Modèles de Ressources Abstraits*

Dominique Larchey-Wendling

Septembre 2015

1 Introduction

Le but de ce rapport est de décrire la problématique des modèles de ressources abstraits liés à la logique de séparation (SL). La notion de ressource peut dans un premier temps être interprétée comme un tas mémoire, c'est à dire un ensemble fini d'adresses et de valeurs à ces adresses, valeurs qui peuvent être des pointeurs vers d'autres adresses : c'est le modèle concret de la logique de séparation.

Toutefois, comme le terme *abstrait* l'indique, on recherche d'autres modèles de la logique de séparation qui permettent de multiples interprétation de la notion de ressources. Il nous importe ici d'isoler et de formaliser les *propriétés importantes* que partagent ou pas ces modèles de la notion de ressource et d'étudier les variantes de SL qui en sont issues.

Ce rapport commence par une présentation de la logique de séparation. Puis il explique pourquoi le modèle concret des tas mémoire conduit à une logique dans laquelle il est très difficile d'automatiser la vérification et/ou la preuve. La conséquence de cette contrainte est l'idée d'approximer SL par le bas, par abstraction du modèle concret. Ceci nous conduit à décrire le noyau logique de SL dénommé BI Booléen (BBI) et les différents variantes de BBI que l'on obtient en restreignant ses modèles abstraits. Ces variantes sont aujourd'hui dénommées *logiques de séparation propositionnelles abstraites* (PASL) et nous présentons succinctement les derniers résultats obtenus par les spécialistes du domaine.

2 La logique de séparation

Au sens le plus général, le terme *logique de séparation* (notée ici SL) désigne un cadre formel fondé sur la logique de Hoare [21]. Il permet d'exprimer, de prouver et/ou vérifier des propriétés de programmes impératifs qui manipulent éventuellement de *pointeurs*, c'est à dire, qui modifie les propriétés de ce

*Rapport de recherche du projet ANR DynRes (projet No. ANR-11-BS02-011).

qu'on appelle la mémoire : l'endroit où sont stockées et modifiées les données que manipulent les programmes informatiques.

Nous décrivons plus précisément les constituants d'une logique de séparation : ils comportent un langage pour décrire les *programmes*, un langage logique pour décrire des *assertions* (i.e. des propriétés des programmes), un ensemble de règles de déduction entre les *triplets de Hoare* et une méthode pour établir les relations (p.ex. d'implication) entre les assertions. Cette dernière peut se présenter sous la forme de règles logiques pour la preuve mais aussi, sous la forme d'outils de vérification de modèle (*Model Checking*).

Un triplet de Hoare s'écrit $\{A\}\ \mathcal{P}\ \{B\}$ où $\mathcal{P}$ est un programme et A, B sont des assertions : A est appelée *pré-condition* et B est appelée *post-condition*. Intuitivement, les règles de déduction et les règles logiques doivent assurer la cohérence du calcul. Ainsi, si le triplet $\{A\}\ \mathcal{P}\ \{B\}$ est déductible, cela implique que quand le programme $\mathcal{P}$ s'exécute en partant d'un état vérifiant la pré-condition A alors, à la fin de son exécution le ou les états d'arrivé vérifie(nt) la post-condition B.

Parmi les nombreuses règles de déduction possibles, nous en présentons deux qui jouent chacune un rôle important; la *frame rule* et la règle de *conséquence*:

$$\frac{\{A\}\ \mathcal{P}\ \{B\}}{\{A * C\}\ \mathcal{P}\ \{B * C\}}\ \langle \text{frame} \rangle \qquad \frac{A' \to A \quad \{A\}\ \mathcal{P}\ \{B\} \quad B \to B'}{\{A'\}\ \mathcal{P}\ \{B'\}}\ \langle \text{conseq.} \rangle$$

