Separation Logic and Logics with Team Semantics

Erich Grädel, Darion Haase and Richard Wilke
RWTH Aachen University, Germany

Separation logic is a successful logical system for formal reasoning about programs that mutate their data structures. It goes back to work by Reynolds, O'Hearn, Pym and others [18, 10, 14] and builds on Hoare logic [8], a system for proving specifications of form {*precondition*}code{*postcondition*} about how a piece of code changes the properties of the states of a computation. Traditional Hoare logic works very well for programs with simple fixed data types, but reasoning about programs with mutable data structures becomes complicated and problematic, and this is the main issue that is addressed by separation logic. Actually, separation logic is part of a larger family of logics with *bunched implications* [13, 16, 15], but to get the point of this paper across, which is the connection with team semantics, we consider a stripped down presentation of separation logic, as used for instance in [6, 17], viewing separation logic as an extension of first-order logic for reasoning about *heaps* (modelled as partial functions $h\colon A \to A^k$) whose expressive power arises from two non-standard logical connectives: the *separating conjunction*, and the *magic wand*. With these new connectives one can write concise specifications of recursive data structures such as doubly linked lists, trees with linked leaves and parent pointers, and so on, and reason in the style of Hoare logic about the semantics of programs with such data structures [18].

More precisely, we define for any $k \geq 1$ the separation logic SL^k as the extension of first-order logic by two new atomic formulae **emp** and $x \mapsto \overline{y}$, and the new connectives $\star$ and $-\!\!*$. A formula $\psi(\overline{z}) \in \mathrm{SL}^k$ of vocabulary τ is interpreted over a triple $(\mathfrak{A}, \mathfrak{h}, s)$, consisting of a τ-structure $\mathfrak{A}$, a finite partial function $\mathfrak{h}\colon A \to A^k$ on the universe of $\mathfrak{A}$, called a heap, and an assignment $s\colon \mathrm{free}(\psi) \to A$ mapping the free variables of ψ to elements of $\mathfrak{A}$. The atom **emp** expresses that the heap is empty, and $x \mapsto \overline{y}$ is true in $(\mathfrak{A}, \mathfrak{h}, s)$ if the heap $\mathfrak{h}$ consist of the single item $s(x) \mapsto s(y_1), \ldots, s(y_k)$. Using the traditional first-order connectives and quantifiers, together with the separating conjunction and the magic wand, one then builds powerful statements describing dynamic transformations of data structures. The separating conjunction $\psi \star \varphi$ asserts that there is a disjoint split of the heap $\mathfrak{h}$ into two disjoint heaps satisfying ψ and φ, respectively, and the *separating implication* or *magic wand* $\psi -\!\!* \varphi$ states that φ is true for any extension $\mathfrak{h} \cup \mathfrak{h}'$ of the given heap $\mathfrak{h}$ by a heap $\mathfrak{h}'$ that satisfies ψ.

Since separation logic is an extension of first-order logic, the fundamental algorithmic problems such as (finite) satisfiability, validity, or entailment are, of course, undecidable. For both theoretical and practical purposes it is interesting to classify fragments of separation logic for which such problems become decidable, and to determine their complexity, and on the other side to identify those fragments that are expressively complete, and thus as difficult to handle as full separation logic. Such work has been done for instance in [3, 4, 5, 6].

Team semantics, on the other side, is the mathematical basis of modern logics for reasoning about dependence, independence, and imperfect information. It originates in the work of Wilfrid Hodges [9], and relies on the idea to evaluate logical formulae $\varphi(x_1, \ldots, x_n)$ not for single assignments $s\colon \{x_1, \ldots, x_n\} \to A$ from the free variables to elements of a structure $\mathfrak{A}$, but for *sets of such assignments*. These sets, which may have arbitrary size, are now called *teams*. Together with the fundamental idea of Väänänen [19] to treat dependencies not as annotations of quantifiers (as in IF-logic), but as atomic properties of teams, this has lead to a lively interdisciplinary research area, involving not just first-order logics, but also logics on the propositional and modal level, see e.g. [1]. Team semantics admits reasoning about large sets of data, modelled by second-order objects such as sets of assignments, with a first-order syntax that does not explicitly refer to higher-order variables. In the presence of appropriate

