Algorithmes d'élimination de quantificateurs pour le calcul des politiques des formules booléennes quantifées

Igor Stéphan

Algorithmes d'élimination de quantificateurs pour le calcul des politiques des formules booléennes quantifiées

Igor Stéphan

LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers Cedex 01
igor.stephan@info.univ-angers.fr

Résumé

Le problème de validité d'une formule booléenne quantifiée est une généralisation du problème de satisfiabilité d'une formule booléenne. Les formules booléennes quantifiées sont utiles pour représenter par exemple des stratégies dans un jeu à deux joueurs mais dans de telles applications c'est une solution au problème de recherche associé qui est nécessaire. La plupart des procédures de décision récentes pour les formules booléennes quantifiées sont des extensions de la procédure de recherche dite de Davis-Putnam et peuvent être aisément étendues au problème de recherche. Ce n'est pas le cas pour les algorithmes basés sur l'élimination de quantificateurs. Dans cet article nous montrons comment des algorithmes d'élimination de quantificateurs peuvent être étendus pour le problème de recherche associé aux formules booléennes quantifiées.

Abstract

The quantified Boolean formulae validity problem is a generalization of the Boolean formulae satisfiability problem. Quantified Boolean formulae are useful to represent strategies in two-player games but in this case solutions (or models) to the quantified Boolean formula are needed. These models are sequences of Skolem functions. Most of the recent decision procedures for quantified Boolean formulae are extensions of the search-based Davis-Putnam procedure and easily extended to computes the models. It is not the case for quantifier-elimination algorithms. In this article, we present how a quantifier-elimination algorithm may be extended to compute the models of quantified Boolean formulae.

1 Introduction

Le problème de validité pour les formules booléennes quantifiées est une généralisation du problème de satisfiabilité pour les formules booléennes. Tandis que décider de la satisfiabilité des formules booléennes est NP-complet, décider de la validité des formules booléennes quantifiées est PSPACE-complet. C'est le prix à payer pour une représentation plus concise pour de très nombreuses classes de formules. Une multitude d'importants problèmes de décision parmi des champs très divers ont des transformations polynomiales vers le problème de validité des formules booléennes quantifiées. C'est pourquoi, l'étude et la mise au point d'outils efficaces pour décider de la validité des formules booléennes quantifiées sont importantes. Les formules booléennes quantifiées sont aussi utiles pour représenter des stratégies dans un jeu à deux joueurs. Dans un tel jeu les adversaires $J_\forall$ et $J_\exists$ jouent alternativement en valuant les variables d'une formule qui doit être valide si le joueur $J_\exists$ est sûr de gagner. Dans ce style d'application, une procédure de décision n'est pas suffisante et une solution au problème de recherche associé à la formule booléenne quantifiée est nécessaire. Une politique [5] est une représentation pour une telle solution : elle explicite à chaque étape du jeu les choix qui doivent être fait par le joueur $J_\exists$ pour qu'il gagne quels que soient les coups que son adversaire $J_\forall$ a joués, joue ou jouera.

Étant donné qu'une formule booléenne quantifiée peut être ramenée à une formule booléenne (non quantifiée), la première solution semble être d'opérer cette mise sous forme de formule booléenne puis d'appliquer un algorithme de test de satisfiabilité. L'inconvénient majeur d'une telle approche est la taille de la formule booléenne qui est dans le pire des cas exponentielle par rapport à la celle de la formule booléenne quantifiée. Le plupart des travaux récents portant sur les procédures de décision pour la validité des formules booléennes quantifiées [16, 15, 9, 10, 3] sont des extensions

de la procédure de recherche dite de Davis-Putnam [6] pour le test de satisfiabilité des formules booléennes. Ceci n'est pas surprenant étant donné que le problème de décision de la validité pour les formules booléennes quantifiées est une généralisation du problème de décision de la satisfiabilité pour les formules booléennes quantifiées. Pour de tels algorithmes, le calcul de la politique nous apparaît comme évident. Il n'en est pas de même pour les algorithmes basés sur l'élimination de quantificateurs [14, 13] qui sont moins étudiés.

