Comparing the expressive power of Separation logic and classical logic

Short Presentation

Lozes Etienne LIP - ENS Lyon
46,allée d'Italie - 69364 Lyon - FRANCE
elozes@ens-lyon.fr

ABSTRACT

This paper compares separation logic to a classical fragment of it. We prove that they are equally expressive, and that the separative power is obtained using only monotonic assertions.

1. INTRODUCTION

Imperative programming languages manipulating pointers allow one to change the value a variable refers to without explicitly mentioning this variable. Such multiple accesses to data make the axiomatic semantics [4] of these programs difficult to handle using classical logic as an assertion language [6]. Separation logic [7] is a proposal for an extension of the assertion language that nicely handles the subtleties of pointer manipulation. It provides two new connectives: a separative conjunction $P * Q$ asserting that P and Q hold in separate parts of the memory, and a separating implication $P \mathrel{-\!\!*} Q$ allowing one to introduce 'spatial hypotheses' about the memory. In [7], the example proof of an in-place reversal of a list turns out to require complex invariants in the standard classical logic, whereas it has a simple formulation in separation logic.

So separation logic offers more concise and meaningful assertions than classical logic. We may raise the question whether it also provides new assertions, that is assertions that cannot be formulated in classical logic. For several examples, classical logic provides a formulation of any given invariant, although usually through costly and poorly scalable methods, as the list reversal example shows. In other words, separation logic could have the same expressive power as classical logic. Our aim in this work is to give a formal account of this intuition, at least for a simple though significant assertion language.

We consider the spatial assertion language presented in [1] for a proof of decidability. We define a *classical* fragment excluding the connectives $*$ and $\mathrel{-\!\!*}$ and adding new primitive assertions, and prove it to be as expressive as the whole language. The proof relies on the use of an intensional model equivalence; such equivalences are common for the study of the expressiveness of a logic (see [3, 5] for spatial logic cases), but were also exploited for decidability issues in [1, 2]. Our intensional equivalence factorize with the equivalence relation presented in [1], which shows its correction respect to Separation logic.

In Section 2 we collect all definitions of the assertion languages; Section 3 defines the intensional equivalence and establishes its correction; Section 4 gives the essential facts to establish the translation from separation logic in our classical fragment, and Section 5 gives some concluding remarks.

2. DEFINITIONS

We consider the assertion language presented in [1].

We assume a countable set Var of variables, ranged over with x, y, and a set Loc of locations such that Loc. Expressions and assertions are defined by the following grammar:

$$\begin{array}{rcl} e & ::= & x \mid \mathsf{nil} \\ P & ::= & (e \mapsto e, e) \mid e = e \mid \mathsf{emp} \mid \bot \mid P \Rightarrow P \\ & & \mid P * P \mid P \mathrel{-\!\!*} P \end{array}$$

We write $\mathsf{v}(P)$ for the set of variables occuring in P. Assertions express properties of memory states, modelled as a pair consisting of a store and a heap, as follows:

$$\begin{array}{rcl} \mathrm{Val} & \stackrel{\mathrm{def}}{=} & \mathrm{Loc} \sqcup \{\mathsf{nil}\} \\ \mathrm{Store} & \stackrel{\mathrm{def}}{=} & \mathrm{Var} \to \mathrm{Val} \\ \mathrm{Heap} & \stackrel{\mathrm{def}}{=} & \mathrm{Loc} \rightharpoonup_{fin} \mathrm{Val} \\ \mathrm{State} & \stackrel{\mathrm{def}}{=} & \mathrm{Stack} \times \mathrm{Heap} \end{array}$$

where $\rightharpoonup_{fin}$ stands for a partial function with finite domain. We range over stores with s, over heaps with h, and over states with σ. We note $\sigma_1 \perp \sigma_2$ for $s_1 = s_2$ and $dom(h_1) \cap dom(h_2) = \emptyset$, and, when this holds, $\sigma_1 * \sigma_2$ is the state defined by keeping the same store and by setting $h_1 * h_2(x) = h_1(x)$ or $h_2(x)$.

