A PROOF SYSTEM FOR SEPARATION LOGIC WITH MAGIC WAND

WONYEOL LEE, JINEON BAEK, AND SUNGWOO PARK

Stanford, USA
e-mail address: wonyeol@stanford.edu

POSTECH, Republic of Korea
e-mail address: gok01172@postech.ac.kr

POSTECH, Republic of Korea
e-mail address: gla@postech.ac.kr

Abstract. Separation logic is an extension of Hoare logic which is acknowledged as an enabling technology for large-scale program verification. It features two new logical connectives, separating conjunction and separating implication, but most of the applications of separation logic have exploited only separating conjunction without considering separating implication. Nevertheless the power of separating implication has been well recognized and there is a growing interest in its use for program verification. This paper develops a proof system for full separation logic which supports not only separating conjunction but also separating implication. The proof system is developed in the style of sequent calculus and satisfies the admissibility of cut. We also propose a proof search strategy based on the proof system.

1. Introduction

Separation logic [26] is an extension of Hoare logic designed to simplify reasoning about programs manipulating mutable data structures with potential pointer aliasing. It features two new logical connectives, separating conjunction $\star$ and separating implication $-\!\!\star$, whose semantics directly assumes memory heaps structured as a monoid. Separating conjunction allows us to describe properties of two disjoint heaps with a single logical formula: $A \star B$ means that a given heap can be divided into two disjoint heaps satisfying A and B respectively. Separating implication, commonly known as magic wand, allows us to reason about hypothetical heaps extending a given heap: $A -\!\!\star B$ means that if a given heap is extended with a disjoint heap satisfying A, the resultant heap satisfies B. The use of the two separating connectives naturally leads to local reasoning in program verification in that we only need to reason locally about those heaps directly affected by the program.

So far, most of the applications of separation logic have exploited only separating conjunction. For example, many existing verification tools based on separation logic, such

1998 ACM Subject Classification: F.4.1. Mathematical Logic – Mechanical theorem proving, Proof theory.
Key words and phrases: Separation logic, Cut elimination, Proof system, Theorem prover.
This is an extended version of the paper that appeared in the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '14) [19].

LOGICAL METHODS
IN COMPUTER SCIENCE
DOI:10.2168/LMCS-???

as Smallfoot [3], Space Invader [8], THOR [21], SLAyer [1], HIP [23], jStar [9], Xisa [7], VeriFast [16], Infer [5], and Predator [11], use a decidable fragment by Berdine *et al.* [2] or its extension which provides only separating conjunction. By virtue of the principle of local reasoning, however, these tools are highly successful in their individual verification domains despite not using separating implication at all.

Although separating implication is not discussed as extensively as separating conjunction in the literature, its power in program verification has nevertheless been well recognized. Just around the inception of separation logic, Yang [27] already gives an elegant proof of the correctness of the Schorr-Waite algorithm which relies crucially on the use of separating implication in the main loop invariant. Krishnaswami [17] shows how to reason abstractly about an iterator protocol with separation logic by exploiting separating implication in the specification of iterators. Maeda *et al.* [20] adopt the idea of separating implication in extending an alias type system in order to express tail-recursive operations on recursive data structures. Recently Hobor and Villard [14] give a concise proof of the correctness of Cheney's garbage collector in a proof system based on the ramify rule, a cousin of the frame rule of separation logic, whose premise checks a logical entailment involving separating implication. These promising results arguably suggest that introducing separating implication alone raises the level of technology for program verification as much as separation logic only with separating conjunction improves on Hoare logic.

Despite the potential benefit of separating implication in program verification, however, there is still no practical theorem prover for full separation logic. The state-of-the-art theorem provers for separation logic such as SeLoger [13] and SLP [22] support only separating conjunction, and the labelled tableau calculus by Galmiche and Méry [12] does not directly give rise to a proof search strategy. Because of the unavailability of such a theorem prover, all proofs exploiting separating implication should be manually checked, which can be time-consuming even with the help of lemmas provided by the proof system (as in [14]). Another consequence is that no existing verification tools based on separation logic can fully support backward reasoning by weakest precondition generation, which requires separating implication whenever verifying heap assignments (see Ishtiaq and O'Hearn [15]).

This paper develops a proof system $\mathbf{P_{SL}}$ for full separation logic which supports not only separating conjunction but also separating implication. Its design is based on the principle of proof by contradiction from classical logic, and we develop its inference rules in the style of sequent calculus. $\mathbf{P_{SL}}$ uses a new form of sequent, called *world sequent*, in order to give a complete description of the world of heaps, and its use of world sequents allows us to treat separating implication in the same way that it treats separating conjunction. The key challenge in the development of $\mathbf{P_{SL}}$ is to devise a set of inference rules for manipulating heap structures so as to correctly analyze separating conjunction and separating implication.

We show that $\mathbf{P_{SL}}$ satisfies the admissibility of cut and that it is sound with respect to separation logic. Although $\mathbf{P_{SL}}$ is not complete with respect to separation logic, it achieves a high degree of completeness in that those valid formulas that practically arise in program verification are usually provable in $\mathbf{P_{SL}}$. We then explain our proof search strategy $\mathcal{SS}$ for $\mathbf{P_{SL}}$ which always terminates and serves as the basis for our prototype implementation of $\mathbf{P_{SL}}$. We show that it is easy to extend $\mathbf{P_{SL}}$ with new logical connectives and predicates, such as an overlapping conjunction $A \circledast B$ by Hobor and Villard [14].

Separating implication has been commonly considered to be much harder to reason about than separating conjunction, as partially evidenced by lack of theorem provers supporting separating implication and abundance of verification systems supporting separating

conjunction. Our development of $\mathbf{P_{SL}}$, however, suggests that a proof system designed in a principled way can support both logical connectives in a coherent way without requiring distinct treatments. Our prototype implementation of $\mathbf{P_{SL}}$ also suggests that such a proof system can develop into a practical theorem prover for separation logic. To the best of our knowledge, $\mathbf{P_{SL}}$ is the first proof system for full separation logic that satisfies the admissibility of cut and provides a concrete proof search strategy.

This paper is organized as follows. Section 2 gives preliminaries on separation logic. Section 3 develops our proof system $\mathbf{P_{SL}}$ and Section 4 gives two examples of proving world sequents. Section 5 proves the admissibility of cut of $\mathbf{P_{SL}}$ and Section 6 proves the soundness of $\mathbf{P_{SL}}$ with respect to separation logic. Section 7 explains our proof search strategy $\mathcal{SS}$ for $\mathbf{P_{SL}}$ and Section 8 discusses the implementation and extension of $\mathbf{P_{SL}}$. Section 9 discusses related work and Section 10 concludes.

2. Semantics of separation logic

Separation logic extends classical first-order logic with multiplicative formulas from intuitionistic linear logic:

$$\begin{array}{rlll}
\text{formula} & A, B, C & ::= & P \mid \bot \mid \neg A \mid A \vee A \mid \\
 & & & \mathsf{I} \mid A \star A \mid A \mathrel{-\!\!\star} A \mid \exists a.A \\
\text{primitive formula} & P & ::= & [l \mapsto E] \mid E = E \mid \cdots \\
\text{expression} & E & ::= & x \mid a \mid \mathsf{L} \mid \cdots \\
\text{location expression} & l & ::= & x \mid a \mid \mathsf{L} \\
\text{value} & V & ::= & \mathsf{L} \mid \cdots \\
\text{location} & \mathsf{L}_1, \mathsf{L}_2, \mathsf{L}_3, \cdots & & \\
\text{stack variable} & x, y, z & & \\
\text{local variable} & a, b, c & &
\end{array}$$

$\bot$, $\neg A$, $A \vee B$, and $\exists a.A$ are from classical first-order logic. I is the multiplicative unit. $A \star B$ is a separating conjunction and $A \mathrel{-\!\!\star} B$ is a separating implication. We define $\top$ as $\neg\bot$, $A \wedge B$ as $\neg(\neg A \vee \neg B)$, and $A \supset B$ as $\neg A \vee B$. We use conventional precedence rules for logical connectives: $\neg > \star > \vee > \mathrel{-\!\!\star} > \exists$. In this work, we do not consider inductively defined predicates.

Primitive formulas include a points-to relation $[l \mapsto E]$ for describing a singleton heap. All other primitive formulas describe relations between expressions; for simplicity, we consider only an equality relation $E = E'$. Expressions denote values which include locations L. Location expressions are a special class of expressions that denote locations. In the present work, we allow only locations as values, but it should be straightforward to introduce additional forms of expressions for new types of values such as booleans and integers. We syntactically distinguish between stack variables which originate from the program being verified (and thus may be called global variables instead) and local variables which are introduced by existential quantifiers (and thus can never appear outside corresponding existential formulas).

We specify the semantics of separation logic with respect to a stack and a heap. A stack S is a finite partial mapping $\mathit{Var} \rightharpoonup \mathit{Val}$ from stack variables to values where Var denotes the set of stack variables and Val denotes the set of values. Given a stack S, we can determine a unique value for every expression E, which we write as $[\![E]\!]_S$. A heap H is a finite partial mapping $\mathit{Loc} \rightharpoonup \mathit{Val}$ from locations to values where Loc denotes the set of locations. We

write $H_1 \# H_2$ to mean that heaps H_1 and H_2 are disjoint, *i.e.*, $dom(H_1) \cap dom(H_2) = \varnothing$. We write $H_1 \circ H_2$ for the union of disjoint heaps H_1 and H_2 where $H_1 \# H_2$ is assumed, and ϵ for an empty heap. Heaps form a commutative cancellative monoid with $\circ$ as the associative operation and ϵ as the identity:

$$\begin{array}{rl} \text{(neutrality)} & H \circ \epsilon = H \\ \text{(commutativity)} & H_1 \circ H_2 = H_2 \circ H_1 \\ \text{(associativity)} & H_1 \circ (H_2 \circ H_3) = (H_1 \circ H_2) \circ H_3 \\ \text{(cancellativity)} & H \circ H_1 = H \circ H_2 \text{ implies } H_1 = H_2. \end{array}$$

Given a stack S and a heap H, we obtain the semantics of separation logic from the satisfaction relation $(S, H) \models A$ for formulas defined as follows:

- $(S, H) \models [l \mapsto E]$ iff. $H = \langle [\![l]\!]_S \mapsto [\![E]\!]_S \rangle$, *i.e.*, H is a singleton heap mapping $[\![l]\!]_S$ to $[\![E]\!]_S$.
- $(S, H) \models E = E'$ iff. $[\![E]\!]_S = [\![E']\!]_S$.
- $(S, H) \models \bot$ iff. never.
- $(S, H) \models \neg A$ iff. $(S, H) \not\models A$.
- $(S, H) \models A \vee B$ iff. $(S, H) \models A$ or $(S, H) \models B$.
- $(S, H) \models \mathsf{I}$ iff. $dom(H) = \varnothing$, *i.e.*, $H = \epsilon$.
- $(S, H) \models A \star B$ iff. $H = H_1 \circ H_2$ and $(S, H_1) \models A$ and $(S, H_2) \models B$ for some heaps H_1 and H_2.
- $(S, H) \models A \mathrel{-\!\!\star} B$ iff. $H_2 = H \circ H_1$ implies $(S, H_1) \not\models A$ or $(S, H_2) \models B$ for any heaps H_1 and H_2.
- $(S, H) \models \exists a.A$ iff. $(S, H) \models [V/a]A$ for some value V.

Note that the definition of $(S, H) \models \exists a.A$ directly substitutes value V for local variable a in formula A (in $[V/a]A$) without extending stack S because we syntactically distinguish between stack variables and local variables.

Although the satisfaction relation $(S, H) \models A$ is enough for specifying the semantics of separation logic, we deliberately derive the definition of its negation $(S, H) \not\models A$, which plays an equally important role in the development of our proof system:

- $(S, H) \not\models [l \mapsto E]$ iff. $H \neq \langle [\![l]\!]_S \mapsto [\![E]\!]_S \rangle$, *i.e.*, $dom(H) \neq \{[\![l]\!]_S\}$ or $H([\![l]\!]_S) \neq [\![E]\!]_S$.
- $(S, H) \not\models E = E'$ iff. $[\![E]\!]_S \neq [\![E']\!]_S$.
- $(S, H) \not\models \bot$ iff. always.
- $(S, H) \not\models \neg A$ iff. $(S, H) \models A$.
- $(S, H) \not\models A \vee B$ iff. $(S, H) \not\models A$ and $(S, H) \not\models B$.
- $(S, H) \not\models \mathsf{I}$ iff. $dom(H) \neq \varnothing$, *i.e.*, $H \neq \epsilon$.
- $(S, H) \not\models A \star B$ iff. $H = H_1 \circ H_2$ implies $(S, H_1) \not\models A$ or $(S, H_2) \not\models B$ for any heaps H_1 and H_2.
- $(S, H) \not\models A \mathrel{-\!\!\star} B$ iff. $H_2 = H \circ H_1$ and $(S, H_1) \models A$ and $(S, H_2) \not\models B$ for some heaps H_1 and H_2.
- $(S, H) \not\models \exists a.A$ iff. $(S, H) \not\models [V/a]A$ for any value V.

We observe that the definition for separating implication is symmetric to the definition for separating conjunction:

- $(S, H) \models A \star B$ should find a certain pair of heaps whereas $(S, H) \not\models A \star B$ should analyze an unspecified pair of heaps.
- $(S, H) \models A \mathrel{-\!\!\star} B$ should analyze an unspecified pair of heaps whereas $(S, H) \not\models A \mathrel{-\!\!\star} B$ should find a certain pair of heaps.

This symmetry suggests that we can incorporate separating implication into the proof system in an analogous way to separating conjunction.

A formula A is valid, written $\models A$, if $(S, H) \models A$ holds for every stack S and heap H.

3. Proof system $\mathbf{P_{SL}}$ for separation logic

This section presents the proof system $\mathbf{P_{SL}}$ for separation logic which is developed in the style of sequent calculus. We first explain *world sequents*, the main judgment in $\mathbf{P_{SL}}$, and then present the inference rules.

3.1. World sequents.

The design of $\mathbf{P_{SL}}$ is based on the principle of proof by contradiction from classical logic. We describe the state of each heap with a set of true formulas and another set of false formulas. A world sequent in $\mathbf{P_{SL}}$ gives a description of the entire world of heaps, and a derivation of it means that the description is self-contradictory. Hence, in order to check the validity of a formula in separation logic, we use it as a false formula about an arbitrary heap w (about which nothing is known) and attempt to produce a logical contradiction by proving a world sequent consisting solely of heap w. The definition of world sequents and the principle of proof by contradiction are inherited from the nested sequent calculus for Boolean BI by Park *et al.* [24].

Since $\mathbf{P_{SL}}$ is designed to check the validity of a formula, it assumes an arbitrary stack, which implies that every stack variable denotes an arbitrary value. This in turn implies that in a derivation of a world sequent, we may use a fresh stack variable to denote an arbitrary value. We exploit this interpretation of stack variables in an inference rule for first-order formulas.

A world sequent consists of expression relations Θ, heap relations Σ, and heap sequents Π:

$$
\begin{array}{rlcl}
\text{expression relation} & \theta & ::= & E = E' \mid E \neq E' \\
\text{expression relations} & \Theta & = & \theta_1, \cdots, \theta_n \\
\text{heap variable} & w, u, v & & \\
\text{heap relation} & \sigma & ::= & w \doteq \epsilon \mid w \not\doteq \epsilon \mid \\
 & & & w \doteq [l \mapsto E] \mid w \not\doteq [l \mapsto E] \mid \\
 & & & w \doteq w_1 \circ w_2 \\
\text{heap relations} & \Sigma & = & \sigma_1, \cdots, \sigma_n \\
\text{truth context} & \Gamma & ::= & \cdot \mid \Gamma, A \\
\text{falsehood context} & \Delta & ::= & \cdot \mid \Delta, A \\
\text{heap sequent} & \pi & ::= & [\Gamma \Longrightarrow \Delta]^w \\
\text{heap sequents} & \Pi & = & \pi_1, \cdots, \pi_n \\
\text{world sequent} & \Theta; \Sigma \parallel \Pi & &
\end{array}
$$

- An expression relation θ is an equality or inequality between two expressions. If we introduce new forms of primitive formulas (*e.g.*, $E < E'$), we should introduce corresponding forms of expression relations.
- We assign a heap variable to each heap, and a heap relation σ relates a heap to an empty heap ($w \doteq \epsilon$ and $w \not\doteq \epsilon$), a singleton heap ($w \doteq [l \mapsto E]$ and $w \not\doteq [l \mapsto E]$), or the union of two disjoint heaps ($w \doteq w_1 \circ w_2$). We refer to those heap relation involving an empty heap or a singleton heap as *atomic heap relations*. As heaps form a commutative (cancellative) monoid, we assume commutativity of $\circ$ and use $w_1 \circ w_2$ and $w_2 \circ w_1$ interchangeably.

- A heap sequent $[\Gamma \Longrightarrow \Delta]^w$ describes heap w with truth context Γ and falsehood context Δ which contain true formulas and false formulas, respectively, about heap w.

In this way, a world sequent $\Theta; \Sigma \parallel \Pi$ gives a complete description of the world of heaps. We require that no local variable appear in expression relations and heap relations, and that a world sequent contain a unique heap sequent for each heap variable.

A world sequent represents a graph of heaps induced by heap relations. Given a heap relation $w \doteq w_1 \circ w_2$, we say that parent heap w has two child heaps w_1 and w_2 which are sibling heaps to each other. We can also extend parent-child relations to derive ancestor-descendant relations. If a heap has no pair of child heaps, we call it a terminal heap (where we ignore such a heap relation as $w \doteq w \circ w_\epsilon$ with $w_\epsilon \doteq \epsilon$); otherwise we call it a non-terminal heap. Note that a heap relation $w \doteq \epsilon$ or $w \doteq [l \mapsto E]$ does not immediately mean that w is a terminal heap because we may have another heap relation $w \doteq w_1 \circ w_2$. $\mathbf{P_{SL}}$, however, allows us to normalize all heap relations and turn w into a terminal heap.

$\mathbf{P_{SL}}$ also uses an expression contradiction judgment $\Theta \vdash \bot$ which is an abbreviation of a particular form of a world sequent $\Theta; \cdot \parallel \cdot$ and means that expression relations Θ produce a logical contradiction. Since the definition of expression relations is extensible, we do not give inference rules for the expression contradiction judgment and just assume a decidable system for it.

$\mathbf{P_{SL}}$ consists of logical rules in Figure 1, structural rules in Figure 2, and heap contradiction rules in Figure 3. The logical rules deal with formulas in heap sequents Π, the structural rules reorganize graphs of heaps induced by heap relations Σ, and the heap contradiction rules detect logical contradictions in heap relations Σ, or *heap contradictions*. $\mathbf{P_{SL}}$ shares the logical rules (for propositional and multiplicative formulas) with the nested sequent calculus for Boolean BI in [24], but the structural rules and the heap contradiction rules are specific to separation logic. We read every inference rule from the conclusion to the premise, and the derivation of a world sequent always terminates with a proof of a logical contradiction. Hence, in order to show the validity of a formula A, we try to prove a world sequent $\cdot; \cdot \parallel [\cdot \Longrightarrow A]^w$.

3.2. **Logical rules of $\mathbf{P_{SL}}$.** Figure 1 shows the logical rules of $\mathbf{P_{SL}}$. Except for the rule $\mathsf{ExpCont}$, a logical rule focuses on a principal formula in a heap sequent and either produces a logical contradiction (in the rule $\bot\mathsf{L}$) or rewrites the world sequent of the conclusion according to the semantics of separation logic in Section 2. For each type of formulas, $\mathbf{P_{SL}}$ has both a left rule, which analyzes a true formula about a heap, and a right rule, which analyzes a false formula about a heap, as in a typical sequent calculus. The rules for points-to relations introduce a corresponding heap relation. The rules for propositional and first-order formulas are from first-order classical logic. In the rule $\exists\mathsf{L}$, the fresh stack variable x denotes an arbitrary value. In the rules $\exists\mathsf{L}$ and $\exists\mathsf{R}$, we write $[E/a]A$ for substituting expression E for local variable a in formula A. The rules $=\mathsf{L}$ and $=\mathsf{R}$ are the only logical rules that add expression relations, and the rule $\mathsf{ExpCont}$ checks if expression relations Θ produce a logical contradiction.

The rules IL and IR use the fact the I is true only at an empty heap. The rules $\star\mathsf{L}$ and $\star\mathsf{R}$ are based on the following interpretation of multiplicative conjunction $\star$ which closely matches the semantics of separation logic in Section 2:

Rules for points-to relations:

$$\frac{\Theta; \Sigma, w \doteq [l \mapsto E] \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma, [l \mapsto E] \Longrightarrow \Delta]^w} \mapsto\mathsf{L} \qquad \frac{\Theta; \Sigma, w \neq [l \mapsto E] \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, [l \mapsto E]]^w} \mapsto\mathsf{R}$$

Rules for propositional formulas:

$$\frac{}{\Theta; \Sigma \parallel \Pi, [\Gamma, \bot \Longrightarrow \Delta]^w} \bot\mathsf{L} \qquad \frac{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma, \neg A \Longrightarrow \Delta]^w} \neg\mathsf{L} \qquad \frac{\Theta; \Sigma \parallel \Pi, [\Gamma, A \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, \neg A]^w} \neg\mathsf{R}$$

$$\frac{\Theta; \Sigma \parallel \Pi, [\Gamma, A \Longrightarrow \Delta]^w \quad \Theta; \Sigma \parallel \Pi, [\Gamma, B \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma, A \vee B \Longrightarrow \Delta]^w} \vee\mathsf{L} \qquad \frac{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A, B]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \vee B]^w} \vee\mathsf{R}$$

Rules for multiplicative formulas:

$$\frac{\Theta; \Sigma, w \doteq \epsilon \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma, \mathsf{I} \Longrightarrow \Delta]^w} \mathsf{I}\mathsf{L} \qquad \frac{\Theta; \Sigma, w \neq \epsilon \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, \mathsf{I}]^w} \mathsf{I}\mathsf{R}$$

$$\frac{\textit{fresh } w_1, w_2 \qquad \Theta; \Sigma, w \doteq w_1 \circ w_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [A \Longrightarrow \cdot]^{w_1}, [B \Longrightarrow \cdot]^{w_2}}{\Theta; \Sigma \parallel \Pi, [\Gamma, A \star B \Longrightarrow \Delta]^w} \star\mathsf{L}$$

$$\frac{w \doteq w_1 \circ w_2 \in \Sigma \qquad \begin{array}{r}\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \star B]^w, [\Gamma_1 \Longrightarrow \Delta_1, A]^{w_1}, [\Gamma_2 \Longrightarrow \Delta_2]^{w_2} \\ \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \star B]^w, [\Gamma_1 \Longrightarrow \Delta_1]^{w_1}, [\Gamma_2 \Longrightarrow \Delta_2, B]^{w_2}\end{array}}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \star B]^w, [\Gamma_1 \Longrightarrow \Delta_1]^{w_1}, [\Gamma_2 \Longrightarrow \Delta_2]^{w_2}} \star\mathsf{R}$$

$$\frac{w_2 \doteq w \circ w_1 \in \Sigma \qquad \begin{array}{r}\Theta; \Sigma \parallel \Pi, [\Gamma, A \mathrel{-\!\!\star} B \Longrightarrow \Delta]^w, [\Gamma_1 \Longrightarrow \Delta_1, A]^{w_1}, [\Gamma_2 \Longrightarrow \Delta_2]^{w_2} \\ \Theta; \Sigma \parallel \Pi, [\Gamma, A \mathrel{-\!\!\star} B \Longrightarrow \Delta]^w, [\Gamma_1 \Longrightarrow \Delta_1]^{w_1}, [\Gamma_2, B \Longrightarrow \Delta_2]^{w_2}\end{array}}{\Theta; \Sigma \parallel \Pi, [\Gamma, A \mathrel{-\!\!\star} B \Longrightarrow \Delta]^w, [\Gamma_1 \Longrightarrow \Delta_1]^{w_1}, [\Gamma_2 \Longrightarrow \Delta_2]^{w_2}} \mathrel{-\!\!\star}\mathsf{L}$$

$$\frac{\textit{fresh } w_1, w_2 \qquad \Theta; \Sigma, w_2 \doteq w \circ w_1 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [A \Longrightarrow \cdot]^{w_1}, [\cdot \Longrightarrow B]^{w_2}}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \mathrel{-\!\!\star} B]^w} \mathrel{-\!\!\star}\mathsf{R}$$

Rules for first-order formulas:

$$\frac{\textit{fresh } x \qquad \Theta; \Sigma \parallel \Pi, [\Gamma, [x/a]A \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma, \exists a.A \Longrightarrow \Delta]^w} \exists\mathsf{L} \qquad \frac{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, [E/a]A, \exists a.A]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, \exists a.A]^w} \exists\mathsf{R}$$

Rules for primitive formulas for expressions:

$$\frac{\Theta, E = E'; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma, E = E' \Longrightarrow \Delta]^w} =\mathsf{L} \qquad \frac{\Theta, E \neq E'; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, E = E']^w} =\mathsf{R} \qquad \frac{\Theta \vdash \bot}{\Theta; \Sigma \parallel \Pi} \mathsf{ExpCont}$$

Figure 1: Logical rules in the proof system $\mathbf{P_{SL}}$ for separation logic

- $A \star B$ is true at heap w iff. $w \doteq w_1 \circ w_2$ and A is true at heap w_1 and B is true at heap w_2 for *some* heaps w_1 and w_2.
- $A \star B$ is false at heap w iff. $w \doteq w_1 \circ w_2$ implies that A is false at heap w_1 or that B is false at heap w_2 for *any* heaps w_1 and w_2.

Hence the rule $\star\mathsf{L}$ creates (*some*) fresh child heaps w_1 and w_2, whereas the rule $\star\mathsf{R}$ chooses (*any*) existing child heaps w_1 and w_2. Similarly the rules $\mathrel{-\!\!\star}\mathsf{L}$ and $\mathrel{-\!\!\star}\mathsf{R}$ are based on the following interpretation of multiplicative implication $\mathrel{-\!\!\star}$:

- $A \mathrel{-\!\!\star} B$ is true at heap w iff. $w_2 \doteq w \circ w_1$ implies that A is false at heap w_1 or that B is true at heap w_2 for *any* heaps w_1 and w_2.
- $A \mathrel{-\!\!\star} B$ is false at heap w iff. $w_2 \doteq w \circ w_1$ and A is true at heap w_1 and B is false at heap w_2 for *some* heaps w_1 and w_2.

Hence the rule $\mathrel{-\!\!\star}\mathsf{L}$ chooses (*any*) existing sibling heap w_1 and parent heap w_2, whereas as the rule $\mathrel{-\!\!\star}\mathsf{R}$ creates (*some*) fresh sibling heap w_1 and parent heap w_2. The rules $\star\mathsf{L}$ and $\mathrel{-\!\!\star}\mathsf{R}$ are the only logical rules that add parent-child heap relations to extend the graph of heaps, and introduce fresh heap variables w_1 and w_2 that are not found in the world sequent in the conclusion. The rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$ are the only logical rules that replicate the principal formula into world sequents in the premise.

In the rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$, we allow equalities between heap variables w_1, w_2, and w. Since an equality between these heap variables invalidates the requirement that a world sequent contain a unique heap sequent for each heap variable, we interpret heap sequents for the same heap variable in the rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$ as follows:

- In the conclusion, we implicitly replicate the same heap sequent as necessary.
- In the premise, we combine all changes made to individual heap sequents for the same heap variable to produce a single heap sequent.

For example, an equality $w = w_1$ in the rule $\star\mathsf{R}$ yields the following special instance:

$$\frac{\begin{array}{c}\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \star B, A]^w, [\Gamma_2 \Longrightarrow \Delta_2]^{w_2} \\ w \doteq w \circ w_2 \in \Sigma \quad \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \star B]^w, [\Gamma_2 \Longrightarrow \Delta_2, B]^{w_2}\end{array}}{\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A \star B]^w, [\Gamma_2 \Longrightarrow \Delta_2]^{w_2}}$$

The rule $\star\mathsf{R}$ has two more special instances (corresponding to heap relations $w \doteq w_1 \circ w_1$ and $w \doteq w \circ w$), and similarly the rule $\mathrel{-\!\!\star}\mathsf{L}$ has a total of three special instances.

Now we can decompose each individual formula by applying its corresponding logical rule, thus accumulating expression relations and heap relations and creating fresh heaps. When expression relations become self-contradictory, we apply the rule $\mathsf{ExpCont}$, thus obtaining a complete derivation tree. In order to achieve a high degree of completeness of $\mathbf{P_{SL}}$ with respect to separation logic, however, we should also be able to: 1) enumerate as many heap relations $w \doteq w_1 \circ w_2$ and $w_2 \doteq w \circ w_1$ for a given heap w as possible for the rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$; 2) produce heap contradictions, for example, from $w \doteq \epsilon$ and $w \doteq [l \mapsto E]$. (We assume that we can make a correct guess on expression E in the rule $\exists\mathsf{R}$.) The remaining challenge is to devise a set of structural rules and another set of heap contradiction rules accomplishing these two goals, which would enable us to enumerate as many feasible heap relations as possible from those generated by the logical rules and detect all types of heap contradictions.

3.3. **Structural rules of $\mathbf{P_{SL}}$.** The structural rules of $\mathbf{P_{SL}}$ are divided into five groups according to their roles in reorganizing graphs of heaps represented by world sequents. The order of the structural rules in Figure 2 roughly follows their use in our proof search strategy $\mathcal{SS}$ described in Section 7. In a certain sense, we design the structural rules so as to maximize the degree of completeness of $\mathbf{P_{SL}}$ with respect to separation logic when the logical rules are already given as in Figure 1. Below we informally discuss the key properties of the structural rules, which we formally present in Section 7.2.