La *frame rule* est spécifique à la logique de séparation. Elle remplace la règle de consistance

$$\frac{\{A\}\ \mathcal{P}\ \{B\}}{\{A \wedge C\}\ \mathcal{P}\ \{B \wedge C\}}\ \langle \text{consistancy} \rangle$$

qui n'est pas valide dans lorsque les programmes manipulent des pointeurs. La règle de *conséquence* contient deux assertions $A' \to A$ et $B \to B'$ et exprime le fait que si on renforce les pré-conditions et/ou on affaiblit les post-conditions, les triplets de Hoare restent valident. Cette règle nécessite d'établir la validité des implications $A' \to A$ et $B \to B'$. C'est le rôle qui est remplit par la logique des assertions et la méthode choisie pour établir des assertions.

Nous n'entrerons pas dans une description plus précise du cadre formel de Hoare qui correspond à SL. Cela nécessite en outre une sémantique opérationnelle pour les programmes, i.e. un modèle d'exécution. Ce qui nous intéresse dans ce rapport, c'est la logique des assertions (également appelée SL par abus de langage) et comment vérifier ou prouver les assertions qui apparaissent lorsqu'on utilise la règle de conséquence.

3 Les difficultés de la vérification dans SL

Décrivons tout d'abord une instance possible pour le langage des assertions de SL. Les formules de 1SL [1] sont définies par la grammaire formelle suivante :

$$\begin{aligned} \mathtt{e} &::= \mathtt{x} \mid \mathtt{u} \quad \text{avec } \mathtt{x} \in \mathtt{PVAR}, \mathtt{u} \in \mathtt{FVAR} \\ \pi &::= \mathtt{e} = \mathtt{e}' \mid \mathtt{e} \hookrightarrow \mathtt{e}' \\ A, B &::= \pi \mid \bot \mid \mathtt{emp} \mid A \wedge B \mid \neg A \mid A * B \mid A \mathrel{-\!\!*} B \mid \exists \mathtt{u}\, A \quad \text{avec } \mathtt{u} \in \mathtt{FVAR} \end{aligned}$$

$\mathtt{e}$ représente une expression, ici soit une variable de programme $\mathtt{x} \in \mathtt{PVAR}$ (qui ne peut pas être l'objet d'une quantification) ou alors une variable quantifiée $\mathtt{u} \in \mathtt{FVAR}$. π représente les assertions (ou formules) atomiques, dans notre cas, le test d'*égalité* entre deux expressions $\mathtt{e} = \mathtt{e}'$ ou le test de la relation *pointe vers* typique de la logique de séparation : $\mathtt{x}_1 \hookrightarrow \mathtt{x}_2$ signifie que la valeur de la variable $\mathtt{x}_1$ (c'est à dire celle que se trouve en mémoire à l'adresse $\mathtt{x}_1$) est un pointeur vers la variable $\mathtt{x}_2$ (c'est à dire l'adresse de $\mathtt{x}_2$). Ces formules atomiques peuvent être combinées en utilisant les connecteurs de la logique du premier ordre (la quantification étant toutefois restreinte aux variables quantifiées de FVAR) auxquels s'ajoutent les connecteurs spécifiques de SL que sont la constante emp, et les connecteurs binaires $*$ (ou *étoile*) et $\mathrel{-\!\!*}$ (ou *baguette magique*).

Décrivons maintenant l'interprétation sémantique des assertions de SL mais nous devons d'abord interpréter les expressions. Nous décrivons donc le *modèle standard* de SL qui est basé sur la notion de *tas*. L'*état mémoire* est une paire (s, h) où :

— s est une valuation $s : \mathtt{PVAR} \to \mathbb{N}$; c'est *l'adressage*;
— h est une fonction partielle $h : \mathbb{N} \rightharpoonup \mathbb{N}$ à domaine fini; c'est *le tas*.