Dans cet article, trois algorithmes d'élimination de quantificateurs pour les formules booléennes quantifiées sont proposés : une procédure de décision générale et deux algorithmes pour calculer l'ensemble des politiques d'une formule booléenne quantifiée ; de ces deux algorithmes l'un est dédié aux formules simplement prénexes et l'autre aux formules en forme normale conjonctive. Nous concluons sur les liens entre nos travaux et les procédures basées sur le principe de résolution ou des méthodes d'élimination de quantificateurs et dressons une succincte description de nos travaux futurs.

2 Préliminaires

Notations. La sémantique des opérateurs booléens qui suivent est définie de manière habituelle sur l'ensemble des valeurs de vérité $\{\mathbf{v}, \mathbf{f}\}$. Une fonction des variables booléennes dans l'ensemble des valeurs de vérité est une interprétation. Les symboles $\bot$ (ce qui est toujours $\mathbf{f}$) et $\top$ (ce qui est toujours $\mathbf{v}$) sont les constantes booléennes. Le symbole $\wedge$ est utilisé pour la conjonction, $\vee$ pour le disjonction, $\neg$ pour la négation, $\rightarrow$pour l'implication et $\leftrightarrow$ pour l'équivalence. Une formule booléenne est construite à partir des constantes booléennes, des variables booléennes et des opérateurs booléens. L'ensemble **PROP** est l'ensemble des formules booléennes et $\mathbf{PROP}_V$ est l'ensemble des formules booléennes comprenant uniquement des occurrences de variables issues de l'ensemble de variables V. Une interprétation v est un modèle pour une formule F si elle rend vrai la formule F (ce qui est noté $v \models F$). Le symbole $\exists$ est utilisé pour le quantificateur existentiel et $\forall$ pour le quantificateur universel (q est utilisée comme une variable de quantification). Toute formule booléenne est aussi une formule booléenne quantifiée (ou QBF). Si F est une formule booléenne quantifiée et x une variable booléenne alors $(\exists x\ F)$ et $(\forall x\ F)$ sont des formules booléennes quantifiées. Nous supposons par la suite que dans une formule booléenne quantifiée les quantificateurs portent sur des variables distinctes. Une substitution $[y \leftarrow F]$ est une fonction de l'ensemble des formules booléennes quantifiées vers lui-même et définie, pour toutes variables booléennes x, y et toutes formules booléennes quantifiées F, A, B, comme suit :

- $\top[y \leftarrow F] = \top$;
- $\bot[y \leftarrow F] = \bot$;
- $x[y \leftarrow F] = x$ si $x \neq y$ et F sinon ;
- $\neg A[y \leftarrow F] = \neg A[y \leftarrow F]$;
- $(A \circ B)[y \leftarrow F] = (A[y \leftarrow F] \circ B[y \leftarrow F])$, pour tout $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$;
- $(qx\ \ A)[y \leftarrow F] = (qx\ \ A[y \leftarrow F])$ si $y \neq x$ et $(qx\ A)$ sinon, pour tout $q \in \{\forall, \exists\}$.

Un lieur Q est une séquence $q_1 x_1 \dots q_n x_n$ avec $x_1, \dots, x_n$ des variables distinctes et $q_1 \dots q_n \in \{\exists, \forall\}$ (n est la taille du lieur). Une occurrence d'une variable y dans une formule F est libre si cette occurrence n'est pas dans la portée d'un quantificateur $\exists y$ ou $\forall y$. Les occurrences de variables qui ne sont pas libres sont liées. Une formule sans variable libre est close. Le symbole ϵ représente la substitution vide. Par définition, pour toute variable booléenne y et toute formule booléenne quantifiée F,

- $(\exists y\ F) = (F[y \leftarrow \top] \vee F[y \leftarrow \bot])$ et,
- $(\forall y\ F) = (F[y \leftarrow \top] \wedge F[y \leftarrow \bot])$.

