A simple separation logic

Andreas Herzig

University of Toulouse, CNRS
Institut de recherche en informatique de Toulouse (IRIT)
Toulouse, France
`www.irit.fr/~Andreas.Herzig`

Abstract

The kinds of models that are usually considered in separation logic are structures such as words, trees, and more generally pointer structures (heaps). In this paper we introduce the separation logic of much simpler structures, viz. sets. The models of our set separation logic are nothing but valuations of classical propositional logic. Separating a valuation V consists in splitting it up into two partial valuations v_1 and v_2. Truth of a formula $\varphi_1 * \varphi_2$ in a valuation V can then be defined in two different ways: first, as truth of φ_1 in *all* total extensions of v_1 and truth of φ_2 in *all* total extensions of v_2; and second, as truth of φ_1 in *some* total extension of v_1 and truth of φ_2 in *some* total extension of v_2. The first is an operator of separation of resources: the update of $\varphi_1 * \varphi_2$ by ψ is the conjunction of the update of φ_1 by ψ and the update of φ_2 by ψ; in other words, $\varphi_1 * \varphi_2$ can be updated independently. The second is an operator of separation of processes: updates by $\psi_1 * \psi_2$ can be performed independently. We show that the satisfiability problem of our logic is decidable in polynomial space (PSPACE). We do so by embedding it into dynamic logic of propositional assignments (which is PSPACE complete). We moreover investigate its applicability to belief update and belief revision, where the separation operators allow to formulate natural requirements on independent pieces of information.

1 Introduction

Separation logics [7, 13, 17] have a modal operator $*$ which allows to talk about the separation of resources. Basically, the formula $\varphi_1 * \varphi_2$ is true in the model M if M can be split into two parts M_1 and M_2 such that φ_1 is true in M_1 and φ_2 is true in M_2. The kinds of models that are usually considered in separation logic are structures such as words, trees, and more generally pointer structures (heaps). The separation logics of such structures are often undecidable. In this paper we investigate the separation logic of much simpler structures, viz. sets. We call our logic *set separation logic*, abbreviated SSL. The models of SSL are nothing but valuations of classical propositional logic. Separating a valuation V consists in splitting it up into two partial valuations v_1 and v_2. Then separability of φ_1 and φ_2 in a valuation V can be defined in two different ways:

first, as truth of φ_1 in *all* total extensions of v_1 and truth of φ_2 in *all* total extensions of v_2; and second, as truth of φ_1 in *some* total extension of v_1 and truth of φ_2 in *some* total extension of v_2. We respectively denote these two separation operators by $\dot\wedge$ and $\dot{\|}$. We chose the symbol $\dot\wedge$ due to its analogy with the symbol of disjoint union $\dot\cup$, and we chose the symbol $\dot{\|}$ because $\|$ denotes parallel execution.

We show that the satisfiability problem of set separation logic is decidable in polynomial space (PSPACE). We do so by embedding SSL into dynamic logic of propositional assignments DL-PA [2], whose star-free fragment is PSPACE complete. This contrasts with separation logics having the implicational connective $-\!\!*$, which are often undecidable even in the proopositional language [4, 11].

Our initial motivation to investigate separation operators was that they can be given an interesting interpretation in the context of the revision and update of propositional belief bases: we consider that when φ_1 and φ_2 are separable then they are independent pieces of information. This naturally leads to the following requirements.

- We suppose that $\dot\wedge$ expresses independence of resources: the update of $\varphi_1 \dot\wedge \varphi_2$ by ψ is the conjunction of the update of φ_1 by ψ and the update of φ_2 by ψ; in other words, $\varphi_1 \dot\wedge \varphi_2$ can be updated independently.

- We suppose that $\dot{\|}$ expresses independence of processes: the update of φ by $\psi_1 \dot{\|} \psi_2$ is the parallel update of φ by ψ_1 and by ψ_2; in other worlds, updates by $\psi_1 \dot{\|} \psi_2$ can be performed independently.

This extends previous approaches by Parikh, Makinson and others that are based on splitting languages [3,10,14]. We investigate the compatibility of existing belief change operations with the above two requirements.

The paper is organised as follows. In Section 2 we introduce set separation logic SSL. In Section 3 we provide a PSPACE upper bound for both its model checking and its satisfiability problem. In Section 4 we discuss the relation between SSL and language splitting-based belief change. Section 5 concludes.

2 Set separation logic SSL

Throughout the paper we use the following conventions.