Deux tas sont *disjoints* (noté $h_1 \perp h_2$) si leurs domaines de définition sont disjoints. Dans ce cas, et dans ce cas seulement, on peut former la composition des tas $h_1 \uplus h_2$ qui est l'unique tas dont le graphe est l'union des graphes de h_1 et de h_2. Nous insistons sur le fait que cette composition est partielle. Elle n'est pas définie si les tas h_1 et h_2 se superposent, et ce, même si les valeurs de h_1 et de h_2 sont identiques sur la partie commune de leur domaine de définition. Le tas vide est représenté par la notation $\square$. Ainsi, l'ensemble des tas muni des opérations $\uplus$ et $\square$ forment un *monoïde commutatif partiel*.

Nous pouvons maintenant décrire comment on interprète les assertions dans le modèle des états mémoire. Étant donné une valuation $\mathfrak{f} : \mathtt{FVAR} \to \mathbb{N}$, un état mémoire (s, h) et une assertion A, nous définissons la relation $(s, h) \models_{\mathfrak{f}} A$ par induction sur A, appelée sémantique de Kripke :

— $(s, h) \models_{\mathfrak{f}} \mathtt{e} = \mathtt{e}'$ ssi $[\![\mathtt{e}]\!] = [\![\mathtt{e}']\!]$ où $[\![\mathtt{x}]\!] = s(\mathtt{x})$ et $[\![\mathtt{u}]\!] = \mathfrak{f}(\mathtt{u})$;
— $(s, h) \models_{\mathfrak{f}} \mathtt{e} \hookrightarrow \mathtt{e}'$ ssi $h([\![\mathtt{e}]\!]) = [\![\mathtt{e}']\!]$;
— $(s, h) \models_{\mathfrak{f}} \mathtt{emp}$ ssi $h = \square$;

1. Le 1 de 1SL indique que les pointeurs pointent vers une seule valeur. Par exemple dans 2SL, le prédicate atomique $\mathtt{e} \hookrightarrow \mathtt{e}_1, \mathtt{e}_2$ est ajouté pour représenter les pointeurs vers les paires qui permettent de coder des arbres binaires.

— $(s,h) \models_{\mathfrak{f}} A * B$ ssi $(s,h_1) \models_{\mathfrak{f}} A$, $(s,h_2) \models_{\mathfrak{f}} B$ pour au moins une paire de tas (h_1,h_2) t.q. $h = h_1 \uplus h_2$;
— $(s,h) \models_{\mathfrak{f}} A \mathrel{-\!\!*} B$ ssi $(s,h') \models_{\mathfrak{f}} A$, $(s, h \uplus h') \models_{\mathfrak{f}} B$ pour tout tas h' t.q. $h \perp h'$;
— $(s,h) \models_{\mathfrak{f}} \exists \mathfrak{u}\, A$ ssi $(s,h) \models_{\mathfrak{f}[\mathfrak{u}\mapsto l]} A$ pour au moins une adresse $l \in \mathbb{N}$, où $\mathfrak{f}[\mathfrak{u} \mapsto l]$ dénote la valuation égale à $\mathfrak{f}$ partout sauf pour $\mathfrak{u}$ où elle vaut l.

Par soucis de simplicité, les définitions (usuelles) des connecteurs de la logique Booléenne propositionnelle sont omises. Lorsque la formule A ne contient pas de variable libre (dans FVAR), autrement dit, lorque A est close, alors la relation $(s,h) \models_{\mathfrak{f}} A$ ne dépend pas du choix de $\mathfrak{f}$ et on note plus simplement $(s,h) \models A$.

Détaillons l'interprétation de l'étoile $*$. Elle permet de représenter la séparation : $A * B$ est valide dans (s,h) si on peut séparer h en deux parties disjointes $h = h_1 \uplus h_2$ de sorte que A soit valide en (s,h_1) et B soit valide en (s,h_2). La baguette magique $\mathrel{-\!\!*}$ quand à elle est un peu moins intuitive. C'est l'opérateur adjoint de $*$: $A \mathrel{-\!\!*} B$ est valide dans (s,h) si toute extension $(s, h \uplus h')$ de (s,h) valide B à partir du moment où (s,h') valide A.