Un littéral est une variable logique ou sa négation. Une clause est une disjonction de littéraux. Une formule booléenne est en forme normale conjonctive (CNF) si c'est une conjonction de clauses. Une formule booléenne quantifiée est sous forme prénexe si elle est composée d'un lieur et d'une formule booléenne dénommée matrice. Une formule booléenne quantifiée est sous forme normale conjonctive si c'est une formule prénexe dont la matrice est en forme normale conjonctive. La relation d'équivalence $\equiv$ est définie de manière standard : deux QBF A et B sont logiquement équivalentes si $A \leftrightarrow B \equiv \top$. Une formule booléenne quantifiée F est valide si $F \equiv \top$. Par exemple $\forall x \exists y (x \leftrightarrow y)$ est valide et $\exists x \forall y (x \leftrightarrow y)$ ne l'est pas. La séquence vide est représentée par ε.

Nous avons choisi une définition des formules booléennes quantifiées plus large que de coutume : elles sont souvent restreintes à des formules prénexes, voire en forme normale conjonctive (en particulier les jeux de tests). Nous pensons que cette dernière restriction fait perdre dans le cas de problèmes structurés des informations précieuses pour la mise en évidence d'une solution.

Le principe de résolution. Le principe de résolution a été introduit dans le célèbre article de J.A. Robinson [17]. L'idée de la méthode est de prouver la validité d'une formule du premier ordre en établissant que sa négation est insatisfaisable. Elle est basée sur l'étape de résolution qui peut être exprimée, au propositionnel, ainsi : soient $A = (l \vee A')$ et $B = (\neg l \vee B')$ deux formules booléennes et $\mathcal{C}$ une conjonction de clauses

alors $(A \wedge B \wedge \mathcal{C}) \equiv ((A' \vee B') \wedge A \wedge B \wedge \mathcal{C})$.

Politique. La représentation d'une solution au problème de recherche associé aux formules booléennes quantifiées est étudiée dans [5] sous le nom de *politique*. La version présentée dans notre article est légèrement différente de l'originale.

Définition 1 (politique [5]) *L'ensemble des politiques (totales) $\mathcal{P}(Q)$ pour un lieur est défini inductivement par :*

$$\begin{aligned}
&\mathcal{P}(\textit{Entrée} : \textit{ un lieur }) \\
&\qquad \textit{Sortie} : \textit{ l'ensemble des politiques} \\
&\mathcal{P}(\varepsilon) = \{\lambda\} \\
&\mathcal{P}(\exists y Q) = \{l; \pi \mid l \in \{y, \neg y\}, \pi \in \mathcal{P}(Q)\} \\
&\mathcal{P}(\forall y Q) = \{y \mapsto \pi, \neg y \mapsto \pi' \mid \pi, \pi' \in \mathcal{P}(Q)\}
\end{aligned}$$

L'opérateur ";" représente la composition séquentielle des politiques. Le symbole λ représente la politique vide. Si π est une politique alors $\pi; \lambda$ est simplifiée en π.

Une politique est une représentation élégante pour des séquences de fonctions de Skolem.

Il est notable de remarquer que les politiques ne sont définies que pour les formules booléennes quantifiées prénexes.

La satisfiabilité d'une formule booléenne quantifiée prénexe pour une politique est définie récursivement.

Définition 2 (satisfiabilité d'une formule booléenne quantifiée prénexe pour une politique [5]) *Une politique π satisfait une formule booléenne quantifiée prénexe QF (noté $\pi \models QF$) si une de ses conditions est satisfaite :*

- $Q = \varepsilon$ *et* $\pi = \lambda$ *et* $F \equiv \top$,
- $Q = \exists x Q'$ *et* $\pi = (x; \pi')$ *avec* $\pi' \models (Q'F)[x \leftarrow \top]$,
- $Q = \exists x Q'$ *et* $\pi = (\neg x; \pi')$ *avec* $\pi' \models (Q'F)[x \leftarrow \bot]$,
- $Q = \forall x Q'$ *et* $\pi = \{x \mapsto \pi', \neg x \mapsto \pi''\}$ *avec* $\pi' \models (Q'F)[x \leftarrow \top]$ *et* $\pi'' \models (Q'F)[x \leftarrow \bot]$.