atomic team properties, such as dependence, inclusion and exclusion, or independence, team semantics can boost the expressiveness of first-order formalisms to the full power of existential second-order logic or, in the presence of further propositional operators such as different variants of implication or negation, even to full second-order logic (SO). There are several reasons for this high expressive power of logics of dependence and independence. One of them is the second-order nature of atomic dependencies in teams. For instance, saying that z depends on y in the team X means that there exists a function which, for all assignments $s \in X$, maps $s(y)$ to $s(z)$. A further reason is that in the context of teams, disjunctions and existential quantification are really second-order operations. Note, however, that only the combination of dependence atoms and disjunction/existential quantification leads to the expressive power of (existential) SO. We write $\mathfrak{A} \models_X \varphi$ to denote that φ is true in the structure $\mathfrak{A}$ for the team X. In this extended abstract we assume that the reader is familiar with the basic definitions of team semantics, and results are given without proofs. Detailed definitions and complete proofs will be given in the full version of this paper.

Connections and differences. Separation logic and team semantics have been introduced with quite different motivations, and are investigated by research communities with rather different backgrounds and objectives. Nevertheless, there are obvious similarities between these formalisms. First of all, both separation logic and logics with team semantics involve the manipulation of second-order objects, such as heaps and teams, by first-order syntax without reference to second-order variables. Moreover, these semantical objects are closely related; it is for instance obvious that a heap, i.e. a partial function $\mathfrak{h}\colon A \to A^k$, can be seen as a team with variables $x, y_1, \ldots, y_k$ satisfying the atom that $\overline{y}$ depends on x. Even more strikingly, the separating conjunction of separation logic is (essentially) the same as the team-semantical disjunction; moreover several notions of implications have been studied for team semantics, so it seems natural to interpret also the magic wand in this context. Based on such similarities, the possible connections between separation logic and team semantics have been raised as a question at several occasions, and lead to informal discussions between these research communities. The objective of this paper is to make this connection precise, and to study its potential but also its obstacles and limitations. We remark that the point of connecting separation logic with team semantics is not just expressive power. Actually, both separation logic and the logics with team semantics that we need here can readily be embedded into second-order logic (SO), and it is not difficult to see that they indeed essentially have the full power of SO. But going through second-order logic does not provide informative and compositional translations between these frameworks, and would thus produce only very limited insights. Rather we aim for a natural set of team-semantical operators that admit us to construct a faithful, complete and compositional representation of separation logic into a suitable logic with team semantics.

At least when we consider logics of dependence and independence in their standard format, there are also important differences to the framework of separation logic. This standard format is based on a collection of atomic dependencies on teams, typically dependence, inclusion, exclusion and/or independence, together with the usual first-order literals, and extends these by conjunction, disjunction, existential and universal quantifiers. In particular, these logics are not closed under negation, which is the first essential difference to separation logic. In fact, in logics of dependence and independence, negation is applied only to first-order atoms, not to dependencies or to compound formulae. Although one can define, for any formula φ, a kind of negation $\varphi^{\neg}$, its meaning is not the same as the classical negation and, in particular, the law of the excluded middle (*tertium non datur*) does not hold, not even for atomic formulae. A second relevant issue is the *empty team property* of these logics:

every formula whatsoever is satisfied by the empty team. This is a source of some technical difficulties in translations from separation logic to logics with team semantics, and excludes in particular the representation of the empty heap by the empty team.