À partir des définitions précédentes, nous pouvons définir trois problèmes liés à la validité des assertions:

— étant donnés un modèle (s,h) et une assertion close A (sans variables libres), peut-on décider automatiquement de la relation $(s,h) \models A$? Ce problème est appelé *Model Checking* (noté MC);
— étant donnée une assertion close A, peut-on décider si il existe un état mémoire (s,h) qui satisfasse $(s,h) \models A$? Ce problème est appelé *Satisfiabilité* (noté SAT);
— l'assertion close A est-elle universellement valide, i.e. la relation $(s,h) \models A$ est-elle vérifiée pour tout état mémoire (s,h) ? Ce problème est celui de la *Validité*.

La présence de la baguette magique $\mathrel{-\!\!*}$ et de l'étoile $*$ a pour conséquence l'interchangeabilité de MC et de SAT dans le contexte de SL. En effet, on peut aisément codé l'un des problèmes avec l'autre ce qui conduit à des transferts de résultats de décidabilité et/ou complexité. La validité est quand à elle duale de la satisfiabilité dans le contexte d'une logique Booléenne comme SL. C'est ce problème que l'on cherche à résoudre de manière automatique dans le cadre de la règle de conséquence de la logique de Hoare de SL. Et la validité (c'est à dire SAT) dans SL sont des problèmes difficiles à résoudre.

Pour en témoigner, nous listons les résultats suivants, certains étant préalable à l'ANR DynRes alors que d'autres ont été montrés au cours de l'ANR par certains de ses partenaires. Tous ces résultats tendent à montrer que les problèmes MC et SAT de SL sont souvent *impraticables*. Il est à noter que les preuves de ces théorèmes sont non triviales et nécessitent des développements longs et complexes auxquels ce petit résumé ne rend absolument pas justice :

— même si on exclu l'usage de l'étoile $*$, les problèmes MC et SAT sont indécidables pour SL [2];

— si on exclu la quantification (hypothèse $\texttt{FVAR} = \varnothing$) alors MC et SAT sont des problèmes PSPACE complets [1];
— si on autorise la quantification sur une seule variable (hypothèse $\texttt{FVAR} = \{\texttt{u}\}$) alors MC et SAT sont des problèmes PSPACE complets [10];
— si on autorise la quantification sur au moins deux variables (hypothèse $\texttt{FVAR} = \{\texttt{u}_1, \texttt{u}_2, \ldots\}$) alors MC et SAT sont des problèmes non récursivement énumérables [9].

La conséquence de ces observations est qu'il faut peut-être envisager des approches alternatives si l'on souhaite automatiser la raisonnement dans la logique des assertions de SL. L'une de ces approches consistent à *approximer* la relation de validité par *abstraction* du modèle. Ainsi, au lieu de considérer le modèle concret des états mémoire, on peut isoler certaines de ses propriétés comme par exemple, le fait qu'il soit un monoïde partiel, ou encore, que ce monoïde soit régulier, que son unité n'est pas divisible, etc. On obtient alors toute une famille de logiques qui approximent inférieurement SL et pour lesquelles on peut raisonnablement espérer que le problème de la validité soit plus praticable. Mais pour cela, il nous faut un cadre formel, celui constitué par la logique BI Booléen.

4 Le noyau logique de SL, BI Booléen

La logique BI est une logique sous-structurelle comparable à la logique linéaire. Elle mixte des opérateurs additifs et des opérateurs multiplicatifs. Dans le cas de BI, et contrairement à la logique linéaire, la partie additive correspond aux logiques qui sont généralement utilisées pour le raisonnement : la logique intuitionniste ou la logique classique Booléenne. BI (intuitionniste) utilise l'interprétation intuitionniste des opérateurs additifs (la négation n'est donc pas involutive) alors que BI Booléen (noté BBI) utilise l'interprétation Booléenne des opérateurs additifs.