Rule for disambiguating heap relations and leaving only disjoint terminal heaps:

$$\frac{\{w \doteq u_1 \circ u_2, w \doteq v_1 \circ v_2\} \subset \Sigma \quad \textit{fresh } w_1, w_2, w_3, w_4 \quad \Theta; \Sigma, \begin{array}{l} u_1 \doteq w_1 \circ w_2, \\ u_2 \doteq w_3 \circ w_4, \\ v_1 \doteq w_1 \circ w_3, \\ v_2 \doteq w_2 \circ w_4 \end{array} \parallel \Pi, \begin{array}{l} [\cdot \Longrightarrow \cdot]^{w_1}, \\ [\cdot \Longrightarrow \cdot]^{w_2}, \\ [\cdot \Longrightarrow \cdot]^{w_3}, \\ [\cdot \Longrightarrow \cdot]^{w_4} \end{array}}{\Theta; \Sigma \parallel \Pi} \ \mathsf{Disj}\star$$

Rules for applying associativity of the union of disjoint heaps:

$$\frac{\{w \doteq u \circ v, u \doteq u_1 \circ u_2\} \subset \Sigma \quad \textit{fresh } u' \quad \Theta; \Sigma, u' \doteq u_2 \circ v, w \doteq u_1 \circ u' \parallel \Pi, [\cdot \Longrightarrow \cdot]^{u'}}{\Theta; \Sigma \parallel \Pi} \ \mathsf{Assoc}$$

Rules for propagating atomic heap relations:

$$\frac{\{w \doteq \epsilon, w \doteq w_1 \circ w_2\} \subset \Sigma \quad \Theta; \Sigma, w_1 \doteq \epsilon, w_2 \doteq \epsilon \parallel \Pi}{\Theta; \Sigma \parallel \Pi} \ \mathsf{Prop}\epsilon$$

$$\frac{\{w \doteq [l \mapsto E], w \doteq w_1 \circ w_2\} \subset \Sigma \quad \begin{array}{l} \Theta; \Sigma, w_1 \doteq [l \mapsto E], w_2 \doteq \epsilon \parallel \Pi \\ \Theta; \Sigma, w_1 \doteq \epsilon, w_2 \doteq [l \mapsto E] \parallel \Pi \end{array}}{\Theta; \Sigma \parallel \Pi} \ \mathsf{Prop}\mapsto$$

Rules for normalizing heap relations:

$$\frac{\Theta; [w/w'](\Sigma, w \doteq u \circ v) \parallel [w/w']\Pi}{\Theta; \Sigma, w \doteq u \circ v, w' \doteq u \circ v \parallel \Pi} \ \mathsf{NormEq}$$

$$\frac{\Theta; [w/u](\Sigma, v \doteq \epsilon) \parallel [w/u]\Pi}{\Theta; \Sigma, w \doteq u \circ v, v \doteq \epsilon \parallel \Pi} \ \mathsf{NormPC} \qquad \frac{\Theta; [w/u](\Sigma, w \doteq \epsilon) \parallel [w/u]\Pi}{\Theta; \Sigma, w \doteq \epsilon, u \doteq \epsilon \parallel \Pi} \ \mathsf{NormEmpty}$$

Rules for creating an empty heap and applying the monoid laws for empty heaps:

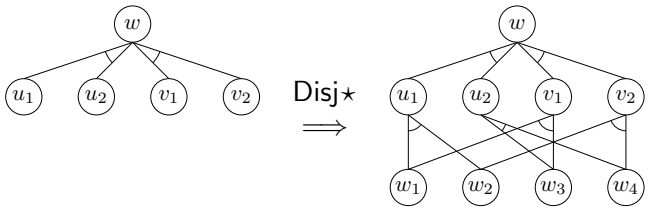
$$\frac{\textit{fresh } w_\epsilon \quad \Theta; \Sigma, w_\epsilon \doteq \epsilon \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}{\Theta; \Sigma \parallel \Pi} \ \mathsf{ENew}$$

$$\frac{w_\epsilon \doteq \epsilon \in \Sigma \quad \Theta; \Sigma, w \doteq w \circ w_\epsilon \parallel \Pi}{\Theta; \Sigma \parallel \Pi} \ \mathsf{EJoin} \qquad \frac{w \doteq w \circ u \in \Sigma \quad \Theta; \Sigma, u \doteq \epsilon \parallel \Pi}{\Theta; \Sigma \parallel \Pi} \ \mathsf{ECancel}$$

Figure 2: Structural rules in the proof system $\mathbf{P_{SL}}$ for separation logic

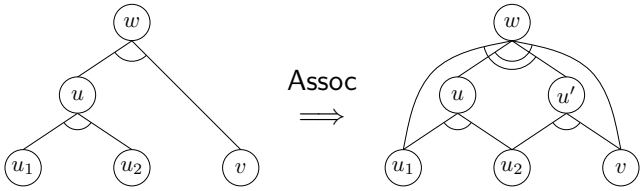
3.3.1. *Rule for disambiguating heap relations.* The rule $\mathsf{Disj}\star$ disambiguates heap relations in order to make disjoint all terminal heaps which are subsumed by a common root heap. Roughly speaking, two heaps are disjoint if they share a common ancestor which has a heap relation separating them. In the premise of the rule $\mathsf{Disj}\star$, child heaps u_i and v_j $(i, j = 1, 2)$ share a common parent heap w, but their exact relations are unknown. For example, heap u_1 may completely subsume, partially overlap with, or be disjoint from heap v_1. In general,

each pair of child heaps u_i and v_j are allowed to share a common child heap, so the rule $\mathsf{Disj}\star$ disambiguates their relations by introducing four fresh terminal heaps, w_1 to w_4, which are all disjoint from each other:



Now, for example, we may assume that the intersection of heaps u_1 and v_1 is represented by heap w_1. Note that if heap u_i or v_j is not a terminal heap, the rule $\mathsf{Disj}\star$ gives rise to unknown relations between the existing child heaps of u_i or v_j and two of the fresh terminal heaps. Thus the rule $\mathsf{Disj}\star$ eliminates unknown relations between child heaps potentially creating similar unknown relations. By repeatedly applying these rules, we can eventually obtain a graph of heaps such that *all terminal heaps subsumed by a common root heap in the graph are disjoint.* The rule $\mathsf{Disj}\star$ corresponds to the cross-split axiom for separation algebras [10].

3.3.2. *Rule for applying associativity of* $\circ$ *.* The rule Assoc creates new heap relations according to associativity of the union of disjoint heaps. Suppose that we have two heap relations $w \doteq u \circ v$ and $u \doteq u_1 \circ u_2$. The rule Assoc introduces a fresh heap u' in order to associate two heaps u_2 and v which are known to be disjoint but do not have a common parent heap yet; it also assigns heap w as the common parent heap of heaps u_1 and u':



Note that unlike the rule $\mathsf{Disj}\star$, the rule Assoc creates no fresh terminal heaps.

The rule Assoc is crucial for enumerating heap relations involving a particular heap. The basic observation is that by repeatedly applying the rule Assoc to a graph of heaps, we can eventually obtain another graph of heaps with the same set of terminal heaps such that *for each combination of terminal heaps subsumed by a common root heap, there is at least one heap subsuming exactly these terminal heaps and no others.* By starting with a graph of heaps obtained by repeatedly applying the rule $\mathsf{Disj}\star$, we can enumerate all feasible heap relations $w \doteq w_1 \circ w_2$ and $w_2 \doteq w \circ w_1$ for a particular heap w where we assume that heaps w_1 and w_2 are in the graph. For the case that w_1 or w_2 is an empty heap, however, we need another set of structural rules for dealing with empty heaps. We should also combine heap sequents for the same heap.

3.3.3. *Rules for propagating atomic heap relations.* The rules for propagating atomic heap relations, or propagation rules, are designed to propagate all atomic heap relations ($w \doteq \epsilon$, $w \not\doteq \epsilon$, $w \doteq [l \mapsto E]$, $w \not\doteq [l \mapsto E]$) from non-terminal heaps to terminal heaps. A propagation rule converts an atomic heap relation for a heap w into semantically equivalent heap

relations for its child heaps w_1 and w_2 (with $w \doteq w_1 \circ w_2$). It rewrites world sequents according to the following fact on atomic heap relations where we assume $w \doteq w_1 \circ w_2$:

- $w \doteq \epsilon$ iff. $w_1 \doteq \epsilon$ and $w_2 \doteq \epsilon$ (for the rule $\mathsf{Prop}\epsilon$).
- $w \doteq [l \mapsto E]$ iff. either $w_1 \doteq [l \mapsto E]$ and $w_2 \doteq \epsilon$, or $w_1 \doteq \epsilon$ and $w_2 \doteq [l \mapsto E]$ (for the rule $\mathsf{Prop}\mapsto$).

Note that although the new heap relations for the child heaps w_1 and w_2 collectively imply the original heap relation σ, we have to preserve σ in every world sequent of the premise because it may still interact with another pair of child heaps w_1' and w_2' (with $w \doteq w_1' \circ w_2'$). After considering all such interactions, however, we may safely discard σ (by the rule Weaken to be introduced in Section 7.1).

The propagation rules are the first step toward a complete procedure for producing heap contradictions (which detect all types of heap contradictions). Suppose that we repeatedly apply the propagation rules until no more new heap relations arise from atomic heap relations. After discarding atomic heap relations for non-terminal heaps, we obtain a set of graphs of heaps (with the same structure as the original graph) in which *atomic heap relations reside only for terminal heaps.* Now, in order to produce heap contradictions from atomic heap relations, we need to inspect only terminal heaps of these graphs, which makes it much easier to develop a complete procedure for producing heap contradictions.

3.3.4. *Rules for normalizing heap relations.* The rules for normalizing heap relations, or normalization rules, merge two identical heaps and isolate empty heaps while simultaneously shrinking the graph of heaps. In the rule NormEq, heaps w and w' are identical and we merge the two heaps by combining their heap sequents. Here we write $[w'/w]\Sigma$ for substituting w' for w in every heap relation in Σ. We also write $[w'/w]\Pi$ for merging a heap sequent for w into a heap sequent for w':

$$[w'/w](\Pi', [\Gamma \Longrightarrow \Delta]^w, [\Gamma' \Longrightarrow \Delta']^{w'}) \quad = \quad \Pi', [\Gamma, \Gamma' \Longrightarrow \Delta, \Delta']^{w'}$$

As a special case, if Π contains a heap sequent for w but not for w', we rename w to w'. Note that the rule NormEq implies that $\circ$ is (partial) deterministic. In the rule NormPC, $v \doteq \epsilon$ implies that heaps w and u are identical. Hence we merge the two heaps by combining their heap sequents and isolate the empty heap v from the graph of heaps. Similarly the rule $\mathsf{NormEmpty}$ merges two empty heaps w and u by combining their heap sequents. In effect, it allows us to collect all empty heaps, which do not need to be distinguished for the purpose of proof search, into a single empty heap. Note that the rule $\mathsf{NormEmpty}$ implies the existence of a single unit of $\circ$. By repeatedly applying the normalization rules to a graph of heaps, we can eventually obtain an equivalent graph which maintains a unique world sequent for each heap and possibly a unique empty heap isolated from the graph.

It is important that the normalization rules shrink the graph of heaps, but preserve all the properties established by the previous structural rules. For example, if the graph satisfies the property that all terminal heaps are disjoint (established by the rule $\mathsf{Disj}\star$), it continues to satisfy the same property after an application of any normalization rule. Hence it is safe to aggressively apply the normalization rules after applying the previous structural rules.

3.3.5. *Rules for dealing with empty heaps.* The last group of structural rules create an empty heap and apply the monoid laws for empty heaps. We use the rule ENew when no rule can directly produce an empty heap. The rule EJoin, which is based on neutrality of ϵ, is sound because extending a heap with an empty heap makes no change. The rule ECancel creates an empty heap when a heap is shown to be a child heap of itself. It is based on cancellativity of $\circ$: we can always generate $w \doteq w \circ w_\epsilon$ and $w_\epsilon \doteq \epsilon$ by the rules ENew and EJoin, and $w \doteq w \circ u$ and $w \doteq w \circ w_\epsilon$ imply $u \doteq \epsilon$ by cancellativity of $\circ$. (Similarly the rule NormPC is based on cancellativity of $\circ$: we can always generate $w \doteq w \circ v$ by the rule EJoin, and $w \doteq u \circ v$ and $w \doteq w \circ v$ imply $w = u$.) It turns out that we need the rule ECancel for the proof of admissibility of cut (Theorem 5.1). On the other hand, the rule ECancel is unnecessary for $\mathcal{SS}$, which searches only for such world sequents that do not contain heap relations of the form $w \doteq w \circ u$.

Now we can, to some extent, achieve a high degree of completeness of $\mathbf{P_{SL}}$. For a given heap w, in order to enumerate as many heap relations of the form $w \doteq w_1 \circ w_2$ and $w_2 \doteq w \circ w_1$ as possible, we first analyze the graph of heaps obtained by repeatedly applying the previous structural rules. This produces all heap relations that involve only non-empty heaps initially present in the graph. Then we apply the rule EJoin as necessary to produce all other heap relations that involve empty heaps.

Note, however, that applying the structural rules in $\mathbf{P_{SL}}$ does not necessarily produce all possible heap relations involving a given heap w. In separation logic, due to the definition of a heap, we can 1) extend an arbitrary heap with a fresh non-empty heap and 2) divide an arbitrary non-empty heap into two disjoint heaps one of which is a singleton heap. In $\mathbf{P_{SL}}$, on the other hand, we cannot produce such heap relations that correspond to 1) or 2).

We may think of the rule EJoin as extending heap relations for heap w with a pair of child heaps w and w_ϵ, or a pair of sibling heap w_ϵ and parent heap w. It is the only rule in $\mathbf{P_{SL}}$ that is capable of creating new heap relations for an arbitrary heap. Thus, whenever an arbitrary heap with no heap relation needs a pair of child heaps or a pair of sibling and parent heaps, we should apply the rule EJoin which inevitably reuses an existing empty heap. For example, we prove the validity of $\top \star \top$ as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdot; w_\epsilon \doteq \epsilon, w \doteq w \circ w_\epsilon \parallel [\bot \Longrightarrow \top \star \top]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\bot\mathsf{L}}{\cdot; w_\epsilon \doteq \epsilon, w \doteq w \circ w_\epsilon \parallel [\cdot \Longrightarrow \top \star \top, \top]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\neg\mathsf{R} \quad \vdots}{\cdot; w_\epsilon \doteq \epsilon, w \doteq w \circ w_\epsilon \parallel [\cdot \Longrightarrow \top \star \top]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\star\mathsf{R}}{\cdot; w_\epsilon \doteq \epsilon \parallel [\cdot \Longrightarrow \top \star \top]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\mathsf{EJoin}}{\cdot; \cdot \parallel [\cdot \Longrightarrow \top \star \top]^w}\,\mathsf{ENew}$$

Note that there is no need to create fresh child heaps w_1 and w_2 with $w \doteq w_1 \circ w_2$: if we can prove the world sequent using fresh child heaps about which nothing is known, we should be able to prove it equally by reusing an existing empty heap. Similarly we prove the validity of $\neg(\top \mathrel{-\!\!\star} \bot)$ as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots \quad \dfrac{}{\cdot; w_\epsilon \doteq \epsilon, w \doteq w \circ w_\epsilon \parallel [\top \mathrel{-\!\!\star} \bot, \bot \Longrightarrow \cdot]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\bot\mathsf{L}}{\cdot; w_\epsilon \doteq \epsilon, w \doteq w \circ w_\epsilon \parallel [\top \mathrel{-\!\!\star} \bot \Longrightarrow \cdot]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\mathrel{-\!\!\star}\mathsf{L}}{\cdot; w_\epsilon \doteq \epsilon \parallel [\top \mathrel{-\!\!\star} \bot \Longrightarrow \cdot]^w, [\cdot \Longrightarrow \cdot]^{w_\epsilon}}\,\mathsf{EJoin}}{\cdot; \cdot \parallel [\top \mathrel{-\!\!\star} \bot \Longrightarrow \cdot]^w}\,\mathsf{ENew}}{\cdot; \cdot \parallel [\cdot \Longrightarrow \neg(\top \mathrel{-\!\!\star} \bot)]^w}\,\neg\mathsf{R}$$

$$\frac{}{\Theta; \Sigma, w \doteq \epsilon, w \doteq [l \mapsto E] \parallel \Pi}\ \mathsf{Cont}\,\epsilon\!\mapsto \qquad \frac{}{\Theta; \Sigma, w \doteq \epsilon, w \neq \epsilon \parallel \Pi}\ \mathsf{Cont}\,\epsilon\!\not\doteq$$

$$\frac{\Theta, l = l', E = E'; \Sigma, w \doteq [l \mapsto E], w \doteq [l' \mapsto E'] \parallel \Pi}{\Theta; \Sigma, w \doteq [l \mapsto E], w \doteq [l' \mapsto E'] \parallel \Pi}\ \mathsf{Cont}\!\mapsto\!\doteq$$

$$\frac{\begin{array}{c}\Theta, l \neq l'; \Sigma, w \doteq [l \mapsto E], w \neq [l' \mapsto E'] \parallel \Pi \\ \Theta, E \neq E'; \Sigma, w \doteq [l \mapsto E], w \neq [l' \mapsto E'] \parallel \Pi\end{array}}{\Theta; \Sigma, w \doteq [l \mapsto E], w \neq [l' \mapsto E'] \parallel \Pi}\ \mathsf{Cont}\!\mapsto\!\not\doteq$$

$$\frac{\Theta, l_1 \neq l_2; \Sigma, w \doteq w_1 \circ w_2, w_1 \doteq [l_1 \mapsto E_1], w_2 \doteq [l_2 \mapsto E_2] \parallel \Pi}{\Theta; \Sigma, w \doteq w_1 \circ w_2, w_1 \doteq [l_1 \mapsto E_1], w_2 \doteq [l_2 \mapsto E_2] \parallel \Pi}\ \mathsf{Cont}\circ\!\mapsto$$

$$\frac{}{\Theta; \Sigma, w \doteq u \circ u, u \doteq [l \mapsto E] \parallel \Pi}\ \mathsf{Cont}\circ\!\mapsto\!2$$

Figure 3: Heap contradiction rules in the proof system $\mathbf{P_{SL}}$ for separation logic

Again we do not create fresh sibling and parent heaps and instead reuse an existing empty heap.

3.4. **Heap contradiction rules of $\mathbf{P_{SL}}$.** The proof system $\mathbf{P_{SL}}$ has six rules, $\mathsf{Cont}\,\epsilon\!\not\doteq$ to $\mathsf{Cont}\circ\!\mapsto\!2$, for producing heap contradictions (Figure 3). In conjunction with the structural rules, these rules enable us to detect all types of heap contradictions in any world sequent only with empty heap sequents.

To see why, assume a world sequent $\Theta; \Sigma \parallel \Pi$ only with empty heap sequents. By repeatedly applying the structural rules in the same order as presented in Section 3.3, we can obtain a semantically equivalent set of world sequents $\Theta_i; \Sigma_i \parallel \Pi_i$ $(i = 1, \cdots, n)$ such that: 1) Σ_i induces a graph of heaps in which all terminal heaps are disjoint; 2) atomic heap relations reside only for terminal heaps and we need to consider only terminal heaps to detect heap contradictions.

- For an empty heap, we use the rules $\mathsf{Cont}\,\epsilon\!\not\doteq$ and $\mathsf{Cont}\,\epsilon\!\mapsto$ which describe the only way to produce heap contradictions from an empty heap with $w \doteq \epsilon$. Note that $w \doteq \epsilon$ and $w \neq [l \mapsto E]$ do not produce a heap contradiction because the former implies the latter.
- For a terminal singleton heap, we use the rules $\mathsf{Cont}\!\mapsto\!\doteq$ and $\mathsf{Cont}\!\mapsto\!\not\doteq$ which describe the only way to extract expression relations from a terminal singleton heap with $w \doteq [l \mapsto E]$. Note that: 1) $w \doteq [l \mapsto E]$ and $w \doteq [l' \mapsto E']$ imply $l = l'$ and $E = E'$; 2) $w \doteq [l \mapsto E]$ and $w \neq [l' \mapsto E']$ imply $l \neq l'$ or $E \neq E'$; and 3) $w \doteq [l \mapsto E]$ implies $w \neq \epsilon$. We do not need to consider other forms of terminal heaps, for example, those with no atomic heap relations.
- Finally the rules $\mathsf{Cont}\circ\!\mapsto$ and $\mathsf{Cont}\circ\!\mapsto\!2$ describe the only way to extract expression relations from two disjoint terminal singleton heaps and to produce heap contradictions from a singleton heap that is disjoint from itself, respectively. Note that: 1)

$w_1 \doteq [l_1 \mapsto E_1]$, $w_2 \doteq [l_2 \mapsto E_2]$, and $w \doteq w_1 \circ w_2$ imply $l_1 \neq l_2$; and 2) a singleton heap is never disjoint from itself.

In this way, we can detect all types of heap contradictions in heap relations Σ_i. We formally state the completeness of the heap contradiction rules with respect to separation logic in Section 7.2.

3.5. **Properties of $\mathbf{P_{SL}}$.** The following propagation rules are admissible:

$$\frac{\{w \not\doteq \epsilon, w \doteq w_1 \circ w_2\} \subset \Sigma \qquad \begin{array}{c}\Theta; \Sigma, w_1 \not\doteq \epsilon \parallel \Pi \\ \Theta; \Sigma, w_2 \not\doteq \epsilon \parallel \Pi\end{array}}{\Theta; \Sigma \parallel \Pi}\ \mathsf{Prop}\epsilon\not\doteq$$

$$\frac{\{w \not\doteq [l \mapsto E], w \doteq w_1 \circ w_2\} \subset \Sigma \qquad \begin{array}{c}\Theta; \Sigma, w_1 \not\doteq \epsilon, w_2 \not\doteq \epsilon \parallel \Pi \\ \Theta; \Sigma, w_1 \not\doteq [l \mapsto E], w_2 \not\doteq [l \mapsto E] \parallel \Pi\end{array}}{\Theta; \Sigma \parallel \Pi}\ \mathsf{Prop}{\mapsto}\not\doteq$$

To derive the rule $\mathsf{Prop}\epsilon\not\doteq$, we use the following relation: $w \not\doteq \epsilon$ iff. $w_1 \not\doteq \epsilon$ or $w_2 \not\doteq \epsilon$. The rule $\mathsf{Prop}{\mapsto}\not\doteq$ is based on the negation of the following relation (which we can easily check):

- $w \doteq [l \mapsto E]$ iff. 1) $w_1 \doteq [l \mapsto E]$ or $w_2 \doteq [l \mapsto E]$; and 2) $w_1 \doteq \epsilon$ or $w_2 \doteq \epsilon$.

We first show that it is safe to merge two arbitrary heap sequents:

Lemma 3.1. If $\Theta; \Sigma \parallel \Pi, [\Gamma_1 \Longrightarrow \Delta_1]^u, [\Gamma_2 \Longrightarrow \Delta_2]^v$, then $\Theta; [u/v]\Sigma \parallel \Pi, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^u$.

Intuitively the second world sequent inherits every heap relation from the first world sequent, so we should be able to prove the second by the same sequence of rules in the proof of the first or its subsequence.

Next we prove the contraction property for heap relations:

Proposition 3.2. If $\Theta; \Sigma, \sigma, \sigma \parallel \Pi$, then $\Theta; \Sigma, \sigma \parallel \Pi$.

The statement in Proposition 3.2 implies that we may apply the rules $\mathsf{Disj}\star$, Assoc, and $\mathsf{Cont}\circ{\mapsto}$ to the same heap relation σ. For the case of applying the rule $\mathsf{Disj}\star$ to the same heap relation σ (which essentially has no effect), we need the rules ENew and EJoin, which are necessary for our proof search strategy $\mathcal{SS}$ anyway. For the case of applying the rule Assoc to the same heap relation σ, we need the rule $\mathsf{ECancel}$, which, however, is unnecessary for $\mathcal{SS}$ because it searches only for such world sequents that do not contain heap relations of the form $w \doteq w \circ u$. Similarly for the rule $\mathsf{Cont}\circ{\mapsto}$, we need the rule $\mathsf{Cont}\circ{\mapsto}2$, which is unnecessary for $\mathcal{SS}$.

Finally we prove the admissibility of the rules $\mathsf{Prop}\epsilon\not\doteq$ and $\mathsf{Prop}{\mapsto}\not\doteq$:

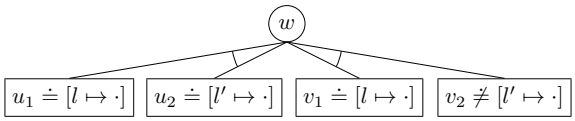
Lemma 3.3. If $\Theta; \Sigma \parallel \Pi$ using the rules $\mathsf{Prop}\epsilon\not\doteq$ and $\mathsf{Prop}{\mapsto}\not\doteq$, then $\Theta; \Sigma \parallel \Pi$.

4. Examples of proving world sequents

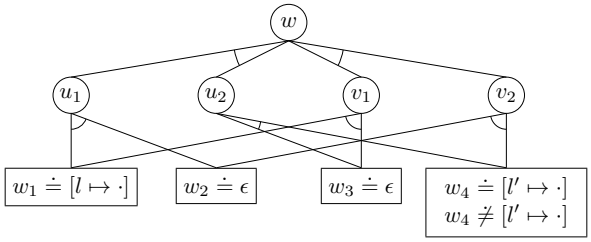
This section presents two examples of proving world sequents in $\mathbf{P_{SL}}$. We write $[l \mapsto \cdot]$ to denote $[l \mapsto E]$ for some expression E and assume two distinct location expressions l and l' ($l \neq l'$).

4.1. $\neg(([l \mapsto \cdot] \star [l' \mapsto \cdot]) \wedge ([l \mapsto \cdot] \star \neg[l' \mapsto \cdot]))$**.** The goal formula implies that given a fragment of a heap, we can uniquely determine the remaining fragment. Its proof illustrates that the rule $\mathsf{Disj}\star$ indirectly applies cancellativity of $\circ$ to two pairs of child heaps.

We begin with a world sequent $\cdot;\cdot \parallel [\cdot \Longrightarrow C]^w$ where C is the goal formula. After applying the logical rules, we obtain the following graph of heaps where heap relations are displayed for child heaps:



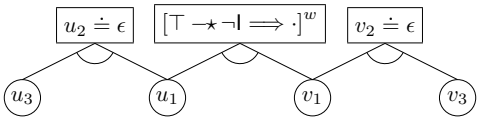
Then we apply the rule $\mathsf{Disj}\star$ and the propagation rules $\mathsf{Prop}\mapsto$ and $\mathsf{Prop}\mapsto\not\doteq$ to generate $2\times2\times2\times2 = 16$ different world sequents as new goals. All these new goals are immediately provable by the rules $\mathsf{Cont}\epsilon\mapsto$, $\mathsf{Cont}\epsilon\not\doteq$, $\mathsf{Cont}\mapsto\not\doteq$, and $\mathsf{ExpCont}$. An example of such a world sequent has heap relations $w_4 \doteq [l' \mapsto \cdot]$, originating from heap u_2 by the rule $\mathsf{Prop}\mapsto$, and $w_4 \not\doteq [l' \mapsto \cdot]$, originating from heap v_2 by the rule $\mathsf{Prop}\mapsto\not\doteq$:



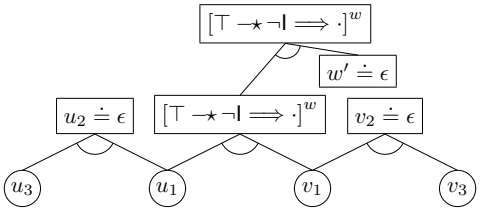
By applying the rules $\mathsf{Cont}\mapsto\not\doteq$ and $\mathsf{ExpCont}$ to heap w_4, we complete the proof.

4.2. $A \star A \supset A$ **where** $A = \neg(\top \mathrel{-\!\!\star} \neg\mathsf{I})$**.** The goal formula is valid in separation logic because heaps form a partial deterministic monoid: $H_1 \circ H_2$ may be undefined (when $H_1 \# H_2$ does not hold), but if it is defined, the result is unique. In contrast, the same formula is not valid in Boolean BI, the underlying theory of separation logic, which assumes a non-deterministic monoid [18].

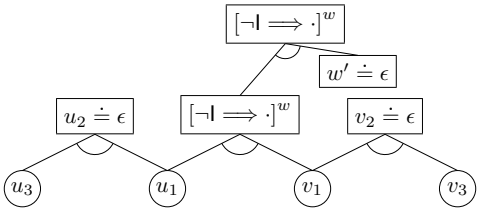
The proof illustrates the use of the rule EJoin in proving a non-trivial formula. After applying the logical rules to a world sequent $\cdot;\cdot \parallel [\cdot \Longrightarrow A \star A \supset A]^w$, we obtain the following graph of heaps:



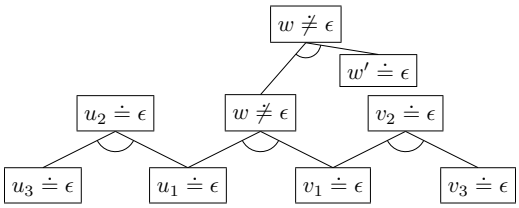
Since heap w has no sibling and parent heaps, we cannot apply the rule $\mathsf{-\!\!\star L}$ to $\top \mathrel{-\!\!\star} \neg\mathsf{I}$ at this point. To make further progress, we apply the rule EJoin after creating an empty heap, which gives us the following graph:



An application of the rule $-\!\!\star\mathsf{L}$ to $\top -\!\!\star \neg\mathsf{I}$ at heap w generates two new goals, and the interesting case produces $\neg\mathsf{I}$ as a true formula at the same heap (where we omit $\top -\!\!\star \neg\mathsf{I}$):



By applying the logical rules to heap w and the propagation rule $\mathsf{Prop}\epsilon$, we obtain the following graph:



Now we can either apply the propagation rule $\mathsf{Prop}\epsilon\not\doteq$ to heap w or use the rule NormPC to complete the proof.

5. Admissibility of cut

We state the admissibility of cut in $\mathbf{P_{SL}}$ as follows:

Theorem 5.1 (Admissibility of cut)**.**
If $\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, C]^w$ and $\Theta; \Sigma \parallel \Pi, [\Gamma, C \Longrightarrow \Delta]^w$, then $\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w$.