We note $[\![e]\!]\sigma$ for the evaluation of e in σ, that is $[\![x]\!]\sigma = s(x)$ and $[\![\mathsf{nil}]\!]\sigma = \mathsf{nil}$. The condition for a state σ to match an assertion P, written $\sigma \models P$, is defined inductively by:

$\sigma \models \bot$	never	
$\sigma \models (e \mapsto e_1, e_2)$	iff	$dom(h) = \{[\![e]\!]\sigma\}$ and $h([\![e]\!]\sigma) = ([\![e_1]\!]\sigma, [\![e_2]\!]\sigma)$
$\sigma \models e_1 = e_2$	iff	$[\![e_1]\!]\sigma = [\![e_2]\!]\sigma$
$\sigma \models \mathsf{emp}$	iff	$dom(h) = \emptyset$
$\sigma \models P_1 \Rightarrow P_2$	iff	$\sigma \models P_1$ implies $\sigma \models P_2$
$\sigma \models P_1 * P_2$	iff	there exist σ_1 and σ_2 such that $\sigma = \sigma_1 * \sigma_2$; $\sigma_1 \models P_1$ and $\sigma_2 \models P_2$
$\sigma \models P_1 -\!\!* P_2$	iff	for all σ_1 such that $\sigma \perp \sigma_1$, $\sigma_1 \models P_1$ implies $\sigma * \sigma_1 \models P_2$

We may define as usual the connectives $\wedge, \vee, \top, \neg, \Leftrightarrow$ in the obvious way. We also introduce three *monotonic*[1] assertions with the following semantic:

$$\begin{array}{rcl}
(e \hookrightarrow e') & \overset{\text{def}}{=} & (e \mapsto e') * \top \\
\sigma \models (e \hookrightarrow e') & \Leftrightarrow & [\![e]\!]\sigma \in dom(h),\ h([\![e]\!]\sigma) = [\![e']\!]\sigma \\
\mathsf{alloc}\, x & \overset{\text{def}}{=} & (x \mapsto \mathsf{nil}) -\!\!* \bot \\
\sigma \models \mathsf{alloc}\, x & \Leftrightarrow & s(x) \in dom(h) \\
\mathsf{size} \geq n & \overset{\text{def}}{=} & \underbrace{\neg\,\mathsf{emp} * \ldots * \neg\,\mathsf{emp}}_{n \text{ times}} \\
\sigma \models \mathsf{size} \geq n & \Leftrightarrow & \sharp dom(h) \geq n
\end{array}$$

Any assertion of this form, or of the form $e = e'$ will be said to be *atomic*. In the remainder, we actually take these as primitive, which allows us to encode $e \mapsto e'$ and emp assertions by boolean combinations (on the contrary, it is not possible to encode $(e \hookrightarrow e')$ and $\mathsf{size} \geq n$ from $e \mapsto e'$ and emp using boolean combinations – this point is also discussed in conclusion). We call *classical fragment* the set of assertions given by the following grammar:

$$\begin{array}{rl}
P ::= & P \Rightarrow P \ \mid\ \bot \\
& \mid\ (e \hookrightarrow e') \ \mid\ \mathsf{alloc}\, x \ \mid\ e_1 = e_2 \ \mid\ \mathsf{size} \geq n\,.
\end{array}$$

We will note $w(P)$ for the maximal n such that $\mathsf{size} \geq n$ is a subassertion of P, and $\mathsf{v}(P)$ for the set of variables of P.

Our main result is the following:

THEOREM 2.1. *For all assertion P, there exists a classical assertion P' such that $\models P \Leftrightarrow P'$.*

At the same time, we also prove the following result: the monotonic (indeed atomic) fragment is as separative as the whole language, that is if two states satisfy the same monotonic assertions, then they satisfy the same assertions.

3. INTENSIONAL EQUIVALENCE

The encoding is based on a notion of equivalence between states that is reminiscent of intensional bisimilarity in the context of process algebras [3]. Let X be a finite set of variables, and w and integer. We say that two states σ and σ' are intensionally equivalent for X, w, written $\sigma \approx_{X,w} \sigma'$, if for all classical assertion P with $\mathsf{v}(P) \subseteq X$ and $w(P) \leq w$, $\sigma \models P \ \Leftrightarrow\ \sigma' \models P$.

