Thesis: Reasoning with Separation Logics
Complexity, Expressive Power, Proof Systems

Alessio Mansutti
CNRS, LSV, ENS Paris-Saclay

Abstract

The thesis focuses on *separation logics*: well-known assertion languages that provide a scalable solution for Hoare-style verification of imperative, heap-manipulating programs [8]. To achieve scalability, separation logics rely in two spatial connectives: the separating conjunction $*$ and the separating implication $-\!\!*$. These operators allow to modularly reason about fragments and extensions of *memory states*, i.e. abstractions of the stack and heap/RAM memory model used as a backbone for the semantics of many programming languages (e.g. Java and C).

Broadly speaking, the goal of the thesis is to methodically analyse the features of separation logics, to improve our understanding of the logic from a computational and proof theoretical point of view. Besides providing an in-depth study of classical decision problems (e.g. satisfiability and validity) and expressiveness of separation logics, the thesis draws interesting connections between separation logic and other logics, most notably modal logics.

The thesis is split in three parts. The first part focuses on the notion of reachability predicates in separation logics. A location ℓ_1 (a.k.a. a memory address) is said to be reachable from a second location ℓ_2 whenever subsequent dereferentiations of ℓ_2 yield ℓ_1. Together with first-order quantification, reachability predicates allow to check for several fundamental properties of the memory, such as acyclicity (i.e. the absence of cycles) and garbage freedom (i.e. the absence of dangling pointers). Our goal is to devise a separation logic featuring these types of predicates, while keeping the complexity of its satisfiability problem in check. This is easier said than done, as we show that several separation logics are largely intractable (undecidable or non-elementary decidable). Nonetheless, after studying the sources of intractability we manage to design a PSPACE fragment of first-order separation logic that is able to express the desired properties. The PSPACE membership is shown through the *core formulae technique*, a model theoretic approach that is reminiscent of Gaifman's locality theorem for first-order logic.

The second part of the thesis tackles the open problem of designing Hilbert-style axiomatisations for separation logics. An axiomatisation is said to be Hilbert-style (or internal) whenever its axioms and inference rules are built solely from formulae of the logic, without any external machinery such as labels or nominals. In particular, we introduce the first sound and complete internal proof system for the quantifier-free fragment of separation logic. The completeness of the system is shown by taking advantage of the core formulae technique introduced in the first part of the thesis. In order to show that this technique can be reused on other logics, we design an axiomatisation for a modal logic enriched with the composition operator from ambient logic (a logic to verify distributed systems). The two proof systems reveal interesting connections between separation logics and ambient logics.

Motivated by the similarities in their proof systems, in the third part of the thesis we dig deep in the connections between separation logic and ambient logic. To carry out our comparison, we devise a suitable framework based on modal logic. This framework gives us the common ground needed in order to study the two logics from the point of view of their spatial connectives: the separating conjunction $*$ for separation logic and the parallel composition $\mid$ for ambient logic. Surprising similarities and differences are discovered, both in terms of expressive power and computational complexity.

Below, we give a more detailed view of the contributions of the thesis.

Part I: Robustness Properties of Logical Assertions

In program analysis, reachability predicates provide the foundation for verifying programs manipulating lists, but in the context of separation logic their use is often very restricted. While being beneficial on a computational level, this severely limits the range of properties we are able to check. In particular, in this part of the thesis we are interested in two properties that are fundamental for pointer programs analysis: *acyclicity* and *garbage freedom*. A memory state is *acyclic* whenever no location can *reach* itself by traversing the heap a positive amount of times. A memory state is $\mathtt{X}$-*garbage free*, where $\mathtt{X}$ is a finite set of program variables, whenever all allocated memory cells are accessible by dereferenciating locations assigned to variables in $\mathtt{X}$. As explained in [6], being able to check whether the acyclicity property holds is useful in analysing the termination of a program, whereas testing for garbage freedom allows us to show that the program does not leak memory by generating unreachable portions of the heap.