Theorem 5.1 assumes a few properties, such as weakening and contraction, of the expression contradiction judgment $\Theta \vdash \bot$ (for which we do not provide inference rules). In particular, we assume its own admissibility of cut: $\Theta_1, \theta \vdash \bot$ and $\Theta_2, \neg\theta \vdash \bot$ imply $\Theta_1, \Theta_2 \vdash \bot$ where $\neg\theta$ denotes the negation of θ.

To prove Theorem 5.1, we generalize its statement as follows:

Proposition 5.2. If $\Theta_1; \Sigma_1 \parallel \Pi_1, [\Gamma_1 \Longrightarrow \Delta_1, C]^w$ and $\Theta_2; \Sigma_2 \parallel \Pi_2, [\Gamma_2, C \Longrightarrow \Delta_2]^w$, then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$.

Here $\Pi_1 \uplus \Pi_2$ denotes the result of combining heap sequents for the same heap variable. In conjunction with the contraction property for formulas, Proposition 5.2 implies Theorem 5.1.

6. Soundness of $\mathbf{P_{SL}}$

This section first proves the soundness of the proof system $\mathbf{P_{SL}}$ with respect to separation logic, and then explains that $\mathbf{P_{SL}}$ is not complete with respect to separation logic. From this section, metavariable W denotes world sequents and heap variables directly refer to heaps.

The soundness property states that a derivation of a world sequent means that its semantic interpretation is self-contradictory. As a special case, we obtain Theorem 6.1:

Theorem 6.1 (Soundness)**.** If $\cdot; \cdot \parallel [\cdot \Longrightarrow A]^w$, then $\models A$.

The key step in the proof of soundness is to show that in any inference rule of $\mathbf{P_{SL}}$, the world sequent in the conclusion is either self-contradictory in itself or semantically implies the disjunction of all world sequents in the premise. Given a stack S, let us write $[\![W]\!]_S$ for the interpretation of world sequent W according to the semantics of separation logic (which is formally defined below). We wish to prove that a derivation of W implies $\neg[\![W]\!]_S$, *i.e.*, $[\![W]\!]_S$ is self-contradictory, for any stack S. Suppose that the last inference rule in the derivation of W is not an axiom and has world sequents $W_1, \cdots, W_n$ in its premise ($n \geq 1$). By induction hypothesis, we have $\neg[\![W_1]\!]_S, \cdots, \neg[\![W_n]\!]_S$, or equivalently, $\bigwedge_{i=1,\cdots,n} \neg[\![W_i]\!]_S$. Then, by proving that $[\![W]\!]_S$ implies $\bigvee_{i=1,\cdots,n} [\![W_i]\!]_S$, we prove that $\bigwedge_{i=1,\cdots,n} \neg[\![W_i]\!]_S$ implies $\neg[\![W]\!]_S$. Now $\neg[\![W]\!]_S$ immediately follows.

Formally we define $[\![W]\!]_S$ using three auxiliary semantic functions $[\![\theta]\!]_S$, $[\![\sigma]\!]_S$, and $[\![\pi]\!]_S$, all of which follow our intuition on world sequents given in Section 3.1:

$$\begin{array}{rcl}
[\![E = E']\!]_S & = & [\![E]\!]_S = [\![E']\!]_S \\
[\![E \neq E']\!]_S & = & [\![E]\!]_S \neq [\![E']\!]_S \\
[\![w \doteq \epsilon]\!]_S & = & w = \epsilon \\
[\![w \not\doteq \epsilon]\!]_S & = & w \neq \epsilon \\
[\![w \doteq [l \mapsto E]]\!]_S & = & w = \langle [\![l]\!]_S \mapsto [\![E]\!]_S \rangle \\
[\![w \not\doteq [l \mapsto E]]\!]_S & = & w \neq \langle [\![l]\!]_S \mapsto [\![E]\!]_S \rangle \\
[\![w \doteq w_1 \circ w_2]\!]_S & = & w = w_1 \circ w_2 \\
[\![[\Gamma \Longrightarrow \Delta]^w]\!]_S & = & \bigwedge_{A \in \Gamma} (S, w) \models A \wedge \bigwedge_{B \in \Delta} (S, w) \not\models B \\
[\![\Theta; \Sigma \parallel \Pi]\!]_S & = & \bigwedge_{\theta \in \Theta} [\![\theta]\!]_S \wedge \bigwedge_{\sigma \in \Sigma} [\![\sigma]\!]_S \wedge \bigwedge_{\pi \in \Pi} [\![\pi]\!]_S
\end{array}$$

Now we prove the key step in the proof of soundness:

Lemma 6.2. For every inference rule with the conclusion W and the premise consisting of $W_1, \cdots, W_n$, it holds that $[\![W]\!]_S$ implies $\bigvee_{i=1,\cdots,n} [\![W_i]\!]_S$ for any stack S. If $n = 0$, we have $\neg[\![W]\!]_S$.

As a corollary, we prove that a derivation of a world sequent means that its semantic interpretation is self-contradictory.

Corollary 6.3. If there is a derivation of a world sequent W in $\mathbf{P_{SL}}$, then $\neg[\![W]\!]_S$ holds for any stack S. For the rule $\mathsf{ExpCont}$, we assume that $\Theta \vdash \bot$ implies $\neg[\![\Theta \vdash \bot]\!]_S$.

Then a derivation of $\cdot; \cdot \parallel [\cdot \Longrightarrow A]^w$ implies $(S, w) \models A$:

$$\neg[\![\cdot; \cdot \parallel [\cdot \Longrightarrow A]^w]\!]_S \;=\; \neg(S, w) \not\models A \;=\; (S, w) \models A$$

Since w denotes an arbitrary heap, we have $\models A$ and Theorem 6.1 follows.

Although $\mathbf{P_{SL}}$ is sound with respect to separation logic, it is not complete. That is, a valid formula in separation logic may not have a proof of its negation in $\mathbf{P_{SL}}$: $\models A$ does not always imply $\cdot; \cdot \parallel [\cdot \Longrightarrow A]^w$. Below we illustrate a few properties of $\mathbf{P_{SL}}$ with such formulas.

First, in $\mathbf{P_{SL}}$, we cannot assume the existence of a non-empty heap or an arbitrary singleton heap outside a given heap. Consider the following formulas:

$$\neg(\neg \mathsf{I} \mathrel{-\!\!\star} \mathsf{I}) \tag{6.1}$$

$$\mathsf{I} \supset \neg([l \mapsto E] \mathrel{-\!\!\star} \neg[l \mapsto E]) \tag{6.2}$$

Formula 6.1 states that any heap can be merged with a non-empty heap, and Formula 6.2 states that an empty heap can be merged with an arbitrary singleton heap. Thus these formulas, all valid in separation logic, essentially state that there always exist a non-empty

heap and an arbitrary singleton heap. They are, however, unprovable in $\mathbf{P_{SL}}$, which lacks the rule capable of creating a non-empty heap or an arbitrary singleton heap then associating it with other heaps. In contrast, the rules ENew and EJoin allow us to create an empty heap and associate it with other heaps:

Moreover, in $\mathbf{P_{SL}}$, we cannot assume the existence of a *prime heap* inside a given non-empty heap, where a heap is prime if and only if it is not empty and cannot be divided into two smaller non-empty heaps. Consider the following formula:

$$\neg \mathsf{I} \supset \Big(\big(\neg \mathsf{I} \wedge \neg (\neg \mathsf{I} \star \neg \mathsf{I}) \big) \star \top \Big) \tag{6.3}$$

Formula 6.3 states that any non-empty heap must contain a prime heap. In separation logic, this formula is valid since any non-empty heap contains a singleton heap, which is prime. It is, however, unprovable in $\mathbf{P_{SL}}$, which lacks the rule capable of creating a singleton heap inside a non-empty heap.

One way to recover the completeness of $\mathbf{P_{SL}}$ with respect to separation logic is to introduce additional sound rules. The following rules are examples of such sound rules:

$$\frac{\mathit{fresh}\ w_1, w_2 \qquad \Theta; \Sigma, w_2 \doteq w \circ w_1, w_1 \not\doteq \epsilon \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_1}, [\cdot \Longrightarrow \cdot]^{w_2}}{\Theta; \Sigma \parallel \Pi}\ \mathsf{R1}$$

$$\frac{\begin{array}{ll} \mathit{fresh}\ w_1, w_2, x & \Theta; \Sigma, w \doteq w_1 \circ w_2, w_1 \doteq [l \mapsto x] \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_1}, [\cdot \Longrightarrow \cdot]^{w_2} \\ \mathit{fresh}\ w_1, w_2 & \Theta; \Sigma, w_2 \doteq w \circ w_1, w_1 \doteq [l \mapsto E] \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_1}, [\cdot \Longrightarrow \cdot]^{w_2} \end{array}}{\Theta; \Sigma \parallel \Pi}\ \mathsf{R2}$$

$$\frac{w \not\doteq \epsilon \in \Sigma \quad \mathit{fresh}\ x, y \quad \Theta; \Sigma, w \doteq w_1 \circ w_2, w_1 \doteq [x \mapsto y] \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_1}, [\cdot \Longrightarrow \cdot]^{w_2}}{\Theta; \Sigma \parallel \Pi}\ \mathsf{R3}$$

The rules $\mathsf{R1}$, $\mathsf{R2}$, and $\mathsf{R3}$ are somewhat extra-logical but still sound because they are based on the following facts in separation logic, respectively: 1) any heap can be merged with a non-empty heap; 2) any location l exists either inside or outside a given heap; 3) any non-empty heap contains a singleton heap. Note that these rules are not admissible in $\mathbf{P_{SL}}$ because Formulas 6.1, 6.2, and 6.3 become provable in $\mathbf{P_{SL}} \cup \{\mathsf{R}i\}$ for $i = 1, 2, 3$, respectively.

As they do not analyze a given world sequent at all, the rules $\mathsf{R1}$ and $\mathsf{R2}$ preserve the admissibility of cut of $\mathbf{P_{SL}}$. On the other hand, the rule $\mathsf{R3}$ does analyze a given world sequent and destroys the admissibility of cut. For example, in $\mathbf{P_{SL}} \cup \{\mathsf{R3}\}$, both world sequents $\cdot; \cdot \parallel \Big[\neg \mathsf{I} \Longrightarrow \big(\neg \mathsf{I} \wedge \neg (\neg \mathsf{I} \star \neg \mathsf{I}) \big) \star \top \Big]^w$ and $\cdot; \cdot \parallel [\mathsf{I}, \neg \mathsf{I} \mathrel{-\!\!\star} \mathsf{I} \Longrightarrow \cdot]^w, [\mathsf{I} \Longrightarrow \neg \mathsf{I} \mathrel{-\!\!\star} \mathsf{I}]^u$ have derivations, but the combined world sequent $\cdot; \cdot \parallel \Big[\neg \mathsf{I} \mathrel{-\!\!\star} \mathsf{I} \Longrightarrow \big(\neg \mathsf{I} \wedge \neg (\neg \mathsf{I} \star \neg \mathsf{I}) \big) \star \top \Big]^w, [\mathsf{I} \Longrightarrow \neg \mathsf{I} \mathrel{-\!\!\star} \mathsf{I}]^u$ does not.

7. Proof search strategy $\mathcal{SS}$ for $\mathbf{P_{SL}}$

This section presents a proof search strategy $\mathcal{SS}$ for $\mathbf{P_{SL}}$, which always terminates and is sound but incomplete with respect to $\mathbf{P_{SL}}$. We first introduce preliminaries necessary to explain $\mathcal{SS}$, and then present the details and incompleteness of $\mathcal{SS}$. The proof search strategy $\mathcal{SS}$ exploits the two propagation rules $\mathsf{Prop}\epsilon\not\doteq$ and $\mathsf{Prop}\mapsto\not\doteq$ (which are shown to be admissible in Section 3.5).

7.1. **Preliminaries of $\mathcal{SS}$.** In order to ensure the termination of $\mathcal{SS}$, we weaken the rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$, at the cost of its completeness with respect to $\mathbf{P_{SL}}$, by discarding their principal formula in the premise. We also introduce an explicit weakening rule (which is admissible) as a new structural rule:

$$\frac{\sigma \text{ is an atomic heap relation} \quad \Theta; \Sigma \parallel \Pi}{\Theta; \Sigma, \sigma \parallel \Pi}\ \mathsf{Weaken}$$

We use the rule Weaken to eliminate all atomic heap relations at non-terminal heaps when the propagation rules can produce no more new heap relations. As explained in Section 5, $\mathcal{SS}$ does not need the rules $\mathsf{ECancel}$ and $\mathsf{Cont} \circ\!\!\mapsto\!2$.

The design of $\mathcal{SS}$ uses two new concepts: *disjunctive derivation states* and *conjunctive proof goals*. A disjunctive derivation state Ψ for a world sequent W is a set of world sequents that constitute all the leaves in a partial derivation of W. That is, a disjunctive derivation state $\Psi = \{W_1, \cdots, W_n\}$ for a world sequent W means that there is a partial derivation of the following form:

$$\begin{array}{c} W_1 \quad \cdots \quad W_n \\ \ddots \vdots \iddots \\ \hline W \end{array}$$

We use a reduction judgment $\Psi \overset{R}{\mapsto} \Psi'$ to mean that such a partial derivation expands to another partial derivation with disjunctive derivation state Ψ' by an application of a logical or structural rule R to some world sequent W_i $(1 \le i \le n)$. That is, we have $\Psi' = \Psi - \{W_i\} \cup \{W_i^1, \cdots, W_i^m\}$ with:

$$\begin{array}{c} W_1 \quad \cdots \quad \dfrac{\begin{array}{c} W_i^1 \quad \cdots \quad W_i^m \\ \hline W_i \end{array} R \quad \cdots \quad W_n}{\begin{array}{c} \ddots \vdots \iddots \\ \hline W \end{array}} \end{array}$$

We write $\Psi \mapsto^* \Psi'$ for the reflexive and transitive closure of $\mapsto$.

A conjunctive proof goal Ω is a set of disjunctive derivation states for a common world sequent, and represents a set of partial derivations that have been found out by some proof search strategy for $\mathbf{P_{SL}}$ until some point. Given a logical or structural rule R, we use a reduction judgment $\Omega \overset{R}{\rightsquigarrow} \Omega'$ to mean that we can generate Ω' by applying the rule R to some disjunctive derivation state Ψ in Ω. That is, we have $\Omega' = \Omega - \{\Psi\} \cup \{\Psi'_1, \cdots, \Psi'_n\}$ and $\Psi \overset{R}{\mapsto} \Psi'_i$ for $i = 1, \cdots, n$. If R is the rule $\star\mathsf{R}$ or $\mathrel{-\!\!\star}\mathsf{L}$, we have $n \ge 1$ and produce each Ψ'_i by focusing on the same formula in the same heap sequent in the same world sequent in Ψ. For all the other rules, we have $n = 1$ and replace Ψ by Ψ'_1. We write $\Omega \rightsquigarrow^* \Omega'$ for the reflexive and transitive closure of $\rightsquigarrow$.

By the definition of $\Omega \rightsquigarrow^* \Omega'$, formulating a proof search strategy for $\mathbf{P_{SL}}$ only needs to explain how to construct a reduction sequence of conjunctive proof goals. That is, in order to explain how $\mathcal{SS}$ searches for a derivation of a world sequent $W = \cdot; \cdot \parallel [\cdot \Longrightarrow A]^w$, we only need to demonstrate how to construct conjunctive proof goals $\Omega_1, \cdots, \Omega_n$ such that $\{\{W\}\} \rightsquigarrow^* \Omega_1 \rightsquigarrow^* \cdots \rightsquigarrow^* \Omega_n$.

In order to concisely describe properties of graphs of heaps, we introduce several notations:

- $w \nearrow u$ means that there is a sequence of zero or more child-parent relations from heap w to heap u in the graph: $w = w_0$, $w_1 \doteq w_0 \circ w'_0$, $\cdots$, $w_n \doteq w_{n-1} \circ w'_{n-1}$, and $w_n = u$ for $n \geq 0$. Hence, if $w \neq u$, heap w is a descendant of heap u, or equivalently, heap u is an ancestor of heap w. Note that we allow $w \nearrow w$.
- $w\downarrow$ means that w is a root heap, *i.e.*, there is no heap relation $u \doteq w \circ v$.
- $w\uparrow$ means that w is a terminal heap, *i.e.*, there is no heap relation $w \doteq u \circ v$.
- $T(w)$ denotes the set of terminal descendants of heap w, *i.e.*, $T(w) = \{v \mid v\uparrow \text{ and } v \nearrow w\}$.

We assume that heap relations in every world sequent induce not only a graph of heaps but also a unique empty heap w_ϵ (with heap relation $w_\epsilon \doteq \epsilon$) that is separate from the graph. This assumption is safe because we can always generate such a unique empty heap with the rule ENew if there is none, and combine multiple empty heaps with the rule $\mathsf{NormEmpty}$ if there are many. We classify world sequents according to the property of graphs of heaps induced by their heap relations (without considering its special empty heap w_ϵ) as follows:

1. Well-formed: if $w \doteq w_1 \circ w_2$, then w, w_1, and w_2 are all distinct.
2. Non-cyclic: $\nearrow$ is a partial ordering on heaps.
3. Elementary: well-formed, non-cyclic, and if $w \doteq w_1 \circ w_2$, then $T(w_1) \cap T(w_2) = \varnothing$.
4. Consistent: elementary, and if $w \doteq u_1 \circ u_2$ and $w \doteq v_1 \circ v_2$, then $T(u_1) \cup T(u_2) = T(v_1) \cup T(v_2)$.
5. Full: consistent, and for any root heap u and any non-empty set $S \subset T(u)$, there exists at least one heap w with $T(w) = S$.
6. $\star$-ready for heap w: full, and for any pair of non-empty sets $S_1, S_2 \subset T(w)$ such that $S_1 \cap S_2 = \varnothing$ and $S_1 \cup S_2 = T(w)$, there exist heaps w_1 and w_2 such that $w \doteq w_1 \circ w_2$ with $T(w_1) = S_1$ and $T(w_2) = S_2$.
7. $-\!\!\star$-ready for heap w: full, and for any root heap u with $w \nearrow u$ and any pair of non-empty sets $S_1, S_2 \subset T(u)$ such that $T(w) \cap S_1 = \varnothing$ and $T(w) \cup S_1 = S_2$, there exist heaps w_1 and w_2 such that $w_2 \doteq w \circ w_1$ with $T(w_1) = S_1$ and $T(w_2) = S_2$.
8. Saturated: full, and applications of the propagation rules produce no more new heap relations.
9. Sanitized: full, and non-terminal heaps have no atomic heap relations.
10. Normalized: sanitized with no empty heaps, and for any root heap u and any non-empty set $S \subset T(u)$, there exists a unique heap w with $T(w) = S$.
11. Expanded: obtained by applying only the logical rules except $\star\mathsf{R}$ and $-\!\!\star\mathsf{L}$ to some consistent world sequent.

7.2. **Proof search strategy $\mathcal{SS}$.** We now explain how $\mathcal{SS}$ searches for a derivation of a world sequent $W = \cdot;\cdot \parallel [\cdot \Longrightarrow A]^w$ (see Figure 4). Let Ω be any conjunctive proof goal such that every world sequent in Ω is consistent. We describe how $\mathcal{SS}$ applies the rules in $\mathbf{P_{SL}}$ to obtain another conjunctive proof goal Ω', consisting only of consistent world sequents, with $\Omega \rightsquigarrow^* \Omega'$.

First we repeatedly apply the logical rules other than the rules $\star\mathsf{R}$, $-\!\!\star\mathsf{L}$, $\bot\mathsf{L}$, and $\mathsf{ExpCont}$ in order to obtain expanded world sequents from initial/consistent to expanded), until we cannot apply the rules anymore. Let Ω^1 be the resultant conjunctive proof goal, consisting of expanded world sequents, with $\Omega \rightsquigarrow^* \Omega^1$.

Suppose that there exist a disjunctive derivation state $\Psi^1 \in \Omega^1$, a world sequent $W^1 \in \Psi^1$, and a heap w in W^1 such that W^1 contains a false formula $A \star B$ (or a true formula $A -\!\!\star B$) about w. In this case, we first apply a series of structural rules to W^1 in order

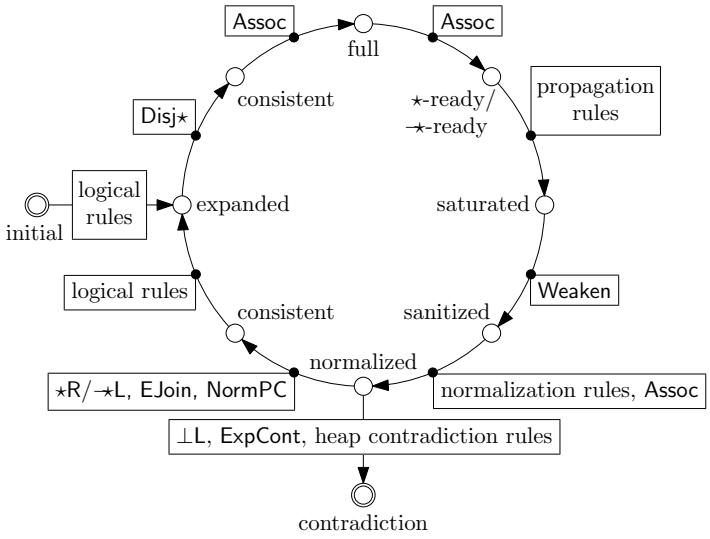


Figure 4: Proof search strategy $\mathcal{SS}$

to obtain normalized world sequents (from expanded to normalized), which yields another disjunctive derivation state Ψ_{W^1} such that: 1) $\{W^1\} \mapsto^* \Psi_{W^1}$; 2) every world sequent in Ψ_{W^1} is normalized and $\star$-ready (or $-\!\!\star$-ready) for w. We can always construct such Ψ_{W^1} because of the following lemmas and Corollary 7.3:

Lemma 7.1. For a world sequent W of a particular kind, there exists a corresponding world sequent W' of another kind such that $\{W\} \mapsto^* \{W'\}$ by applying only the structural rules, where one of the following holds:

- W is expanded and W' is consistent (11. to 4.);
- W is consistent and W' is full (4. to 5.);
- W is full and W' is $\star$-ready for a given heap w (5. to 6.);
- W is full and W' is $-\!\!\star$-ready for a given heap w (5. to 7.);
- W is saturated and $\star$-ready ($-\!\!\star$-ready) for a given heap w, and W' is sanitized and $\star$-ready ($-\!\!\star$-ready) for heap w (8. to 9.);
- W is sanitized and $\star$-ready ($-\!\!\star$-ready) for a given heap w, and W' is normalized and $\star$-ready ($-\!\!\star$-ready) for heap w (9. to 10.).

Lemma 7.2. For a world sequent W of a particular kind, there exists a disjunctive derivation state Ψ such that $\{W\} \mapsto^* \Psi$ by applying only the propagation rules, where one of the following holds:

- W is $\star$-ready for a given heap w, and every world sequent in Ψ is saturated and $\star$-ready for heap w (6. to 8.);
- W is $-\!\!\star$-ready for a given heap w, and every world sequent in Ψ is saturated and $-\!\!\star$-ready for heap w (7. to 8.).

Corollary 7.3. For any expanded world sequent W and heap w, there exists a disjunctive derivation state Ψ such that:

- $\{W\} \mapsto^* \Psi$ by applying only the structural rules;
- Ψ contains only normalized world sequents that are also $\star$-ready or $-\!\!\star$-ready for heap w.

By letting $\Psi^2 := \Psi^1 - \{W^1\} \cup \Psi_{W^1}$, we have $\Psi^1 \mapsto^* \Psi^2$. Next we choose a world sequent $W^2 \in \Psi_{W^1}$ and apply the rule $\star\mathsf{R}$ (or $\mathsf{-\!\!\star L}$) to W^2, in order to obtain consistent world sequents (from normalized to consistent), in the following way:

- Suppose that $w = w_\epsilon$ holds and W^2 contains a true formula $A \mathrel{-\!\!\star} B$ about w. Let $w_1, \cdots, w_n$ denote all heaps in W^2 (including w). After applying the rule EJoin to create heap relations $w_i \doteq w_i \circ w$ $(i = 1, \cdots, n)$, we apply the rule $\mathsf{-\!\!\star L}$ to the true formula $A \mathrel{-\!\!\star} B$ for $w_i \doteq w_i \circ w$, and then apply the rule NormPC to remove $w_i \doteq w_i \circ w$ so that we have $\{W^2\} \mapsto^* \Psi_{W^2,i}$.
- Otherwise we apply the rule $\star\mathsf{R}$ (or $\mathsf{-\!\!\star L}$) to a false formula $A \star B$ (or a true formula $A \mathrel{-\!\!\star} B$) about heap w for each heap relation of the form $w \doteq u_i \circ v_i$ in W^2 $(i = 1, \cdots, n-1)$ so that we have $\{W^2\} \mapsto \Psi_{W^2,i}$. Moreover, after applying the rule EJoin to create a heap relation $w \doteq w \circ w_\epsilon$, we apply the rule $\star\mathsf{R}$ (or $\mathsf{-\!\!\star L}$) to the same formula for $w \doteq w \circ w_\epsilon$, and then apply the rule NormPC to remove $w \doteq w \circ w_\epsilon$ so that we have $\{W^2\} \mapsto^* \Psi_{W^2,n}$.

By letting $\Psi^3_i := \Psi^2 - \{W^2\} \cup \Psi_{W^2,i}$ $(i = 1, \cdots, n)$ and $\Omega' := \Omega^1 - \{\Psi^1\} \cup \{\Psi^3_1, \cdots, \Psi^3_n\}$, we have $\Psi^2 \mapsto^* \Psi^3_i$ $(i = 1, \cdots, n)$ and thus $\Omega^1 \rightsquigarrow^* \Omega'$. Since every world sequent in Ω' is consistent, we can repeat the above process of rule applications.

Suppose now that every world sequent in Ω^1 contains no false formula $A \star B$ and no true formula $A \mathrel{-\!\!\star} B$. In this case, we first apply a series of structural rules to Ω^1 as in Corollary 7.3 in order to obtain normalized world sequents (from expanded to normalized), which yields Ω^2 satisfying that: 1) $\Omega^1 \rightsquigarrow^* \Omega^2$; 2) every world sequent in Ω^2 is normalized. As no formulas other than $\bot$ remain in Ω^2, we attempt to generate a logical contradiction by applying the rules $\bot\mathsf{L}$, $\mathsf{ExpCont}$, and the heap contradiction rules (from normalized to contradiction). After checking whether there is a logical contradiction, $\mathcal{SS}$ completes the proof search. We remark that for any normalized world sequent, there is a simple way to apply the rules $\bot\mathsf{L}$, $\mathsf{ExpCont}$, and the heap contradiction rules to the world sequent, which always terminates and is complete with respect to $\mathbf{P_{SL}}$. We also remark that the heap contradiction rules are, in fact, complete with respect to separation logic in the following sense:

Proposition 7.4 (Completeness of the heap contradiction rules)**.**

For a normalized world sequent W with no formulas other than $\bot$, if $\neg[\![W]\!]_S$ holds for any stack S, then we can construct its derivation using only the rules $\bot\mathsf{L}$, $\mathsf{ExpCont}$, and the heap contradiction rules. For the rule $\mathsf{ExpCont}$, we assume that $\neg[\![\Theta \vdash \bot]\!]_S$ implies $\Theta \vdash \bot$.

In this way, $\mathcal{SS}$ constructs a reduction sequence of conjunctive proof goals, starting from $\{\{W\}\}$. Moreover $\mathcal{SS}$ always terminates because it eventually decomposes all formulas in W other than $\bot$.

7.3. **Incompleteness of $\mathcal{SS}$.** Although $\mathcal{SS}$ is a sound proof search strategy which always terminates, it is incomplete with respect to $\mathbf{P_{SL}}$. That is, there exists some formula that has a proof in $\mathbf{P_{SL}}$ but is not provable with $\mathcal{SS}$. The following is an example among such formulas (where $l \neq l'$):

$$(\neg\mathsf{I} \star \neg\mathsf{I}) \supset (A \star A), \text{ where } A = \neg\mathsf{I} \wedge \neg([l \mapsto E] \star [l' \mapsto E]). \tag{7.1}$$

Formula 7.1 is valid in separation logic because: (1) if a given heap consists of 2 (or ≥ 4) singleton heaps, then it can be divided into two disjoint heaps, a singleton heap and a heap consisting of 1 (or ≥ 3) singleton heap(s); (2) if a given heap consists of 3 singleton heaps,

then it can be divided into two disjoint heaps, a singleton heap and a heap containing exactly 2 locations different from $\{l, l'\}$. It is easy to check that Formula 7.1 has a proof in $\mathbf{P_{SL}}$ but is not provable with $\mathcal{SS}$.

The incompleteness of $\mathcal{SS}$ with respect to $\mathbf{P_{SL}}$ is mainly due to its use of the weakened rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$. For some formula (*e.g.*, Formula 7.1), the only way to build its proof in $\mathbf{P_{SL}}$ is by applying the original rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$ more than once to the same formula. In such a case, $\mathcal{SS}$ cannot find its proof because it uses the weakened rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$ that discard their principal formula in the premise, thus forestalling repeated applications of the same rule to the same formula. If we use the original rules $\star\mathsf{R}$ and $\mathrel{-\!\!\star}\mathsf{L}$ instead, the proof search space of $\mathcal{SS}$ expands, but at the cost of its termination property.

8. Discussion

Our prototype implementation of $\mathbf{P_{SL}}$ (without first-order formulas) is based on $\mathcal{SS}$, but with a few changes. In particular, it internally uses a different type of normalized world sequents which maintain a unique heap corresponding to each non-empty set of terminal heaps, but permit unknown relations between heaps. The decision is based on the observation that it is the rule $\mathsf{Disj}\star$ (for eliminating unknown relations between heaps) that contributes the most to the complexity of graphs of heaps. Thus it selectively applies the rule $\mathsf{Disj}\star$ only when it cannot complete the proof search otherwise.