Remarks:

[1]or *intuitionistic*, using the terminology of [7], that is assertions P such that $\sigma \models P$ implies $\sigma' \models P$ for all $\sigma' \geq \sigma$.

- This definition amounts to say that σ and σ' satisfy the same atomic classical assertions P with $\mathsf{v}(P) \subseteq X$ and $w(P) \leq w$.
- Let us write $w(\sigma) = \sharp dom(h)$. Given three natural numbers a, b, w, we write $a =_w b$ if either $a = b$ or $a, b \geq w$. Then for any σ, σ' such that $\sigma \approx_{X,w} \sigma'$, $w(\sigma) =_w w(\sigma')$.
- Equality assertions $x = y$ only depend on the store. We note $s =_X s'$ if these stores satisfy the same equality assertions with variables in X. Then for any σ, σ' such that $\sigma \approx_{X,w} \sigma'$, $s =_X s'$.
- Let V be some set of values. We note $v =_V v'$ if either $v = v'$ or $\{v, v'\} \cap V = \emptyset$. Then for any s, h, h' such that $(s, h) \approx_{X,w} (s, h')$, $dom(h) \cap s(X) = dom(h') \cap s(X)$ due to assertions $\mathsf{alloc}\, x$, and for all $l \in s(X) \cap dom(h)$, $h(l) =_{s(X) \cup \mathsf{nil}} h'(l)$ due to assertions $e \hookrightarrow e'$.

Let say more about store equivalence. Consider a store s_0 and a state $\sigma = (s, h)$ such that $s_0 =_{\mathsf{Var}} s$. Then we may define a new state $\mathsf{shift}_{s_0}\sigma$ of store s_0 and heap h' defined such that

- $dom(h') = s_0\big(s^{-1}(dom(h)) \cap \mathsf{Var}\big) \cup B$ with B some arbitrary set of locations such that $\sharp dom(h) = \sharp dom(h')$ and $B \cap s_0(\mathsf{Var}) = \emptyset$.
- for all $l \in dom(h')$, if $l = s_0(x)$ and $hs(x) = (s(y), s(z))$ for some $x, y, z \in \mathsf{Var}$, $h' s_0(x)$ is set to be $(s_0(y), s_0(z))$, otherwise $h'(l)$ is arbitrarily defined out of $s_0(\mathsf{Var})$.

This is easy to check that σ and $\mathsf{shift}_{s_0}\sigma$ satisfy the same atomic assertions. Moreover, this transformation is compositional, in the sense that $\mathsf{shift}_{s_0}(\sigma * \sigma') = \mathsf{shift}_{s_0}\sigma * \mathsf{shift}_{s_0}\sigma'$. This transformation is not completely deterministic, but assuming that every choice of a "fresh" value is made different at each time and at each call to shift, $\sigma \perp \tau$ will imply $\mathsf{shift}_{s_0}\sigma \perp \mathsf{shift}_{s_0}\tau$. We actually have the following stronger result:

LEMMA 3.1. *For all assertion P (of Separation logic), $\sigma \models P$ iff $\mathsf{shift}_{s_0}\sigma \models P$.*

The proof is by induction on the assertion P exploiting the previous remarks.

We now recall the equivalence relation defined by Yang in [8] for the decidability proof, and use it to derive the correction of $\approx_{X,w}$.

DEFINITION 3.2 ($\sim_{s,n,X}$[8]). *Given a stack s, a natural number n and a set X of variables, $\sim_{s,n,X}$ is a relation between heaps such that $h \sim_{s,n,X} h'$ iff*

1. *$s(X) \cap dom(h) = s(X) \cap dom(h')$;*
2. *for all $l \in s(X) \cap dom(h)$, $h(l) =_{s(X)} h'(l)$;*
3. *$\sharp\big(dom(h) - s(X)\big) \ =_n\ \sharp\big(dom(h') - s(X)\big)$.*

The first step of the correction proof is to factorize $\approx_{X,w}$ in $\sim_{s,n,X}$.

