{s}, b_1, h_1)$ (Definition 23), hence both occurrences of $x$ have been introduced during the unfolding. But each time a variable $x$ is introduced in $\text{roots}_{p_i}(\psi_1)$, there is another occurrence of the same variable $x$ that is introduced in $\text{roots}_{p_i}(\psi_2)$, hence $\psi_2$ is unsatisfiable, which contradicts the fact that $(\hat{s}, b_2) \models_{\epsilon_S} \psi_2$.

(B) Let $\psi = \exists_b x_1 \forall_y \exists_y \chi$, where $\chi$ is a quantifier-free core formula in $C_p(\hat{s}, b_1, h_1)$ and let $\hat{s}$ be an injective store. Note that, since $\psi \in C_p(\hat{s}, b_1, h_1)$ and $\text{dom}(\hat{s}) \subseteq V_{p_i}^j$, we have $(x \cup y) \cap \text{dom}(\hat{s}) = \emptyset$. Because $\psi \in C_p(\hat{s}, b_1, h_1)$, by Definition 23, there exists a witness $\hat{s} \in W_S(\hat{s}, b, \psi)$, satisfying the three points of Definition 23, and such that:

$$\hat{s}(\text{roots}_{p_i}(\psi)) \cap \text{dom}(b) = \emptyset. (\ddagger \ddagger \ddagger)$$

Note that $\hat{s}$ is injective by Definition 23, and we can assume w.l.o.g. that it is bijective.

To this aim, we consider any bijection $\ell \mapsto x_\ell$ between $L \setminus \text{rng}(\hat{s})$ and $V \setminus \text{dom}(\hat{s})$. Such a bijection
exists because both $L \setminus \text{rng}(\tilde{x})$ and $V \setminus \text{dom}(\tilde{x})$ are infinitely countable. Let $\tilde{x}'$ be the extension of $\tilde{x}$ with the set of pairs $((x_1, t), t) \notin \tilde{x}$. It is easy to check that $\tilde{x}'$ is bijective.

Since $(\tilde{s}, h) \not\equiv_{\text{eq}} \varphi$ by point 1 of Definition 23 and $\varphi$ is quantifier-free, we have $\tilde{s} \in \mathcal{W}_{\text{eq}}(\tilde{s}, h, \varphi)$, hence $\varphi \in C_{\text{eq}}(\tilde{s}, h)$, because $\text{roots}_{\text{eq}}(\varphi) = \text{roots}_{\text{eq}}(\tilde{s}, h) \cap \text{dom}(h) = \emptyset$ follows from $(\tilde{s})$.

By case (A), there exist quantifier-free core formulæ $\varphi_1, \varphi_2$, such that $\varphi_1, \varphi_2 \not\equiv_{\text{D}} \varphi$, $\tilde{s} \in \mathcal{W}_{\text{eq}}(\tilde{s}, h, \varphi_i)$ and $\text{roots}(\varphi_i) \subseteq \mathcal{V}_{\text{eq}} \cup C \cup \text{roots}(\varphi)$, for $i = 1, 2$. Let $\tilde{x}_i$ be the restriction of $\tilde{s}$ to $\text{fv}(\varphi_i) \cup C$ and define the following sets, for $i = 1, 2$:

$$x_i \overset{\text{def}}{=} \left\{ x \in \text{dom}(\tilde{s}) \setminus \text{dom}(\tilde{x}_i) \mid \tilde{x}(x) \in \text{loc}(h_i) \right\}, \quad y_i \overset{\text{def}}{=} \left\{ x \in \text{dom}(\tilde{x}_i) \setminus \text{dom}(\tilde{s}) \mid \tilde{x}(x) \not\in \text{loc}(h_i) \right\}.$$

Note that we do not know at this point whether $x_i \subseteq \text{dom}(h_i)$ (this will be established later), while $y_i \subseteq \text{dom}(h_i)$ holds by definition.

We prove that for all $x \in x_i$, there exists a subformula $\delta$ occurring in $\varphi'$ such that $x \in \text{fv}(\delta)$, and either $\delta$ is a points-to atom or $\delta = a \rightarrow b$ with $x \in \text{fv}(b) \setminus \text{fv}(a)$. To this aim, we begin by proving that if some formula $\varphi'$ is obtained from the initial formula $\varphi$ by a sequence of unfoldings as defined in Part (A) and if $x \in \text{fv}(\varphi')$, then $\varphi'$ contains a formula of the form above. The proof is by induction on the length of the unfolding:

1. If $\varphi = \varphi'$, then by the hypothesis $x \not\in \text{dom}(\tilde{x})$ and $x \in \text{fv}(\varphi)$, thus $x \in x \cup y$. Since $\tilde{x}_i(x) \in \text{loc}(h_i)$, hence by Condition (3) of Definition 23, necessarily $x \in x_i$. Then the proof follows immediately from Condition (ii) in Definition 19.