Our experience with the prototype implementation of $\mathbf{P_{SL}}$ shows that it allows us to incorporate new logical connectives and predicates in a principled way without having to introduce additional structural rules. As an example, consider an overlapping conjunction $A \owedge B$ by Hobor and Villard [14] which can be defined in the framework of $\mathbf{P_{SL}}$ as follows:

- $A \owedge B$ is true at heap w iff. $w \doteq w_1 \circ v_2$, $w \doteq v_1 \circ w_2$, $w_1 \doteq v_1 \circ u$, $w_2 \doteq u \circ v_2$, and A is true at heap w_1 and B is true at heap w_2 for *some* heaps w_1, w_2, v_1, v_2, and u.
- $A \owedge B$ is false at heap w iff. $w \doteq w_1 \circ v_2$, $w \doteq v_1 \circ w_2$, $w_1 \doteq v_1 \circ u$, and $w_2 \doteq u \circ v_2$ implies that A is false at heap w_1 or that B is false at heap w_2 for *any* heaps w_1, w_2, v_1, v_2, and u.

We directly translate this definition into two inference rules for $\owedge$:

$$\frac{\begin{array}{l}\textit{fresh } w_1, w_2, v_1, v_2, u \\ \qquad\qquad\qquad\qquad\qquad\qquad [A \Longrightarrow \cdot]^{w_1}, \\ \quad w \doteq w_1 \circ v_2, \qquad\qquad\qquad\quad [B \Longrightarrow \cdot]^{w_2}, \\ \quad w \doteq v_1 \circ w_2, \qquad\qquad\qquad\quad [\cdot \Longrightarrow \cdot]^{v_1}, \\ \quad w_1 \doteq v_1 \circ u, \qquad\qquad\qquad\quad [\cdot \Longrightarrow \cdot]^{v_2}, \\ \Theta; \Sigma, \;\; w_2 \doteq u \circ v_2 \quad \parallel \Pi, [\Gamma \Longrightarrow \Delta]^{w}, \;\; [\cdot \Longrightarrow \cdot]^{u}\end{array}}{\Theta; \Sigma \parallel \Pi, [\Gamma, A \owedge B \Longrightarrow \Delta]^{w}} \owedge\mathsf{L}$$

$$\frac{\begin{array}{lll} w \doteq w_1 \circ v_2, & & \\ w \doteq v_1 \circ w_2, & [\Gamma \Longrightarrow \Delta, A \owedge B]^{w}, & [\Gamma \Longrightarrow \Delta, A \owedge B]^{w}, \\ w_1 \doteq v_1 \circ u, & [\Gamma_1 \Longrightarrow \Delta_1, A]^{w_1}, & [\Gamma_1 \Longrightarrow \Delta_1]^{w_1}, \\ \{\, w_2 \doteq u \circ v_2 \,\} \subset \Sigma \quad \Theta; \Sigma \parallel \Pi, & [\Gamma_2 \Longrightarrow \Delta_2]^{w_2} & \Theta; \Sigma \parallel \Pi, \;\; [\Gamma_2 \Longrightarrow \Delta_2, B]^{w_2} \end{array}}{\begin{array}{l}[\Gamma \Longrightarrow \Delta, A \owedge B]^{w}, \\ \Theta; \Sigma \parallel \Pi, \;\; [\Gamma_1 \Longrightarrow \Delta_1]^{w_1}, \\ [\Gamma_2 \Longrightarrow \Delta_2]^{w_2}\end{array}} \owedge\mathsf{R}$$

Note that we obtain the rules $\uplus\mathsf{L}$ and $\uplus\mathsf{R}$ exactly in the same way that we derive the rules $\star\mathsf{L}$ and $\star\mathsf{R}$ from the interpretation of multiplicative conjunction $\star$. The only difference is that we create five fresh heaps in the rule $\uplus\mathsf{L}$ and try to detect a subgraph consisting of six existing heaps in the rule $\uplus\mathsf{R}$. Equally important is that we need no additional structural or heap contradiction rules because overlapping conjunction does not require new forms of heap relations. Thus, in principle, it is relatively easy to incorporate overlapping conjunction into our prototype implementation of $\mathbf{P_{SL}}$. Overall we may think of $\mathbf{P_{SL}}$ as a highly extensible proof system for separation logic.

9. Related work

9.1. **Automated verification tools based on separation logic.** Separation logic has been the basis for a number of automated verification tools targeting programs using mutable data structures. The first such tool is Smallfoot by Berdine *et al.* [3] which aims to test the feasibility of automated verification using separation logic. To achieve full automation, it permits no pointer arithmetic and verifies only shape properties of linked lists and trees. Space Invader by Distefano *et al.* [8] permits pointer arithmetic by integrating the abstract interpretation method into the symbolic execution method in [4]. THOR by Magill *et al.* [21] is an extension of Space Invader which is capable of tracking the length of linked lists. SLAyer by Berdine *et al.* [1] is another extension of Space Invader which uses higher-order predicates to express common properties of nodes in linked lists. The use of higher-order predicates enables SLAyer to verify shape properties of composite linked lists such as linked lists of circular linked lists.

There are also several tools supporting arbitrary data structures. HIP by Nguyen and Chin [23] allows users to specify invariants on arbitrary data structures in terms of inductive predicates. Since checking these invariants usually relies on basic properties of inductive predicates that are easy to prove but difficult to discover automatically, HIP requires users to explicitly state such properties in the form of lemmas, which are automatically proven and then applied as necessary. Similarly to HIP, VeriFast by Jacobs *et al.* [16] relies on user-supplied inductive predicates and lemmas. Unlike HIP, however, VeriFast requires users to provide proofs of these lemmas and specify when to apply them. jStar by Distefano and Parkinson [9] is an extension of Space Invader which exploits user-supplied abstraction rules in order to support arbitrary data structures. Its distinguishing feature is the ability to infer loop invariants automatically. Xisa by Chang and Rival [7] takes a different approach by indirectly specifying invariants on data structures with validation code. Xisa analyzes validation code to extract inductive predicates for describing invariants as well as lemmas for describing their basic properties. Since validation code can be written in common programming languages, users of Xisa do not need the expertise to specify invariants of interest in terms of inductive predicates.

All these tools use as their logical foundation not full separation logic but only its decidable fragment by Berdine *et al.* [2], which does not include separating implication $-\!\!\star$. As shown by Ishtiaq and O'Hearn [15], lack of separating implication implies no support for backward reasoning by weakest precondition generation for those programs performing heap assignments or allocation. As a result, these tools allow only forward reasoning based on symbolic execution as in [4] and do not demonstrate the full potential of separation logic in program verification.

9.2. **Proof search in full separation logic.** Despite the practical importance of separating implication, proof search in full separation logic has not drawn much attention from researchers. Calcagno *et al.* [6] present a translation from propositional separation logic to first-order logic (with only propositional connectives and no multiplicative connectives) for which a decision procedure already exists. The labelled tableau calculus for separation logic by Galmiche and Méry [12] supports both separating conjunction and separating implication. Similarly to our proof system $\mathbf{P_{SL}}$, their calculus combines both syntactic (tableau) and semantic (labelled) formulations and uses labels to directly refer to heaps. Although it is shown to be sound and complete, their calculus does not give rise to a proof search strategy. Specifically, in order to check that all branches in a tableau are logically or structurally inconsistent, we need two semantic functions, a measure and an interpretation, for each branch. Their calculus, however, does not explain how to construct such semantic functions for each branch and it is not clear how to extract a concrete proof search strategy.

The closest proof system to ours is the nested sequent calculus $\mathbf{S_{BBI}}$ for Boolean BI by Park *et al.* [24], which inspired the overall design of $\mathbf{P_{SL}}$. Similarly to world sequents in $\mathbf{P_{SL}}$, sequents in $\mathbf{S_{BBI}}$ use a truth context consisting of true formulas and a falsehood context consisting of false formulas, and both systems are based on the principle of proof by contradiction. Because of the similarity in syntactic formulations, their approach to dealing with separating conjunction and separating implication in $\mathbf{S_{BBI}}$ equally applies to our setting for $\mathbf{P_{SL}}$, which is not surprising considering that separation logic is just an instance of Boolean BI with additional restrictions on the semantic structure. The structural rules of $\mathbf{P_{SL}}$, however, are specific to separation logic and are designed independently of $\mathbf{S_{BBI}}$. Since $\mathbf{S_{BBI}}$ allows propositional variables, we may use its theorem prover as a supplementary system for our implementation of $\mathbf{P_{SL}}$.

For theorem provers based on the decidable fragment of separation logic by Berdine *et al.* [2] (without separating implication), see, for example, SeLoger [13] and SLP [22]. For an isomorphism between (intuitionistic) separation logic and implicit dynamic frames, see [25].

10. Conclusion

We have presented a proof system $\mathbf{P_{SL}}$ for full separation logic with separating implication. Considering the potential benefit of separating implication, we envision that program verification systems in the future will provide separating implication and support backward reasoning by weakest precondition generation for their scalability in program verification. We also envision that proof assistants can interface with theorem provers for separation logic and provide a powerful automation tactic for dealing with logical connectives from separation logic. When extended with inductively defined predicates, $\mathbf{P_{SL}}$ may serve as a practical foundation for such systems.

References

[1] Josh Berdine, Cristiano Calcagno, Byron Cook, Dino Distefano, Peter W. O'Hearn, Thomas Wies, and Hongseok Yang. Shape analysis for composite data structures. In *Proceedings of the 19th International Conference on Computer Aided Verification (CAV)*, pages 178–192, 2007.

[2] Josh Berdine, Cristiano Calcagno, and Peter W. O'Hearn. A decidable fragment of separation logic. In *Proceedings of the 24th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 97–109, 2004.

[3] Josh Berdine, Cristiano Calcagno, and Peter W. O'Hearn. Smallfoot: Modular automatic assertion checking with separation logic. In *Proceedings of the 4th International Conference on Formal Methods for Components and Objects (FMCO)*, pages 115–137, 2005.

[4] Josh Berdine, Cristiano Calcagno, and Peter W. O'Hearn. Symbolic execution with separation logic. In *Proceedings of the Third Asian Conference on Programming Languages and Systems (APLAS)*, pages 52–68, 2005.

[5] Cristiano Calcagno and Dino Distefano. Infer: an automatic program verifier for memory safety of C programs. In *Proceedings of the Third international conference on NASA Formal methods*, pages 459–465, 2011.

[6] Cristiano Calcagno, Philippa Gardner, and Matthew Hague. From separation logic to first-order logic. In *Proceedings of the 8th International Conference on Foundations of Software Science and Computation Structures (FOSSACS)*, pages 395–409, 2005.

[7] Bor-Yuh Evan Chang and Xavier Rival. Relational inductive shape analysis. In *Proceedings of the 35th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 247–260, 2008.

[8] Dino Distefano, Peter W. O'Hearn, and Hongseok Yang. A local shape analysis based on separation logic. In *Proceedings of the 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 287–302, 2006.

[9] Dino Distefano and Matthew J. Parkinson. jStar: towards practical verification for Java. In *Proceedings of the 23rd ACM SIGPLAN Conference on Object-oriented Programming Systems Languages and Applications (OOPSLA)*, pages 213–226, 2008.

[10] Robert Dockins, Aquinas Hobor, and Andrew W. Appel. A fresh look at separation algebras and share accounting. In *Proceedings of the 7th Asian Symposium on Programming Languages and Systems (APLAS)*, pages 161–177, 2009.

[11] Kamil Dudka, Petr Müller, Petr Peringer, and Tomáš Vojnar. Predator: a tool for verification of low-level list manipulation. In *Proceedings of the 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 627–629, 2013.

[12] Didier Galmiche and Daniel Méry. Tableaux and resource graphs for separation logic. *Journal of Logic and Computation*, 20:189–231, 2010.

[13] Christoph Haase, Samin Ishtiaq, Joël Ouaknine, and Matthew J. Parkinson. SeLoger: A tool for graph-based reasoning in separation logic. In *Proceedings of the 25th International Conference on Computer Aided Verification (CAV)*, pages 790–795, 2013.

[14] Aquinas Hobor and Jules Villard. The ramifications of sharing in data structures. In *Proceedings of the 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 523–536, 2013.

[15] Samin S. Ishtiaq and Peter W. O'Hearn. BI as an assertion language for mutable data structures. In *Proceedings of the 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 14–26, 2001.

[16] Bart Jacobs, Jan Smans, and Frank Piessens. VeriFast: Imperative programs as proofs. In *In VSTTE Workshop on Tools & Experiments*, pages 59–68, 2010.

[17] Neelakantan R. Krishnaswami. Reasoning about iterators with separation logic. In *Proceedings of the 2006 Conference on Specification and Verification of Component-based Systems (SAVCBS)*, pages 83–86, 2006.

[18] Dominique Larchey-Wendling and Didier Galmiche. The undecidability of boolean BI through phase semantics. In *Proceedings of the 2010 25th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 140–149, 2010.

[19] Wonyeol Lee and Sungwoo Park. A proof system for separation logic with magic wand. In *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 477–490, 2014.

[20] Toshiyuki Maeda, Haruki Sato, and Akinori Yonezawa. Extended alias type system using separating implication. In *Proceedings of the 7th ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI)*, pages 29–42, 2011.

[21] Stephen Magill, Josh Berdine, Edmund M. Clarke, and Byron Cook. Arithmetic strengthening for shape analysis. In *Proceedings of the 14th International Static Analysis Symposium (SAS)*, pages 419–436, 2007.

[22] Juan Antonio Navarro Pérez and Andrey Rybalchenko. Separation logic + superposition calculus = heap theorem prover. In *Proceedings of the 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pages 556–566, 2011.

[23] Huu Hai Nguyen and Wei-Ngan Chin. Enhancing program verification with lemmas. In *Proceedings of the 20th International Conference on Computer Aided Verification (CAV)*, pages 355–369, 2008.

[24] Jonghyun Park, Jeongbong Seo, and Sungwoo Park. A theorem prover for Boolean BI. In *Proceedings of the 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 219–232, 2013.

[25] Matthew J. Parkinson and Alexander J. Summers. The relationship between separation logic and implicit dynamic frames. In *Proceedings of the 20th European Conference on Programming Languages and Systems (ESOP)*, pages 439–458, 2011.

[26] John C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 55–74, 2002.

[27] Hongseok Yang. An example of local reasoning in BI pointer logic: the Schorr-Waite graph marking algorithm. In *Proceedings of the 1st Workshop on Semantics, Program Analysis, and Computing Environments for Memory Management*, pages 41–68, 2001.

Appendix A. Properties of $\mathbf{P_{SL}}$

A.1. **Preliminaries and weakening.** In the following sections, we assume that the expression contradiction judgment satisfies the followings:

- Weakening: $\Theta \vdash \bot$ implies $\Theta, \theta \vdash \bot$.
- Contraction: $\Theta, \theta, \theta \vdash \bot$ implies $\Theta, \theta \vdash \bot$.
- Admissibility of cut: $\Theta_1, \theta \vdash \bot$ and $\Theta_2, \neg\theta \vdash \bot$ imply $\Theta_1, \Theta_2 \vdash \bot$.
- $\Theta, E \neq E \vdash \bot$ always holds, and $\Theta, E = E \vdash \bot$ implies $\Theta \vdash \bot$.
- $\Theta, E_1 = E_2, E_1 = E_3 \vdash \bot$ implies $\Theta, E_1 = E_2, E_2 = E_3 \vdash \bot$.
  $\Theta, E_1 = E_2, E_1 \neq E_3 \vdash \bot$ implies $\Theta, E_1 = E_2, E_2 \neq E_3 \vdash \bot$.

We maintain the invariant that a world sequent contains a unique heap sequent for each heap variable.

For simplicity, let $\mathsf{ERules} := \{\mathsf{ENew}, \mathsf{EJoin}, \mathsf{ECancel}\}$ and $\mathsf{NormRules} := \{\mathsf{NormEq}, \mathsf{NormPC}, \mathsf{NormEmpty}\}$.

Let $\mathcal{D}$ be any proof in $\mathbf{P_{SL}}$. We define $\mathcal{L}(\mathcal{D})$ as the rule which is lastly applied in $\mathcal{D}$. We define $\|\mathcal{D}\|$, the size of $\mathcal{D}$, as the number of rule applications along the longest path in $\mathcal{D}$ without counting the number of applications of the rules in ERules; any rule in ERules does not increase the complexity of a proof. We define $\lfloor\mathcal{D}\rfloor$ as the number of the rule applications in $\mathcal{D}$ before we encounter the application of a rule which is not in ERules.

We define the heap sequent union operation $\Pi \uplus \Pi'$ as follows:

$$\begin{aligned}\Pi \uplus \Pi' \quad = \quad & \{[\Gamma, \Gamma' \Longrightarrow \Delta, \Delta']^w \mid [\Gamma \Longrightarrow \Delta]^w \in \Pi \text{ and } [\Gamma' \Longrightarrow \Delta']^w \in \Pi'\} \cup \\ & \{[\Gamma \Longrightarrow \Delta]^w \mid [\Gamma \Longrightarrow \Delta]^w \in \Pi \text{ or } [\Gamma \Longrightarrow \Delta]^w \in \Pi' \text{ but not both}\}\end{aligned}$$

Since the following Lemma A.1 is frequently exploited in this section, we omit its use for simplicity.

Lemma A.1.

- If R is not in $\mathsf{NormRules}$ and $\dfrac{\Theta_1; \Sigma_1 \parallel \Pi_1 \quad \cdots \quad \Theta_n; \Sigma_n \parallel \Pi_n}{\Theta; \Sigma \parallel \Pi}\ R$ holds for some $n \geq 0$,
  then for any Θ', Σ', Π' we have $\dfrac{\Theta_1, \Theta'; \Sigma_1, \Sigma' \parallel \Pi_1 \uplus \Pi' \quad \cdots \quad \Theta_n, \Theta'; \Sigma_n, \Sigma' \parallel \Pi_n \uplus \Pi'}{\Theta, \Theta'; \Sigma, \Sigma' \parallel \Pi, \Pi'}\ R$.
- If R is in $\mathsf{NormRules}$ and $\dfrac{\Theta; [u/v](\Sigma \backslash \{\sigma\}) \parallel [u/v]\Pi}{\Theta; \Sigma \parallel \Pi}\ R$ holds for some $\sigma \in \Sigma$,
  then for any Θ', Σ', Π' we have $\dfrac{\Theta, \Theta'; [u/v](\Sigma \backslash \{\sigma\}, \Sigma') \parallel [u/v](\Pi \uplus \Pi')}{\Theta, \Theta'; \Sigma, \Sigma' \parallel \Pi \uplus \Pi'}\ R$.

Proof. By case analysis of the rule R. □

Proposition A.2 (Weakening for expression relations and heap relations)**.**
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi$, then $\mathcal{E} :: \Theta, \theta; \Sigma \parallel \Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi$, then $\mathcal{E} :: \Theta; \Sigma, \sigma \parallel \Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $(\|\mathcal{D}\|, \lfloor\mathcal{D}\rfloor)$. In the proof, we use the weakening property of $\Theta \vdash \bot$. □

Proposition A.3 (Weakening)**.**
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi$, then $\mathcal{E} :: \Theta; \Sigma \parallel \Pi, \pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w$, then $\mathcal{E} :: \Theta; \Sigma \parallel \Pi, [\Gamma, A \Longrightarrow \Delta]^w$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w$, then $\mathcal{E} :: \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A]^w$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $(\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. □

A.2. **Merging lemma.**

Lemma A.4.

If R is not in $\mathsf{NormRules}$ and $\dfrac{\Theta_1; \Sigma_1 \parallel \Pi_1 \quad \cdots \quad \Theta_n; \Sigma_n \parallel \Pi_n}{\Theta; \Sigma \parallel \Pi}\, R$ $\cdots$ (*) holds for some $n \geq 0$,

then for any heaps u and v ($u \neq v$), we have $\dfrac{\Theta_1; [u/v]\Sigma_1 \parallel [u/v]\Pi_1 \quad \cdots \quad \Theta_n; [u/v]\Sigma_n \parallel [u/v]\Pi_n}{\Theta; [u/v]\Sigma \parallel [u/v]\Pi}\, R$.

Proof. By case analysis of the rule R. Most cases are immediate and non-trivial cases are as follows.

- **Case** $R \in \{\star\mathsf{R}, \mathsf{-\!\!\star L}\}$:
  In this case, we should use the special instances of the rule R. For example, when $R = \star\mathsf{R}$ and R focuses on $u \doteq v \circ w$ (where $w \neq u, v$), the proof is as follows:
  – In this case, (*) is equal to

$$\frac{u \doteq v \circ w \in \Sigma \qquad \begin{array}{c}\Theta; \Sigma \parallel \Pi', [\Gamma_1 \Longrightarrow \Delta_1, A \star B]^u, [\Gamma_2 \Longrightarrow \Delta_2, A]^v, [\Gamma \Longrightarrow \Delta]^w \\ \Theta; \Sigma \parallel \Pi', [\Gamma_1 \Longrightarrow \Delta_1, A \star B]^u, [\Gamma_2 \Longrightarrow \Delta_2]^v, [\Gamma \Longrightarrow \Delta, B]^w\end{array}}{\Theta; \Sigma \parallel \Pi', [\Gamma_1 \Longrightarrow \Delta_1, A \star B]^u, [\Gamma_2 \Longrightarrow \Delta_2]^v, [\Gamma \Longrightarrow \Delta]^w}\, \star\mathsf{R}$$

  Thus, by the special instance of the rule $\star\mathsf{R}$, we obtain

$$\frac{u \doteq u \circ w \in [u/v]\Sigma \qquad \begin{array}{c}\Theta; [u/v]\Sigma \parallel \Pi', [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2, A \star B, A]^u, [\Gamma \Longrightarrow \Delta]^w \\ \Theta; [u/v]\Sigma \parallel \Pi', [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2, A \star B]^u, [\Gamma \Longrightarrow \Delta, B]^w\end{array}}{\Theta; \Sigma \parallel \Pi', [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2, A \star B]^u, [\Gamma \Longrightarrow \Delta]^w}\, \star\mathsf{R}$$

- **Case** R is one of the structural rules and the heap contradiction rules:
  In this case, the proof is immediate by using the fact that (*) is of the following form:

$$\frac{\Sigma' \subset \Sigma \qquad \begin{array}{c}\Theta, f_1(\Sigma); \Sigma, g_1(\Sigma') \parallel \Pi, \Pi'_1 \\ \cdots \\ \Theta, f_n(\Sigma); \Sigma, g_n(\Sigma') \parallel \Pi, \Pi'_n\end{array}}{\Theta; \Sigma \parallel \Pi}\, R$$

  where $\Theta_i = \Theta, f_i(\Sigma)$, $\Sigma_i = \Sigma, g_i(\Sigma')$, and $\Pi_i = \Pi, \Pi'_i$ such that $[u/v]g_i(\Sigma') = g_i([u/v]\Sigma')$ and Π'_i is heap sequents for fresh heaps ($i = 1, \cdots, n$).

□

Lemma 3.1 (Merging lemma)**.**
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi$, then $\mathcal{E} :: \Theta; [u/v]\Sigma \parallel [u/v]\Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $(n, m) := (\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. We assume that $u \neq v$ (since the proof is trivial when $u = v$). Most cases are immediate by Lemma A.4. The only non-trivial case is when $n > 1$ and $\mathcal{L}(\mathcal{D}) \in \mathsf{NormRules}$, and we use Proposition A.2 in proving the case. For example, when $\mathcal{L}(\mathcal{D}) = \mathsf{NormPC}$ and R focuses on $\{u \doteq v \circ w, w \doteq \epsilon\}$, the proof is as follows:

- In this case, we have $\Sigma = \Sigma', u \doteq v \circ w, w \doteq \epsilon$ and
$$\mathcal{D} = \dfrac{\mathcal{D}' :: \Theta; [u/v](\Sigma', w \doteq \epsilon) \parallel [u/v]\Pi}{\Theta; \Sigma \parallel \Pi}\ \mathsf{NormPC}$$
$\mathcal{E} :: \Theta; [u/v](\Sigma', u \doteq v \circ w, w \doteq \epsilon) \parallel [u/v]\Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}'\|$ [Proposition A.2]

□

A.3. **Contraction and inversion.**

Proposition A.5 (Contraction for expression relations)**.**
If $\mathcal{D} :: \Theta, \theta, \theta; \Sigma \parallel \Pi$, then $\mathcal{E} :: \Theta, \theta; \Sigma \parallel \Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $(\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. In the proof, we use the contraction property of $\Theta \vdash \bot$.

□

Lemma A.6.
If $\mathcal{D} :: \Theta, E = E; \Sigma \parallel \Pi$, then $\mathcal{E} :: \Theta; \Sigma \parallel \Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $(\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. In the proof, we use the fact that $\Theta, E = E \vdash \bot$ implies $\Theta \vdash \bot$. □

Proposition 3.2 (Contraction for heap relations)**.**
If $\mathcal{D} :: \Theta; \Sigma, \sigma, \sigma \parallel \Pi$, then $\mathcal{E} :: \Theta; \Sigma, \sigma \parallel \Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $(n, m) := (\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. In the proof, we use Proposition A.2 and A.3, and Lemma 3.1. Most cases are immediate and non-trivial cases are when $n > 1$ and $\mathcal{L}(\mathcal{D}) \in \{\mathsf{Cont}{\mapsto}{\doteq}, \mathsf{Cont} \circ{\mapsto}, \mathsf{Disj}\star, \mathsf{Assoc}, \mathsf{NormPC}\}$. The proof for the non-trivial cases are as follows.

- **Case** $\mathcal{L}(\mathcal{D}) \in \{\mathsf{Cont}{\mapsto}{\doteq}, \mathsf{Cont} \circ{\mapsto}\}$:
In this case, we should use Lemma A.6 and the rule $\mathsf{Cont} \circ{\mapsto}2$.
- **Case** $\mathcal{L}(\mathcal{D}) = \mathsf{Disj}\star$:
The only non-trivial case is when the rule $\mathsf{Disj}\star$ focuses on $\{\sigma, \sigma\}$ (where $\sigma = w \doteq u \circ v$). In this case, we have

$$\mathcal{D} = \dfrac{\{\sigma, \sigma\} \subset \Sigma, \sigma, \sigma \quad \textit{fresh } w_1, w_2, w_3, w_4 \quad \mathcal{D}' :: \Theta; \Sigma, \sigma, \sigma, \begin{array}{l} u \doteq w_1 \circ w_2, \\ v \doteq w_3 \circ w_4, \\ u \doteq w_1 \circ w_3, \\ v \doteq w_2 \circ w_4 \end{array} \parallel \Pi, \begin{array}{l} [\cdot \Longrightarrow \cdot]^{w_1}, \\ [\cdot \Longrightarrow \cdot]^{w_2}, \\ [\cdot \Longrightarrow \cdot]^{w_3}, \\ [\cdot \Longrightarrow \cdot]^{w_4} \end{array}}{\Theta; \Sigma, \sigma, \sigma \parallel \Pi}\ \mathsf{Disj}\star$$

$$\mathcal{E}'_1 :: \Theta; \Sigma, \sigma, \sigma,\ \begin{array}{c} u \doteq u \circ w_2, \\ v \doteq w_3 \circ w_4, \end{array}\ \begin{array}{c} u \doteq u \circ w_3, \\ v \doteq w_2 \circ w_4 \end{array}\ \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_2},\ [\cdot \Longrightarrow \cdot]^{w_3},\ [\cdot \Longrightarrow \cdot]^{w_4}$$ and $\|\mathcal{E}'_1\| \leq \|\mathcal{D}'\|$

[Lemma 3.1 on u, w_1]

$$\mathcal{E}'_2 :: \Theta; \Sigma, \sigma, \sigma,\ \begin{array}{c} u \doteq u \circ w_2, \\ v \doteq w_3 \circ v, \end{array}\ \begin{array}{c} u \doteq u \circ w_3, \\ v \doteq w_2 \circ v \end{array}\ \parallel \Pi,\ [\cdot \Longrightarrow \cdot]^{w_2},\ [\cdot \Longrightarrow \cdot]^{w_3}$$ and $\|\mathcal{E}'_2\| \leq \|\mathcal{E}'_1\|$

[Lemma 3.1 on v, w_4]

$$\mathcal{E}'_3 :: \Theta; \Sigma, \sigma, \sigma, w_\epsilon \doteq \epsilon,\ \begin{array}{c} u \doteq u \circ w_2, \\ v \doteq v \circ w_3, \end{array}\ \begin{array}{c} u \doteq u \circ w_3, \\ v \doteq v \circ w_2 \end{array}\ \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_\epsilon},\ [\cdot \Longrightarrow \cdot]^{w_2},\ [\cdot \Longrightarrow \cdot]^{w_3}$$ for *fresh* w_ϵ and $\|\mathcal{E}'_3\| \leq \|\mathcal{E}'_2\|$

[Proposition A.2, A.3]

$$\mathcal{E}'_4 :: \Theta; \Sigma, \sigma, \sigma, w_\epsilon \doteq \epsilon,\ \begin{array}{c} u \doteq u \circ w_\epsilon, \\ v \doteq v \circ w_3, \end{array}\ \begin{array}{c} u \doteq u \circ w_3, \\ v \doteq v \circ w_\epsilon \end{array}\ \parallel \Pi,\ [\cdot \Longrightarrow \cdot]^{w_\epsilon},\ [\cdot \Longrightarrow \cdot]^{w_3}$$ and $\|\mathcal{E}'_4\| \leq \|\mathcal{E}'_3\|$

[Lemma 3.1 on w_ϵ, w_2]

$$\mathcal{E}'_5 :: \Theta; \Sigma, \sigma, \sigma, w_\epsilon \doteq \epsilon,\ \begin{array}{c} u \doteq u \circ w_\epsilon, \\ v \doteq v \circ w_\epsilon, \end{array}\ \begin{array}{c} u \doteq u \circ w_\epsilon, \\ v \doteq v \circ w_\epsilon \end{array}\ \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_\epsilon}$$ and $\|\mathcal{E}'_5\| \leq \|\mathcal{E}'_4\|$

[Lemma 3.1 on w_ϵ, w_3]

$\mathcal{E}'_6 :: \Theta; \Sigma, \sigma, \sigma, w_\epsilon \doteq \epsilon \parallel \Pi, [\cdot \Longrightarrow \cdot]^{w_\epsilon}$ and $\|\mathcal{E}'_6\| = \|\mathcal{E}'_5\|$ [the rule EJoin]

$\mathcal{E}'_7 :: \Theta; \Sigma, \sigma, \sigma \parallel \Pi$ and $\|\mathcal{E}'_7\| = \|\mathcal{E}'_6\|$ [the rule ENew]

$\mathcal{E} :: \Theta; \Sigma, \sigma \parallel \Pi$ and $\|\mathcal{E}\| \leq \|\mathcal{E}'_7\|$ [IH on $\mathcal{E}'_7$]

- **Case** $\mathcal{L}(\mathcal{D}) = \mathsf{Assoc}$:
  The only non-trivial case is when the rule Assoc focuses $\{\sigma, \sigma\}$ (where $\sigma = w \doteq w \circ v$) and a fresh heap created by the rule is the union of heaps v and v. In this case, we have

$$\mathcal{D} = \frac{\{\sigma, \sigma\} \subset \Sigma, \sigma, \sigma \quad \textit{fresh } u' \quad \mathcal{D}' :: \Theta; \Sigma, \sigma, \sigma, u' \doteq v \circ v, w \doteq w \circ u' \parallel \Pi, [\cdot \Longrightarrow \cdot]^{u'}}{\Theta; \Sigma, \sigma, \sigma \parallel \Pi}\ \mathsf{Assoc}$$

  $\mathcal{E}'_1 :: \Theta; \Sigma, \sigma, \sigma, v \doteq v \circ v, \sigma \parallel \Pi$ and $\|\mathcal{E}'_1\| \leq \|\mathcal{D}'\|$ [Lemma 3.1 on v, u']

  $\mathcal{E}'_2 :: \Theta; \Sigma, \sigma, \sigma, v \doteq \epsilon, v \doteq v \circ v, \sigma \parallel \Pi$ and $\|\mathcal{E}'_2\| \leq \|\mathcal{E}'_1\|$ [Proposition A.2]

  $\mathcal{E}'_3 :: \Theta; \Sigma, \sigma, \sigma, \sigma, v \doteq \epsilon \parallel \Pi$ and $\|\mathcal{E}'_3\| = \|\mathcal{E}'_2\|$ [the rule EJoin]

  $\mathcal{E}'_4 :: \Theta; \Sigma, \sigma, \sigma, v \doteq \epsilon \parallel \Pi$ and $\|\mathcal{E}'_4\| \leq \|\mathcal{E}'_3\|$ [IH on $\mathcal{E}'_3$]

  $\mathcal{E}'_5 :: \Theta; \Sigma, \sigma, v \doteq \epsilon \parallel \Pi$ and $\|\mathcal{E}'_5\| \leq \|\mathcal{E}'_4\|$ [IH on $\mathcal{E}'_4$]

  $\mathcal{E} :: \Theta; \Sigma, \sigma \parallel \Pi$ and $\|\mathcal{E}\| = \|\mathcal{E}'_5\|$ [the rule ECancel]

- **Case** $\mathcal{L}(\mathcal{D}) = \mathsf{NormPC}$:
  The only non-trivial case is when the rule NormPC focuses on σ (where $\sigma = w \doteq u \circ v$). In this case, we have $\Sigma = \Sigma', v \doteq \epsilon$ and

$$\mathcal{D} = \frac{\mathcal{D}' :: \Theta; [w/u](\Sigma', w \doteq u \circ v, v \doteq \epsilon) \parallel [w/u]\Pi}{\Theta; \Sigma, \sigma, \sigma \parallel \Pi}\ \mathsf{NormPC}$$

  Then we have

$$\mathcal{E} := \frac{\dfrac{\mathcal{D}' :: \Theta; [w/u](\Sigma', w \doteq u \circ v, v \doteq \epsilon) \parallel [w/u]\Pi}{\Theta; [w/u](\Sigma', v \doteq \epsilon) \parallel [w/u]\Pi}\ \mathsf{EJoin}}{\Theta; \Sigma, \sigma \parallel \Pi}\ \mathsf{NormPC}$$

  and $\|\mathcal{E}\| = \|\mathcal{D}\|$.