LEMMA 3.3. *For any X, w, n such that $n + \sharp X \leq w$, for any $\sigma, \sigma', s, h, h'$ such that $\sigma = (s, h)$, $\sigma \approx_{X,w} \sigma'$, and $\mathsf{shift}_s\sigma' = (s, h')$, it holds that $h \sim_{s,n,X} h'$.*

Proof. By Lemma 3.1, $(s, h) \approx_{X,w} (s, h')$. Then conditions 1 and 2 in Definition 3.2 holds by the previous remark, so the proof follows from the verification of the condition 3 on the heap size: let assume first that $\sharp(dom(h) - s(X)) < n$; then $\sharp dom(h) = k < n + \sharp X \leq w$, so $\sigma \models P = \mathsf{size} \geq k \wedge \neg\,\mathsf{size} \geq k+1$, and $w(P) = k+1 \leq w$. By definition of $\approx_{X,w}$, $\sigma' \models P$, so $\sharp dom(h') = k = \sharp dom(h)$, and finaly $\sharp(dom(h) - s(X)) = \sharp(dom(h') - s(X))$. (since $s(X) \cap dom(h) = s(X) \cap dom(h')$). Let assume now that $\sharp(dom(h) - s(X)) \geq n$; then $\sharp dom(h) \geq k \geq n + \sharp(dom(h) \cap s(X))$, where $k = \min(\sharp dom(h), w)$. So $\sigma \models \mathsf{size} \geq k$, and by definition of $\approx_{X,w}$, $\sigma' \models \mathsf{size} \geq k$, so that finaly $\sharp dom(h') \geq n + \sharp(dom(h') \cap s(X))$. $\square$

We recall now the correction result obtained by Yang and derive our correction from it. First we recall the notion of formula's size used by Yang:

$$\begin{array}{rcl@{\qquad}rcl}
\mid (e \mapsto e_1, e_2) \mid & = & 1 & \mid e_1 = e_2 \mid & = & 0 \\
\mid P \Rightarrow Q \mid & = & \max(\mid P \mid, \mid Q \mid) & \mid \perp \mid & = & 0 \\
\mid P * Q \mid & = & \mid P \mid + \mid Q \mid & \mid P -\!\!* \, Q \mid & = & \mid Q \mid \\
\mid emp \mid & = & 1 & & &
\end{array}$$

Lemma 3.4 ([8]). *For all s, h, h', n, X, P such that $\mathsf{v}(P) \subseteq X$, $\mid P \mid \leq n$ and $h \sim_{s,n,X} h'$, $(s,h) \models P$ iff $(s,h') \models P$.*

Corollary 3.5 (Correction). *For all σ, σ', w, X, P such that $\mathsf{v}(P) \subseteq X$, $\mid P \mid + \sharp X \leq w$ and $\sigma \approx_{X,w} \sigma'$, $\sigma \models P$ iff $\sigma' \models P$.*

Proof. By Lemma 3.3, $h \approx_{s,n,X} h'$ with $\sigma = (s,h)$, $\mathsf{shift}_s \sigma' = (s, h')$, and $n = w - \sharp X$. Then $\sigma \models P$ implies $\mathsf{shift}_s \sigma' \models P$ by Lemma 3.4, which implies $\sigma' \models P$ by Lemma 3.1. $\square$

4. TRANSLATION

We deals now with the translation of Separation logic into the classical fragment. Whereas for decidability issues, the concluding argument was to exhibit an effective enumeration of the classes of $\sim_{s,n,X}$, the property required here is more logical; we actually need only the finiteness of the classes of $\approx_{X,w}$ together with their expressibility in the classical fragment.