Motivations.

In separation logic, both acyclicity and garbage freedom can be expressed with first-order quantification and reachability predicates. For instance, acyclicity holds for memory states satisfying the formula $\neg\exists \mathtt{x}\, \mathtt{x} \hookrightarrow^{+} \mathtt{x}$. Here, $\neg$ stands for negation, the first-order quantifier $\exists \mathtt{x}$ quantifies over locations, and the *reach-plus* predicate $\mathtt{x} \hookrightarrow^{+} \mathtt{y}$ states that the location assigned to the variable $\mathtt{y}$ can be retrieved by repeatedly dereferenciating (at least once) the address assigned to $\mathtt{x}$. Unfortunately, no known separation logic with an elementary decidability status allows for these types of formulae. This leads us to journey through various fragments of first-order separation logic featuring reachability predicates, with the goal of designing a separation logic that is expressive enough to capture the notions of acyclicity and garbage freedom, while still having a nice computational status. More precisely, we aim for its satisfiability (alt. validity and entailment) problem to be decidable in PSPACE, exactly as it is the case for quantifier-free separation logic ($\mathsf{SL}(*, -\!\!*)$). This goal, which we eventually reach in Chapter 5, reveals to be quite ambitious, as in both Chapters 3 and 4 we show how small extensions of $\mathsf{SL}(*, -\!\!*)$ lead to negative results in terms of computational complexity.

Contribution of Chapter 3.

Our journey starts by simply adding to $\mathsf{SL}(*, -\!\!*)$ standard reachability predicates (e.g. $\mathtt{x} \hookrightarrow^{+} \mathtt{y}$) as well as bounded reachability predicates $\mathtt{x} \hookrightarrow^{\delta} \mathtt{y}$ ($\delta \in \mathbb{N}$) that are satisfied by a memory state if the location assigned to $\mathtt{y}$ can be obtained by dereferencing $\mathtt{x}$ exactly δ times. Very surprisingly, we show that as soon as the bounded reachability predicates $\mathtt{x} \hookrightarrow^{2} \mathtt{y}$ and $\mathtt{x} \hookrightarrow^{3} \mathtt{y}$ are added to $\mathsf{SL}(*, -\!\!*)$, the satisfiability problem jumps from PSPACE to non recursively enumerable. This result extends to several separation logics featuring reachability predicates. The main cause of the computational blow-up is traced back to the interactions between the reachability predicates and the separating implication $-\!\!*$, which allow to encode first-order quantifications by means of heap updates.

This chapter covers the first part of the work published in:

▷ S. Demri, E. Lozes, and A. Mansutti, "The effects of adding reachability predicates in propositional separation logic." *Foundations of Software Science and Computational Structures*. 2018.

An extended version of the paper above has been accepted to the journal *ACM Transactions on Computational Logic* (DOI 10.1145/3448269, paper in the process of being published).

Contribution of Chapter 4.

In view of the results in Chapter 3, we remove for the time being the separating implication and introduce first-order quantification over a single variable name (renaming is not allowed). In particular, we focus on the separation logic $\mathsf{SL}([\exists]_1, *, \mathtt{x} \hookrightarrow \_, \hookrightarrow^{+})$ featuring one quantified variable name, the separating conjunction $*$, the reach-plus predicate $\mathtt{x} \hookrightarrow^{+} \mathtt{y}$, and the predicate *alloc* $\mathtt{x} \hookrightarrow \_$ stating that the location assigned to $\mathtt{x}$ is allocated in the memory. We show that this logic is already TOWER-complete, hence non elementary decidable. In the chapter, this result is proved in a more

general settings: we discuss a set of features centred around reachability and submodel reasoning which causes logics interpreted on trees to be TOWER-hard. These features are formally described through a new modal logic which we call ALT (short for Auxiliary Logic on Trees). After looking at the expressive power of ALT by defining a suitable notion of Ehrenfeucht-Fraïssé games, we show that the satisfiability problem of ALT is TOWER-complete. Apart from $\mathsf{SL}([\exists]_1, *, \mathtt{x} \hookrightarrow \_, \hookrightarrow^{+})$, ALT is captured by several logics that were independently found to be TOWER-hard, as quantified computation tree logic [7], modal separation logics [4] and modal logic of heaps [3]. Thanks to ALT, new strict fragments of these logics are discovered to be non elementary.