□

Proposition A.7 (Inversion)**.**

Except for the rule $\mathsf{ExpCont}$, every logical rule R of $\mathbf{P_{SL}}$ is invertible, *i.e.*, if the conclusion of the rule R has a proof $\mathcal{D}$, then every premise of the rule R has a proof $\mathcal{E}$ with $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. By induction on $\|\mathcal{D}\|$. In the proof, we use Proposition A.3. □

Proposition A.8 (Contraction)**.**

If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi, [\Gamma, A, A \Longrightarrow \Delta]^w$, then $\mathcal{E} :: \Theta; \Sigma \parallel \Pi, [\Gamma, A \Longrightarrow \Delta]^w$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A, A]^w$, then $\mathcal{E} :: \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, A]^w$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$.

Proof. We prove the two statements simultaneously by induction on $(n, m) := (\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. In the proof, we use Proposition A.5, 3.2, and A.7, and Lemma 3.1. For example, the proof of the first statement when $n > 1$, $\mathcal{L}(\mathcal{D}) = \star\mathsf{L}$, and $\mathcal{L}(\mathcal{D})$ focuses on A is as follows:

- In this case, we have $A = B \star C$ and

$$\mathcal{D} = \dfrac{\mathit{fresh}\ w_1, w_2 \quad \mathcal{D}' :: \Theta; \Sigma, w \doteq w_1 \circ w_2 \parallel \Pi, [\Gamma, B \star C \Longrightarrow \Delta]^w, [B \Longrightarrow \cdot]^{w_1}, [C \Longrightarrow \cdot]^{w_2}}{\Theta; \Sigma \parallel \Pi, [\Gamma, A, A \Longrightarrow \Delta]^w}\ \star\mathsf{L}$$

$\mathcal{E}'_1 :: \Theta; \Sigma, w \doteq w_1 \circ w_2, w \doteq w'_1 \circ w'_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w,\ [B \Longrightarrow \cdot]^{w_1},\ [B \Longrightarrow \cdot]^{w'_1},\ [C \Longrightarrow \cdot]^{w_2},\ [C \Longrightarrow \cdot]^{w'_2}$
for *fresh* w'_1, w'_2, and $\|\mathcal{E}'_1\| \leq \|\mathcal{D}'\|$ [Proposition A.7]

$\mathcal{E}'_2 :: \Theta; \Sigma, w \doteq w_1 \circ w_2, w \doteq w_1 \circ w'_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [B, B \Longrightarrow \cdot]^{w_1}, [C \Longrightarrow \cdot]^{w_2}, [C \Longrightarrow \cdot]^{w'_2}$

and $\|\mathcal{E}'_2\| \leq \|\mathcal{E}'_1\|$ [Lemma 3.1 on w_1, w'_1]
$\mathcal{E}'_3 :: \Theta; \Sigma, w \doteq w_1 \circ w_2, w \doteq w_1 \circ w_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [B, B \Longrightarrow \cdot]^{w_1}, [C, C \Longrightarrow \cdot]^{w_2}$
and $\|\mathcal{E}'_3\| \leq \|\mathcal{E}'_2\|$ [Lemma 3.1 on w_2, w'_2]
$\mathcal{E}'_4 :: \Theta; \Sigma, w \doteq w_1 \circ w_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [B, B \Longrightarrow \cdot]^{w_1}, [C, C \Longrightarrow \cdot]^{w_2}$ and
$\|\mathcal{E}'_4\| \leq \|\mathcal{E}'_3\|$ [Proposition 3.2]
$\mathcal{E}'_5 :: \Theta; \Sigma, w \doteq w_1 \circ w_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [B \Longrightarrow \cdot]^{w_1}, [C, C \Longrightarrow \cdot]^{w_2}$ and $\|\mathcal{E}'_5\| \leq \|\mathcal{E}'_4\|$ [IH on $\mathcal{E}'_4$]
$\mathcal{E}'_6 :: \Theta; \Sigma, w \doteq w_1 \circ w_2 \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w, [B \Longrightarrow \cdot]^{w_1}, [C \Longrightarrow \cdot]^{w_2}$ and $\|\mathcal{E}'_6\| \leq \|\mathcal{E}'_5\|$ [IH on $\mathcal{E}'_5$]
$\mathcal{E} :: \Theta; \Sigma \parallel \Pi, [\Gamma, B \star C \Longrightarrow \Delta]^w$ and $\|\mathcal{E}\| \leq \|\mathcal{D}\|$ [the rule $\star\mathsf{L}$]

□

A.4. **Admissibility of cut (for expression relations) and main lemma.**

Proposition A.9 (Cut for expression relations)**.**

If $\mathcal{D} :: \Theta_1, E \neq E'; \Sigma_1 \parallel \Pi_1$ and $\mathcal{E} :: \Theta_2, E = E'; \Sigma_2 \parallel \Pi_2$, then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$.

Proof. By induction on $(n_1, n_2, m_1, m_2) := (\|\mathcal{D}\|, \|\mathcal{E}\|, \lfloor \mathcal{D} \rfloor, \lfloor \mathcal{E} \rfloor)$. In the proof, we use the admissibility of cut of $\Theta \vdash \bot$. □

Lemma A.10. $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$ satisfies the following properties: 1) the weakening and contraction properties for expression relations, heap relations, and formulas; 2) the cut property for expression relations; and 3) the merging lemma.

Proof. The proof is similar to that of Proposition A.2, A.3, A.5, 3.2, A.8, and A.9, and Lemma 3.1. □

Lemma A.11.

$$\text{If we have } \begin{cases} \mathcal{D} :: \Theta_1; \Sigma_1, w_1 \not\doteq \epsilon \parallel \Pi_1 \text{ in } \mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}, \\ \mathcal{E} :: \Theta_2; \Sigma_2, w_2 \not\doteq \epsilon \parallel \Pi_2 \text{ in } \mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}, \\ \{w \not\doteq \epsilon, w \doteq w_1 \circ w_2\} \subset \Sigma_1, \text{ and} \\ \{w \not\doteq \epsilon, w \doteq w_1 \circ w_2\} \subset \Sigma_2, \end{cases}$$

then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$.

Proof. By induction on $(n_1, n_2, m_1, m_2) := (\|\mathcal{D}\|, \|\mathcal{E}\|, \lfloor\mathcal{D}\rfloor, \lfloor\mathcal{E}\rfloor)$. In the proof, we use Lemma A.10. Most cases are immediate and some of non-trivial cases are as follows.

- **Case** $n_1 = 1$, $n_2 = 1$, and $\mathcal{L}(\mathcal{D})$ (resp. $\mathcal{L}(\mathcal{E})$) focuses on $w_1 \not\doteq \epsilon$ (resp. $w_2 \not\doteq \epsilon$):
  In this case, we have $\mathcal{L}(\mathcal{D}) = \mathcal{L}(\mathcal{E}) = \mathsf{Cont}\,\epsilon \not\doteq$ and $\{w_1 \doteq \epsilon, w_2 \doteq \epsilon\} \subset \Sigma_1, \Sigma_2$. Thus, $\Sigma_1, \Sigma_2 = \Sigma', w \not\doteq \epsilon, w \doteq w_1 \circ w_2, w_1 \doteq \epsilon, w_2 \doteq \epsilon$ and we obtain
  $$\dfrac{\dfrac{}{\Theta_1, \Theta_2; [w/w_1](\Sigma', w \not\doteq \epsilon, w_1 \doteq \epsilon, w_2 \doteq \epsilon) \parallel [w/w_1](\Pi_1 \uplus \Pi_2)}\,\mathsf{Cont}\,\epsilon \not\doteq}{\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2}\,\mathsf{NormPC}$$
- **Case** $n_2 > 1$ and $\mathcal{L}(\mathcal{E}) \in \mathsf{NormRules}$:
  In this case, for some $\sigma \in \Sigma_2$ we have
  $$\mathcal{E} = \dfrac{\mathcal{E}' :: \Theta_2; [u/v](\Sigma_2 \backslash \{\sigma\}, w_2 \not\doteq \epsilon) \parallel [u/v]\Pi_2}{\Theta_2; \Sigma_2, w_2 \not\doteq \epsilon \parallel \Pi_2}\,\mathcal{L}(\mathcal{E})$$
  where $\mathcal{L}(\mathcal{E})$ does not focus on $w_2 \not\doteq \epsilon$.
  $\mathcal{E}'' :: \Theta_2; [u/v](\Sigma_2, w_2 \not\doteq \epsilon) \parallel [u/v]\Pi_2$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$ and $\|\mathcal{E}''\| \leq \|\mathcal{E}'\|$ [Lemma A.10]
  $\mathcal{D}' :: \Theta_1; [u/v](\Sigma_1, w_1 \not\doteq \epsilon) \parallel [u/v]\Pi_1$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$ and $\|\mathcal{D}'\| \leq \|\mathcal{D}\|$ [Lemma A.10 on $\mathcal{D}$ with u, v]
  $\Theta_1, \Theta_2; [u/v](\Sigma_1, \Sigma_2) \parallel [u/v](\Pi_1 \uplus \Pi_2)$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$ [IH on $\mathcal{D}'$ and $\mathcal{E}''$]
  $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2, \sigma \parallel \Pi_1 \uplus \Pi_2$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$ [the rule $\mathcal{L}(\mathcal{E})$]
  $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop} \mapsto \not\doteq\}$ [Lemma A.10]

□

Lemma A.12.

$$\text{If we have } \begin{cases} \mathcal{D} :: \Theta_1; \Sigma_1, w_1 \not\doteq [l \mapsto E], w_2 \not\doteq [l \mapsto E] \parallel \Pi_1, \\ \mathcal{E} :: \Theta_2; \Sigma_2, w_1 \not\doteq \epsilon, w_2 \not\doteq \epsilon \parallel \Pi_2, \\ \{w \not\doteq [l \mapsto E], w \doteq w_1 \circ w_2\} \subset \Sigma_1, \text{ and} \\ \{w \not\doteq [l \mapsto E], w \doteq w_1 \circ w_2\} \subset \Sigma_2, \end{cases}$$

then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$.

Proof. By induction on $(n_1, n_2, m_1, m_2) := (\|\mathcal{D}\|, \|\mathcal{E}\|, \lfloor\mathcal{D}\rfloor, \lfloor\mathcal{E}\rfloor)$. In the proof, we use Proposition A.2, A.5, 3.2, A.8, and A.9, and Lemma 3.1. Most cases are immediate and some of non-trivial cases are as follows.

- **Case** $n_1 > 1$, $n_2 = 1$, and $\mathcal{L}(\mathcal{D})$ (resp. $\mathcal{L}(\mathcal{E})$) focuses on $w_1 \neq [l \mapsto E]$ (resp. $w_2 \neq \epsilon$):
  In this case, we have $\mathcal{L}(\mathcal{D}) = \mathsf{Cont}{\mapsto}{\neq}$ (since $\mathcal{D}$ does not contain an application of the rule $\mathsf{Prop}{\mapsto}{\neq}$), $\mathcal{L}(\mathcal{E}) = \mathsf{Cont}\epsilon{\neq}$, and $\{w_1 \doteq [l' \mapsto E'], w_2 \doteq \epsilon\} \subset \Sigma_1, \Sigma_2$. Thus we have $\Sigma_1, \Sigma_2 = \Sigma', w \neq [l \mapsto E], w \doteq w_1 \circ w_2, w_1 \doteq [l' \mapsto E'], w_2 \doteq \epsilon$ and
  $$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1' :: \Theta_1, l \neq l'; \Sigma_1, w_1 \neq [l \mapsto E], w_2 \neq [l \mapsto E] \parallel \Pi_1 \\ \mathcal{D}_2' :: \Theta_1, E \neq E'; \Sigma_1, w_1 \neq [l \mapsto E], w_2 \neq [l \mapsto E] \parallel \Pi_1\end{array}}{\Theta_1; \Sigma_1, w_1 \neq [l \mapsto E], w_2 \neq [l \mapsto E] \parallel \Pi_1}\ \mathsf{Cont}{\mapsto}{\neq}$$
  Since $\Theta, E_1 = E_2, E_1 \neq E_2 \vdash \bot$ hold for any Θ, E_1, E_2 (from the assumptions of $\Theta \vdash \bot$), we obtain
  $$\mathcal{F} := \dfrac{\dfrac{\dfrac{\Theta_1, \Theta_2, l = l', E = E', l \neq l' \vdash \bot}{\Theta_1, \Theta_2, l = l', E = E', l \neq l'; \cdots \parallel \cdots}\ \mathsf{ExpCont} \quad \dfrac{\Theta_1, \Theta_2, l = l', E = E', E \neq E' \vdash \bot}{\Theta_1, \Theta_2, l = l', E = E', E \neq E'; \cdots \parallel \cdots}\ \mathsf{ExpCont}}{\Theta_1, \Theta_2, l = l', E = E'; [w/w_1](\Sigma', w \neq [l \mapsto E], w_1 \doteq [l' \mapsto E'], w_2 \doteq \epsilon) \parallel [w/w_1](\Pi_1 \uplus \Pi_2)}\ \mathsf{Cont}{\mapsto}{\neq}}{\Theta_1, \Theta_2, l = l', E = E'; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2}\ \mathsf{NormPC}$$
  $\mathcal{G}_1 :: \Theta_1, \Theta_2, l \neq l'; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [IH on $\mathcal{D}_1'$ and $\mathcal{E}$]
  $\mathcal{G}_2 :: \Theta_1, \Theta_2, E \neq E'; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [IH on $\mathcal{D}_2'$ and $\mathcal{E}$]
  $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [Proposition A.9 (with $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{F}$), A.5, 3.2, A.8]
- **Case** $n_1 > 1$ and $\mathcal{L}(\mathcal{D}) \in \mathsf{NormRules}$:
  In this case, for some $\sigma \in \Sigma_1$ we have
  $$\mathcal{D} = \dfrac{\mathcal{D}' :: \Theta_1; [u/v](\Sigma_1 \backslash \{\sigma\}, w_1 \neq [l \mapsto E], w_2 \neq [l \mapsto E]) \parallel [u/v]\Pi_1}{\Theta_1; \Sigma_1, w_1 \neq [l \mapsto E], w_2 \neq [l \mapsto E] \parallel \Pi_1}\ \mathcal{L}(\mathcal{D})$$
  where $\mathcal{L}(\mathcal{D})$ does not focus on both $w_1 \neq [l \mapsto E]$ and $w_2 \neq [l \mapsto E]$.
  $\mathcal{D}'' :: \Theta_1; [u/v](\Sigma_1, w_1 \neq [l \mapsto E], w_2 \neq [l \mapsto E]) \parallel [u/v]\Pi_1$ and $\|\mathcal{D}''\| \leq \|\mathcal{D}'\|$ [Proposition A.2]
  $\mathcal{E}' :: \Theta_2; [u/v](\Sigma_2, w_1 \neq \epsilon, w_2 \neq \epsilon) \parallel [u/v]\Pi_2$ and $\|\mathcal{E}'\| \leq \|\mathcal{E}\|$ [Lemma 3.1 on $\mathcal{E}$ with u, v]
  $\Theta_1, \Theta_2; [u/v](\Sigma_1, \Sigma_2) \parallel [u/v](\Pi_1 \uplus \Pi_2)$ [IH on $\mathcal{D}''$ and $\mathcal{E}'$]
  $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2, \sigma \parallel \Pi_1 \uplus \Pi_2$ [the rule $\mathcal{L}(\mathcal{D})$]
  $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [Proposition 3.2]

□

Lemma 3.3.
If $\Theta; \Sigma \parallel \Pi$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop}\epsilon{\neq}, \mathsf{Prop}{\mapsto}{\neq}\}$, then $\Theta; \Sigma \parallel \Pi$.

Proof. It suffices to show the following two statements separately:

- If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop}\epsilon{\neq}, \mathsf{Prop}{\mapsto}{\neq}\}$, then $\Theta; \Sigma \parallel \Pi$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop}{\mapsto}{\neq}\}$.
- If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi$ in $\mathbf{P_{SL}} \cup \{\mathsf{Prop}{\mapsto}{\neq}\}$, then $\Theta; \Sigma \parallel \Pi$.

The proof of each of the above statements proceeds by induction on $(\|\mathcal{D}\|, \lfloor \mathcal{D} \rfloor)$. In the proof, we use Lemma A.10, A.11, and A.12, and Proposition A.5, 3.2, and A.8. □

Appendix B. Admissibility of cut

Lemma B.1.
If $\mathcal{D} :: \Theta; \Sigma, w \doteq [l \mapsto E] \parallel \Pi$, then

$$\langle l', E'/l, E\rangle\Theta, l = l', E = E'; \langle [l' \mapsto E']/[l \mapsto E]\rangle\Sigma, w \doteq [l' \mapsto E'] \parallel \Pi.$$

Here we write $\langle l', E'/l, E\rangle\Theta$ for substituting l (resp. E) for l' (resp. E') in *some* expression relations in Θ; moreover we write $\langle [l' \mapsto E']/[l \mapsto E]\rangle\Sigma$ for substituting $[l \mapsto E]$ for $[l' \mapsto E']$ in *some* heap relations in Σ.

Proof. By induction on $(\|\mathcal{D}\|, \lfloor\mathcal{D}\rfloor)$. In the proof, we use the weakening property of $\Theta \vdash \bot$ and the fact that: $\Theta, E_1 = E_2, E_1 = E_3 \vdash \bot$ implies $\Theta, E_1 = E_2, E_2 = E_3 \vdash \bot$; and $\Theta, E_1 = E_2, E_1 \neq E_3 \vdash \bot$ implies $\Theta, E_1 = E_2, E_2 \neq E_3 \vdash \bot$. □

Proposition B.2 (Cut for heap relations)**.**
If $\Theta_1; \Sigma_1, w \not\doteq \epsilon \parallel \Pi_1$ and $\Theta_2; \Sigma_2, w \doteq \epsilon \parallel \Pi_2$, then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$.
If $\Theta_1; \Sigma_1, w \not\doteq [l \mapsto E] \parallel \Pi_1$ and $\Theta_2; \Sigma_2, w \doteq [l \mapsto E] \parallel \Pi_2$, then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$.

Proof. The proof of each statement proceeds by induction on $(n, m) := (\|\mathcal{D}\|, \lfloor\mathcal{D}\rfloor)$. In the proof, we use Proposition A.2, A.3, A.5, 3.2, A.8, and A.9, and Lemma B.1 which is used in proving the second statement. For example, the proof of the second statement when $n > 1$ and $\mathcal{L}(\mathcal{D})$ focuses on $w \not\doteq [l \mapsto E]$ is as follows:

- In this case, we have $\mathcal{L}(\mathcal{D}) = \mathsf{Cont}{\mapsto}{\not\doteq}$ (since $\mathcal{D}$ does not contain an application of the rule $\mathsf{Prop}{\mapsto}{\not\doteq}$), $w \doteq [l' \mapsto E'] \in \Sigma_1$, and

$$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1' :: \Theta_1, l \neq l'; \Sigma_1, w_1 \not\doteq [l \mapsto E] \parallel \Pi_1 \\ \mathcal{D}_2' :: \Theta_1, E \neq E'; \Sigma_1, w_1 \not\doteq [l \mapsto E] \parallel \Pi_1\end{array}}{\Theta_1; \Sigma_1, w_1 \not\doteq [l \mapsto E] \parallel \Pi_1}\ \mathsf{Cont}{\mapsto}{\not\doteq}$$

  $\mathcal{F}_1 :: \Theta_1, \Theta_2, l \neq l'; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [IH on $\mathcal{D}_1'$]
  $\mathcal{F}_2 :: \Theta_1, \Theta_2, E \neq E'; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [IH on $\mathcal{D}_2'$]
  $\mathcal{E}' :: \Theta_2, l = l', E = E'; \Sigma_2, w \doteq [l' \mapsto E'] \parallel \Pi_2$ [Lemma B.1]
  $\mathcal{G} :: \Theta_1, \Theta_2, l = l', E = E'; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [Proposition A.2, A.3]
  $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2$ [Proposition A.9 (with $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{G}$), A.5, 3.2, A.8]

  □

Lemma B.3.
If $\mathcal{D} :: \Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta, \bot]^w$, then $\Theta; \Sigma \parallel \Pi, [\Gamma \Longrightarrow \Delta]^w$.

Proof. By induction on $(\|\mathcal{D}\|, \lfloor D\rfloor)$. □

Proposition 5.2.
If $\mathcal{D} :: \Theta_1; \Sigma_1 \parallel \Pi_1, [\Gamma_1 \Longrightarrow \Delta_1, C]^w$ and $\mathcal{E} :: \Theta_2; \Sigma_2 \parallel \Pi_2, [\Gamma_2, C \Longrightarrow \Delta_2]^w$,
then $\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$.

Proof. By induction on $(k, n_1, n_2, m_1, m_2) := (|C|, \|\mathcal{D}\|, \|\mathcal{E}\|, \lfloor\mathcal{D}\rfloor, \lfloor\mathcal{E}\rfloor)$, where $|C|$ is the size of cut formula C. In the proof, we use Proposition A.2, A.3, A.5, 3.2, A.8, A.9, and B.2, and Lemma 3.1 and B.3. Most cases are immediate and some of non-trivial cases are as follows.

- **Case** $n_1 > 1$, $n_2 > 1$, $\mathcal{L}(\mathcal{D})$ (resp. $\mathcal{L}(\mathcal{E})$) focuses on C, and $\mathcal{L}(\mathcal{D}) = \star\mathsf{R}$:
In this case, we have $C = A \star B$ and $\mathcal{L}(\mathcal{E}) = \star\mathsf{L}$. For a simple argument, let us assume that $\mathcal{L}(\mathcal{D})$ is not a special instance of $\star\mathsf{R}$. Then we have $\Pi_1 = \Pi_1', [\Gamma_{1u} \Longrightarrow \Delta_{1u}]^u, [\Gamma_{1v} \Longrightarrow \Delta_{1v}]^v$ and

$$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1' :: \Theta_1; \Sigma_1 \parallel \Pi_1', [\Gamma_{1u} \Longrightarrow \Delta_{1u}, A]^u, [\Gamma_{1v} \Longrightarrow \Delta_{1v}]^v, [\Gamma_1 \Longrightarrow \Delta_1, A \star B]^w \\ w \doteq u \circ v \in \Sigma_1 \quad \mathcal{D}_2' :: \Theta_1; \Sigma_1 \parallel \Pi_1', [\Gamma_{1u} \Longrightarrow \Delta_{1u}]^u, [\Gamma_{1v} \Longrightarrow \Delta_{1v}, B]^v, [\Gamma_1 \Longrightarrow \Delta_1, A \star B]^w\end{array}}{\Theta_1; \Sigma_1 \parallel \Pi_1, [\Gamma_1 \Longrightarrow \Delta_1, A \star B]^w}\ \star\mathsf{R}$$

$$\mathcal{E} = \dfrac{\textit{fresh } w_1, w_2 \quad \mathcal{E}' :: \Theta_2; \Sigma_2, w \doteq w_1 \circ w_2 \parallel \Pi_2,\ [\Gamma_2 \Longrightarrow \Delta_2]^w, [A \Longrightarrow \cdot]^{w_1}, [B \Longrightarrow \cdot]^{w_2}}{\Theta_2; \Sigma_2 \parallel \Pi_2, [\Gamma_2, A \star B \Longrightarrow \Delta_2]^w}\ \star\mathsf{L}$$

$$\mathcal{F}_1 :: \quad \Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \left(\Pi_1',\ \begin{array}{l}[\Gamma_{1u}, \Gamma_{2u} \Longrightarrow \Delta_{1u}, \Delta_{2u}, A]^u, \\ [\Gamma_{1v}, \Gamma_{2v} \Longrightarrow \Delta_{1v}, \Delta_{2v}]^v\end{array}\right) \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$$

[IH on $A \star B, \mathcal{D}_1', \mathcal{E}$]

$$\mathcal{F}_2 :: \quad \Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \left(\Pi_1',\ \begin{array}{l}[\Gamma_{1u}, \Gamma_{2u} \Longrightarrow \Delta_{1u}, \Delta_{2u}]^u, \\ [\Gamma_{1v}, \Gamma_{2v} \Longrightarrow \Delta_{1v}, \Delta_{2v}, B]^v\end{array}\right) \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$$

[IH on $A \star B, \mathcal{D}_2', \mathcal{E}$]

$$\mathcal{E}'' :: \Theta_2; \Sigma_2, w \doteq u \circ v \parallel (\Pi_2, [\Gamma_2 \Longrightarrow \Delta_2]^w) \uplus ([A \Longrightarrow \cdot]^u, [B \Longrightarrow \cdot]^v)$$

[Lemma 3.1 on $\mathcal{E}'$ with u, w_1 and v, w_2]

$$\mathcal{G} :: \Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \left(\Pi_1',\ \begin{array}{l}[\Gamma_{1u}, \Gamma_{2u} \Longrightarrow \Delta_{1u}, \Delta_{2u}]^u, \\ [\Gamma_{1v}, \Gamma_{2v}, B \Longrightarrow \Delta_{1v}, \Delta_{2v}]^v\end{array}\right) \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$$

[IH on $A, \mathcal{F}_1, \mathcal{E}''$ and Proposition A.5, 3.2, A.8]

$$\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \left(\Pi_1',\ \begin{array}{l}[\Gamma_{1u}, \Gamma_{2u} \Longrightarrow \Delta_{1u}, \Delta_{2u}]^u, \\ [\Gamma_{1v}, \Gamma_{2v} \Longrightarrow \Delta_{1v}, \Delta_{2v}]^v\end{array}\right) \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$$

[IH on $B, \mathcal{F}_2, \mathcal{G}$ and Proposition A.5, 3.2, A.8]

- **Case** $n_1 > 1$, $n_2 > 1$, $\mathcal{L}(\mathcal{D})$ does not focus on C, and $\mathcal{L}(\mathcal{D}) \in \mathsf{NormRules}$:
In this case, for some $\sigma \in \Sigma_1$ we have

$$\mathcal{D} = \dfrac{\mathcal{D}' :: \Theta_1; [u/v](\Sigma_1 \backslash \{\sigma\}) \parallel [u/v](\Pi_1, [\Gamma_1 \Longrightarrow \Delta_1, C]^w)}{\Theta_1; \Sigma_1 \parallel \Pi_1, [\Gamma_1 \Longrightarrow \Delta_1, C]^w}\ \mathcal{L}(\mathcal{D})$$

$\mathcal{E}' :: \Theta_2; [u/v]\Sigma_2 \parallel [u/v](\Pi_2, [\Gamma_2, C \Longrightarrow \Delta_2]^w)$ [Lemma 3.1 on $\mathcal{E}$ with u, v]
$\Theta_1, \Theta_2; [u/v](\Sigma_1 \backslash \{\sigma\}, \Sigma_2) \parallel [u/v](\Pi_1 \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w)$ [IH on $C, \mathcal{D}', \mathcal{E}'$]
$\Theta_1, \Theta_2; \Sigma_1, \Sigma_2 \parallel \Pi_1 \uplus \Pi_2, [\Gamma_1, \Gamma_2 \Longrightarrow \Delta_1, \Delta_2]^w$ [the rule $\mathcal{L}(\mathcal{D})$]

□

Appendix C. Soundness of $\mathbf{P_{SL}}$

Lemma 6.2. For every inference rule with the conclusion W and the premise consisting of $W_1, \cdots, W_n$, it holds that $[\![W]\!]_S$ implies $\bigvee_{i=1,\cdots,n} [\![W_i]\!]_S$ for any stack S. If $n = 0$, we have $\neg[\![W]\!]_S$.