La partie multiplicative inclue les opérateurs *étoile* $*$ et *baguette magique* $-\!\!*$ ainsi que *unité multiplicative* $\mathbb{I}$, comme dans SL ou encore dans la logique linéaire. Leur interprétation correspond au fragment multiplicatif intuitionniste de la logique linéaire et constitue une généralisation de SL. Ainsi, BBI *est généralement considéré comme le noyau logique de* SL, c'est à dire SL sans les prédicats atomiques d'égalité $=$ ou « pointe vers » $\hookrightarrow$.

L'interprétation sémantique des formules de BBI se fait dans un modèle abstrait. La présentation la plus générale est fondée sur le notion de *monoïde non-déterministe* [12] (noté ND-monoïde).
Un ND-monoïde est un triplet $\mathfrak{M} = (M, \circ, U)$ où $U \subseteq M$ est un sous-ensemble d'*unités* et $\circ : M \times M \longrightarrow \mathcal{P}(M)$ est un opérateur de *composition* (non-déterministe) pour lesquelles les lois de

neutralité $\forall x \in M\ x \circ U = \{x\}$

commutativité $\forall x, y \in M\ x \circ y = y \circ x$

associativité $\forall x, y, z \in M\ (x \circ y) \circ z = x \circ (y \circ z)$

sont satisfaites. La neutralité et l'associativité doivent être interprétées en utilisant l'*extension de* $\circ$ à $\mathcal{P}(M)$ définie par $X \circ Y = \bigcup\{x \circ y \mid x \in X, y \in Y\}$. Étant donnés un ND-monoïde $\mathfrak{M} = (M, \circ, U)$ et un élément $x \in M$, nous pouvons interpréter les formules de BBI à la Kripke de la manière suivante :

$$\mathfrak{M}, x \Vdash_\delta v \text{ ssi } x \in \delta(v) \qquad \mathfrak{M}, x \Vdash_\delta \mathbb{I} \text{ ssi } x \in U \qquad \mathfrak{M}, x \Vdash_\delta \neg A \text{ ssi } \mathfrak{M}, x \nVdash_\delta A$$
$$\mathfrak{M}, x \Vdash_\delta A \wedge B \text{ ssi } \mathfrak{M}, x \Vdash_\delta A \text{ et } \mathfrak{M}, x \Vdash_\delta B$$
$$\mathfrak{M}, x \Vdash_\delta A * B \text{ ssi } \exists a, b,\ x \in a \circ b \text{ et } \mathfrak{M}, a \Vdash_\delta A \text{ et } \mathfrak{M}, b \Vdash_\delta B$$
$$\mathfrak{M}, x \Vdash_\delta A \mathrel{-\!\!*} B \text{ ssi } \forall a, b,\ (b \in x \circ a \text{ et } \mathfrak{M}, a \Vdash_\delta A) \Rightarrow \mathfrak{M}, b \Vdash_\delta B$$

Notons que lorsque l'on considère le modèle des tas mémoire comme un cas particulier de ND-monoïde,[2] on retrouve exactement la même interprétation que celle de la logique des assertions de SL, avec une correspondance évidente entre les opérateurs des deux logiques. On note $\mathsf{BBI}_{\mathrm{ND}}$ l'ensemble des formules universellement valides de BBI dans la classe générale des ND-monoïdes.

Pour cette interprétation sémantique de BBI, on peut donner plusieurs calculs de preuves corrects et complets :

— un système axiomatique *à la Hilbert* [12];
— un système basé sur le *Display Calculus* [3];
— un calcul des *séquents imbriqués* (nested sequents) [20];
— un calcul des *séquents avec labels* et relations [14].