This chapter covers the work published in:

▷ A. Mansutti, "An auxiliary logic on trees: on the tower-hardness of logics featuring reachability and submodel reasoning". *Foundations of Software Science and Computational Structures*. 2020.

Contribution of Chapter 5.

The negative results of Chapters 3 and 4 guide us to the definition of $\mathsf{SL}([\exists]_1, *, [-\!\!*, \hookrightarrow^{\dagger}]^{\mathcal{S}}_{\mathcal{W}})$, a separation logic that satisfies all our desiderata: (I) it extends $\mathsf{SL}(*, -\!\!*)$ with reachability predicates, (II) it can express the properties of acyclicity and garbage freedom, and (III) its satisfiability problem is PSPACE-complete, exactly as for $\mathsf{SL}(*, -\!\!*)$. At the time of writing, $\mathsf{SL}([\exists]_1, *, [-\!\!*, \hookrightarrow^{\dagger}]^{\mathcal{S}}_{\mathcal{W}})$ is the only separation logic satisfying these three conditions. The logic is a syntactical fragment of first-order separation logic crafted to avoid the interactions between reachability and the connectives $*$ and $-\!\!*$ that, in the previous two chapters, were discovered causing computational blow-ups. To prove the PSPACE upper bound of its satisfiability problem, we rely on techniques from finite model theory, and establish a result that is reminiscent of the first-order locality theorem proved by H. Gaifman in [5]: every formula of $\mathsf{SL}([\exists]_1, *, [-\!\!*, \hookrightarrow^{\dagger}]^{\mathcal{S}}_{\mathcal{W}})$ is shown to be equivalent to a Boolean combination of "core" formulae. This equivalence, which yields a polynomial small model property for the logic, is proved via Ehrenfeucht-Fraïssé games and by relying on a technique that we call *game hopping*, as it is inspired by the homonymous technique for indistinguishability games in computational security [1].

This chapter covers the second part of the work published in:

▷ A. Mansutti, "Extending propositional separation logic for robustness properties". *Foundations of Software Technology and Theoretical Computer Science*. 2018.

Part II: Internal calculi for spatial logics.

An Hilbert-style proof system $\mathcal{H}$ for a logic $\mathcal{L}$ is a set of formulae of $\mathcal{L}$ (the axioms), together with rules of the form $\varphi_1, \ldots, \varphi_n \vdash \psi$, which should be read as "if $\varphi_1, \ldots, \varphi_n$ are all valid formulae, then so is ψ". Here, $\varphi_1, \ldots, \varphi_n, \psi$ are all formulae from $\mathcal{L}$. Designing a sound and complete Hilbert-style proof system is usually quite challenging, but beneficial:

- the system gives an exhaustive understanding of the expressive power of the logic, by discarding the use of any external artifact referring to semantical objects,
- thanks to the insights on the expressive power, the system can improve existing decision procedures, by means of better abstractions or more refined calculi,
- the proof of completeness can be theoretically novel, thus paving the way towards Hilbert-style proof systems of other logics.

Because of these features, many logics related to program verification have been axiomatised, often requiring non-trivial completeness proofs. Unfortunately, this is not the case for separation logic, as the separating connectives $*$ and $-\!\!*$, together with the memory model used by the logic, break standard techniques used to prove the completeness of Hilbert-style proof systems.

Motivations.