Proof. The proof for the logical rules is immediate from the definition of $[\![W]\!]_S$ and from the semantics of separation logic. Moreover the proof for the structural rules and the heap contradiction rules, the sketch of which is described in Section 3.3 and 3.4, are based on neutrality, associativity, cancellativity, and partial-determinism of $\circ$, and on the existence

and uniqueness of an empty heap. Since the proof is not difficult at all, we present the proof for the rule $\mathsf{Disj}\star$ only:

- Let $W = \Theta; \Pi \parallel \Sigma$. Then $\{w \doteq u_1 \circ u_2, w \doteq v_1 \circ v_2\} \subset \Sigma$.
- Suppose $[\![W]\!]_S$ holds. By assumption, we get $w = u_1 \circ u_2$ and $w = v_1 \circ v_2$.
- Let $D_i := dom(u_i), D'_j := dom(v_j)$, and $E_{ij} := D_i \cap D'_j$ $(i, j = 1, 2)$.
- Let w_{ij} be the heap subsumed by w such that $dom(w_{ij}) = E_{ij}$ $(i, j = 1, 2)$.
- Since $D_1 \cup \mathcal{D}_2 = D'_1 \cup \mathcal{D}'_2$ and $D_1 \cap D_2 = D'_1 \cap D'_2 = \varnothing$, we know that $D_i = E_{i1} \cup E_{i2}$, $D'_j = E_{1j} \cup E_{2j}$ and $E_{ij} \cap E_{kl} = \varnothing$ for any $i, j, k, l = 1, 2$ with $(i, j) \neq (k, l)$.
- Thus we have $u_i = w_{i1} \circ w_{i2}$ and $v_j = w_{1j} \circ w_{2j}$ for any $i, j = 1, 2$, which implies $[\![W_1]\!]_S$.

□

Appendix D. Proof search strategy $\mathcal{SS}$ for $\mathbf{P_{SL}}$

Todo. In this section, $\mathbf{P_{SL}}$ means '$\mathbf{P_{SL}}$ with the two propagation rules $\mathsf{Prop}{\epsilon}{\not=}$ and $\mathsf{Prop}{\mapsto}{\not=}$.'

D.1. **Preliminary definitions and lemmas.** In this section, for the sake of simplicity, we use the following notational abuse:

- For a heap w and a world sequent W, $w \in W$ means that W contains a heap sequent for w.
- For a heap relation σ and a world sequent W, $\sigma \in W$ means that heap relations of W contain σ.
- "WLOG" means "without loss of generality," and $\lightning$ means that we obtain a contradiction.

Moreover, we assume that every world sequent has no duplicate heap relations. This assumption can always be achieved by applying the following rule $\mathsf{Weaken'}$ (which is admissible) whenever we obtain duplicate heap relations:

$$\frac{\Theta; \Sigma, \sigma \parallel \Pi}{\Theta; \Sigma, \sigma, \sigma \parallel \Pi}\ \mathsf{Weaken'}$$

We introduce some more definitions about the properties of graphs of heaps, and prove basic lemmas regarding the properties of graphs of heaps. Since Lemma D.2 is frequently exploited in the remaining part of the appendix, we omit its use for simplicity.

Definition D.1.

- $w \not\nearrow u \iff w \nearrow u$ does not hold.
- $w \not\curlywedge u \iff w \curlywedge u$ does not hold.
- $w \Uparrow u$ means that heaps w and u have a common ancestor v such that both $w \nearrow v$ and $u \nearrow v$ hold. We write $w \Uparrow^v u$ to indicate that heap v is such a common ancestor of heaps w and u.
- $w \curlywedge u$ means that heaps w and u are disjoint: there is at least one common ancestor v of w and u along with two heaps w' and u' such that $w \nearrow w'$, $u \nearrow u'$, and $v \doteq w' \circ u'$. We write $w \curlywedge^v u$ to indicate that heap v is such a common ancestor of heaps w and u.
- Whenever we are unclear about a world sequent that we are talking about, we will add a subscripted world sequent to existing definitions such as $w \nearrow_W u$ and $T_W(w)$.
- W is consistent at $w \iff$ if $w \doteq u_1 \circ u_2$ and $w \doteq v_1 \circ v_2$ then, $T(u_1) \cup T(u_2) = T(v_1) \cup T(v_2)$.
- $\mathcal{R}(W)$ denotes the set of all root heaps in W.
- $\mathcal{T}(W)$ denotes the set of all terminal heaps in W.

Lemma D.2. Let W be any world sequent. Then, the followings hold:

- Suppose that W is non-cyclic. Then, $|T(w)| \geq 1$ for any heap w.
- Suppose that W is elementary. Then, $w \not\curlywedge w$ for any heap w.
- Suppose that W is consistent at w. Then, $w \doteq u \circ v$ implies $T(w) = T(u) \cup T(v)$.

Proof. The proof is straightforward. □

Lemma D.3. Let W be any world sequent. Then, the followings hold:

(1) Suppose that W is non-cyclic and well-formed. Then, $T(u) = \{v\}$ implies $u = v$.
(2) Suppose that W is consistent, and $u \neq v$ and $u \Uparrow^w v$ hold for some heap w and terminal heaps u, v.
Then, $u \wedge^{w'} v$ for some heap w' such that $w' \nearrow w$.

Proof.

(1)
- Suppose $u \neq v$. Let $u_1 := u$. We want to get a contradiction.
- Since $v \in T(u_1)$, we have $w_1 \doteq v \circ u_2 \in W$ for some $w_1, u_2 \in W$ with $w_1 \nearrow u_1$.
- Since W is well-formed and non-cyclic, $v \neq u_2$ and $u_1 \neq u_2$.
- Since W is non-cyclic, $T(u_2) = \{v\}$.
- Since $v \in T(u_2)$, we have $w_2 \doteq v \circ u_3 \in W$ for some $w_2, u_3 \in W$ with $w_2 \nearrow u_2$.
- Since W is well-formed and non-cyclic, $v \neq u_3$ and $u_i \neq u_3$ $(i = 1, 2)$.
- In this way, we get infinite distinct heaps $u_1, u_2, \cdots$ in W. ※

(2)
- Since $u \neq v$ and u is a terminal heap, $u \neq w$ and $v \not\nearrow u$.
- Since $v \not\nearrow u$ and $v \nearrow w$, we have $w' \doteq u' \circ v' \in W$ for some heaps $u', v', w' \in W$ with $u \nearrow u'$, $w' \nearrow w$, $v \not\nearrow u'$, and $v \nearrow w'$.
- Since v is a terminal heap, $v \in T(w') - T(u')$.
- Since W is consistent at w', $v \in T(v')$ and $u \wedge^{w'} v$.

□

We now prove auxiliary lemmas which tells us what properties of world sequents are preserved after we apply each of the structural rules. The next subsection will be substantially based on these lemmas.

Lemma D.4 (Preservation lemma for the logical rules)**.**

(1) Every logical rule preserves elementary-ness.
(2) Every logical rule except the rule $\star\mathsf{L}$ preserves consistency.

Proof. The proof is straightforward. □

Lemma D.5 (Preservation lemma for the rule $\mathsf{Disj}\star$)**.**

(1) Let W, W' be any world sequents such that W is elementary and $\{W\} \mapsto \{W'\}$ by applying the rule $\mathsf{Disj}\star$. Then, for any heaps $u, v \in W$, $u \nearrow_W v \iff u \nearrow_{W'} v$ and $u \wedge_W v \iff u \wedge_{W'} v$.
(2) The rule $\mathsf{Disj}\star$ preserves non-cyclic-ness and well-formed-ness but not elementary-ness.

Proof. The proof of (1) is immediate. The proof of (2) uses (1), and an example that the rule $\mathsf{Disj}\star$ does not preserve elementary-ness is: applying the rule $\mathsf{Disj}\star$ to

$$\{w \doteq w_{12} \circ w_3, w \doteq w_1 \circ w_{23}, w_{12} \doteq w_1 \circ w_2, w_{23} \doteq w_2 \circ w_3\}$$

with $\{w \doteq w_{12} \circ w_3, w \doteq w_1 \circ w_{23}\}$. □

Lemma D.6 (Preservation lemma for the rule Assoc)**.**

(1) Let W, W' be any world sequents such that $\{W\} \mapsto \{W'\}$ by applying the rule Assoc. Then, $\mathcal{T}(W) = \mathcal{T}(W')$ and $\mathcal{R}(W) = \mathcal{R}(W')$.
Moreover, for any heaps $u, v \in W$, $u \nearrow_W v \iff u \nearrow_{W'} v$ and $T_W(u) = T_{W'}(u)$.

(2) The rule Assoc preserves consistency, full-ness, $\star$-ready-ness, $-\!\!\star$-ready-ness, and sanitizedness.

Proof. The proof of (1) is immediate.

(2)
- Let W, W' be any world sequents such that W is elementary and $\{W\} \mapsto \{W'\}$ by applying the rule Assoc to $\{w \doteq u \circ v,\ u \doteq u_1 \circ u_2\}$.
- Since W is non-cyclic, it is easy to check that W' is non-cyclic.
- Since W is elementary, w, u, v, u_1, u_2 are all distinct. Thus, W' is well-formed.
- Finally, W' is elementary by (1).

By using the above observations together with (1), we can show that the rule Assoc preserves consistency, full-ness, $\star$-ready-ness, $-\!\!\star$-ready-ness, and sanitizedness.

□

Lemma D.7 (Preservation lemma for the propagation rules and the rule Weaken)**.**

(1) The propagation rules preserve $\star$-ready-ness and $-\!\!\star$-ready-ness.

(2) The rule Weaken preserves $\star$-ready-ness and $-\!\!\star$-ready-ness.

Proof. The proof is straightforward. □

Lemma D.8 (Preservation lemma for the rule NormEq)**.**

Let W and W' be any world sequents such that $\{W\} \mapsto \{W'\}$ by applying the rule NormEq to $\{w \doteq w_1 \circ w_2,\ \widehat{w} \doteq w_1 \circ w_2\}$. Then, we have the followings.

(1)
- If W is well-formed, then $\mathcal{T}(W) = \mathcal{T}(W')$ and $\mathcal{R}(W) \supset \mathcal{R}(W')$.
- If $u'_1 \doteq u'_2 \circ u'_3 \in W'$ for some heaps $u'_i \in W'$ $(i = 1, 2, 3)$,
then $u_1 \doteq u_2 \circ u_3 \in W$ for some heaps $u_i \in W$ such that $[w/\widehat{w}]u_i = u'_i$ $(i = 1, 2, 3)$.
- If $u_1 \doteq u_2 \circ u_3 \in W$ for some heaps $u_i \in W$ $(i = 1, 2, 3)$, then
$$[w/\widehat{w}]u_1 \doteq [w/\widehat{w}]u_2 \circ [w/\widehat{w}]u_3 \in W'.$$
Moreover, for any heaps $\alpha, \beta \in W$, $\alpha \nearrow_W \beta$ implies $[w/\widehat{w}]\alpha \nearrow_{W'} [w/\widehat{w}]\beta$.

(2)
- If W is consistent, then for any heap $\alpha \in W$, $T_W(\alpha) = T_{W'}([w/\widehat{w}]\alpha)$.

(3)
- If W is sanitized, then W' is sanitized.
- If W is $\star$-ready for heap $\alpha \in W$, then W' is $\star$-ready for heap $[w/\widehat{w}]\alpha \in W'$.
- If W is $-\!\!\star$-ready for heap $\alpha \in W$ and if $\alpha \nearrow_{W'} r'$ implies $\alpha \nearrow_W r'$ for any $r' \in \mathcal{R}(W')$,
then W' is $-\!\!\star$-ready for heap $[w/\widehat{w}]\alpha \in W'$.
- Even if W is $-\!\!\star$-ready for heap $\alpha \in W$, W' is not $-\!\!\star$-ready for heap $[w/\widehat{w}]\alpha \in W'$ in general.

Proof. In the following proof, we assume $w \neq \widehat{w}$ since this lemma becomes trivial when $w = \widehat{w}$. Moreover, in the proof of (2) and (3), we will frequently use the fact that $T_W(w) = T_W(\widehat{w})$ which holds since W is consistent and $\{w \doteq w_1 \circ w_2,\ \widehat{w} \doteq w_1 \circ w_2\} \subset W$. The proof of (1) is immediate so we omit it.

(3) Before proving (2), we first prove that: if W is consistent, then W' is well-formed and non-cyclic.

1. W' is well-formed:
   - Suppose that W' is not well-formed.
   - Then, we have the following cases since W is well-formed.
   - **Case** $u \doteq w \circ \widehat{w} \in W$ for some heap $u \in W$:
     - $T_W(w) = T_W(\widehat{w})$ contradicts to that W is elementary and $u \doteq w \circ \widehat{w} \in W$. ※
   - **Case** $w \doteq \widehat{w} \circ u \in W$ or $\widehat{w} \doteq w \circ u \in W$ for some heap $u \in W$:
     - WLOG, assume $w \doteq \widehat{w} \circ u \in W$.
     - Since W is consistent and $w \doteq \widehat{w} \circ u \in W$, $T_W(w) = T_W(\widehat{w}) = T_W(\widehat{w}) \cup T_W(u)$.
     - Thus, $T_W(u) = \varnothing$ and this contradicts to that W is non-cyclic. ※
2. W' is non-cyclic:
   - Suppose that W' is cyclic.
   - Then, $u'_1 \nearrow_{W'} u'_2$ and $u'_2 \nearrow_{W'} u'_1$ for some heaps $u'_1, u'_2 \in W'$ such that $u'_1 \neq u'_2$.
   - By (1) and that W is non-cyclic, $w \nearrow_{W'} u' \nearrow_{W'} w$ for some heap $u' \in W'$ such that $u' \neq w$.
     (Here, $\alpha_1 \nearrow \alpha_2 \nearrow \alpha_3$ denotes that $\alpha_1 \nearrow \alpha_2$ and $\alpha_2 \nearrow \alpha_3$ hold.)
   - By (1) and that W is non-cyclic, $w \nearrow_W u' \nearrow_W \widehat{w}$ or $\widehat{w} \nearrow_W u' \nearrow_W w$.
   - This contradicts to that W is non-cyclic and $T_W(w) = T_W(\widehat{w})$. ※

(2) Let W be a consistent world sequent and let α be a heap in W. We have the following cases.
- **Case** $\alpha \neq \widehat{w}$:
  As $\mathcal{T}(W) = \mathcal{T}(W')$ by (1), it suffices to show that: for any $t \in \mathcal{T}(W)$, $t \nearrow_W \alpha \iff t \nearrow_{W'} \alpha$.
  - ($\Rightarrow$) – This direction is easy by (1).
  - ($\Leftarrow$) – Suppose $t \in \mathcal{T}(W)$ and $t \nearrow_{W'} \alpha$.
    - Then, $t = u'_0$, $\{u'_1 \doteq u'_0 \circ v'_0, \cdots, u'_n \doteq u'_{n-1} \circ v'_{n-1}\} \subset W'$, and $u'_n = \alpha$ hold for some $n \geq 0$ and some heaps $u'_i, v'_j \in W'$ ($i = 0, \cdots, n$ and $j = 0, \cdots, n-1$).
    - Since $t \in \mathcal{T}(W)$ and $w \notin \mathcal{T}(W)$, we have $u'_0 \neq w$.
    - (Subcase) $u'_i \neq w$ for all $1 \leq i \leq n$: By (1), we have $t \nearrow_W \alpha$.
    - (Subcase) $u'_k = w$ for some $1 \leq k \leq n$:
      - As W' is well-formed and non-cyclic by (3) proven above, $u'_i \neq w$ for all $1 \leq i \leq n$ with $i \neq k$.
      - Hence, we have $\begin{cases} t \nearrow_W w \text{ or } t \nearrow_W \widehat{w} \text{ (by (1) and } t \nearrow_{W'} u'_k), \text{ and} \\ w \nearrow_W \alpha \text{ or } \widehat{w} \nearrow_W \alpha \text{ (by (1) and } u'_k \nearrow_{W'} \alpha). \end{cases}$
      - Since $T_W(w) = T_W(\widehat{w})$, $t \nearrow_W w \iff t \nearrow_W \widehat{w}$.
      - Therefore, $t \nearrow_W w$ and $t \nearrow_W \widehat{w}$ hold, and thus $t \nearrow_W \alpha$ holds.
- **Case** $\alpha = \widehat{w}$: By the above case when $\alpha = w$, we have $T_W(\widehat{w}) = T_W(w) = T_{W'}(w)$.

(3) First, let W be a sanitized world sequent. Let us prove that W' is sanitized as well.

3. W' is elementary:
   - We already proved that W' is well-formed and non-cyclic.
   - Suppose $u'_1 \doteq u'_2 \circ u'_3 \in W'$ for some heaps $u'_i \in W'$ ($i = 1, 2, 3$).

- By (1), $u_1 \doteq u_2 \circ u_3 \in W$ for some heaps $u_i \in W$ such that $[w/\widehat{w}]u_i = u'_i$ $(i = 1, 2, 3)$.
- By (2) and that W is elementary, $T_{W'}(u'_2) \cap T_{W'}(u'_3) = T_W(u_2) \cap T_W(u_3) = \varnothing$.

4. W' is consistent: The proof is similar to that of 3., using (1), (2) and that W is consistent.
5. W' is full: The proof is similar to that of 3., using (1), (2) and that W is full.
9. W' is sanitized: The proof is easy by using (1) and that W is sanitized.

Next, let W be a world sequent which is $\star$-ready for heap $\alpha \in W$. The proof of that W' is $\star$-ready for heap $[w/\widehat{w}]\alpha \in W'$ is similar to that of 3., using (1), (2) and that W is $\star$-ready for α. The third claim is similarly proved.

Finally, let W^0 be a world sequent whose heap relations are $\{w \doteq \alpha \circ \beta, w \doteq w_1 \circ w_2, \widehat{w} \doteq w_1 \circ w_2, \gamma \doteq \widehat{w} \circ \delta\}$, where every heap is distinct. It is easy to show that there exist world sequents W and W' such that:

- $\{W^0\} \mapsto^* \{W\}$ by applying only the rules $\mathsf{Disj}\star$ and Assoc;
- $\{W\} \mapsto \{W'\}$ by applying the rule NormEq to $\{w \doteq w_1 \circ w_2, \widehat{w} \doteq w_1 \circ w_2\}$;
- W is $-\!\!\star$-ready for α while W' is not $-\!\!\star$-ready for α.

The reason why the rule NormEq does not preserve $-\!\!\star$-ready-ness is that the rule does not preserve the relation $\not\nearrow$ in general. For instance, for the world sequents W and W' described above, we have $\alpha \not\nearrow_W \gamma$ and $\alpha \nearrow_{W'} \gamma$.

□

The preservation lemma for the rule NormPC is slightly different from the previous preservation lemmas because even if a world sequent W is elementary, the world sequent obtained by applying the rule NormPC to W is not elementary in general (see Lemma D.12). After introducing the following additional definitions about the properties of graphs of heaps, we prove that the newly defined properties are preserved after we apply the rule NormPC. Even though the newly defined properties are weaker than the previous corresponding ones, it turns out that they are strong enough to establish Lemma 7.1.

Definition D.9. For a world sequent W and $t \in \mathcal{T}(W)$, we define the following properties of W.

3. t-Elementary: well-formed, non-cyclic, and if $w \doteq w_1 \circ w_2$, then $(T(w_1) - \{t\}) \cap (T(w_2) - \{t\}) = \varnothing$.
4. t-Consistent: t-elementary and if $w \doteq u_1 \circ u_2$ and $w \doteq v_1 \circ v_2$, then $(T(u_1) - \{t\}) \cup (T(u_2) - \{t\}) = (T(v_1) - \{t\}) \cup (T(v_2) - \{t\})$.
5. t-Full: t-consistent and for any $r \in \mathcal{R}(W)$ and any non-empty set $S \subset T(r) - \{t\}$, there exists at least one heap w with $T(w) - \{t\} = S$.
6. t- $\star$-ready for heap w: t-full and for any pair of non-empty sets $S_1, S_2 \subset T(w) - \{t\}$ such that $S_1 \cap S_2 = \varnothing$ and $S_1 \cup S_2 = T(w) - \{t\}$, there exist heaps w_1 and w_2 such that $w \doteq w_1 \circ w_2$ with $T(w_1) - \{t\} = S_1$ and $T(w_2) - \{t\} = S_2$.
9. t-Sanitized: t-full and non-terminal heaps have no atomic heap relations.

Lemma D.10.

Suppose that W is t-consistent world sequent, where $t \in \mathcal{T}(W)$.

Then, $w \doteq u \circ v$ implies $T(w) - \{t\} = (T(u) - \{t\}) \cup (T(v) - \{t\})$.

Proof. The proof is straightforward. □

Lemma D.11. Let $A, B \subset U$ be sets and $x, y \in U$. Then, the followings hold.

(1) • If $x \notin A$, then $[y/x][x/y]A = A$.

(2) • If $x \notin A$ and $y \notin B$, then $[x/y]A = B \iff A = [y/x]B \iff \begin{cases} a \in A \Rightarrow [x/y]a \in B, \text{ and} \\ b \in B \Leftarrow [y/x]b \in A. \end{cases}$

(3) • $([x/y]A) \cap ([x/y]B) = [x/y](A \cap B)$ if $x \notin A$ and $x \notin B$.
• $([x/y]A) \cup ([x/y]B) = [x/y](A \cup B)$.

Proof. The proof is straightforward. □

Lemma D.12 (Preservation lemma for the rule NormPC)**.**

Let W and W' be any world sequents such that $\{W\} \mapsto \{W'\}$ by applying the rule NormPC to $\{w \doteq w_1 \circ w_2, w_2 \doteq \epsilon\}$, where $w_2 \in T(W)$. Assume that for any $n \geq 2$, there do not exist heaps $u_0, \cdots, u_n \in W$ such that $w_1 = u_0, \{u_1 \doteq u_0 \circ w_2, \cdots, u_n \doteq u_{n-1} \circ w_2\}, u_n = w \quad \cdots$ (*). Then, we have the followings.

(1) • If W is w_2-consistent, then $[w/w_1]\mathcal{T}(W) = \mathcal{T}(W')$ and $\mathcal{R}(W) \supset \mathcal{R}(W') - \{w_2\}$.
• If $u'_1 \doteq u'_2 \circ u'_3 \in W'$ for some heaps $u'_i \in W'$ $(i = 1, 2, 3)$,
then $u_1 \doteq u_2 \circ u_3 \in W$ for some heaps $u_i \in W$ such that $[w/w_1]u_i = u'_i$ $(i = 1, 2, 3)$.
• If $u_1 \doteq u_2 \circ u_3 \in W$ for some heaps $u_i \in W$ $(i = 1, 2, 3)$ with $u_2 \neq w_2$ and $u_3 \neq w_2$,
then $[w/w_1]u_1 \doteq [w/w_1]u_2 \circ [w/w_1]u_3 \in W'$.
Moreover, for any heaps $\alpha, \beta \in W$ with $\alpha \neq w_2$, $\alpha \nearrow_W \beta$ implies $[w/w_1]\alpha \nearrow_{W'} [w/w_1]\beta$.

(2) • If W is w_2-consistent, then for any heap $\alpha \in W$, $[w/w_1](T_W(\alpha) - \{w_2\}) = T_{W'}([w/w_1]\alpha) - \{w_2\}$.

(3) • If W is w_2-sanitized, then W' is w_2-sanitized.
• If W is w_2- $\star$-ready for heap $\alpha \in W$, then W' is w_2- $\star$-ready for heap $[w/w_1]\alpha \in W'$.
• Even if W is consistent, W' is not elementary in general.

Proof. In the following proof, for simplicity we omit the use of Lemma D.10 and that W is well-formed. Moreover, we will frequently use the fact that $T_W(w) - \{w_2\} = T_W(w_1) - \{w_2\}$ which holds since W is w_2-consistent and $w \doteq w_1 \circ w_2 \in W$.

(1) As the proof of the second and the third claim is immediate, we prove only the first claim. By Lemma D.11-(2) with $w \notin \mathcal{T}(W)$ and $w_1 \notin \mathcal{T}(W')$, it suffices to show that:
(a) For any $t \in \mathcal{T}(W)$, $[w/w_1]t \in \mathcal{T}(W')$; and
(b) For any $t' \in \mathcal{T}(W')$, $[w_1/w]t' \in \mathcal{T}(W)$.
The proof is as follows.
(a) • Let $t \in \mathcal{T}(W)$.
• **Case** $t \neq w_1$: This case is easy by using $[w/w_1]t = t$ and $t \neq w$ (since $w \notin \mathcal{T}(W)$).
• **Case** $t = w_1$:
It suffices to show that $w \doteq w_1 \circ w_2$ is the only non-atomic heap relation at w in W.
– Suppose not. Since W has no duplicate heap relations, $w \doteq \eta_1 \circ \eta_2 \in W$ for some heaps $\eta_1, \eta_2 \in W$ such that $\{\eta_1, \eta_2\} \neq \{w_1, w_2\}$.
– Since $w_1 \in \mathcal{T}(W)$ and $T_W(w) - \{w_2\} = T_W(w_1) - \{w_2\}$, $T_W(w) = \{w_1, w_2\}$.

- (Subcase) $\eta_1 = w_1$: (We omit the proof for $\eta_2 = w_1$)
  - Since $T_W(\eta_2) \subset T_W(w) = \{w_1, w_2\}$ and W is w_2-elementary, $T_W(\eta_2) \subset \{w_2\}$.
  - Since W is non-cyclic, $T_W(\eta_2) = \{w_2\}$ and thus $\eta_2 = w_2$. ※ [Lemma D.3-(1)]
- (Subcase) $\eta_1 \neq w_1$ and $\eta_2 \neq w_1$:
  - Since W is w_2-consistent, $T_W(w) - \{w_2\} = (T_W(\eta_1) - \{w_2\}) \cup (T_W(\eta_2) - \{w_2\})$.
  - Since $w_1 \in T_W(w) - \{w_2\}$, $w_1 \nearrow_W \eta_1$ or $w_1 \nearrow_W \eta_2$. WLOG, assume $w_1 \nearrow_W \eta_1$.
  - Thus, $w_1 = u_0$, $\{u_1 \doteq u_0 \circ v_0, \cdots, u_n \doteq u_{n-1} \circ v_{n-1}\} \subset W$, and $u_n = \eta_1$ hold for some $n \geq 0$ and some heaps $u_i, v_j \in W$ ($i = 0, \cdots, n$ and $j = 0, \cdots, n-1$).
  - Since $\eta_1 \neq w_1$, we have $n \geq 1$.
  - Let $u_{n+1} := w$ and $v_n = \eta_2$. Then, since $w_1 \in T_W(u_j)$ for all $0 \leq j \leq n$, we have $T_W(v_j) = \{w_2\}$ for all $0 \leq j \leq n$ by a similar proof to that of the above subcase.
  - Hence, $v_j = w_2$ for all $0 \leq j \leq n$, and this contradicts to (*). ※ [Lemma D.3-(1)]

(b)
- Let $t' \in \mathcal{T}(W')$.
- **Case** $t' \neq w$: This case is easy by using $[w_1/w]t' = t'$ and $t' \neq w_1$.
- **Case** $t' = w$: $w \in \mathcal{T}(W')$ implies that there is no non-atomic heap relation at w_1 in W.

(3) Before proving (2), we prove that: if W is w_2-consistent, then W' is well-formed and non-cyclic.

1. W' is well-formed:
   - Suppose that W' is not well-formed.
   - Then, we have the following cases since W is well-formed.
   - **Case** $u \doteq w \circ w_1 \in W$ for some heap $u \in W$:
     - Since W is non-cyclic and $w_1 \neq w_2$, $T_W(w_1) - \{w_2\} \neq \varnothing$. [Lemma D.3-(1)]
     - This contradicts to that W is w_2-elementary. ※
   - **Case** $w \doteq w_1 \circ u \in W$ for some heap $u \in W$:
     - Since W has no duplicate heap relations, $u \neq w_2$.
     - Since W is w_2-consistent, $(T_W(u) - \{w_2\}) \cup (T_W(w_1) - \{w_2\}) = (T_W(w_1) - \{w_2\})$.
     - Since W is w_2-elementary, $(T_W(u) - \{w_2\}) \cap (T_W(w_1) - \{w_2\}) = \varnothing$.
     - Thus, $T_W(u) = \{w_2\}$ since W is non-cyclic, and thus $u = w_2$. ※ [Lemma D.3-(1)]
   - **Case** $w_1 \doteq w \circ u \in W$ for some heap $u \in W$:
     - This case contradicts to that W is non-cyclic. ※
2. W' is non-cyclic:
   - Suppose that W' is cyclic.
   - As in the proof of Lemma D.8, we obtain that $w \nearrow_W u' \nearrow_W w_1$ or $w_1 \nearrow_W u' \nearrow_W w$ for some heap $u' \in W'$ such that $u' \neq w$, by using (1) and that W is non-cyclic.
   - Since $w \nearrow_W u' \nearrow_W w_1$ contradicts to that W is non-cyclic, assume $w_1 \nearrow_W u' \nearrow_W w$.