We write $\Phi_{X,w}$ for the set of atomic assertions P such that $\mathsf{v}(P) \subseteq X$ and $w(P) \leq w$. For X finite, $\Phi_{X,w}$ is finite as well. This has two important consequences:

Proposition 4.1 (Precompactness). *For all w and all finite X, $\approx_{X,w}$ has only finitely many classes.*

Proof. A class is represented by a subset $\Phi \subseteq \Phi_{X,w}$ of atomic assertions that are the ones satisfied by any state of the class. So there are less than $2^{\sharp \Phi_{X,w}}$ distinct classes. $\square$

Proposition 4.2 (Characteristic formula). *For all states σ, for all X, w, there is a classical assertion $F_\sigma^{(X,w)}$ such that*

$$\forall \sigma'. \quad \sigma' \models F_\sigma^{(X,w)} \ \textit{ iff } \ \sigma \approx_{X,w} \sigma'.$$

Proof. Take

$$\bigwedge_{\sigma \models P, P \in \Phi_{X,w}} P \quad \wedge \bigwedge_{\sigma \not\models P, P \in \Phi_{X,w}} \neg P.$$

$\square$

We may now establish Theorem 2.1 noticing that any assertion P is equivalent to the classical assertion:

$$\bigvee_{\sigma \in \mathrm{State}_{/\approx_{X,w}}, \sigma \models P} F_\sigma^{(X,n)},$$

where the finiteness of this disjunction is ensured by Proposition 4.1.

5. CONCLUSION

We defined a classical fragment of the separation logic, excluding both $*$ and $-\!\!*$, and proved it to be as expressive as the full separation logic. Our approach shows also that all the separative power of the logic lies in the monotonic fragment.

The elimination of the connective $\triangleright$ (equivalent to $-\!\!*$) has been established for another spatial logic [5]. For Separation logic, it is even possible to eliminate spatial conjunction, which cannot hold for other spatial logics where multiple copies of the same structure may coexist. The use of equality assertions is essential for that (relation $=_X$ and Lemma 3.1), since the $*$ connective does express distinctions between pointers. For instance, $x \hookrightarrow - * y \hookrightarrow -$ says that $x \neq y$. Equalities play also an essential role to handle quantifiers, as a counterexample in [5] tends to show.

When defining our classical fragment, we had to move from the assertions $e \mapsto e'$ and emp to $e \hookrightarrow e'$ and $\mathsf{size} \geq n$ in order to capture the $*$ connective, and we needed as well to define the extra primitive $\mathsf{alloc}\, x$ to capture the expressiveness of $-\!\!*$. This would not happen for an assertion language with lookup and quantifiers, where the only necessary atomic assertions are equality assertions.

We do not study the effectiveness of the translation, but it could certainly be proved. However, our approach seems independent from decidability issues (see [5]).

Yang proposed a clever counterexample to the elimination of $-\!\!*$ in a Separation logic with quantifiers; an equivalent result was established for the static Ambient logic in [5], but the example of Yang seems of deeper meaning. We do not know wether our result remains true for the assertion language with only $*$ and quantifiers.

Acknowledgements: I would like to thank Daniel Hirschkoff for his support in the preparation of this work. I merely thank Hongseok Yang for the rich discussions he took the time to offer me about the first, approximative version of this work. Without his contribution, this work would be a lot different.

6. REFERENCES

[1] C. Calcagno, H. Yang, and P. O'Hearn. Computability and Complexity Results for a Spatial Assertion Language for Data Structures. In *Proceedings of FSTTCS '01*, volume 2245 of *LNCS*. Springer Verlag, 2001.

[2] C.Calcagno, L. Cardelli, and A. Gordon. Deciding Validity in a Spatial Logic for Trees. In *Proc. of TLDI'03*, pages 62–73. ACM, 2003.

[3] D. Hirschkoff, E. Lozes, and D. Sangiorgi. Separability, Expressiveness and Decidability in the Ambients Logic. In *17th IEEE Symposium on Logic in Computer Science*, pages 423–432. IEEE Computer Society, 2002.

[4] C.A.R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, pages 12(10):576–580, october 1969.

[5] E. Lozes. Adjunct elimination in the static ambient logic. In *Proceedings of Express '03*, 2003.

[6] J. Reynolds. Intuitionistic reasoning about shared mutable data structure, 2000.

[7] J. Reynolds. Separation logic: a logic for shared mutable data structures. 2002.

[8] Hongseok Yang. *Local Reasoning for Stateful programs.* PhD thesis, University of Illinois at Urbana Champaign, 2001.