$\mathbb{P} = \{p, q, \ldots\}$ is a countable set of propositional variables. The set $\{P_1, P_2\}$ is a partition of $\mathbb{P}$ iff $P_1 \cup P_2 = \mathbb{P}$ and $P_1 \cap P_2 = \emptyset$.

A *valuation* is a total function from $\mathbb{P}$ to $\{0, 1\}$. We use V, V_1, ... for valuations. Two valuations V and V' *agree on the set of variables* $P \subseteq \mathbb{P}$, if both give the same truth value to each of the variables in P: $V \sim_P V'$ iff $V(p) = V'(p)$ for every $p \in P$.

A *partial valuation* is a partial function from $\mathbb{P}$ to $\{0, 1\}$. For a valuation $V : \mathbb{P} \longrightarrow \{0, 1\}$ and a set of propositional variables $P \subseteq \mathbb{P}$, the *restriction of* V *to* P is the partial function whose domain is P, noted $V|_P$. We use v, v_1, ... for partial valuations. The total valuation V is an *extension* of the partial valuation v if $V(p) = v(p)$ for every $p \in \mathsf{dom}(v)$.[1]

[1]We might as well define valuations to be sets of propositional variables. However, it would have been less elegant to account for partial valuations under such a presentation.

The *language of* SSL is defined by the following grammar:

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \mathbin{\dot\wedge} \varphi \mid \varphi \mathbin{\dot{\|}} \varphi$$

where p ranges over the set of propositional variables $\mathbb{P}$. The formula $\varphi \mathbin{\dot\wedge} \psi$ may be read "φ and ψ are statically separable" and $\varphi \mathbin{\dot{\|}} \psi$ may be read "φ and ψ are dynamically separable". Our intuition is the following: when $\varphi \mathbin{\dot\wedge} \psi$ is true then the conjunction of φ and ψ can be updated separately; and when $\varphi \mathbin{\dot{\|}} \psi$ is true then updating by the conjunction of φ and ψ can be performed in parallel.

We abbreviate the logical connectives $\wedge$, $\rightarrow$ and $\leftrightarrow$ in the usual way.

The truth conditions are as follows:

$$\begin{array}{rll}
V \models p & \text{iff} & V(p) = 1; \\
V \models \neg\varphi & \text{iff} & V \not\models \varphi; \\
V \models \varphi_1 \wedge \varphi_2 & \text{iff} & V \models \varphi_1 \text{ and } V \models \varphi_2; \\
V \models \varphi_1 \mathbin{\dot\wedge} \varphi_2 & \text{iff} & \text{there is a partition } \{P_1, P_2\} \text{ of } \mathbb{P} \text{ such that} \\
& & V_1 \models \varphi_1 \text{ for every valuation } V_1 \text{ agreeing with } V \text{ on } P_1 \text{ and} \\
& & V_2 \models \varphi_2 \text{ for every valuation } V_2 \text{ agreeing with } V \text{ on } P_2; \\
V \models \varphi_1 \mathbin{\dot{\|}} \varphi_2 & \text{iff} & \text{there is a partition } \{P_1, P_2\} \text{ of } \mathbb{P} \text{ such that} \\
& & V_1 \models \varphi_1 \text{ for some valuation } V_1 \text{ agreeing with } V \text{ on } P_1 \text{ and} \\
& & V_2 \models \varphi_2 \text{ for some valuation } V_2 \text{ agreeing with } V \text{ on } P_2.
\end{array}$$

The conditions for the two separation operators can be reformulated in terms of partial valuations as follows:

$$\begin{array}{rll}
V \models \varphi_1 \mathbin{\dot\wedge} \varphi_2 & \text{iff} & \text{there is a partition } \{P_1, P_2\} \text{ of } \mathbb{P} \text{ such that} \\
& & V_1 \models \varphi_1 \text{ for every extension } V_1 \text{ of } V|_{P_1} \text{ and} \\
& & V_2 \models \varphi_2 \text{ for every extension } V_2 \text{ of } V|_{P_2}; \\
V \models \varphi_1 \mathbin{\dot{\|}} \varphi_2 & \text{iff} & \text{there is a partition } \{P_1, P_2\} \text{ of } \mathbb{P} \text{ such that} \\
& & V_1 \models \varphi_1 \text{ for some extension } V_1 \text{ of } V|_{P_1} \text{ and} \\
& & V_2 \models \varphi_2 \text{ for some extension } V_2 \text{ of } V|_{P_2}.
\end{array}$$

Some observations:

- In the truth condition for $\dot\wedge$, the exhaustiveness condition $P_1 \cup P_2 = \mathbb{P}$ can be dropped. If we dropped the disjointness condition $P_1 \cap P_2 = \emptyset$ then $\varphi \mathbin{\dot\wedge} \psi$ trivialises to the conjunction $\varphi \wedge \psi$.