Team logic for separation logic. To make translations from separation logic into team semantics possible, we consider the syntactic extension of separation logic by the dual connectives to the separating conjunction and the magic wand, the *separating disjunction* $\psi \circ \varphi$ and the *septraction* $\psi \multimap \varphi$, so that we can write all formulae of separation logic in negation normal form. This is a conservative extension that does not change the expressive power of the logic. We then discuss which of the ingredients of logics with team semantics are needed for achieving the expressive power of separation logic, and in particular, how the standard framework should be extended so that all of separation logic can be translated in a natural way. Of specific importance for the the translation that we propose are the *nonemptiness atom* NE, the uniform quantifiers $\exists^1$ and $\forall^1$, classical and dependent disjunctions and the intuitionistic implication. Although these operators are not part of what we call the standard framework of logics of dependence and independence, they have been studied quite thoroughly in team semantics, for instance in [2, 7, 12, 20].

Nonemptiness: $\mathfrak{A} \models_X \mathsf{NE}$ if $X \neq \varnothing$.

Finiteness: $\mathfrak{A} \models_X \mathrm{Fin}(\overline{x})$ if $X(\overline{x})$ is finite.

Equiextension: $\mathfrak{A} \models_X \overline{x} \bowtie \overline{y}$ if $X(\overline{x}) = X(\overline{y})$.

Classical disjunction: $\mathfrak{A} \models_X \varphi \sqcup \psi$ if $\mathfrak{A} \models_X \varphi$ or $\mathfrak{A} \models_X \psi$.

Uniform quantification: These are the usual quantifiers of FO, lifted to the team level:
$\mathfrak{A} \models_X \exists^1 x\varphi$ if $\mathfrak{A} \models_{X[x\mapsto\{a\}]} \varphi$ for some $a \in A$, and
$\mathfrak{A} \models_X \forall^1 x\varphi$ if $\mathfrak{A} \models_{X[x\mapsto\{a\}]} \varphi$ for all $a \in A$.

Dependent disjunction: $\mathfrak{A} \models_X \varphi \curlyvee_{\overline{x}} \psi$ if there is a disjoint decomposition $X = X_1 \cup X_2$ satisfying $\mathfrak{A} \models_{X_1} \varphi$ and $\mathfrak{A} \models_{X_2} \psi$ such that for all $s, s' \in X$, if $s \in X_i$ and $s(\overline{x}) = s'(\overline{x})$ then also $s' \in X_i$.

Intuitionistic implication: $\mathfrak{A} \models_X \varphi \to \psi$ if, for all teams $Y \subseteq X$ with $\mathfrak{A} \models_Y \varphi$, also $\mathfrak{A} \models_Y \psi$.

We remark that in dependence logic, i.e. first-order logic with dependence atoms $\mathrm{dep}(\overline{x};\overline{y})$ some of these connectives are expressible. The addition of NE, classical and dependent disjunctions, and the uniform quantifiers produces an expressively modest extension of dependence logic that remains inside the existential fragment of second-order logic. The further addition of the intuitionistic implication $\varphi \to \psi$ changes this, but it is needed for expressing the magic wand and the separating disjunction, which are universal second-order connectives. We note that the finiteness atom (which is necessary when we consider finite heaps that take values in an infinite structure) is easily expressible through dependence, equiextension, and intuitionistic implication, by means of Dedekind-finiteness. To summarize, the specific logic with team semantics that we are going to use for a compositional translation of separation logic is defined as follows.

▶ **Definition 1.** *Team logic for separation logic, abbreviated* TLfSL*, is the extension of dependence logic by* $\mathsf{NE}, \sqcup, \forall^1$*, and the intuitionistic implication* $\to$*. Note that* $\exists^1$*, dependent disjunction, equiextension, and the finiteness atoms are definable in it, and will also be used.*

From heaps to teams. We next discuss how the semantic objects for separation logic, i.e. triples $(\mathfrak{A}, \mathfrak{h}, s)$ consisting of a structure $\mathfrak{A}$, a heap $\mathfrak{h}$, and an assignment s, should be represented by the semantic objects in team semantics, i.e. pairs $(\mathfrak{B}, X)$ consisting of a structure $\mathfrak{B}$ and a team X. We will then want to provide translations, mapping any formula

$\varphi \in \mathrm{SL}$ (in negation normal form) to a formula $\varphi^* \in \mathrm{TLfSL}$ such that, whenever $(\mathfrak{B}, X)$ represents $(\mathfrak{A}, \mathfrak{h}, s)$, we have that $\mathfrak{A}, \mathfrak{h} \models_s \varphi$ if, and only if, $\mathfrak{B} \models_X \varphi^*$.