2. Otherwise, according to the construction above, $\varphi'$ is obtained from an unfolding $\varphi''$ of $\varphi$, by replacing some formula $\lambda_i = k_{j=1}^k q_j(u_j) \rightarrow p_i(t_i)$ in $\varphi''$ by $\lambda_{i, 1} = k_{j=1}^h (\delta_{j} \rightarrow r_j(v_j))$, with

$$\lambda_{i, 1} = k_{j=1}^h (\delta_{j} \rightarrow r_j(v_j)) \rightarrow p_i(t_i),$$

and all atoms in $\delta_{j}$ occur in $k_{j=1}^h q_j(u_j)$.

If $x$ occurs in $\varphi''$, then by the induction hypothesis $\varphi''$ contains a formula $\delta$ satisfying the condition above. If $\delta$ is distinct from $\lambda_i$, then $\delta$ occurs in $\varphi'$ and the proof is completed. Otherwise, we have $\delta = a \rightarrow b$ with $b = p_i(t_i)$, $\alpha = k_{j=1}^h q_j(u_j)$ and $x \in \text{fv}(b) \setminus \text{fv}(a)$. We distinguish two cases: If $x \in \text{fv}(r_j(v_j))$, for some $j \in [1 \ldots h]$, then $x \in \text{fv}(r_j(v_j)) \setminus \text{fv}(\delta_j)$ (since $x \not\in \text{fv}(a) \setminus \text{fv}(\delta_j)$), thus the formula $k_{j=1}^h (\delta_{j} \rightarrow r_j(v_j))$ fulfills the required property.

Otherwise, $x \in \text{fv}(p_i(t_i)) \setminus \text{fv}(k_{j=1}^h r_j(v_j))$ and $\lambda_{i, 1}$ fulfills the property.

Now assume that $x$ does not occur in $\varphi''$. This necessarily entails that $x \in \text{fv}(r_j(v_j))$, for some $j \in [1 \ldots h]$, and that $x \not\in \text{fv}(\delta_j)$, thus $x \in \text{fv}(r_j(v_j)) \setminus \text{fv}(\delta_j)$ and the formula $\delta_j \rightarrow r_j(v_j)$ fulfills the required property.

We show that such a formula $\delta$ cannot occur in $\varphi_{3-i}$, hence necessarily occurs in $\varphi_i$, which entails that $x \in \text{dom}(\tilde{x}_i)$, and also that $x_1 \cap x_2 = \emptyset$. This is the case because if $\delta$ occurs in $\varphi_{3-i}$, then there exists a subheap $h'_{3-i}$ of $h_{3-i}$ such that $(\tilde{s}, h'_{3-i}) \models \delta$. By Lemma 45, since $x \in \text{fv}(\beta) \setminus \text{fv}(\alpha)$ when $\delta$ is of the form $\alpha \rightarrow \beta$, we have $\tilde{x}(x) \in \text{loc}(h_{3-i})$. Furthermore, by hypothesis $x \in x_i$, hence $\tilde{x}(x) \in \text{loc}(h_i)$. Therefore $\tilde{x}(x) \in \text{Fr}(h_1, h_2) \subseteq \text{rng}(\tilde{x})$ by the hypothesis (2) of the Lemma. Since $\tilde{x}$ is injective, this entails that $x \in \text{dom}(\tilde{x})$, which contradicts the definition of $x_i$.

Let $\psi_i \overset{\text{def}}{=} \exists y_i. \exists y_i. \varphi_i$, for $i = 1, 2$. Due to the previous property, $\psi_i$ satisfies Condition (ii) of Definition 19. By definition of $y_i$, we have $y_i \subseteq \text{dom}(\tilde{s}_i)$ and by definition of $\tilde{s}_i$, we have $\text{dom}(\tilde{s}_i) \subseteq \text{fv}(\varphi_i) \cup C$, thus $\psi_i$ also fulfills Condition (i) of the same definition. By part (A) $\varphi_i$ is a core formula, hence Condition (iii) is satisfied, which entails that $\psi_i$ is a core formula. Still by part (A) of the proof, $(\tilde{s}, h_i) \not\equiv_{\text{eq}} \varphi_i$, thus we also have also $(\tilde{s}_1, h_1) \not\equiv_{\text{eq}} \varphi_i$, for $i = 1, 2$. By the definition of $x_i$ and $y_i$, we have $\tilde{x}_i \in \mathcal{W}_{\text{eq}}(\tilde{s}_i, h_i, \varphi_i)$ and since $\tilde{s}_i(\text{roots}_{\text{eq}}(\varphi_i)) = \tilde{s}_i(\text{roots}_{\text{eq}}(\psi_i))$ and $\text{roots}_{\text{eq}}(\psi_i) \cap \text{dom}(h_i) = \emptyset$, we obtain $\psi_i \in C_{\text{eq}}(\tilde{s}_i, h_i)$, for $i = 1, 2$.