- In the truth condition for $\dot{\|}$, if we drop the exhaustiveness condition $P_1 \cup P_2 = \mathbb{P}$ then $\varphi \mathbin{\dot{\|}} \psi$ trivialises to the consistency of both φ and ψ.

Here are some examples. Let V_{pq} be a valuation such that $V_{pq}(p) = V_{pq}(q) = 1$ and let $V_{p\bar{q}}$ be a valuation such that $V_{p\bar{q}}(p) = 1$ and $V_{p\bar{q}}(q) = 0$. Then we have:

$$\begin{array}{lll}
V_{pq} \models p \mathbin{\dot\land} q & V_{pq} \models p \mathbin{\dot{\|}} q & V_{pq} \models (\neg p) \mathbin{\dot{\|}} (\neg q) \\
V_{pq} \models p \mathbin{\dot\land} (p \lor q) & V_{pq} \models p \mathbin{\dot{\|}} (\neg p \land \neg q) & V_{pq} \not\models \neg p \mathbin{\dot{\|}} (\neg p \land \neg q) \\
V_{p\bar{q}} \not\models p \mathbin{\dot\land} (p \lor q) & V_{p\bar{q}} \models p \mathbin{\dot{\|}} (p \lor q) & V_{p\bar{q}} \models \neg p \mathbin{\dot{\|}} (p \lor q) \\
V_{p\bar{q}} \models p \mathbin{\dot\land} (p \lor \neg q) & V_{p\bar{q}} \models p \mathbin{\dot{\|}} (p \lor \neg q) & V_{p\bar{q}} \not\models \neg p \mathbin{\dot{\|}} \neg (p \lor q)
\end{array}$$

Satisfiability and validity are defined as usual. The following formula schemas are valid:

$$\begin{array}{ll}
\varphi_1 \mathbin{\dot\land} \varphi_2 \leftrightarrow \varphi_2 \mathbin{\dot\land} \varphi_1 & \varphi_1 \mathbin{\dot{\|}} \varphi_2 \leftrightarrow \varphi_2 \mathbin{\dot{\|}} \varphi_1 \\
\varphi_1 \mathbin{\dot\land} \varphi_2 \to \varphi_2 \land \varphi_1 & \varphi_1 \land \varphi_2 \to \varphi_2 \mathbin{\dot{\|}} \varphi_1 \\
\top \mathbin{\dot\land} \varphi \leftrightarrow \varphi & \top \mathbin{\dot{\|}} \varphi \leftrightarrow \begin{cases} \top & \text{if } \varphi \text{ is satisfiable} \\ \bot & \text{otherwise} \end{cases}
\end{array}$$

As the last line shows, consistency of a formula φ can be expressed in the language of SSL by the formula $\top \mathbin{\dot{\|}} \varphi$. Here are two inference rules preserving validity:

$$\frac{\varphi \to \psi}{(\varphi \mathbin{\dot\land} \chi) \to (\psi \mathbin{\dot\land} \chi)} \qquad \frac{\varphi \to \psi}{(\varphi \mathbin{\dot{\|}} \chi) \to (\psi \mathbin{\dot{\|}} \chi)}$$

The following equivalences are valid, where the propositional variables p and q are supposed to be different:

$$\begin{array}{rr}
p \mathbin{\dot\land} p \leftrightarrow \bot & p \mathbin{\dot{\|}} p \leftrightarrow p \\
p \mathbin{\dot\land} \neg p \leftrightarrow \bot & p \mathbin{\dot{\|}} \neg p \leftrightarrow \top \\
p \mathbin{\dot\land} q \leftrightarrow p \land q & p \mathbin{\dot{\|}} q \leftrightarrow \top \\
p \mathbin{\dot\land} (p \lor q) \leftrightarrow p \land q & p \mathbin{\dot{\|}} (p \lor q) \leftrightarrow \top \\
(p \lor q) \mathbin{\dot\land} (p \lor q) \leftrightarrow p \land q & (p \lor q) \mathbin{\dot{\|}} (p \lor q) \leftrightarrow \top
\end{array}$$

3 Complexity

In this section we establish an upper bound for the complexity of both model checking and satisfiability checking of set separation logic. We prove this by showing that both $\varphi_1 \mathbin{\dot\land} \varphi_2$ and $\varphi_1 \mathbin{\dot{\|}} \varphi_2$ can be expressed in dynamic logic of propositional assignments DL-PA (that we have recently proposed with Philippe Balbiani and Nicolas Troquard in [2]) by equivalent formulas whose length is polynomial in the length of $\varphi_1 \mathbin{\dot\land} \varphi_2$ and $\varphi_1 \mathbin{\dot{\|}} \varphi_2$, respectively.