- This contradicts to (*) by a similar proof to that of (1), since $u' \neq w$ and $u' \neq w_1$. ※

(2) Let W be a w_2-consistent world sequent and α be a heap in W. We have the following cases.

- **Case** $\alpha \neq w_1$:
  By Lemma D.11-(2) with $w \notin T_W(\alpha) - \{w_2\}$ and $w_1 \notin T_{W'}(\alpha) - \{w_2\}$, it suffices to show that:
  (a) For any $t \in T_W(\alpha) - \{w_2\}$, $[w/w_1]t \in T_{W'}(\alpha) - \{w_2\}$; and
  (b) For any $t' \in T_{W'}(\alpha) - \{w_2\}$, $[w_1/w]t' \in T_W(\alpha) - \{w_2\}$.
  The proof is as follows.
  (a)
  - Let $t \in T_W(\alpha) - \{w_2\}$. By (1), $[w/w_1]t \in \mathcal{T}(W')$ and $[w/w_1]t \nearrow_{W'} \alpha$.
  - Since $t \neq w_2$ and $w \neq w_2$, we conclude that $[w/w_1]t \in T_{W'}(\alpha) - \{w_2\}$.

  (b)
  - Let $t' \in T_{W'}(\alpha) - \{w_2\}$. By (1), $[w_1/w]t' \in \mathcal{T}(W)$.
  - Since $t' \neq w_2$ and $w_1 \neq w_2$, it suffices to show that $[w_1/w]t' \nearrow_W \alpha$.
  - Since $t' \nearrow_{W'} \alpha$, we have $t' = u'_0$, $\{u'_1 \doteq u'_0 \circ v'_0, \cdots, u'_n \doteq u'_{n-1} \circ v'_{n-1}\} \subset W'$, and $u'_n = \alpha$ for some $n \geq 0$ and some heaps $u'_i, v'_j \in W'$ $(i = 0, \cdots, n$ and $j = 0, \cdots, n-1)$.
  - (Subcase) $t' = w$:
    - As W' is well-formed and non-cyclic by (3) proven above, $u'_i \neq w$ for all $1 \leq i \leq n$.
    - By (1), $w \nearrow_W \alpha$ or $w_1 \nearrow_W \alpha$. Since $w_1 \nearrow_W w$, we have $w_1 \nearrow_W \alpha$ in both cases.
  - (Subcase) $t' \neq w$:
    - If $u'_i \neq w$ for all $1 \leq i \leq n$, then by (1) we immediately have $t' \nearrow_W \alpha$.
    - Hence, suppose that $u'_k = w$ for some $1 \leq k \leq n$.
    - As W' is well-formed and non-cyclic by (3) proven above, $u'_i \neq w$ for all $1 \leq i \leq n$ with $i \neq k$.
    - Thus, we have $\begin{cases} t' \nearrow_W w \text{ or } t' \nearrow_W w_1 \text{ (by (1) and } t' \nearrow_{W'} u'_k), \text{ and} \\ w \nearrow_W \alpha \text{ or } w_1 \nearrow_W \alpha \text{ (by (1) and } u'_k \nearrow_{W'} \alpha). \end{cases}$
    - Since W is w_2-consistent, $t' \neq w_2$, and $w_2 \in \mathcal{T}(W)$, we have $t' \nearrow_W w_1$.
    - Since $w_1 \nearrow_W w$, we finally have $t' \nearrow_W \alpha$.
- **Case** $\alpha = w_1$:
  - By the above case when $\alpha = w$, and by using that $T_W(w_1) - \{w_2\} = T_W(w) - \{w_2\}$,
    we have $[w/w_1](T_W(w_1) - \{w_2\}) = [w/w_1](T_W(w) - \{w_2\}) = T_{W'}(w) - \{w_2\}$.

(3) First, let W be a w_2-sanitized world sequent. Let us prove that W' is w_2-sanitized as well.

3. W' is w_2-elementary:
   - We already proved that W' is well-formed and non-cyclic.
   - Suppose $u'_1 \doteq u'_2 \circ u'_3 \in W'$ for some heaps $u'_i \in W'$ $(i = 1, 2, 3)$.
   - By (1), $u_1 \doteq u_2 \circ u_3 \in W$ for some heaps $u_i \in W$ such that $[w/w_1]u_i = u'_i$ $(i = 1, 2, 3)$.

- By (2) and that W is w_2-elementary, we have [Lemma D.11-(3) with $w \notin \mathcal{T}(W)$]

$$\begin{aligned}&(T_{W'}(u'_2) - \{w_2\}) \cap (T_{W'}(u'_3) - \{w_2\})\\ &= \big([w/w_1](T_W(u_2) - \{w_2\})\big) \cap \big([w/w_1](T_W(u_3) - \{w_2\})\big)\\ &= [w/w_1]\big((T_W(u_2) - \{w_2\}) \cap (T_W(u_3) - \{w_2\})\big)\\ &= [w/w_1]\varnothing = \varnothing.\end{aligned}$$

4. W' is w_2-consistent:
   - Suppose $\{u'_1 \doteq u'_2 \circ u'_3, v'_1 \doteq v'_2 \circ v'_3\} \subset W'$ for some heaps $u'_i, v'_i \in W'$ $(i = 1, 2, 3)$ with $u'_1 = v'_1$.
   - By (1), $\{u_1 \doteq u_2 \circ u_3, v_1 \doteq v_2 \circ v_3\} \subset W$ for some heaps $u_i, v_i \in W$ such that $[w/w_1]u_i = u'_i$ and $[w/w_1]v_i = v'_i$ $(i = 1, 2, 3)$.
   - By (2) and that W is w_2-consistent, we have [Lemma D.11-(3)]

$$\begin{aligned}&(T_{W'}(u'_2) - \{w_2\}) \cup (T_{W'}(u'_3) - \{w_2\})\\ &= \big([w/w_1](T_W(u_2) - \{w_2\})\big) \cup \big([w/w_1](T_W(u_3) - \{w_2\})\big)\\ &= [w/w_1]\big((T_W(u_2) - \{w_2\}) \cup (T_W(u_3) - \{w_2\})\big)\\ &= [w/w_1](T_W(u_1) - \{w_2\}).\end{aligned}$$

   - Similarly, we have $(T_{W'}(v'_2) - \{w_2\}) \cup (T_{W'}(v'_3) - \{w_2\}) = [w/w_1](T_W(v_1) - \{w_2\})$.
   - Since $T_W(w) - \{w_2\} = T_W(w_1) - \{w_2\}$, $T_W(u_1) - \{w_2\} = T_W(v_1) - \{w_2\}$.
5. W' is w_2-full:
   - Let $r' \in \mathcal{R}(W')$ and let $S' \subset T_{W'}(r') - \{w_2\}$ with $S' \neq \varnothing$.
   - **Case** $r' = w_2$: Since $w_2 \in \mathcal{T}(W')$ by (1), we have $S' = \varnothing$. ⨳
   - **Case** $r' \neq w_2$:
     - By (1) and $r' \neq w_2$, $r' \in \mathcal{R}(W)$.
     - By (2) and $r' \neq w_1$, $S' \subset [w/w_1](T_W(r') - \{w_2\})$.
     - Moreover, $[w_1/w]S' \subset [w_1/w][w/w_1](T_W(r') - \{w_2\}) = T_W(r') - \{w_2\}$ [Lemma D.11-(1) with $w \notin \mathcal{T}(W)$]
     - Since W is w_2-full, $T_W(u) - \{w_2\} = [w_1/w]S'$ for some heap $u \in W$.
     - By (2), $T_{W'}([w/w_1]u) - \{w_2\} = [w/w_1](T_W(u) - \{w_2\}) = [w/w_1][w_1/w]S' = S'$. [Lemma D.11-(1) with $w_1 \notin \mathcal{T}(W')$]

9. W' is w_2-sanitized: The proof is easy by using (1) and that W is w_2-sanitized.

Next, let W be a world sequent which is w_2-$\star$-ready for heap $\alpha \in W$. The proof of that W' is w_2-$\star$-ready for heap $[w/w_1]\alpha \in W'$ is similar to that of 5., using (1), (2) and that W is w_2-$\star$-ready for α.

Finally, let W be a world sequent whose heap relations are $\{u \doteq u_{123} \circ u_4, u \doteq u_1 \circ u_{234}, u_{123} \doteq u_1 \circ u_{23}, u_{234} \doteq u_{23} \circ u_4, u_{234} \doteq u_2 \circ u_{34}, u_{23} \doteq u_2 \circ u_3, u_{34} \doteq u_3 \circ u_4, u_4 \doteq \epsilon\}$, where every heap is distinct. After letting $w := u_{234}$, $w_1 := u_{23}$, and $w_2 := u_4$, define W' be a world sequent such that $\{W\} \mapsto \{W'\}$ by applying the rule NormPC to $\{w \doteq w_1 \circ w_2, w_2 \doteq \epsilon\}$. Then, W' is not elementary while W is consistent.

□

D.2. **Proof search strategy $\mathcal{SS}$.** Before we prove Lemma 7.1, we introduce several auxiliary lemmas about how to apply structural rules to obtain a world sequent having specific properties.

Lemma D.13. Let W be any elementary world sequent with a heap w such that:

(1) $w \doteq u_{1,1} \circ u_{1,2}, \cdots, w \doteq u_{n,1} \circ u_{n,2}, w \doteq v_1 \circ v_2$ are all the heap relations in W at w for some $n \geq 1$;
(2) v_1, v_2 are terminal heaps in W;
(3) $w \succ v_j$ $(j = 1, 2)$, where $\alpha \succ \beta \iff \begin{cases} p_1 \text{ is the only parent heap of } \beta, \text{ and } \alpha \nearrow p_1, \text{ or} \\ p_1, p_2 \text{ are the only parent heaps of } \beta, \text{ and } \alpha \nearrow p_1 \text{ and } \alpha \succ p_2; \end{cases}$
(4) $\bigcup_{j=1,2} T_W(u_{i,j}) = \{t_1, \cdots, t_m\}$ for some $m \geq 1$ $(i = 1, \cdots, n)$;
(5) $\{v_1, v_2\} \cap \{t_1, \cdots, t_m\} = \varnothing$;
(6) For any heap $\alpha \in W$ with $\alpha \nearrow_W w$ and $\alpha \neq w$, W is consistent at α.

Let $u_j := u_{1,j}$ $(j = 1, 2)$. Then, there exists an elementary world sequent W' such that:

(a) $\{W\} \mapsto^* \{W'\}$ by applying only the rule $\mathsf{Disj}\star$;
(b) For any heap relation $\alpha' \doteq \cdot \circ \cdot \in W' - W$, we have $\alpha' \nearrow_{W'} w$;
(c) For any heap $\alpha \in W$ with $\alpha \nearrow_W w$ and $\alpha \neq v_1, v_2$, and for any terminal heap $t' \in W' - W$ in W',
$t' \nearrow_{W'} \alpha$ implies $t' \nearrow_{W'} t_k$ and $t_k \nearrow_{W'} \alpha$ for some $0 < k \leq m$;
(d) $v_1, v_2, t_1, \cdots, t_m$ are not terminal heaps in W';
(e) For any heap $\alpha' \in W'$ with $\alpha' \nearrow_{W'} w$, W' is consistent at α'.

Proof. The proof proceeds by induction on m. Note that since W is elementary, $m \geq 2$.

i) $m = 2$:
We construct W' as follows:
- Let W' be a world sequent obtained by applying $\mathsf{Disj}\star$ to W with $\{w \doteq u_1 \circ u_2, w \doteq v_1 \circ v_2\}$.
- Then, $W' = W \cup \{u_1 \doteq \eta'_1 \circ \eta'_2, u_2 \doteq \eta'_3 \circ \eta'_4, v_1 \doteq \eta'_1 \circ \eta'_3, v_2 \doteq \eta'_2 \circ \eta'_4\}$ for fresh heaps $\eta'_1, \cdots, \eta'_4$.

Let us show that W' is elementary.
- W' is non-cyclic and well-formed. [Lemma D.5-(2)]
- Suppose that W' is not elementary. Then $\alpha' \wedge_{W'} \alpha'$ for some $\alpha' \in W'$.
- Since W is elementary, $\alpha' \in W' - W$. WLOG, We can assume $\alpha' = \eta'_1$. [Lemma D.5-(1)]
- Since W is elementary, we have $u_1 \wedge_{W'} v_1$ and thus $u_1 \wedge_W v_1$. [Lemma D.5-(1)]
- By (3) and that W is elementary, we get a contradiction. ※

Now, let us show that W' satisfies (a)-(e). First, (a)-(b) are clearly true. Moreover, (c)-(e) are true because the followings hold:
- By (4) and that W is elementary, $\{u_{i,1}, u_{i,2}\} = \{t_1, t_2\}$ $(i = 1, \cdots, n)$.
- By (2), (5) and that W is well-formed, u_1, u_2, v_1, v_2 are all distinct terminal heaps in W.
- Thus, there is a unique heap relation in W' at each of u_1, u_2, v_1, v_2.

ii) $m > 2$:
We first construct W^1 as follows:
- Let W^1 be a world sequent obtained by applying $\mathsf{Disj}\star$ to W with $\{w \doteq u_1 \circ u_2, w \doteq v_1 \circ v_2\}$.
- Then, $W^1 = W \cup \{u_1 \doteq \eta^1_1 \circ \eta^1_2, u_2 \doteq \eta^1_3 \circ \eta^1_4, v_1 \doteq \eta^1_1 \circ \eta^1_3, v_2 \doteq \eta^1_2 \circ \eta^1_4\}$ for fresh heaps $\eta^1_1, \cdots, \eta^1_4$.
- W^1 is elementary by the same argument above.

Suppose that u_1 and u_2 are not terminal heaps in W. (The other cases are similarly proved, so we omit them.) Then, $\{u_1 \doteq \zeta_1 \circ \zeta_2, u_2 \doteq \zeta_3 \circ \zeta_4\} \subset W$ for some heaps $\zeta_1, \cdots, \zeta_4 \in W$. WLOG, we can assume that $\bigcup_{j=1,2} T_W(\zeta_j) = \{t_1, \cdots, t_l\}$ and $\bigcup_{j=3,4} T_W(\zeta_j) = \{t_{l+1}, \cdots, t_m\}$ for some $0 < l < m$.

In order to apply IH, let us show that W^1 with u_1 satisfies the assumptions $(2)^1$-$(6)^1$ of this Lemma.

$(2)^1$ This assumption is clearly true.

$(3)^1$
- Let $1 \leq j \leq 2$. Then, u_1, v_j are the only parent heaps of η_j^1 in W^1.
- By (3) and that $\cdot \doteq \alpha^1 \circ \cdot \in W^1 - W$ implies $\alpha^1 \notin W$, $w \succ v_j$ in W and thus in W^1.
- By using $u_1 \nearrow_{W^1} w$, we have $u_1 \succ v_j$ in W^1. Thus, $u_1 \succ \eta_j^1$ in W^1.

$(*)^1$ Before proving the remaining assumptions, we prove that: for any heap $\alpha^1 \in W^1$ with $\alpha^1 \nearrow_{W^1} u_1$ and $\alpha^1 \neq u_1, \eta_1^1, \eta_2^1$, we have $\alpha^1 \in W$ and $T_{W^1}(\alpha^1) = T_W(\alpha^1)$.

It is clear that $\alpha^1 \in W$, and the proof of $T_{W^1}(\alpha^1) = T_W(\alpha^1)$ is as follows:

$(\supset)$
- Suppose $t \in T_W(\alpha^1)$. Then, t is a terminal heap in W^1 since $t \neq v_1, v_2$ by (5).

$(\subset)$
- Suppose $t^1 \in T_{W^1}(\alpha^1)$.
- **Case** $t^1 \in W$:
  - t^1 is a terminal heap in W, and $t^1 \nearrow_W \alpha^1$ since $\alpha^1 \in W$. [Lemma D.5-(1)]
- **Case** $t^1 \in W^1 - W$:
  - $t^1 = \eta_j^1$ for some $1 \leq j \leq 4$.
  - Thus, by (3) and that W is elementary, $t^1 \nearrow_{W^1} \alpha^1$ holds. ※

$(4)^1$-$(6)^1$ We can easily prove these assumptions by using $(*)^1$, (6), and $\eta_1^1, \eta_2^1 \notin W$.

Since $l < m$ holds and W^1 with u_1 satisfies the assumptions of this Lemma, we can apply IH on W^1 with u_1. Hence, there exists an elementary world sequent W^2 such that:

$(a)^1$ $\{W^1\} \mapsto^* \{W^2\}$ by applying only the rule $\mathsf{Disj}\star$;

$(b)^1$ For any heap relation $\alpha^2 \doteq \cdot \circ \cdot \in W^2 - W^1$, we have $\alpha^2 \nearrow_{W^2} u_1$;

$(c)^1$ For any heap $\alpha^1 \in W^1$ with $\alpha^1 \nearrow_{W^1} u_1$ and $\alpha^1 \neq \eta_1^1, \eta_2^1$, and for any terminal heap $t^2 \in W^2 - W^1$ in W^2, $t^2 \nearrow_{W^2} \alpha^1$ implies $t^2 \nearrow_{W^2} t_k$ and $t_k \nearrow_{W^2} \alpha^1$ for some $0 < k \leq l$;

$(d)^1$ $\eta_1^1, \eta_2^1, t_1, \cdots, t_l$ are not terminal heaps in W^2;

$(e)^1$ For any heap $\alpha^2 \in W^2$ with $\alpha^2 \nearrow_{W^2} u_1$, W^2 is consistent at α^2.

In order to apply IH, let us show that W^2 with u_2 satisfies the assumptions $(2)^2$-$(6)^2$ of this Lemma.

$(2)^2$ η_3^1, η_4^1 are terminal heaps in W^1, and so in W^2 by $(b)^1$ and that W^2 is elementary.

$(3)^2$
- Let $3 \leq j \leq 4$. Then, similarly to $(3)^1$, we can prove that $u_2 \succ \eta_j^1$ in W^1.
- Since $\cdot \doteq \alpha^2 \circ \cdot \in W^2 - W^1$ implies $\alpha^2 \notin W^1$, $u_2 \succ \eta_j^1$ in W^2.

$(*)^2$ Before proving the remaining assumptions, we prove that: for any heap $\alpha^2 \in W^2$ with $\alpha^2 \nearrow_{W^2} u_2$ and $\alpha^2 \neq u_2, \eta_3^1, \eta_4^1$, we have $\alpha^2 \in W^1$ and $T_{W^2}(\alpha^2) = T_W(\alpha^2)$.

By $(b)^1$ and that W^2 is elementary, we have $\alpha^2 \in W^1$. Moreover, similarly to $(*)^1$, we can prove that $T_{W^1}(\alpha^2) = T_W(\alpha^2)$. Finally, the proof of $T_{W^2}(\alpha^2) = T_{W^1}(\alpha^2)$ is as follows:

($\supset$)
- Suppose $t^1 \in T_{W^1}(\alpha^2)$.
- Then, t^1 is a terminal heap in W^2 by $(b)^1$ and that W^2 is elementary.

($\subset$)
- Suppose $t^2 \in T_{W^2}(\alpha^2)$.
- **Case** $t^2 \in W^1$:
  - t^2 is a terminal heap in W^1, and $t^2 \nearrow_{W^1} \alpha^2$ since $\alpha^2 \in W^1$. [Lemma D.5-(1)]
- **Case** $t^2 \in W^2 - W^1$:
  - $t^2 \nearrow_{W^2} \alpha^2$ by $(b)^1$ and that W^2 is elementary. ※

$(4)^2$-$(6)^2$ We can easily prove these assumptions by using $(*)^2$, (6), and $\eta_3^1, \eta_4^1 \notin W$.

Since $m - l < m$ holds and W^2 with u_2 satisfies the assumptions of this Lemma, we can apply IH on W^2 with u_2. Hence, there exists an elementary world sequent W^3 such that:

- $(a)^2$ $\{W^2\} \mapsto^* \{W^3\}$ by applying only the rule $\mathsf{Disj}\star$;
- $(b)^2$ For any heap relation $\alpha^3 \doteq \cdot \circ \cdot \in W^3 - W^2$, we have $\alpha^3 \nearrow_{W^3} u_2$;
- $(c)^2$ For any heap $\alpha^2 \in W^2$ with $\alpha^2 \nearrow_{W^2} u_2$ and $\alpha^2 \neq \eta_3^1, \eta_4^1$, and for any terminal heap $t^3 \in W^3 - W^2$ in W^3, $t^3 \nearrow_{W^3} \alpha^2$ implies $t^3 \nearrow_{W^3} t_k$ and $t_k \nearrow_{W^3} \alpha^2$ for some $l < k \leq m$;
- $(d)^2$ $\eta_3^1, \eta_4^1, t_{l+1}, \cdots, t_m$ are not terminal heaps in W^3;
- $(e)^2$ For any heap $\alpha^3 \in W^3$ with $\alpha^3 \nearrow_{W^3} u_2$, W^3 is consistent at α^3.

Finally, let us show that $W' := W^3$ satisfies (a)-(e).

- (a) It is clear from the construction of W^1, $(a)^1$, and $(a)^2$.
- (b) It is clear from the construction of W^1, $(b)^1$, and $(b)^2$.
- (d) It is clear from the construction of W^1, $(d)^1$, and $(d)^2$
- (c)
  - Let $\alpha \in W$ be any heap with $\alpha \nearrow_W w$ and $\alpha \neq v_1, v_2$.
  - Let $t' \in W' - W$ be any terminal heap in W', and assume that $t' \nearrow_{W'} \alpha$.
  - Then, by $(d)^1$ and $(d)^2$, $t' \notin W^1$.
  - **Case** $\alpha \nearrow_W u_1$:
    - $\alpha \neq \eta_1^1, \eta_2^1$ since $\alpha \in W$.
    - Moreover, we know $t' \in W^2$ by $(b)^2$ and that W' is elementary.
    - Hence, $t' \nearrow_{W^2} \alpha$ holds, and by using $(c)^1$ we get a desired result. [Lemma D.5-(1)]
  - **Case** $\alpha \nearrow_W u_2$:
    - Then, $\alpha \neq \eta_3^1, \eta_4^1$ since $\alpha \in W$.
    - Moreover, we know $t' \notin W^2$ by $(b)^1$ and that W' is elementary.
    - Hence, by using $(c)^2$, we get a desired result.
  - **Case** $\alpha \not\nearrow_W u_1$ and $\alpha \not\nearrow_W u_2$:
    - Since $\alpha \in W$ and $t' \in W' - W$, there exist heaps $\beta \in W$ and $\beta' \in W' - W$ such that $t' \nearrow_{W'} \beta'$, $\beta \nearrow_{W'} \alpha$, and $\beta \doteq \beta' \circ \cdot \in W' - W$.
    - By $(b)^1$ and $(b)^2$, we have $\beta \nearrow_W u_j$ for some $1 \leq j \leq 2$. [Lemma D.5-(1)]
    - Hence, by using the above cases, we get a desired result.
- (*) Before proving (e), we prove that: for any heap $\alpha^3 \in W^3$ with $\alpha^3 \nearrow_{W^3} u_1$ and $\alpha^3 \neq u_1, \eta_1^1, \eta_2^1$, we have $\alpha^3 \in W^2$ and $T_{W^3}(\alpha^3) = T_{W^2}(\alpha^3)$.

The proof of this claim is similar to that of $(*)^2$, so we omit it. (The only difference is that we use $(b)^2$ and that W^3 is elementary, instead of $(b)^1$ and that W^2 is elementary.)

(#) In addition to (*), we prove that: for any heap $\alpha \in W$ with $\alpha \nearrow_W w$ and $\alpha \neq w, v_1, v_2$,

$$T_{W'}(\alpha) = \bigcup_{t \in T_W(\alpha)} T_{W'}(t).$$

The proof is as follows:

($\supset$)
- This direction is obvious.

($\subset$)
- Let $t' \in T_{W'}(\alpha)$. We want to show $t' \in \bigcup_{t \in T_W(\alpha)} T_{W'}(t)$.
- **Case** $t' \in W$:
  - Since $t' \nearrow_W \alpha$ and t' is a terminal heap in W, $t' \in T_W(\alpha)$. [Lemma D.5-(1)]
  - Since $T_{W'}(t') = \{t'\}$, we get a desired result.
- **Case** $t' \in W' - W$:
  - By (c), $t' \nearrow_{W'} t_k$ and $t_k \nearrow_{W'} \alpha$ for some $0 < k \leq m$.
  - Since $t_k \nearrow_W \alpha$, we have $t_k \in T_W(\alpha)$ so we are done. [Lemma D.5-(1)]

(e)
- Let $\alpha' \in W'$ be any heap with $\alpha' \nearrow_{W'} w$.
- **Case** $\alpha' \nearrow_{W'} u_1$:
  - By $(b)^2$ and that W^3 is elementary, heap relations at u_1 in W^2 are the same as that in W^3.
  - Hence, W' is consistent at α' by $(e)^1$ with (*).
- **Case** $\alpha' \nearrow_{W'} u_2$: Then, W' is consistent at α' by $(e)^2$.
- **Case** $\alpha' = v_1$: (We omit the proof for v_2.)
  - By $(b)^1$, $(b)^2$, (5), $v_1 \doteq \eta_1^1 \circ \eta_2^1$ is the only heap relation at v_1 in W'. [Lemma D.5-(1)]
  - Hence, it is obvious that W' is consistent at α'.
- **Case** $\alpha' \not\nearrow_{W'} u_1$, $\alpha' \not\nearrow_{W'} u_2$, and $\alpha' \neq w, v_1, v_2$:
  - Let $\{\alpha' \doteq \alpha'_1 \circ \alpha'_2, \alpha' \doteq \alpha'_3 \circ \alpha'_4\} \subset W'$.
  - By $(b)^1$ and $(b)^2$, $\alpha' \in W$ and heap relations at α' in W are the same as that in W'.
  - Thus, $\{\alpha' \doteq \alpha'_1 \circ \alpha'_2, \alpha' \doteq \alpha'_3 \circ \alpha'_4\} \subset W$.
  - By (#) and (6), W' is consistent at α' since we have

$$\bigcup_{j=1,2} T_{W'}(\alpha'_j) = \bigcup_{j=1,2} \bigcup_{t \in T_W(\alpha_j)} T_{W'}(t) = \bigcup_{j=3,4} \bigcup_{t \in T_W(\alpha_j)} T_{W'}(t) = \bigcup_{j=3,4} T_{W'}(\alpha'_j).$$

- **Case** $\alpha' = w$:
  - By (#) and that W' is consistent at u_1 and u_2, W' is consistent at α' since we have

$$\bigcup_{j=1,2} T_{W'}(u_{i,j}) = \bigcup_{j=1,2} \bigcup_{t \in T_W(u_{i,j})} T_{W'}(t) = \bigcup_{1 \leq k \leq m} T_{W'}(t_k)$$

$$\text{and } \bigcup_{j=1,2} T_{W'}(v_j) = \bigcup_{1 \leq j \leq 4} T_{W'}(\eta_j^1) = \bigcup_{j=1,2} T_{W'}(u_j)$$

for any $1 \leq i \leq n$.

□

Lemma D.14. Let W be any world sequent with heaps u, v, w satisfying one of the followings: 1) $u \wedge^w_W v$; 2) $u \nearrow_W v$ and $u \neq v$. Then, there exists a world sequent W' with a heap w' such that:

- $\{W\} \mapsto^* \{W'\}$ by applying only the rule Assoc;
- The corresponding one of the followings holds:
  1) $w' \doteq u \circ v \in W'$ and $w' \nearrow_{W'} w$;
  2) $v \doteq u \circ w' \in W'$ for some heap $w' \in W'$.

Proof. The proof is straightforward. □

Lemma D.15. Let W be any consistent world sequent and $w \in W$ be any heap. Suppose $T_W(w) = \{w_1, \cdots, w_n\}$ for some $n \geq 2$. Then, there exists a world sequent W' with a heap w' such that:

- $\{W\} \mapsto^* \{W'\}$ by applying only the rule Assoc;
- $w \doteq w_1 \circ w' \in W'$;
- For any heap $u' \in W' - W$, $|T_{W'}(u')| < n$.

Proof. The proof is straightforward. □

Lemma D.16. Let W be a t-sanitized world sequent, where $t \in \mathcal{T}(W)$ and $t \doteq \epsilon \in W$. Then, there exists a t-sanitized world sequent W' such that:

- $\{W\} \mapsto^* \{W'\}$ by applying only the rule NormPC;
- There is no heap relation of the form $\cdot \doteq \cdot \circ t$ in W'.

Proof. The proof proceeds by induction on $m :=$ (the number of heap relations in W of the form $\cdot \doteq \cdot \circ t$). In the proof, we exploit Lemma D.12-(3), and a key step is to deduce that when $m > 0$, there exist heaps $w, w_1 \in W$ such that: $w \doteq w_1 \circ t \in W$ and for any $n \geq 2$, there do not exist heaps $u_0, \cdots, u_n \in W$ with $w_1 = u_0, \{u_1 \doteq u_0 \circ t, \cdots, u_n \doteq u_{n-1} \circ t\} \subset W, u_n = w_2$. We can always find such heaps w and w_1 in W because W is well-formed and non-cyclic and there are only finitely many heaps in W. □

Lemma D.17. Let W be a sanitized world sequent and w_ϵ be the special empty heap in W. Then, there exists a sanitized world sequent W' such that:

- $\{W\} \mapsto^* \{W'\}$ by applying only the rules NormPC and NormEmpty;
- There is no empty heap except w_ϵ in W'.

Proof. The proof proceeds by induction on $n :=$ (the number of empty heaps in W except w_ϵ).

i) $n = 0$: There is nothing to prove.
ii) $n > 0$:
  - Let $w \doteq \epsilon \in W$ where $w \neq w_\epsilon$.
  - Since W is sanitized, $w \in \mathcal{T}(W)$ and W is w-sanitized.
  - There exists a w-sanitized world sequent W^1 such that: [Lemma D.16]
    - ▷ $\{W\} \mapsto^* \{W^1\}$ by applying only the rule NormPC;
    - ▷ There is no heap relation of the form $\cdot \doteq \cdot \circ w$ in W^1.
  - Let W^2 be a world sequent obtained by applying NormEmpty to W^1 with $\{w_\epsilon \doteq \epsilon, w \doteq \epsilon\}$.
  - Then, W^2 is sanitized and the number of empty heaps in W^2 except w_ϵ is $n-1$.