Toutefois, le modèle des tas mémoire de SL est un cas particulier de ND-monoïde qui vérifie d'autres propriétés comme par exemple :

(PD)	Déterministe partiel	$\forall x, y, a, b\ \ \{x, y\} \subseteq a \circ b \Rightarrow x = y$
(SU)	Une seule unité	$\exists u\ \ U = \{u\}$
(CA)	Régularité	$\forall k, a, b\ \ (k \circ a) \cap (k \circ b) \neq \varnothing \Rightarrow a = b$
(IU)	Unité indivisible	$\forall x, y\ \ x \circ y \cap U \neq \varnothing \Rightarrow x \in U$
(DI)	Sans carré	$\forall x\ \ x \circ x \neq \varnothing \Rightarrow x \in U$

Les propriétés ci-dessus (et d'autres) ont été singularisées pour la première fois de manière systématique par Dockins, Hobor et Appel aussi nous les appelleront pas le suite *propriétés DHA*.

On note BBI_X l'ensemble des formules universellement valides dans la sous-classe $X \subseteq \mathrm{ND}$. Par exemple $\mathsf{BBI}_{\mathrm{PD+SU+CA}}$ est l'ensemble des formules de BBI qui sont toujours vraies quand on les interprète dans un monoïde partiel régulier. Lorsque $X \subsetneq \mathrm{ND}$ est une sous-classe stricte de ND, alors BBI_X n'est généralement pas le même ensemble que $\mathsf{BBI}_{\mathrm{ND}}$; seule la relation $\mathsf{BBI}_{\mathrm{ND}} \subseteq \mathsf{BBI}_X$ est triviale car plus on restreint les classes de modèles, plus il y a de formules universellement valides.

Une série de résultats importants d'indécidabilité viennent confirmer, à l'instar de SL, la difficulté de la preuve dans BBI. Pour toutes les sous-classes X de ND-monoïdes obtenue par conjonction des propriétés DHA l'ensemble BBI_X

2. c'est à dire avec les définitions $U = \{(s, \square) \mid s : \mathtt{PVAR} \longrightarrow \mathbb{N}\}$ et $(s, h) \in (s_1, h_1) \circ (s_2, h_2)$ ssi $s = s_1 = s_2$ et $h = h_1 \uplus h_2$.

n'est pas décidable [5, 4, 17, 18]. Toutefois, contrairement à SL (avec quantification), BBI_X est *semi-décidable*. On a déjà vu ce résultat précédemment dans le cas $\mathsf{BBI}_{\mathrm{ND}}$, on le retrouve pour des sous-classes de ND-monoïdes. Ainsi, il existe des systèmes de preuves corrects et complets (mais hélas non-terminants) pour BBI_X dans plusieurs cas significatifs :

$\mathsf{BBI}_{\mathrm{PD+SU}}$ pas de calcul à la Hilbert car PD n'est pas axiomatisable dans BBI [6]. Mais des calculs à base de tableaux avec labels [16, 15] ou des calculs à base de séquents avec labels [13];

BBI_X pour X égale à de nombreuses conjonctions de propriétés DHA, des *calculs hybrides* à la Hilbert [6] ou des calculs des séquents avec labels et relations ternaires [13].

L'étude systématique des propriétés DHA et des variantes de BBI associées à ces propriétés est identifié sous le terme *logique de séparation abstraite*. Sa description plus précise est l'objet de la section suivante.

5 Logique de séparation abstraite

Le terme *logique de séparation abstraite* a été proposé par [7] mais l'étude axiomatique des modèles abstraits de la logique de séparation a été initiée par [11]. Ces derniers ont identifiés certaines propriétés « importantes » du modèle concret des tas mémoire. Nous nommerons ces propriétés du symbole DHA. On y trouve celles citées à la section précédente : déterminisme partiel (PD), unique unité (SU), régularité (CA), unité indivisible (IU), sans carré (DI). On y trouve en outre deux autres propriétés :

Divisibilité $\forall w \notin U, \exists x, y \notin U, w \in x \circ y$;

Cross split $\forall a, b, x, y \in M, (a \circ b) \cap (x \circ y) \neq \varnothing \Rightarrow \exists r, s, t, u \in M, a \in r \circ t, b \in s \circ u, x \in r \circ s, y \in t \circ u$;

La divisibilité modélise le fait qu'une ressource non vide puisse toujours être coupée en deux parties strictes, autrement dit il n'y a pas de ressource atomique autre que les unités. Cette propriété n'est pas valide dans le modèle de tas mémoire mais elle peut-être requise par d'autres modèles de la logique de séparation [11].