Since $\varphi_1 \cdot \varphi_2 \not\equiv_{\text{D}} \varphi$ and $\varphi_1, \varphi_2$ are quantifier-free, we have, by definition of $\not\equiv_{\text{D}}$:

$$\psi_1 \cdot \psi_2 \not\equiv_{\text{D}} \exists x'. \forall y'. \varphi(x', y'),$$

where $x' = (x_1 \cup x_2) \setminus \text{fv}(\varphi)$ and $y' = ((y_1 \cup y_2) \setminus \text{fv}(\varphi)) \setminus x'$.
To complete the proof, it is sufficient to show that $x' = x$ and that $y = y'$, so that $\exists x' \forall y' \varphi = \psi$.

$x' = x$ “⇒” Let $x \in x'$. We have $x \in x$, for some $i = 1, 2$, and $x \in \text{fv}(\varphi)$. By definition of $x$, this entails that $x \in \text{dom}(\tilde{S}) \cap \text{dom}(\tilde{S})$ and that $\tilde{S}(x) \in \text{loc}(\tilde{S}) \subseteq \text{loc}(\text{loc}(\tilde{S}))$. Since $x \in \text{fv}(\varphi)$ and $x \in \text{dom}(\tilde{S}) \setminus \text{dom}(\tilde{S})$, necessarily $x \in \text{dom}(\tilde{S}) \cap \text{dom}(\tilde{S})$. Hence $x \in x$.

“⇐” Let $x \in x$. We have $x \in \text{fv}(\varphi)$ by Definition 19 (ii), and $\tilde{S}(x) \in \text{loc}(\tilde{S})$ by Definition 23 (1), thus $\tilde{S}(x) \in \text{loc}(\tilde{S}(x))$, for some $i = 1, 2$, so that $x \in x'$. Consequently $x \in x'$.

$y = y'$ “⇒” Let $y \in y'$. By definition, we have $y \in y$, for some $i = 1, 2$, $y \in \text{fv}(\varphi)$, and $y \notin x = x'$. Since $y \in y$, we have $y \notin \text{dom}(\tilde{S})$, thus $y \notin \text{fv}(\varphi)$, hence $y \notin x \cup y$. Since $y \notin x$, we deduce that $y \in y$.

“⇐” Let $y \in y$. By definition, $y \notin \text{dom}(\tilde{S})$ and $y \notin x$, moreover $y \in \text{fv}(\varphi)$, by Definition 19 (i). By Definition 23 (3), we have $\tilde{S}(y) \notin \text{loc}(\tilde{S})$. By definition of $\text{loc}(\tilde{S})$, necessarily $y \notin \text{fv}(\varphi)$, for some $i = 1, 2$. Since $y \notin \text{dom}(\tilde{S})$, we deduce that $y \in x \cup y'$. Since $\tilde{S}(y) \notin \text{loc}(\tilde{S})$, we have $\tilde{S}(y) \notin \text{loc}(\tilde{S}(y))$, hence $y \notin y'$. Consequently, $y \notin y'$.

Proof of Lemma 31 (Section 7)

Let $\phi \in \text{Core}(\mathcal{F})$ be a core formula. Then $\phi$ can be viewed as a formula built over atoms of the form $p(t)$ and $t_0 \mapsto (t_1, \ldots, t_r)$ using the connectives $\ast$, $\rightarrow$ and the quantifiers $\exists$, $\forall$.
Definition 19 (iii), $\phi$ contains at most $|V_P|$ occurrences of such atoms. Since, by points (i) and (ii) of Definition 19, all the variables in $\phi$ necessary occur in an atom, this entails that $\phi$ contains at most $|V_P| \times \alpha$ (bound or free) variables, where $\alpha = \max(\{#p \mid p \in P\} \cup \{R + 1\})$ denotes the maximal arity of the relation symbols (including $\rightarrow$) in $\phi$. Since each atom is of size at most $\alpha + 1$ and since there is at most one connective $\ast$ or $\rightarrow$ for each atom, we deduce that $\text{size}(\phi) \leq 2 \times |V_P| \times \alpha + |V_P| \times (\alpha + 2)$. By definition, we have $\alpha \leq \text{width}(P)$, and $V_P$ is chosen such that $|V_P| = 2 \times \text{width}(P)$, thus $\text{size}(\phi) = O(\text{width}(P)^2)$. The symbols that may occur in the formula include the set of free and bound variables, the predicate symbols and the symbols $\rightarrow$, $\ast$, $\forall_h$, $\exists_h$, yielding at most ($|V_P| \times \alpha + \text{size}(P) + 5 \leq \text{width}(P)^2 + \text{size}(P) + 5$) symbols. Thus there are at most $(\text{width}(P)^2 + \text{size}(P) + 5) \times O(\text{width}(P)^2) = 2O(\text{width}(P)^3 \times \log(\text{size}(P)))$ core formulæ in $\text{Core}(P)$. □