We start with the natural idea to view a heap $\mathfrak{h}\colon A \rightharpoonup_{\mathrm{fin}} A^k$ as a team over the variables $x, y_1, \dots, y_k$, and to represent a triple $(\mathfrak{A}, \mathfrak{h}, s)$ by a pair $(\mathfrak{A}, Y_{\mathfrak{h},s})$, leaving the structure $\mathfrak{A}$ unchanged and expanding the team representing the heap by the values representing the assignment s, to obtain an expanded team $Y_{\mathfrak{h},s}$.

▶ **Definition 2.** *For a heap $\mathfrak{h}\colon A \rightharpoonup_{fin} A^k$ and an assignment $s\colon \{z_1, \dots, z_m\} \to A$, the team $Y_{\mathfrak{h},s}$ consists of all assignments $t\colon \{x, y_1, \dots y_k\} \cup \{z_1, \dots, z_m\} \to A$ such that $t(x) \in \mathrm{dom}(\mathfrak{h})$, $t(\overline{y}) = \mathfrak{h}(t(x))$ and $t(z_i) = s(z_i)$. Notice that the team $Y_{\mathfrak{h},s}$ fulfils the dependence atom $\mathrm{dep}(x; y_1, \dots, y_k)$ and the constancy atoms $\mathrm{dep}(z_1), \dots, \mathrm{dep}(z_m)$.*

Although $Y_{\mathfrak{h},s}$ is a natural representation of the pair $(\mathfrak{h}, s)$, this idea is too simple to work well. Any pair $(\mathfrak{h}, s)$ where the heap is empty is represented by the empty team, so all information about the assignment s is lost. Moreover, the standard logics of dependence and independence have the empty team property, and even logics as strong as team logic, which have classical negation (and hence do not have the empty team property) cannot express anything useful about the given structure in the presence of the empty team [11]. To take care of the problems arising with the empty team we add a dummy element δ to $\mathfrak{A}$ to obtain the structure $\mathfrak{A}^\delta$ with universe $A \cup \{\delta\}$ such that, for every relation symbol $R \in \tau$, we set $R^{\mathfrak{A}^\delta} := R^{\mathfrak{A}}$ and for any function symbol $f \in \tau$, we let $f^{\mathfrak{A}^\delta}$ coincide with $f^{\mathfrak{A}}$ on all tuples from $\mathfrak{A}$, and map all other tuples to δ. We extend the vocabulary by a new constant symbol δ interpreting the dummy element and add a dummy assignment to the team.

▶ **Definition 3.** *Given a structure $\mathfrak{A}$ and some element $\delta \notin A$, a triple $(\mathfrak{A}, \mathfrak{h}, s)$ is now represented by the pair $(\mathfrak{A}^\delta, X_{\mathfrak{h},s})$ where $X_{\mathfrak{h},s} := Y_{\mathfrak{h},s} \cup \{s^\delta\}$ with $s^\delta(x) = s^\delta(y_1) = \cdots = s^\delta(y_k) = \delta$ and $s^\delta(z_i) = s(z_i)$ for $i = 1, \dots, m$. Note that for any assignment $s\colon \{z_1, \dots, z_m\} \to A$ we have that $X_{\varnothing,s} = \{s^\delta\}$.*

Based on the presentation of triples $(\mathfrak{A}, \mathfrak{h}, s)$ by $(\mathfrak{A}^\delta, X_{\mathfrak{h},s})$ it is not difficult to translate the first-order part of separation logic to the extension of dependence logic with the uniform quantifiers $\exists^1$ and $\forall^1$, the classical disjunction $\sqcup$, the non-empty split disjunction and the non-emptiness predicate NE.