La propriété "cross-split" modélise le fait que deux découpages en deux parties d'une même ressource proviennent nécessairement de deux fusions (différentes) d'un même découpage de la ressource en quatre parties. Cette propriété est vérifiée dans le modèle des tas mémoire mais ne l'est pas dans le modèle syntaxique des contraintes sur les mots que l'on trouve dans le calcul de tableaux pour $\mathsf{BBI}_{\mathrm{PD}}$ [16, 15].

Les variantes de BBI obtenues par ajout de propriétés DHA supplémentaires sont qualifiées de *logiques de séparation propositionnelles abstraites* (PASL) par [13]. Un série de résultats récents portent sur les PASL:

— la plupart des propriétés DHA ne peuvent pas être représentées par une formule de BBI, à l'exception notable de IU [6]. On dit qu'elles ne sont pas axiomatisables. Ainsi, il n'y a pas de système de Hilbert pour les variantes de BBI correspondantes. Par conséquent, établir la correction et la complétude de méthodes de preuve pour ces variantes ne peut se faire uniquement par traduction de preuves et doit passer par des méthodes sémantiques comme celle proposée par [15] et étendue par [8] ou par [13];
— certaines propriétés comme SU, CA ou DI ne modifient pas l'ensemble des formules valident ce qui est un résultat plus fort que l'impossibilité de les axiomatiser [19]. Pour illustrer ces résultats, les équivalences suivantes on été montrées (en gris sur la figure)

Ainsi, la notion de validité est la même pour les classes PD et SU + CA (ainsi que toutes les classes intermédiaires). Ces résultats découlent d'une étude fine des propriétés des modèles syntaxiques engendrés par les calculs à base de tableaux sémantiques;
— on peut obtenir des méthodes de preuves (semi-décision) de manière *paramétrique* pour certaines PASL issues des propriétés DHA. Par exemple, [6] propose une technique basée sur les logiques modales hybrides, c'est à dire dont le langage permet de parler explicitement des éléments du modèle. Par ailleurs, l'approche de [13] permet de coder certaines propriétés DHA par des règles structurelles supplémentaires dans le calcul des séquents avec labels et contraintes relationnelles.

Du point de vue de la recherche de preuve dans PASL, on voit donc que même si elle reste un problème indécidable, des systèmes de preuves corrects et complets existent. Toutefois, la logique hybride n'offre pas pour l'instant un cadre qui permette l'automatisation de la recherche de preuve. Du côté des calculs de séquents avec labels, les développements sont plus avancés avec des prototypes relativement efficaces [13] lorsqu'une preuve est attendue. Par contre, lorsqu'une formule n'est pas prouvable, ces prouveurs expérimentaux mettent du temps à s'en rendre compte. Il faudrait sans doute apporter des améliorations sur l'aspect génération de contre-modèles.

Références

[1] Rémi Brochenin, Stéphane Demri, and Étienne Lozes. Reasoning about sequences of memory states. *Annals of Pure and Applied Logic*, 161(3):305–323, 2009.

[2] Rémi Brochenin, Stéphane Demri, and Étienne Lozes. On the almighty wand. *Information and Computation*, 211:106–137, 2012.

[3] James Brotherston. Bunched Logics Displayed. *Studia Logica*, 100(6):1223–1254, 2012.

[4] James Brotherston and Max Kanovich. Undecidability of Propositional Separation Logic and Its Neighbours. In *LICS*, pages 130–139. IEEE Computer Society, 2010.