- Thus, by IH on W^2, we obtain a desired world sequent W'.

□

Lemma D.18. Let W be a sanitized world sequent which has no empty heap except its special empty heap w_ϵ. Then, for any $n \geq 1$, there exists a sanitized world sequent W' such that:

(a) $\{W\} \mapsto^* \{W'\}$ by applying only the rules NormEq and Assoc;
(b) There is no empty heap except w_ϵ in W';
(c) For any $r' \in \mathcal{R}(W')$ and any non-empty set $S' \subset T_{W'}(r')$ such that $|S'| \leq n$, there exists a unique heap $w' \in W'$ with $T_{W'}(w') = S'$;
(d) For any heap $\alpha' \in W' - W$, $|T_{W'}(\alpha')| < n$.

Proof. The proof proceeds by nested induction on n and $\chi(W; n)$, where $\chi(W; n)$ is defined as:

$$\chi(W;n) := \sum_{r \in \mathcal{R}(W)} \sum_{\substack{S \subset T_W(r) \\ |S|=n}} \{(\#\text{ of heaps } w \in W \text{ such that } T_W(w) = S) - 1\}.$$

Note that $\chi(W; n) \geq 0$ for any $n \geq 1$ since W is full.

i) $n = 1$:
- We want to show that $W' := W$ satisfies (a)-(d). Trivially, W satisfies (a), (b), and (d).
- To show that W satisfies (c), let $S = \{t\} \subset \mathcal{T}(W)$ and $\alpha \in W$ be any heap with $T_W(\alpha) = S$.
- Since W is elementary, $\alpha \in \mathcal{T}(W)$ and thus $\alpha = t$. Hence, W satisfies (c).

ii) $n > 1$:
   i. $\chi(W; n) = 0$:
      - By IH on $n - 1$, we easily obtain a desired world sequent W'. [Lemma D.6-(1) and D.8-(1), (2)]
   ii. $\chi(W; n) > 0$:
      - In the following proof, we apply only the rules NormEq and Assoc. Thus, all the following world sequents are sanitized and satisfy (a) and (b). [Lemma D.6-(2) and D.8-(3)]
      - Since $\chi(W; n) > 0$, there exist $r \in \mathcal{R}(W)$, $S \subset T_W(r)$, and heaps $u, v \in W$ such that $u \neq v$ and $T_W(u) = T_W(v) = S$, where $S = \{t_1, \cdots, t_n\}$.
      - There exists a world sequent W^1 with heaps u^1 and v^1 such that: [Lemma D.15]
        (a)^1 $\{W\} \mapsto^* \{W^1\}$ by applying only the rule Assoc;
        (*)^1 $\{u \doteq t_1 \circ u^1, v \doteq t_1 \circ v^1\} \subset W^1$;
        (d)^1 For any heap $\alpha^1 \in W^1 - W$, $|T_{W^1}(\alpha^1)| < n$.
      - $\chi(W^1; n) = \chi(W; n)$ by (d)^1, and $T_{W^1}(r) = T_W(r)$. [Lemma D.6-(1)]
      - Since $T_{W^1}(u) = S = T_{W^1}(v)$, $T_{W^1}(u^1) = S - \{t_1\} = T_{W^1}(v^1)$. [Lemma D.6-(1)]
      - By IH on W^1 with $n - 1$, there exists a world sequent W^2 such that:
        (a)^2 $\{W^1\} \mapsto^* \{W^2\}$ by applying onlythe rules NormEq and Assoc;
        (c)^2 For any $r^2 \in \mathcal{R}(W^2)$ and any non-empty set $S^2 \subset T_{W^2}(r^2)$ such that $|S^2| \leq n - 1$,
        there exists a unique heap $w^2 \in W^2$ with $T_{W^2}(w^2) = S^2$;
        (d)^2 For any heap $\alpha^2 \in W^2 - W^1$, $|T_{W^2}(\alpha^2)| < n$.

- $\chi(W^2; n) \leq \chi(W^1; n)$ by $(\mathrm{d})^2$, and $T_{W^2}(r) = T_{W^1}(r)$. [Lemma D.6-(1) and D.8-(1), (2)]
- WLOG, assume that $\chi(W^2; n) = \chi(W^1; n)$.
- Then, we have $\{u \doteq t_1 \circ \eta^2, v \doteq t_1 \circ \eta^2\} \subset W^2$ for some heap $\eta^2 \in W^2$ by using $(*)^1$ and $(\mathrm{c})^2$ with $S - \{t_1\} \subset T_W(r) = T_{W^2}(r)$.
- Let W^3 be a world sequent obtained by applying NormEq to $\{u \doteq t_1 \circ \eta^2, v \doteq t_1 \circ \eta^2\}$.
- Then, $\chi(W^3; n) < \chi(W^2; n) = \chi(W; n)$.
- Hence, by IH on W^3 with n and $\chi(W^3; n)$, we get a desired world sequent W'.

□

We now prove the remaining lemmas and propositions.

Lemma 7.1. For a world sequent W of a particular kind, there exists a world sequent W' of another kind such that $\{W\} \mapsto^* \{W'\}$ by applying only the structural rules, where one of the following holds:

11. $\to$ 4. W is expanded and W' is consistent;
4. $\to$ 5. W is consistent and W' is full;
5. $\to$ 6. W is full and W' is $\star$-ready for a given heap w;
5. $\to$ 7. W is full and W' is $-\!\!\star$-ready for a given heap w;
8. $\to$ 9. W is saturated and $\star$-ready ($-\!\!\star$-ready) for a given heap w, and W' is sanitized and $\star$-ready ($-\!\!\star$-ready) for heap w;
9. $\to$10. W is sanitized and $\star$-ready ($-\!\!\star$-ready) for a given heap w, and W' is normalized and $\star$-ready ($-\!\!\star$-ready) for heap w.

Proof. In the proof, we use the structural rules $\mathsf{Disj}\star$, Assoc, NormPC, Weaken, NormEq, $\mathsf{NormEmpty}$, and NormEq. The proof for each step is as follows.

11.$\to$4. In this step, we use only the rule $\mathsf{Disj}\star$.

By Lemma D.4, it suffices to show that: for any world sequents W, W^1 satisfying that W is consistent and $\{W\} \mapsto \{W^1\}$ by applying the rule $\star\mathsf{L}$, we can construct a consistent world sequent W' by applying only the rule $\mathsf{Disj}\star$ to W^1. The proof is as follows.

- We have $W^1 = W \cup \{w \doteq \cdot \circ \cdot\}$ for some heap $w \in W$, and W^1 is elementary. [Lemma D.4]
- There exists an elementary world sequent W' such that: [Lemma D.13]
  - ▷ $\{W^1\} \mapsto^* \{W'\}$ by applying only the rule $\mathsf{Disj}\star$;
  - ▷ For any heap relation $\alpha' \doteq \cdot \circ \cdot \in W' - W^1$, we have $\alpha' \nearrow_{W'} w$;
  - ▷ For any heap $\alpha' \in W'$ with $\alpha' \nearrow_{W'} w$, W' is consistent at α'.
- By slightly extending the proof of Lemma D.13, we obtain that: for any heap $\alpha \in W$,

$$T_{W'}(\alpha) = \bigcup_{t \in T_W(\alpha)} T_{W'}(t).$$

- From the above observations and that W is consistent, it is easy to check that W' is consistent.

Figure 5 shows an example of applying the rule $\mathsf{Disj}\star$ to get a consistent world sequent from a expanded world sequent.

However, the way to apply the rule $\mathsf{Disj}\star$ to get a consistent world sequent, which is described above, is not obvious at all. Figure 6 illustrates that if we carelessly apply the rule $\mathsf{Disj}\star$ even once, it would be impossible to obtain a consistent world sequent. Suppose that we apply the rule $\mathsf{Disj}\star$ to $\{w \doteq w_1 \circ w_2, w \doteq w_5 \circ w_6\}$ in the third world sequent only because $T(w_1) \cup T(w_2) \neq T(w_5) \cup T(w_6)$, as shown in the figure. Then, this application makes us impossible to get a consistent world sequent (no matter how we apply the rule $\mathsf{Disj}\star$ from that point) since: the situation of w_i $(i = 1, \cdots, 6)$ in the fourth world sequent exactly happens to u_j $(j = 1, \cdots, 12)$ and to their descendant heaps in the following world sequents, whenever we apply the rule $\mathsf{Disj}\star$.

4.→5. In this step, we use only the rule Assoc.

Let W be any consistent world sequent. By Lemma D.6, it suffices to show that: for any $r \in \mathcal{R}(W)$ and any non-empty $S \subset T_W(r)$, we can construct a world sequent W' with a heap w', by applying only the rule Assoc to W, such that $T_{W'}(w') = S$ and $w' \nearrow_{W'} r$. The proof proceeds by induction on $n = |S|$. (We omit the use of Lemma D.6 in the following proof.)

i) $n = 1, 2$:
- There is nothing to prove for $n = 1$, so let $n = 2$ and $S = \{t_1, t_2\}$.
- Since $t_1 \Uparrow^r_W t_2$, $t_1 \wedge^w_W t_2$ and $w \nearrow_W r$ for some heap $w \in W$. [Lemma D.3-(2)]
- There exists a world sequent W' with a heap w' such that: [Lemma D.14]
  - ▷ $\{W\} \mapsto^* \{W'\}$ by applying only the rule Assoc;
  - ▷ $w' \doteq t_1 \circ t_2 \in W'$ and $w' \nearrow_{W'} w$.

ii) $n > 2$:
- Let $S = \{t_1, \cdots, t_n\}$.
- By IH on $S - \{t_n\}$, there exists a world sequent W^1 with a heap w^1 such that:
  - ▷ $\{W\} \mapsto^* \{W^1\}$ by applying only the rule Assoc;
  - ▷ $T_{W^1}(w^1) = S - \{t_n\}$ and $w^1 \nearrow_{W^1} r$.
- There exists a world sequent W^2 with a heap w^2 such that: [Lemma D.14]
  - ▷ $\{W^1\} \mapsto^* \{W^2\}$ by applying only the rule Assoc;
  - ▷ $r \doteq w^1 \circ w^2 \in W^2$.
- Since $t_n \in T_{W^2}(r) - T_{W^2}(w^1)$ and W^2 is consistent, $t_n \in T_{W^2}(w^2)$. Thus, $w^1 \wedge^r_{W^2} t_n$.
- There exists a world sequent W' with a heap w' such that: [Lemma D.14]
  - ▷ $\{W^2\} \mapsto^* \{W'\}$ by applying only the rule Assoc;
  - ▷ $w' \doteq w^1 \circ t_n \in W'$ and $w' \nearrow_{W'} r$.

5.→6. In this step, we use only the rule Assoc.

Let W be any full world sequent and w be any heap in W. By Lemma D.6, it suffices to show that: for any non-empty sets $S_1, S_2 \subset T_W(w)$ such that $S_1 \cap S_2 = \varnothing$ and $S_1 \cup S_2 = T_W(w)$, we can construct a world sequent W' with heaps w'_1 and w'_2, by applying only the rule Assoc to W, such that $w \doteq w'_1 \circ w'_2 \in W'$ and $T_{W'}(w'_i) = S_i$ $(i = 1, 2)$. The proof proceeds by induction on $n = |T_W(w)|$. (We omit the use of Lemma D.6 in the following proof.)

i) $n = 2$:
- Since $n > 1$, $w \doteq w_1 \circ w_2 \in W$ for some heaps $w_1, w_2 \in W$.

- Since $n = 2$, we have $S_i = \{t_i\}$ $(i = 1, 2)$ for some heaps $w_1 \neq w_2$ in W.
- Since W is elementary, $\{T_W(w_1), T_W(w_2)\} = \{\{t_1\}, \{t_2\}\}$ so $\{w_1, w_2\} = \{t_1, t_2\}$ holds.

ii) $n > 2$:

- Since $n > 1$, $w \doteq w_1 \circ w_2 \in W$ for some heaps $w_1, w_2 \in W$.
- Let $U_i := T_W(w_i)$ and $S_{ij} := S_i \cap U'_j$ $(i, j = 1, 2)$.
- Since W is consistent, $U_1 \cap U_2 = \varnothing$ and $U_1 \cup U_2 = T_W(w)$.
- We can assume that $S_i \neq U'_j$ for any $1 \leq i, j \leq 2$. (Otherwise, there is nothing to prove.)
- **Case** $S_{ij} = \varnothing$ for some $1 \leq i, j \leq 2$:
  - WLOG, let $i = 1$ and $j = 2$. Note that $S_1 \neq \varnothing$ and $S_2 - U_2 \neq \varnothing$.
  - By IH on w_1, there exists a world sequent W^1 with heaps w^1_1, w^1_2 such that:
    - $\{W\} \mapsto^* \{W^1\}$ by applying only the rule Assoc;
    - $w_1 \doteq w^1_1 \circ w^1_2 \in W^1$, $T_{W^1}(w^1_1) = S_1$, and $T_{W^1}(w^1_2) = S_2 - U_2$.
  - By applying the rule Assoc to W^1, we get a desired world sequent W'.
- **Case** $S_{ij} \neq \varnothing$ for any $1 \leq i, j \leq 2$:
  - By IH on w_1 and w_2, there exists a world sequent W^1 with heaps w^1_{ij} $(i, j = 1, 2)$ such that:
    - $\{W\} \mapsto^* \{W^1\}$ by applying only the rule Assoc;
    - $w_i \doteq w^1_{1j} \circ w^1_{2j}$ and $T_{W^1}(w^1_{ij}) = S_{ij}$ $(i, j = 1, 2)$.
  - It is easy to check that there exists a world sequent W' with heaps w'_1, w'_2 such that:
    - $\{W^1\} \mapsto^* \{W'\}$ by applying only the rule Assoc;
    - $w \doteq w'_1 \circ w'_2 \in W'$ and $w'_i \doteq w^1_{i1} \circ w^1_{i2} \in W'$ $(i = 1, 2)$.
  - Since $S_i = S_{i1} \cup S_{i2}$, we have $T_{W'}(w'_i) = S_i$ $(i = 1, 2)$.

5.→7. In this step, we use only the rule Assoc.

Let W be any full world sequent and w be any heap in W. By Lemma D.14, it suffices to show that: for any $r \in \mathcal{R}(W)$ with $w \nearrow_W r$ and for any non-empty sets $S_1, S_2 \subset T_W(r)$ such that $T_W(w) \cap S_1 = \varnothing$ and $T_W(w) \cup S_1 = S_2$, we can construct a world sequent W' with heaps w'_1 and w'_2, by applying only the rule Assoc to W, such that $w'_2 \doteq w \circ w'_1 \in W'$ and $T_{W'}(w'_i) = S'_i$ $(i = 1, 2)$. The proof is as follows. (We omit the use of Lemma D.14 in the following proof.)

- There exists a world sequent W^1 with a heap w^1 such that: [Lemma D.14]
  - $\{W\} \mapsto^* \{W^1\}$ by applying only the rule Assoc;
  - $r \doteq w \circ w^1 \in W^1$.
- Since $S_1 \subset T_{W^1}(r) - T_{W^1}(w)$ and W^1 is consistent, $S_1 \subset T_{W^1}(w^1)$.
- By the proof of 4, there exists a world sequent W^2 with a heap w^2_1, w^2_2 such that:
  - $\{W^1\} \mapsto^* \{W^2\}$ by applying only the rule Assoc;
  - $w^1 \doteq w^2_1 \circ w^2_2 \in W^2$ and $T_{W^2}(w^2_1) = S_1$.
- By applying the rule Assoc to W^2, we get a desired world sequent W'.

8.→9. In this step, we use only the rule Weaken.

By Lemma D.7 and by the fact that the way to apply the rule Weaken to obtain a sanitized world sequent is straightforward, we omit the proof for this step.

9.→10. In this step, we use only the rules NormPC, NormEmpty, NormEq and Assoc.

- Let W be a sanitized world sequent and w_ϵ be the special empty heap in W.
- There exists a sanitized world sequent W^1 such that: [Lemma D.17]
  - ▷ $\{W\} \mapsto^* \{W^1\}$ by applying only the rules NormPC and NormEmpty;
  - ▷ There is no empty heap except w_ϵ in W^1.
- There exists a sanitized world sequent W^2 such that: [Lemma D.18 with $n = |\mathcal{T}(W^1)|$]
  - ▷ $\{W^1\} \mapsto^* \{W^2\}$ by applying only the rules NormEq and Assoc;
  - ▷ There is no empty heap except w_ϵ in W^2;
  - ▷ For any $r^2 \in \mathcal{R}(W^2)$ and any non-empty set $S^2 \subset T_{W^2}(r^2)$ such that $|S^2| \leq |\mathcal{T}(W^1)|$, there exists a unique heap $w^2 \in W^2$ with $T_{W^2}(w^2) = S^2$.
- Since $|\mathcal{T}(W^1)| = |\mathcal{T}(W^2)|$, W^2 is normalized. [Lemma D.6-(1) and D.8-(2)]

At this point, it is easy to check that if W is $\star$-ready for heap w, then so does $W := W^2$. This is because:

- Assoc, NormEq, and NormPC preserve $\star$-ready-ness. [Lemma D.6-(2), D.8-(3), and D.12-(3)]
- In Lemma D.17, we applied the rule NormEmpty to empty heaps w_ϵ and u ($\neq w_\epsilon$) only when there is no heap relation of the form $u \doteq \cdot \circ \cdot$ and $\cdot \doteq \cdot \circ u$.

On the other hand, even if W is $-\!\!\star$-ready for heap w, W^2 may not be $-\!\!\star$-ready for heap w as the rule NormEq does not preserve $-\!\!\star$-ready-ness in general by Lemma D.8-(3). Fortunately, by applying Lemma D.18 and 5.' of Lemma 7.1 to W^2 alternately and finitely many times, we obtain a desired world sequent W' which is normalized as well as $-\!\!\star$-ready for heap w. This is possible due to Lemma D.6-(1) and D.8-(1), (3).

□

Lemma 7.2. For a world sequent W of a particular kind, there exists a disjunctive derivation state Ψ such that $\{W\} \mapsto^* \Psi$ by applying only the propagation rules, where one of the following holds:

6. →8. W is $\star$-ready for a given heap w, and every world sequent in Ψ is saturated and $\star$-ready for heap w;
7. →8. W is $-\!\!\star$-ready for a given heap w, and every world sequent in Ψ is saturated and $-\!\!\star$-ready for heap w.

Proof. In the proof, we use only the propagation rules. By Lemma D.7, it suffices to show that: for any full world sequent W, we can construct a disjunctive derivation state Ψ such that $\{W\} \mapsto^* \Psi$ and every world sequent in Ψ is saturated.

Before starting to prove it, we introduce several definitions to simplify our arguments. First, for any $a_i, a_i' \in \mathbb{N}$ $(i = 1, 2, 3)$, define an ordering $<$ on $\mathbb{N} \times \mathbb{N}$ and $(\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$ as follows:

- ▷ $(a_1, a_2) < (a_1', a_2')$ iff $a_1 < a_1'$ or $(a_1 = a_1'$ and $a_2 < a_2')$;
- ▷ $((a_1, a_2), a_3) < ((a_1', a_2'), a_3')$ iff $(a_1, a_2) < (a_1', a_2')$ or $((a_1, a_2) = (a_1', a_2')$ and $a_3 < a_3')$.

Next, for any world sequent W, define $\alpha(W)$ be the set of atomic heap relations in W such that the application of some propagation rule to them produces new heap relations. Note

that heap relations in $\alpha(W)$ must reside for non-terminals heaps. Finally, for any two world sequents W and W', we write $W \equiv W'$ iff. non-atomic heap relations of W and that of W' are the same.

Let W be any full world sequent and $w_1, \cdots, w_N$ be a topological ordering of non-terminal heaps in W with respect to the partial ordering $\nearrow$. Then, w_1 is a heap which is a parent of some terminal heap and does not have $w_2, \cdots, w_N$ as its descendant, and w_N is one of root heaps in W. Now, for any world sequent W' with $W \equiv W'$, define $\chi(W') \in \mathbb{N} \times \mathbb{N}$ as:

- $\chi(W'; i) := |\{\sigma : \sigma \in \alpha(W') \text{ and } \sigma \text{ resides for } w_i\}|$
- $\chi(W') := \max\{(i, \chi(W'; i)) : i \in \{1, \cdots, N\} \text{ and } \chi(W'; i) > 0\}$ (here $\max \varnothing := (0, 0)$).

Moreover, for any disjunctive derivation state Ψ' satisfying that $W \equiv W'$ for all $W' \in \Psi'$, define $\chi(\Psi') \in (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$ as:

- $\chi(\Psi'; (i, j)) := |\{W' : W' \in \Psi' \text{ and } \chi(W') = (i, j)\}|$
- $\chi(\Psi') := \max\{\big((i, j), \chi(\Psi'; (i, j))\big) : (i, j) = \chi(W') \text{ for } W' \in \Psi'\}$

Let us now prove this Lemma. It suffices to show the following: for any disjunctive derivation state Ψ^1 with $\{W\} \mapsto^* \Psi^1$ by applying only the propagation rules, there exists a disjunctive derivation state Ψ such that:

(a) $\Psi^1 \mapsto^* \Psi$ by applying only the propagation rules;
(b) Every world sequent in Ψ is saturated.

The proof proceeds by induction on $\chi(\Psi^1)$.

i) $\chi(\Psi^1) = ((0, 0), k)$ for any $k > 0$:
   - $\chi(\Psi^1) = ((0, 0), k)$ means that every world sequent in Ψ^1 is already saturated. Hence $\Psi = \Psi^1$.

ii) $\chi(\Psi^1) \geq ((1, 1), 1)$:
   - Let $((i, j), k) := \chi(\Psi^1)$ and $W^1 \in \Psi^1$ be a world sequent with $\chi(W^1) = (i, j)$.
   - Since $i > 0$, there exists an atomic heap relation $\sigma_i \in \alpha(W^1)$ at w_i.
   - By applying one of the propagation rules to σ_i at w_i in W^1, we get $\Psi^2 = \Psi^1 - \{W^1\} \cup \{W^2_1, \cdots, W^2_M\}$ for some $M \geq 1$ and some world sequents $W^2_1, \cdots, W^2_M$.
   - Then, we have $\sigma_i \in \alpha(W^1)$ and $\sigma_i \notin \alpha(W^2_m)$ for all $1 \leq m \leq M$.
   - Moreover, we see that every heap relation produced by the above application of a propagation rule resides for a heap which has a topological order less than i.
   - Hence, $\chi(W^2_m) < \chi(W^1)$ for all $1 \leq m \leq M$ and thus $\chi(\Psi^2) < \chi(\Psi^1)$.
   - By IH on Ψ^2, we get a desired disjunctive derivation state Ψ.

□

Proposition 7.4 (Completeness of the heap contradiction rules)**.**

For a normalized world sequent W with no formulas other than $\bot$, if $\neg[\![W]\!]_S$ holds for any stack S, then we can construct its derivation using only the rules $\bot\mathsf{L}$, $\mathsf{ExpCont}$, and the heap contradiction rules. For the rule $\mathsf{ExpCont}$, we assume that $\neg[\![\Theta \vdash \bot]\!]_S$ implies $\Theta \vdash \bot$.

Proof. The sketch of the proof is given in Section 3.4. □

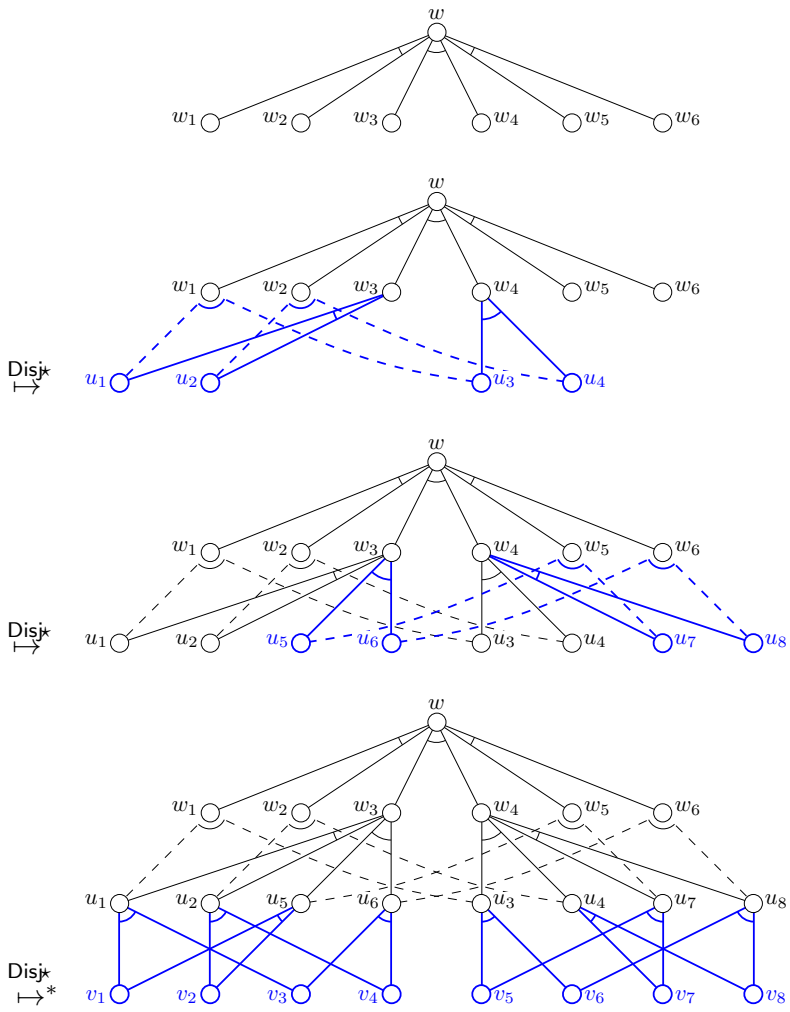


Figure 5: An example of applying the rule Disj⋆ to get a consistent world sequent. (The blue means newly created heap relations and the dashed is used just for easy visualization.)

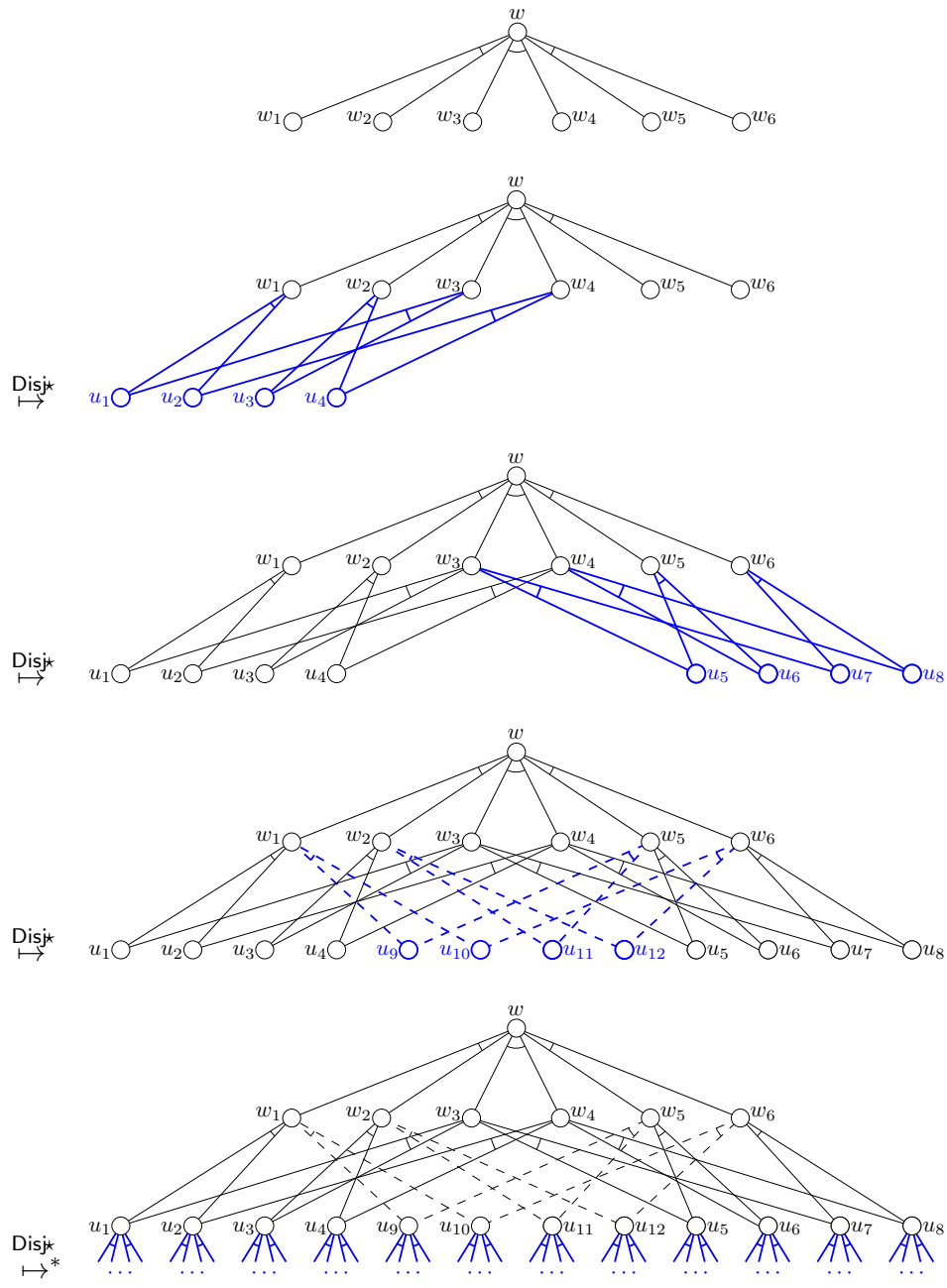


Figure 6: An example of carelessly applying the rule Disj⋆. (The blue means newly created heap relations and the dashed is used just for easy visualization.)