Splitting and extending heaps and teams. The translation of the separating conjunction $\psi \star \varphi$ into team semantics requires that we are able to talk about splits of a heap $\mathfrak{h}$ on the level of teams $X^*_{\mathfrak{h},s}$. We do this by defining the formula

$$\mathrm{split}(x, c) := [(c = \delta \wedge \mathsf{NE}) \vee (c \neq \delta \wedge \mathrm{dep}(c) \wedge \mathsf{NE})] \wedge [(x = \delta \wedge (\mathsf{NE} \curlyvee_c \mathsf{NE})) \vee (x \neq \delta \wedge \mathrm{dep}(x; c))].$$

For any triple $(\mathfrak{A}, \mathfrak{h}, s)$ and any function $F\colon X_{\mathfrak{h},s} \to \mathcal{P}^+(A \cup \{\delta\})$, we have that $\mathfrak{A}^\delta \models_{X_{\mathfrak{h},s}[c \mapsto F]} \mathrm{split}(x, c)$ if, and only if, there is an element $a \in A$ and a split $(\mathfrak{h}_1, \mathfrak{h}_2)$ of $\mathfrak{h}$ such that $X_{\mathfrak{h},s}[c \mapsto F] = X_{\mathfrak{h}_1,s}[c \mapsto \delta] \cup X_{\mathfrak{h}_2,s}[c \mapsto a]$. The separating conjunction is translated as

$$(\varphi \star \psi)^* := \exists c \Big(\mathrm{split}(x, c) \wedge \big((c = \delta \wedge \varphi^*) \vee (c \neq \delta \wedge \psi^*) \big) \Big).$$

We next discuss the translation of the septraction $\psi \mathrel{-\!\!\circ} \varphi$. Recall that $\mathfrak{A}, \mathfrak{h} \models_s \psi \mathrel{-\!\!\circ} \varphi$ if there exists a disjoint extension $\mathfrak{h} \cup \mathfrak{h}'$ with $\mathfrak{A}, \mathfrak{h}' \models_s \psi$ and $\mathfrak{A}, \mathfrak{h} \cup \mathfrak{h}' \models_s \varphi$. We have to represent extensions of the given heap $\mathfrak{h}$ by appropriate extensions of the encoding team

$X_{\mathfrak{h},s}$. We need a formula that says that a team Y, restricted to variables $(x, \overline{y})$, correctly encodes a heap. This is achieved by

$$\mathrm{heap}(x, \overline{y}) := \mathrm{dep}(x; \overline{y}) \wedge \mathrm{Fin}(x) \wedge \big((x = \delta \wedge \overline{y} = \overline{\delta} \wedge \mathsf{NE}) \vee (x \neq \delta \wedge \bigwedge_{i=1}^{k} y_i \neq \delta)\big).$$

For a given formula φ of separation logic, and its translation to φ^* to a formula with team semantics, let $\varphi^*[u, \overline{v}]$ be obtained from φ^* by renaming $x, \overline{y}$ to new variables $u, \overline{v}$ (whereas the variables $\overline{z}$ representing the assignment s are left unchanged). We now construct a formula to talk about disjoint extensions of the given heap by a heap that satisfies φ.

$$\varphi_{\mathrm{ext}}(x, \overline{y}, u, \overline{v}) := \Big((x = \delta \wedge \mathrm{heap}(u, \overline{v}) \wedge \varphi^*[u, \overline{v}]) \vee (x \neq \delta \wedge u = x \wedge \overline{v} = \overline{y})\Big) \wedge \mathrm{dep}(u; x)$$

The translation of $\varphi \multimap \psi$ now asserts that ψ is true in some disjoint extension of the given heap by a heap that satisfies φ:

$$(\varphi \multimap \psi)^* := \exists u \exists \overline{v} (\varphi_{\mathrm{ext}}(x, \overline{y}, u, \overline{v}) \wedge \psi^*[u, \overline{v}]).$$