3.1 DL-PA: dynamic logic of propositional assignments

The language of DL-PA is defined by the following grammar:

$$\begin{array}{lll}
\pi & ::= & p{\leftarrow}\top \mid p{\leftarrow}\bot \mid \varphi? \mid \pi;\pi \mid \pi \cup \pi \mid \pi^* \\
\varphi & ::= & p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \lor \varphi \mid \langle\pi\rangle\varphi
\end{array}$$

$$\begin{aligned}
||p{\leftarrow}\top|| &= \{(V,V') \ : \ V'(p) = 1 \text{ and } V' \text{ agrees with } V \text{ on } \mathbb{P} \setminus \{p\}\} \\
||p{\leftarrow}\bot|| &= \{(V,V') \ : \ V'(p) = 0 \text{ and } V' \text{ agrees with } V \text{ on } \mathbb{P} \setminus \{p\}\} \\
||\pi;\pi'|| &= ||\pi|| \circ ||\pi'|| \\
||\pi \cup \pi'|| &= ||\pi|| \cup ||\pi'|| \\
||\pi^*|| &= \bigcup_{k \in \mathbb{N}_0} (||\pi||)^k \\
||\varphi?|| &= \{(V,V) \ : \ V \in ||\varphi||\} \\
||p|| &= \{V \ : \ V(p) = 1\} \\
||\top|| &= 2^{\mathbb{P}} \\
||\bot|| &= \emptyset \\
||\neg\varphi|| &= 2^{\mathbb{P}} \setminus ||\varphi|| \\
||\varphi \vee \psi|| &= ||\varphi|| \cup ||\psi|| \\
||\langle\pi\rangle\varphi|| &= \{V \ : \text{ there is } V' \text{ s.t. } (V,V') \in ||\pi|| \text{ and } V' \in ||\varphi||\}
\end{aligned}$$

Table 1: Interpretation of the DL-PA connectives

where p ranges over $\mathbb{P}$. So an *atomic program* of the language of DL-PA is a program of the form $p{\leftarrow}\varphi$. The operators of sequential composition (";"), nondeterministic composition ("$\cup$"), unbounded iteration ("*", the Kleene star), and test ("?") are familiar from PDL.

We define $\mathbb{P}_\varphi$ to be the set of variables from $\mathbb{P}$ occurring in formula φ, and we define $\mathbb{P}_\pi$ to be the set of variables from $\mathbb{P}$ occurring in program π. For example, $\mathbb{P}_{p\leftarrow q \cup p \leftarrow \neg q} = \{p, q\} = \mathbb{P}_{\langle p\leftarrow\bot\rangle q}$.

We abbreviate the logical connectives $\wedge$, $\rightarrow$ and $\leftrightarrow$ in the usual way. Moreover, $[\pi]\varphi$ abbreviates $\neg\langle\pi\rangle\neg\varphi$. The program `skip` abbreviates $\top?$ ("nothing happens") and the program $p{\leftarrow}q$ abbreviates $(q?; p{\leftarrow}\top) \cup (\neg q?; p{\leftarrow}\bot)$ ("p gets the truth value of q").

DL-PA programs are interpreted by means of a (unique) *relation between valuations*: atomic programs $p{\leftarrow}\top$ and $p{\leftarrow}\bot$ update valuations in the obvious way, and complex programs are interpreted just as in PDL by mutual recursion. Table 1 gives the interpretation of the DL-PA connectives.

A formula φ is DL-PA *valid* if $||\varphi|| = 2^{\mathbb{P}}$, and φ is DL-PA *satisfiable* if $||\varphi|| \neq \emptyset$. For example, the formulas $\langle p{\leftarrow}\top\rangle\top$ and $\langle p{\leftarrow}\top\rangle\varphi \leftrightarrow \neg\langle p{\leftarrow}\top\rangle\neg\varphi$ are DL-PA valid. Other examples of DL-PA validities are $\langle p{\leftarrow}\top\rangle p$ and $\langle p{\leftarrow}\bot\rangle\neg p$. Observe that if p does not occur in φ then both $\varphi \rightarrow \langle p{\leftarrow}\top\rangle\varphi$ and $\varphi \rightarrow \langle p{\leftarrow}\bot\rangle\varphi$ are valid. This is due to the following semantical property.