Since the birth of separation logics, there has been a lot of interest in the study of decidability and computational complexity issues, and comparatively less attention to the design of proof systems, and even less with the puristic approach that consists in discarding any external feature such as nominals or labels in the calculi. In particular, an open question in the field is how to develop sound and complete Hilbert-style proof systems for separation logics. In the second part of the thesis, we tackle this problem and introduce a methodology to axiomatise separation logics and similar logics, based on the core formulae technique introduced in Chapter 5.

Contribution of Chapter 6.

We present the first Hilbert-style proof system for the quantifier-free separation logic $\mathsf{SL}(*, -\!\!*)$, featuring Boolean connectives and both the separating conjunction $*$ and implication $-\!\!*$. As already stated, to design the proof system we rely on the core formulae technique. More precisely, we design an axiomatisation for arbitrary Boolean combinations of core formulae, and add axioms and rules that allow to syntactically transform every formula of $\mathsf{SL}(*, -\!\!*)$ into such Boolean combinations. Schematically, for a valid formula φ of $\mathsf{SL}(*, -\!\!*)$, the proof system is able to derive whether $\vdash \varphi$ holds by showing $\vdash \varphi'$ and $\vdash \varphi' \Leftrightarrow \varphi$, where φ' is a Boolean combination of core formulae. Our methodology leads to a modular calculus that is divided in three parts: (1) the axiomatisation of Boolean combinations of core formulae, (2) axioms and inference rules to simulate a bottom-up elimination of the separating conjunction $*$, and (3) axioms and inference rules to simulate a bottom-up elimination of the separating implication $-\!\!*$. A nice property of this methodology is that only the completeness of the axiomatisation of Boolean combinations of core formulae has to be proven by semantical means. The bottom-up elimination of $*$ and $-\!\!*$ is showed completely syntactically, which leads to a less error-prone proof of completeness of the full calculus.

This chapter covers the first part of the work published in:

▷ S. Demri, E. Lozes, and A. Mansutti, "Internal calculi for separation logics". *Computer Science Logic*. 2020.

An extended version of the first part of the paper above has been accepted with minor revisions (provided at the beginning of March) to the journal *Logical Methods in Computer Science*.

Contribution of Chapter 7.

To show that our methodology is reusable in practice, we apply it to the ambient logic $\mathsf{ML}(\mid)$. Ambient logics [2] are modal logics introduced to verify properties of distributed systems specified in the calculus of Mobile Ambients. Their main feature of ambient logics is given by the composition operator $\varphi \mid \psi$ that, similarly to the separating conjunction, asks to spatially split the distributed process into two pieces, one satisfying the formula φ and the other satisfying the formula ψ. The logic $\mathsf{ML}(\mid)$ extends the modal logic K with said composition operator. The axiomatisation of $\mathsf{ML}(\mid)$ follows the same principles as the one of $\mathsf{SL}(*, -\!\!*)$, based on core formulae, and shows that $\mathsf{ML}(\mid)$ is as expressive as graded modal logic (GML), a well-known extension of modal logic K.

Part III : Mixing multiplicative connectives and modalities.

The proof systems for $\mathsf{SL}(*, -\!\!*)$ and $\mathsf{ML}(\mid)$ introduced in the second part of the thesis show interesting connections between separation logics and ambient logics. Despite the novelty of the two proof systems, connections between these two families of logics were already established in the early 2000s, when the two logics were firstly introduced. For instance, the decidability of the static ambient logic SAL considered in [2] is based on the proof technique firstly used to show the decidability of $\mathsf{SL}(*, -\!\!*)$. Nonetheless, a direct comparison between separation logics and ambient logics is missing or limited to a superficial analysis on the different classes of models and spatial connectives.

Motivations.

