Biabduction for Separation Logic

Deliverable D1-4

ANR project VECOLIB

September 2017

Abstract

This deliverable reports on the bi-abduction procedures for Separation Logic developped during the VECOLIB project. This includes work on Separation Logic with and without inductive predicates, carried out at IRIF and VERIMAG, respectively.

1 Definition of Separation Logic

1.1 Syntax

We consider a signature Σ, such that $\Sigma^{\mathrm{s}} = \{\mathsf{Loc}, \mathsf{Bool}\}$ and $\Sigma^{\mathrm{f}} = \emptyset$, i.e. the only sorts are the boolean and *location* sort, with no function symbols defined on it, other than equality. Observe that, in this case $\mathcal{T}_\Sigma(\mathbf{x}) = \mathbf{x}$, for any $\mathbf{x} \subseteq \mathsf{Var}$, i.e. the only terms occurring in a formula are variables of sort Loc. In the rest of this section we consider systems whose constraints are Separation Logic formulae, generated by the following syntax:

$$\varphi ::= \bot \mid x \approx y \mid \mathsf{emp} \mid x \mapsto (y_1, \ldots, y_k) \mid \varphi_1 * \varphi_2 \mid \varphi_1 \mathrel{-\!\!*} \varphi_2 \mid \neg\varphi_1 \mid \varphi_1 \wedge \varphi_2 \mid \exists x \,.\, \varphi_1$$

where $k > 0$ is a fixed constant denoting the number of outgoing selector fields from a memory cell. We consider the following shorthand notations:

- $\top \equiv \neg\bot$
- $x \hookrightarrow (y_1, \ldots, y_k) \equiv x \mapsto (y_1, \ldots, y_k) * \top$
- $\mathsf{alloc}(x) \equiv x \mapsto (x, \ldots, x) \mathrel{-\!\!*} \bot$
- for any $n \in \mathbb{Z}$: $|h| \geq n \equiv \begin{cases} \top \text{ if } n \leq 0 \\ (|h| \geq n-1) * \neg\mathsf{emp} \text{ otherwise} \end{cases}$

1.2 Semantics

We interpret Loc as a countably infinite set L. A *heap* is a finite partial mapping $h : \mathsf{L} \rightharpoonup_{\mathsf{fin}} \mathsf{L}^k$ associating locations with k-tuples of locations. We denote by $\mathrm{dom}(h)$ the set of locations on which h is defined, by $\mathrm{img}(h)$ the set of locations occurring in the range of h, and by Heaps the set of heaps. Two heaps h_1 and h_2 are disjoint if $dom(h_1) \cap dom(h_2) = \emptyset$. In this case $h_1 \uplus h_2$ denotes their union, which is undefined if h_1 and h_2 are not disjoint. Given a valuation $\nu : \mathsf{Var} \to \mathsf{L}$ and a heap h, the semantics of

SL formulae is defined as follows:

$$
\begin{array}{lcl}
\nu, h \models x \approx y & \iff & \nu(x) = \nu(y) \\
\nu, h \models \mathsf{emp} & \iff & h = \emptyset \\
\nu, h \models x \mapsto (y_1, \ldots, y_k) & \iff & h = \{\langle \nu(x), (\nu(y_1), \ldots, \nu(y_k)) \rangle\} \\
\nu, h \models \phi_1 * \phi_2 & \iff & \text{there exist } h_1, h_2 \in \mathsf{Heaps}: \ h = h_1 \uplus h_2 \text{ and } \mathcal{I}, h_i \models_{\mathsf{sl}} \phi_i, i = 1,2 \\
\nu, h \models \phi_1 \mathrel{-\!\!*} \phi_2 & \iff & \text{for all } h' \in \mathsf{Heaps}: \text{ if } \mathrm{dom}(h') \cap \mathrm{dom}(h) = \emptyset \text{ and } \nu, h' \models \phi_1 \text{ then } \nu, h' \uplus h \models \phi_2 \\
\nu, h \models \exists x \,.\, \varphi(x) & \iff & \nu[x \leftarrow \ell], h \models \varphi(x), \text{ for some } \ell \in \mathsf{L}
\end{array}
$$

The semantics of boolean connectives is the usual one, omitted for space reasons. A formula ϕ is *satisfiable* if there exists a valuation ν and a heap h such that $\nu, h \models \phi$. Given formulae φ and ψ, we say that ϕ *entails* ψ, denoted $\phi \models \psi$ iff $\nu, h \models \varphi$ implies $\nu, h \models \psi$, for each valuation ν and each heap h.

1.2.1 Bi-Abduction Problems

Consider two formulae φ and ψ of Separation Logic (SL). The problems of abduction, frame inference and bi-abduction are defined below:

- the *abduction* problem asks for a formula X such that $\varphi * X \models \psi$,
- the *frame inference* problem asks for a formula Y such that $\varphi \models \psi * Y$,
- the *bi-abduction* problem asks for formulae X and Y such that $\varphi * X \models \psi * Y$.

2 Separation Logic without Inductive Definitions

We address first the three problems above in case φ and ψ do not contain predicate atoms. It is not difficult to prove that:

1. the weakest solution of the abduction problem is $X \equiv \varphi \mathrel{-\!\!*} \psi$, and
2. the strongest solution of the frame inference problem is $Y \equiv \neg(\varphi \mathrel{-\!\!*} \neg\psi)$, also denoted as $\varphi \multimap \psi$.

Based on these observations, a possible solution to the bi-abduction problem can be defined as follows:

$$
\begin{array}{lcl}
X & \equiv & \varphi \mathrel{-\!\!*} (\psi * \top) \\
Y & \equiv & (\varphi * X) \multimap \psi \equiv (\varphi * (\varphi \mathrel{-\!\!*} (\psi * \top))) \multimap \psi
\end{array}
$$

The main difficulty with the above solutions is the use of the magic wand $\mathrel{-\!\!*}$ connective, which poses important problems for automated reasoning. The solution we consider is to translate any SL formula as a boolean combination of SL-minterms, defined in the following.

Definition 1 *An* SL-minterm *is either* $\bot$ *or a formula of the form:*

$$\phi^{eq} \wedge \ \phi^{pt} \wedge \ \phi^{a} \wedge \ |h| \geq \min_\phi \wedge \ |h| < \max_\phi$$

where:

- ϕ^{eq} *is a conjunction of equalities and disequalities,*
- ϕ^{pt} *is a conjunction of literals of the form* $x \hookrightarrow (y_1, \ldots, y_k)$ *or* $x \not\hookrightarrow (y_1, \ldots, y_k)$*,*
- ϕ^{a} *is a conjunction of literals of the form* $\mathsf{alloc}(x)$ *or* $\neg\mathsf{alloc}(x)$*,*
- $\min_\phi \in \mathbb{N}$ *and* $\max_\phi \in \mathbb{N} \cup \{\infty\}$.

The main result here is that any quantifier-free SL formula is equivalent to a boolean combination of SL-minterms. This result is obtained by giving translations for the formulae $\phi_1 * \phi_2$ and $\phi_1 \mathrel{-\!\!*} \phi_2$, where ϕ_1 and ϕ_2 are SL-minterms satisfying a few additional restrictions, such as completness w.r.t.

equalities and allocations. This latter conditions can always be enforced by considering a finite disjunction of cases, stating which variables are equal, not equal, allocated and not allocated, respectively. The translation can be implemented in polynomial space, which means that the bi-abduction problems above can be solved in polynomial space and exponential time.