Proposition 1. *Suppose $p \notin \mathbb{P}_\varphi$, i.e., p does not occur in φ. Then $\varphi \in ||V \cup \{p\}||$ iff $\varphi \in ||V \setminus \{p\}||$.*

Theorem 1 ([2])**.** *For the full language, both the* DL-PA *satisfiability problem and the* DL-PA *model checking problem are EXPTIME complete.*

For the star-free fragment, both the DL-PA *satisfiability problem and the* DL-PA *model checking problem are PSPACE complete.*

3.2 Embedding set separation logic into DL-PA

We now give a polynomial transformation mapping set separation logic formulas φ_0 into DL-PA formulas.

Let P' be the set of variables p' such that p is in P and p' is fresh: p' does not occur in the formula φ_0 under consideration. The following abbreviations will be useful:

$$\begin{aligned}
\pm p &= p{\leftarrow}\top \cup p{\leftarrow}\bot \\
\mathsf{changeSome}(\{p_1, \cdots, p_n\}) &= \pm p_1; \cdots; \pm p_n \\
\mathsf{store}(\{p_1, \cdots, p_n\}) &= p'_1{\leftarrow}p_1; \cdots; p'_n{\leftarrow}p_n \\
\mathsf{retrieve}(\{p_1, \cdots, p_n\}) &= p_1{\leftarrow}p'_1; \cdots; p_n{\leftarrow}p'_n \\
\mathsf{changeSomeMarked}(\{p_1, \cdots, p_n\}) &= \neg(p_1 \leftrightarrow p'_1)? \cup (p_1 \leftrightarrow p'_1?; \pm p_1); \\
&\quad \cdots \\
&\quad \neg(p_n \leftrightarrow p'_n)? \cup (p_n \leftrightarrow p'_n?; \pm p_n) \\
\mathsf{changeSomeUnmarked}(\{p_1, \cdots, p_n\}) &= p_1 \leftrightarrow p'_1? \cup (\neg(p_1 \leftrightarrow p'_1)?; \pm p_1); \\
&\quad \cdots \\
&\quad p_n \leftrightarrow p'_n? \cup (\neg(p_n \leftrightarrow p'_n)?; \pm p_n) \\
\mathsf{changeRestAndRestore}(\{p_1, \cdots, p_n\}) &= ((p_1 \leftrightarrow p'_1?; \pm p_1) \cup (\neg(p_1 \leftrightarrow p'_1)?; p_1{\leftarrow}p'_1)); \\
&\quad \cdots \\
&\quad ((p_n \leftrightarrow p'_n?; \pm p_n) \cup (\neg(p_n \leftrightarrow p'_n)?; p_n{\leftarrow}p'_n))
\end{aligned}$$

The program $\mathsf{changeSome}(P)$ nondeterministically changes the truth value of some variables in P. The program $\mathsf{store}(P)$ stores the truth value of each variable p by means of a fresh variable p', and $\mathsf{retrieve}(P)$ reestablishes that 'old' value. When p and p' have different truth values then we say that p is marked; else we say that p is unmarked. The program $\mathsf{changeSomeMarked}(P)$ arbitrarily changes only the unmarked variables. The other way round, the program $\mathsf{changeSomeUnmarked}(P)$ leaves every unmarked $p \in P$ unchanged and arbitrarily changes the marked p's.

Observe that each of the above programs has length linear in the cardinality n of the set of propositional variables $\{p_1, \ldots, p_n\}$.

The next two propositions provide an embedding of set separation logic into DL-PA.

Proposition 2. *Let φ_1 and φ_2 be two propositional formulas. Let $P = \mathbb{P}_{\varphi_1} \cap \mathbb{P}_{\varphi_2}$. Let P' be the set of variables p' such that p is in P and p' is fresh: p' does not occur in the formula under consideration. Then the formula $\varphi_1 \mathbin{\|} \varphi_2$ is equivalent to the* DL-PA *formula*

$$\langle \mathit{store}(P); \mathit{changeSome}(P)\rangle \; (\; \langle \mathit{changeSome}(\mathbb{P}_{\varphi_1} \setminus P)\rangle\varphi_1 \; \wedge \\ \langle \mathit{changeRestAndRestore}(P)\rangle\langle \mathit{changeSome}(\mathbb{P}_{\varphi_2} \setminus P)\rangle\varphi_2 \;)$$