[5] James Brotherston and Max Kanovich. Undecidability of Propositional Separation Logic and its Neighbours. *Journal of the ACM*, 61(2), 2014.

[6] James Brotherston and Jules Villard. Parametric Completeness for Separation Theories. In *POPL*, pages 453–464. ACM, 2014.

[7] Cristiano Calcagno, Peter W. O'Hearn, and Hongseok Yang. Local Action and Abstract Separation Logic. In *LICS*, pages 366–378. IEEE Computer Society, 2007.

[8] Jean-René Courtault and Didier Galmiche. A modal BI logic for dynamic resource properties. In Sergei N. Artëmov and Anil Nerode, editors, *Logical Foundations of Computer Science, International Symposium, LFCS 2013, San Diego, CA, USA, January 6-8, 2013. Proceedings*, volume 7734 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 2013.

[9] Stéphane Demri and Morgan Deters. Expressive Completeness of Separation Logic with Two Variables and No Separating Conjunction. In *CSL-LICS'14*. ACM Press, 2014.

[10] Stéphane Demri, Didier Galmiche, Dominique Larchey-Wendling, and Daniel Méry. Separation Logic with One Quantified Variable. In *CSR'14*, volume 8476 of *Lecture Notes in Computer Science*, pages 125–138. Springer, 2014.

[11] Robert Dockins, Aquinas Hobor, and Andrew W. Appel. A Fresh Look at Separation Algebras and Share Accounting. In *APLAS*, volume 5904 of *Lecture Notes in Computer Science*, pages 161–177. Springer, 2009.

[12] Didier Galmiche and Dominique Larchey-Wendling. Expressivity properties of Boolean BI through Relational Models. In *FSTTCS*, volume 4337 of *Lecture Notes in Computer Science*, pages 358–369. Springer, 2006.

[13] Zhé Hóu, Ranald Clouston, Rajeev Goré, and Alwen Tiu. Proof search for propositional abstract separation logics via labelled sequents. In *POPL*, pages 465–476. ACM, 2014.

[14] Zhé Hóu, Alwen Tiu, and Rajeev Goré. A Labelled Sequent Calculus for BBI: Proof Theory and Proof Search. In *TABLEAUX*, volume 8123 of *Lecture Notes in Computer Science*, pages 172–187. Springer, 2013.

[15] Dominique Larchey-Wendling. The formal strong completeness of partial monoidal Boolean BI. *Journal of Logic and Computation*, 2014. doi:10.1093/logcom/exu031.

[16] Dominique Larchey-Wendling and Didier Galmiche. Exploring the relation between Intuitionistic BI and Boolean BI: an unexpected embedding. *Mathematical Structures in Computer Science*, 19(3):435–500, 2009.

[17] Dominique Larchey-Wendling and Didier Galmiche. The Undecidability of Boolean BI through Phase Semantics. In *LICS*, pages 140–149. IEEE Computer Society, 2010.

[18] Dominique Larchey-Wendling and Didier Galmiche. Nondeterministic Phase Semantics and the Undecidability of Boolean BI. *ACM Transactions on Computational Logic*, 14(1):6, 2013.

[19] Dominique Larchey-Wendling and Didier Galmiche. Looking at separation algebras with boolean bi-eyes. In Josep Diaz, Ivan Lanese, and Davide Sangiorgi, editors, *Theoretical Computer Science - 8th IFIP TC 1/WG 2.2 International Conference, TCS 2014, Rome, Italy, September 1-3, 2014. Proceedings*, volume 8705 of *Lecture Notes in Computer Science*, pages 326–340. Springer, 2014.

[20] Jonghyun Park, Jeongbong Seo, and Sungwoo Park. A theorem prover for Boolean BI. In *POPL*, pages 219–232. ACM, 2013.

[21] John Reynolds. Separation Logic: A Logic for Shared Mutable Data Structures. In *LICS'02*, pages 55–74. IEEE, 2002.