Translating the separating disjunction and the magic wand. We finally discuss the translation of the two connectives that involve a universal quantification about heaps. It is clear that these are not definable in the existential fragment of second-order logic, and the natural way to go is to extend dependence logic by the intuitionistic implication $\psi \to \varphi$. But notice that this is a quite different kind of implication than the magic wand, and the translation is far from obvious. Recall that

$$\begin{aligned} \mathfrak{A}, \mathfrak{h} \models_s \varphi \mathrel{-\!\!*} \psi &\iff \text{for every heap } \mathfrak{h}' \text{ with } \mathfrak{h} \# \mathfrak{h}' \text{ and } \mathfrak{A}, \mathfrak{h}' \models_s \psi, \text{also } \mathfrak{A}, (\mathfrak{h} \cup \mathfrak{h}') \models_s \psi \\ \mathfrak{A} \models_X \varphi \to \psi &\iff \text{for every subteam } Y \subseteq X \text{ with } \mathfrak{A} \models_Y \varphi, \text{ also } \mathfrak{A} \models_Y \psi \end{aligned}$$

where $\mathfrak{h} \# \mathfrak{h}'$ means that $\mathfrak{h}$ and $\mathfrak{h}'$ are disjoint, i.e. $\mathrm{dom}(\mathfrak{h}) \cap \mathrm{dom}(\mathfrak{h}') = \varnothing$.

We start with the idea to use a universal variant of the translation of septraction, i.e. $\forall u \forall \overline{v} (\varphi_{\mathrm{ext}}(x, \overline{y}, u, \overline{v}) \to \psi^*[u, \overline{v}])$. Intuitively, in the evaluation of this formula over $X_{\mathfrak{h},s}$, the universal quantification generates a team Y that represents a maximal extension of the given heap $\mathfrak{h}$. The implication then says that *all subteams* of Y that represent an extension by a heap $\mathfrak{h}'$ that satisfies φ must also satisfy ψ. But this is not really correct, because left side of the implication can also be true for subteams that do not contain all the data present in $\mathfrak{h}$ i.e. represent an extension not of $\mathfrak{h}$, but of some subheap. We thus have to restrict the left side of the implication so that it talks only about those subteams that contain the full information of the team $X_{\mathfrak{h},s}$. To achieve this, we construct a formula in two variables x, x' that enforces a cyclic permutation of the values of x in the team. We set

$$\mathrm{cycle}(x, x') := x \bowtie x' \wedge \mathrm{dep}(x; x') \wedge \mathrm{dep}(x'; x) \wedge (x \bowtie x' \to ((x = \delta \wedge \mathsf{NE}) \vee x \neq \delta)).$$

A finite team Z with $\mathrm{dom}(Z) = \{x, x'\}$ is a model of $\mathrm{cycle}(x, x')$ if, and only if, there is a cyclic permutation $(a_0, \ldots, a_{m-1})$ of $Z(x) = Z(x')$ such that $Z = \{s_0, \ldots, s_{m-1}\}$ with $s_i(x) = a_i$ and $s_i(x') = a_{i+1 (\mathrm{mod}\ m)}$. As a consequence, if $Y \subseteq Z$ is a non-empty subteam of such a model with $\models_Y x \bowtie x'$ then $Y = Z$. We then translate the magic wand $\varphi \mathrel{-\!\!*} \psi$ by

$$(\varphi \mathrel{-\!\!*} \psi)^* := \exists x' \Big(\mathrm{cycle}(x, x') \wedge \forall u \forall \overline{v} \big((\mathsf{NE} \wedge x \bowtie x' \wedge \varphi_{\mathrm{ext}}(x, \overline{y}, u, \overline{v})) \to \psi^*[u, \overline{v}]\big)\Big).$$

For the translation of the separating disjunction $\varphi \circ \psi$, a similar idea is used, based on the formula $\mathrm{split}(x, c)$. We put

$$(\varphi \circ \psi)^* := \exists x' \Big(\mathrm{cycle}(x, x') \wedge \forall c \big((\mathsf{NE} \wedge x \bowtie x' \wedge \mathrm{split}(x, c)) \rightarrow \big(((c = \delta \wedge \varphi^*) \vee c \neq \delta) \sqcup ((c \neq \delta \wedge \psi^*) \vee c = \delta) \big) \Big).$$