Proposition 3. *Let φ_1 and φ_2 be two propositional formulas. Let $P = \mathbb{P}_{\varphi_1} \cap \mathbb{P}_{\varphi_2}$. Let P' be the set of variables p' such that p is in P and p' is fresh: p' does not occur in the formula under consideration. Then the formula $\varphi_1 \dot{\wedge} \varphi_2$ is equivalent to the following* DL-PA *formula:*

$$\langle \mathit{store}(P); \mathit{changeSome}(P') \rangle \; (\; [\mathit{changeSomeMarked}(P)]\varphi_1 \wedge$$
$$[\mathit{changeSomeUnmarked}(P)]\varphi_2 \;)$$

Intuitively, after the program store(P) has stored the value of each of the elements of P, the program changeSome(P') allows to (nondeterministically) identify a subset of P: those p whose value differs from its copy p'. We consider that these 'marked' variables are those of the partial valuation for φ_1, while the complementary, unmarked variables make up the partial valuation for φ_2.

This can be turned more formally into a transformation from the language of set separation logic into the language of DL-PA. The transformation is clearly linear in the size of the original formula φ_0.

The codomain of the transformation is the star-free fragment of DL-PA. As both model checking and satisfiability checking in DL-PA are PSPACE complete, it follows that model checking and satisfiability checking in set separation logic are in PSPACE. It remains to investigate the lower bounds.

4 Separability in the context of belief change operations

As we have mentioned in the introduction, one can use the SSL operators to formulate new postulates for belief change operations such as AGM belief revision operators [1,6] and KM update operators [8, 9]. We investigate this now in more depth.

4.1 The basic belief change postulates

Let $\circ$ be a belief change operator and let β and ψ be boolean formulas. (We use β for the base and ψ for the input.) $\beta \circ \psi$ is the result of incorporating the input ψ into the base β. Both revision and update operations were mainly studied from a semantical perspective: $\beta \circ \psi$ is viewed as a set of valuations.

Katsuno and Mendelzon promoted the distinction between belief update and belief revision [9]. Their idea is that update keeps track of changes in the world while revision corrects errors about an unchanged world. This can be illustrated by the revised and updated edition of a dictionary: we say that it has been revised because past errors have been corrected, and we say that it has been updated because new usages of existing words have been added to it and outdated usages have been dropped. Traditionally, $\beta \diamond \psi$ denotes the update of the base β by the input ψ and $\beta * \psi$ denotes the revision of the base β by the input ψ.

Alchourrón, Gärdenfors and Makinson designed a set of postulates for belief revision operations (the so-called AGM postulates), and Katsuno and Mendelzon designed a set of postulates for belief update operations (the so-called KM-postulates). The following postulates are common to both kinds of operations:

(RE)	if $\|\beta_1\| = \|\beta_2\|$ and $\|\psi_1\| = \|\psi_2\|$ then $\beta_1 \circ \psi_1 = \beta_2 \circ \psi_2$
(SUCCESS)	$\beta \circ \psi \subseteq \|\psi\|$
(PRES$_w$)	if $\|\beta\| \subseteq \|\psi\|$ then $\beta \circ \psi = \|\beta\|$

where $||\varphi||$ is the set of valuations where φ is true (just as in Section 3.1). We call the above the *basic belief change postulates*.

RE is a postulate of insensitivity to syntax. SUCCESS says that belief change is successful: the input has priority. PRES$_w$ is a weak preservation postulate: if the input is already in the base then the base should not change. AGM revision operations moreover satisfy a strengthening of PRES$_w$:

$$\text{(PRES)} \quad \text{if } ||\beta|| \cap ||\psi|| \neq \emptyset \text{ then } \beta \circ \psi = ||\beta \wedge \psi||$$

4.2 Belief change operations and language splitting

It has been observed by many that the drastic update operation defined as

$$\beta \circ \psi = \begin{cases} ||\beta|| & \text{if } ||\beta|| \subseteq ||\psi|| \\ ||\psi|| & \text{otherwise} \end{cases}$$

satisfies the KM postulates. Similarly, the following drastic revision operation

$$\beta \circ \psi = \begin{cases} ||\beta \wedge \psi|| & \text{if } ||\beta|| \cap ||\psi|| \neq \emptyset \\ ||\psi|| & \text{otherwise} \end{cases}$$

satisfies the AGM postulates. In order to exclude such operations, Parikh, Makinson and others argued for a further postulate of relevance [3, 10, 14]. Its formulation refers to the syntax of the base and the input.

$$\text{(REL)} \quad (\beta_1 \wedge \beta_2) \circ \psi = (\beta_1 \circ \psi) \cap (\beta_2 \circ \psi) \qquad \text{if } \mathbb{P}_{\beta_1} \cap \mathbb{P}_{\beta_2} = \emptyset$$

Just as in Section 2, $\mathbb{P}_\varphi$ denotes the set of propositional variables occurring in the boolean formula φ. Therefore $\mathbb{P}_{\beta_1} \cap \mathbb{P}_{\beta_2} = \emptyset$ means that the signatures of β_1 and β_2 are disjoint: the languages of β_1 and β_2 can be split. Each of the above drastic operations violates the postulate REL.