We aim for an in-depth comparison between the composition operator $\mid$ from ambient logics and the separating conjunction $*$ from separation logics by identifying a common ground in terms of logical languages and models. As a consequence, we are able to study the effects of having these operators as far as expressivity and complexity are concerned. To carry out our comparison, we consider $\mathsf{ML}(\mid)$ as an ambient logic and introduce the modal logic $\mathsf{ML}(*)$ which is obtained from $\mathsf{ML}(\mid)$ by replacing the composition operator with the separating conjunction. This framework is sufficiently fundamental to give us the possibility to take advantage of model theoretical tools from modal logics, and sets a common ground for comparison that may lead to further connections with other logics. Despite the semantical similarities between $\mathsf{ML}(*)$ and $\mathsf{ML}(\mid)$, we identify surprising differences in terms of their expressiveness and complexity.

Part III roughly covers the work published in:

▷ B. Bednarczyk, S. Demri, R. Fervari, and A. Mansutti, "Modal logics with composition on finite forests: Expressivity and complexity". *Logic in Computer Science*. 2020.

Contribution of Chapter 8.

We continue the analysis of $\mathsf{ML}(\mid)$ started in Chapter 7, by looking at its computational complexity. We show that its satisfiability problem is AExp_{Pol}-complete, where AExp_{Pol} is the class of decision problems solvable by an alternating Turing machine with exponential runtime and polynomial number of alternations between existential and universal states. The AExp_{Pol} upper bound is derived from a refined translation to GML, using fundamental properties of the proof system designed in Chapter 7. Furthermore, we relate $\mathsf{ML}(\mid)$ to the intensional fragment of static ambient logic $\mathsf{SAL}(\mid)$ from [2] by providing polynomial-time reductions between their satisfiability problems. Consequently, we establish AExp_{Pol}-completeness for the satisfiability problem of $\mathsf{SAL}(\mid)$, refuting hints from [2, Section 6], were the problem was conjectured to be PSpace-complete.

Contribution of Chapter 9.

We introduce $\mathsf{ML}(*)$, the logic obtained from $\mathsf{ML}(\mid)$ by replacing the composition operator $\mid$ by the separating conjunction $*$. We show that $\mathsf{ML}(*)$ is strictly less expressive than $\mathsf{ML}(\mid)$ and GML. Interestingly, this development partially reuses the result for $\mathsf{ML}(\mid)$, hence showing that our framework allows us to transpose results between the two logics. To show that GML is strictly more expressive than $\mathsf{ML}(*)$, we define an ad-hoc notion of Ehrenfeucht-Fraïssé games for the logic. Very surprisingly, although $\mathsf{ML}(*)$ is strictly less expressive than $\mathsf{ML}(\mid)$, its complexity is much higher. More precisely, we show that the satisfiability problem for $\mathsf{ML}(*)$ is Tower-complete.

References

[1] B. Blanchet and D. Pointcheval, "Automated security proofs with sequences of games," in *International Cryptology Conference*, ser. LNCS, vol. 4117. Springer, 2006, pp. 537–554.

[2] C. Calcagno, L. Cardelli, and A. D. Gordon, "Deciding validity in a spatial logic for trees," in *International Workshop on Types in Languages Design and Implementation*. ACM, 2003, pp. 62–73.

[3] S. Demri and M. Deters, "Two-variable separation logic and its inner circle," *Transactions on Computational Logic*, vol. 16, pp. 15:1–15:36, 2015.

[4] S. Demri and R. Fervari, "On the complexity of modal separation logics," in *Advances in Modal Logic*. College Publications, 2018, pp. 179–198.

[5] H. Gaifman, "On local and non-local properties," *Studies in Logic and the Foundations of Mathematics*, vol. 107, pp. 105–135, 1982.

[6] S. Genaim and D. Zanardini, "Inference of field-sensitive reachability and cyclicity," *Transactions on Computational Logic*, vol. 15, pp. 1–41, 2014.

[7] F. Laroussinie and N. Markey, "Quantified CTL: expressiveness and complexity," *Logical Methods in Computer Science*, vol. 10, 2014.

[8] P. W. O'Hearn, "Separation logic," *Communications of the Association for Computing Machinery*, vol. 62, pp. 86–95, 2019.