We summarize our findings:

▶ **Theorem 4.** *There is a compositional translation that maps any formula $\varphi \in \mathrm{SL}$ into a formula $\varphi^* \in \mathrm{TLfSL}$ such that $(\mathfrak{A}, \mathfrak{h}) \models_s \varphi \iff \mathfrak{A}^\delta \models_{X_{\mathfrak{h}}, s} \varphi^*$, for every triple $\mathfrak{A}, \mathfrak{h}, s$.*

References

1 S. Abramsky, J. Kontinen, J. Väänänen, and H. Vollmer, editors. *Dependence Logic. Theory and Applications*. Birkhäuser, 2016.

2 S. Abramsky and J. Väänänen. From IF to BI. *Synthese*, 167(2):207–230, Mar 2009.

3 R. Brochenin, S. Demri, and É. Lozes. On the almighty wand. *Information and Computation*, 211:106–137, 2012.

4 S. Demri and M. Deters. Expressive completeness of separation logic with two variables and no separating conjunction. *ACM Trans. Comput. Log.*, 17(2):12:1–12:44, 2016.

5 S. Demri, D. Galmiche, D. Larchey-Wendling, and D. Méry. Separation logic with one quantified variable. *Theory of Computing Systems*, 61(2):371–461, 2017.

6 M. Echenim, R. Iosif, and N. Peltier. The Bernays-Schönfinkel-Ramsey Class of Separation Logic on Arbitrary Domains. In *FOSSACS*, pages 242–259, 2019.

7 P. Galliani. On strongly first-order dependencies. In S. Abramsky et al., editor, *Dependence Logic. Theory and Applications*. Birkhäuser, 2016.

8 C. A. R. Hoare. An Axiomatic Basis for Computer Programming. *Commun. ACM*, 12(10):576–580, 1969.

9 W. Hodges. Compositional semantics for a logic of imperfect information. *Logic Journal of IGPL*, 5:539–563, 1997.

10 S. Ishtiaq and P. O'Hearn. BI as an Assertion Language for Mutable Data Structures. In *Proceedings of POPL 2001*, pages 14–26, 2001.

11 J. Kontinen and V. Nurmi. Team Logic and Second-Order Logic. *Fundamenta Informaticae*, 106(2-4):259–272, 2011.

12 J. Kontinen and J. Väänänen. On Definability in Dependence Logic. *Journal of Logic, Language and Information*, 18(3):317–332, Jul 2009.

13 P. O'Hearn and D. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, 1999. doi:10.2307/421090.

14 P. O'Hearn, J. Reynolds, and H. Yang. Local reasoning about programs that alter data structures. In *Computer Science Logic, CSL 2001*, pages 1–19, 2001.

15 D. Pym. Resource semantics: logic as a modelling technology. *SIGLOG News*, 6(2):5–41, 2019.

16 D. Pym, P. O'Hearn, and H. Yang. Possible worlds and resources: the semantics of BI. *Theoretical Computer Science*, 315(1):257–305, 2004.

17 A. Reynolds, R. Iosif, and C. Serban. Reasoning in the bernays-schönfinkel-ramsey fragment of separation logic. In *Verification, Model Checking, and Abstract Interpretation - 18th International Conference, VMCAI*, pages 462–482, 2017.

18 J. Reynolds. Separation logic: A logic for shared mutable data structures. In *17th IEEE Symposium on Logic in Computer Science (LICS 2002), 22-25 July 2002, Copenhagen, Denmark, Proceedings*, pages 55–74, 2002.

19 J. Väänänen. *Dependence logic: A new approach to independence friendly logic*, volume 70. Cambridge University Press, 2007.

20 F. Yang. *On Extensions and Variants of Dependence Logic – A study of intuitionistic connectives in the team semantics setting.* PhD thesis, Department of Mathematics and Statistics, Faculty of Science, University of Helsinki, 2014.