4.3 Separation-based belief change operations

In the same spirit and as already stated informally in the introduction, the SSL operators enable us to go beyond such syntax-based postulates and strengthen the above relevance postulate REL. The strengthening comes in a static version and in a dynamic version:

$$\begin{array}{lllll} \text{(REL}_s\text{)} & (\beta_1 \dot{\wedge} \beta_2) \circ \psi & = & (\beta_1 \circ \psi) \cap (\beta_2 \circ \psi) \\ \text{(REL}_d\text{)} & \beta \circ (\psi_1 \dot{\|} \psi_2) & = & (\beta \circ \psi_1) \circ \psi_2 \\ & & = & (\beta \circ \psi_2) \circ \psi_1 \end{array}$$

where $\circ$ is any belief change operation, be it update or revision.[2] The static relevance postulate REL$_s$ says that when the bases β_1 and β_2 are statically separable then they can be updated separately. Its dynamic counterpart REL$_d$ says that when the inputs ψ_1 and ψ_2 are dynamically separable then the update can be performed in parallel (or rather, in an interleaving fashion).

It turns out that both postulates are violated by any AGM revision operation and and any KM update operation.

Proposition 4. *There is no operation $\circ$ satisfying both the basic belief change postulates and REL$_s$.*

Proof. Suppose $\circ$ satisfies the basic belief change postulates and REL$_s$. Consider the base $\beta = (p \vee q) \mathbin{\dot{\wedge}} (p \vee q)$ and the input $\psi = p \vee q$. We have seen above that β is equivalent to $p \wedge q$, and we therefore have:

$$\begin{aligned} \beta \circ \psi &= (p \vee q) \mathbin{\dot{\wedge}} (p \vee q) \circ p \vee q & \\ &= p \wedge q \circ p \vee q & \text{(by RE)} \\ &= \|p \wedge q\| & \text{(by PRES}_w\text{)} \end{aligned}$$

This is incompatible with what postulate REL$_s$ gives us:

$$\begin{aligned} \beta \circ \psi &= (p \vee q) \mathbin{\dot{\wedge}} (p \vee q) \circ p \vee q & \\ &= p \vee q \circ p \vee q \cap p \vee q \circ p \vee q & \text{(by REL}_s\text{)} \\ &= \|p \vee q\| \cap \|p \vee q\| & \text{(by PRES}_w\text{)} \\ &= \|p \vee q\| & \end{aligned}$$

□

Proposition 5. *There is no operation $\circ$ satisfying both the basic belief change postulates and REL$_d$.*

Proof. Suppose $\circ$ satisfies the KM postulates and REL$_d$. Consider the base $\beta = \neg p$ and the input $\psi = \neg p \mathbin{\dot{\|}} p$. We have seen above that ψ is equivalent to $\top$, and we therefore have:

$$\begin{aligned} \beta \circ \psi &= \neg p \circ \neg p \mathbin{\dot{\|}} p & \\ &= \neg p \circ \top & \text{(by RE)} \\ &= \|\neg p\| & \text{(by PRES}_w\text{)} \end{aligned}$$

This is incompatible with what postulate REL$_d$ gives us:[3]

$$\begin{aligned} \beta \circ \psi &= \neg p \circ \neg p \mathbin{\dot{\|}} p & \\ &= (\neg p \circ \neg p) \diamond p & \text{(by REL}_d\text{)} \\ &= \neg p \circ p & \text{(by PRES}_w\text{)} \\ &\subseteq \|p\| & \text{(by SUCCESS)} \end{aligned}$$

Incompatibility is the case because the set of valuations where $\neg p$ is true is non empty.

□

[2]Strictly speaking, REL$_d$ requires to build a formula representing the updates $\beta \circ \psi_1$ and $\beta \circ \psi_2$, as usually done in the KM framework.

[3]We recall that the second line of the proof is formulated sloppily: instead of the set of valuations $\neg p \circ \neg p$ there should be a formula representing that valuation.

5 Discussion and conclusion

We have introduced a simple version of separation logic working on sets (alias propositional valuations) that we have called set separation logic, SSL. Our logic has two separation operators: $\dot{\land}$ allows to separate resources, and $\dot{\|}$ allows to separate updates. We have shown that in our logic, both model checking and satisfiability checking can be done in polynomial space. We conjecture that the PSPACE upper bound that we have established coincides with the lower bound, but this remains to be proved. We would also like to provide an axiomatisation.

In the last part of the paper we have investigated the relation between SSL and belief change operations. We have formulated two postulates that appear to be natural and have shown that they are nevertheless incompatible with both AGM belief revision operations and KM belief update operations.

The problem of belief change respecting separation that we have studied in the last section is related to the frame problem in artificial intelligence [12]. Reiter's solution to that problem [15, 16] is by now widely accepted for actions without ramifications, i.e., without side effects. In joint work with Hans van Ditmarsch and Tiago de Lima [5] we have recently shown that Reiter's solution can be mapped to dynamic logics with propositional assignments DL-PA. Given that set separation logic can be embedded into DL-PA, it is immediate to extend it by propositional assignments.

It would be interesting to add the implicational connective $-\!\!*$ of separation logic to SSL (which should lead to undecidability given the results of [4, 11]). it is however not clear how the semantics of $-\!\!*$ can be defined in the framework of valuations.

Acknowledgements

The work in this paper was done in the framework of the ANR project DynRes (Dynamic Resources and Separation and Update Logics, project no. ANR-11-BS02-011). I would like to thank the members of the project for their comments of a (very preliminary) presentation of the ideas that are worked out in more detail here. Thanks are also due to the reviewers of WoLLIC 2013 whose comments I took into account as far as I could.

References

[1] Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. of Symbolic Logic*, 50:510–530, 1985.

[2] Philippe Balbiani, Andreas Herzig, and Nicolas Troquard. Dynamic logic of propositional assignments: a well-behaved variant of PDL. In Orna Kupferman, editor, *Logic in Computer Science (LICS), New Orleans, June 25-28, 2013*, http://www.ieee.org/, juin 2013. IEEE.

[3] Meghyn Bienvenu, Andreas Herzig, and Guilin Qi. Prime implicate-based belief revision operators. In Malik Ghallab, Constantine D. Spyropoulos, Nikos Fakotakis, and Nikos Avouris, editors, *European Conference on Artificial Intelligence (ECAI)*, pages 741–742, Patras, Greece, july 2008. IOS Press.

[4] James Brotherston and Max I. Kanovich. Undecidability of propositional separation logic and its neighbours. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, pages 130–139. IEEE Computer Society, 2010.

[5] Hans van Ditmarsch, Andreas Herzig, and Tiago de Lima. From Situation Calculus to Dynamic Logic. *Journal of Logic and Computation*, 21(2):179–204, 2011. http://logcom.oxfordjournals.org/content/21/2/179.abstract?etoc.

[6] Peter Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. MIT Press, 1988.

[7] Samin S. Ishtiaq and Peter W. O'Hearn. BI as an assertion language for mutable data structures. In Chris Hankin and Dave Schmidt, editors, *POPL*, pages 14–26. ACM, 2001.

[8] Hirofumi Katsuno and Alberto O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52:263–294, 1991.

[9] Hirofumi Katsuno and Alberto O. Mendelzon. On the difference between updating a knowledge base and revising it. In Peter Gärdenfors, editor, *Belief revision*, pages 183–203. Cambridge University Press, 1992. (preliminary version in Allen, J.A., Fikes, R., and Sandewall, E., eds., Principles of Knowledge Representation and Reasoning: Proc. 2nd Int. Conf., pages 387–394. Morgan Kaufmann Publishers, 1991).

[10] G. Kourousias and D. Makinson. Parallel interpolation, splitting, and relevance in belief change. *Journal of Symbolic Logic*, 72(3):994–1002, 2007.

[11] Dominique Larchey-Wendling and Didier Galmiche. The undecidability of boolean bi through phase semantics. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, pages 140–149. IEEE Computer Society, 2010.

[12] J. McCarthy and P.J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Mitchie, editors, *Machine Intelligence*, volume 4, pages 463–502. Edinburgh University Press, 1969.

[13] Peter W. O'Hearn, John C. Reynolds, and Hongseok Yang. Local reasoning about programs that alter data structures. In Laurent Fribourg, editor, *CSL*, volume 2142 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2001.

[14] R. Parikh. Beliefs, belief revision, and splitting languages. In Lawrence S. Moss, Jonathan Ginzburg, and Maarten de Rijke, editors, *Logic, Language, and Computation, vol. 2*, pages 266–278. CSLI Publications, 1999.

[15] Ray Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 359–380. Academic Press, San Diego, CA, 1991.

[16] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, 2001.

[17] John C. 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. IEEE Computer Society, 2002.