Une politique est dite *valide* pour une QBF si elle satisfait celle-ci. Dans [5], il est établi qu'une QBF prénexe et close QF est valide si et seulement si il existe une politique π telle que $\pi \models QF$.

Le même symbole a été utilisé pour noter qu'une interprétation satisfait une formule booléenne et qu'une politique satisfait une QBF. Cette ambiguïté, qui n'en sera pas une puisque le sens sera toujours clair par le contexte, n'en est pas une sur le fond car une interprétation qui satisfait une formule booléenne est aussi une politique valide pour la clôture existentielle de cette même formule booléenne et réciproquement, une politique valide pour une QBF prénexe dont le lieur est uniquement constitué de quantificateurs existentiels est aussi un modèle pour la matrice de cette QBF.

3 Un exemple

Nous présentons un exemple qui recouvre l'ensemble des résultats de cet article.

Soit la formule booléenne quantifiée

$$F = \forall a \exists b \forall c((c \vee b) \wedge \exists d((b \rightarrow ((c \rightarrow d) \wedge (c \vee (a \leftrightarrow \neg d))))$$

Cette QBF est-elle valide ? Nous obtenons immédiatement

$$\begin{aligned}
F \equiv \forall a \exists b \forall c((c \vee b) \wedge \exists d(\; & (d \vee (b \rightarrow \neg(a \rightarrow c))) \wedge \\
& (\neg d \vee (b \rightarrow (a \rightarrow c))))).
\end{aligned}$$

Grâce à la définition de la sémantique du quantificateur existentiel

$$\begin{aligned}
F \equiv \quad & \forall a \exists b \forall c((c \vee b) \wedge ((b \rightarrow \neg(a \rightarrow c)) \vee (b \rightarrow (a \rightarrow c)))) \\
\equiv \quad & \forall a \exists b \forall c(c \vee b).
\end{aligned}$$

Enfin, grâce à la définition de la sémantique du quantificateur universel

$$\begin{aligned}
F \equiv \quad & \forall a \exists b \forall c((c \vee b) \wedge (\neg c \vee \top)) \\
\equiv \quad & \forall a \exists b (b \wedge \top) \equiv \forall a \exists b (b) \equiv \forall a (\top) \equiv \top.
\end{aligned}$$

La formule F est donc valide. Ainsi la définition de la sémantique des quantificateurs appliquée à une forme particulière de QBF donne directement un algorithme de décision. La section 4.1 formalise cette forme particulière ; la section 4.2 définit l'algorithme de décision.

Notre objectif étant de définir des algorithmes qui calculent les politiques valides pour des QBF prénexes, il est important de manipuler les QBF de telle manière à ce qu'elles conservent les mêmes politiques. La seule politique valide pour la formule F est la politique

$$\{a \mapsto b; \{c \mapsto d, \neg c \mapsto \neg d\}, \neg a \mapsto b; \{c \mapsto d, \neg c \mapsto d\}\}.$$

Elle correspond à la séquence $\overline{b}_a; \overline{d}_{a,c}$ de fonctions de Skolem définies ainsi : $\overline{b}_a(\mathbf{v}) = \overline{b}_a(\mathbf{f}) = \mathbf{v}$, $\overline{d}_{a,c}(\mathbf{v}, \mathbf{f}) = \mathbf{f}$ et $\overline{d}_{a,c}(\mathbf{v}, \mathbf{v}) = \overline{d}_{a,c}(\mathbf{f}, \mathbf{v}) = \overline{d}_{a,c}(\mathbf{f}, \mathbf{f}) = \mathbf{v}$. Mais cette politique n'est pas valide pour la QBF $\top$ qui est équivalente à F. La notion d'équivalence qui préserve la validité des QBF n'est donc plus adaptée pour la préservation des politiques. La section 4.3 introduit une nouvelle relation d'équivalence entre les QBF qui préserve les politiques.

Chaque formule booléenne portant sur un ensemble de variables $\{x_1, \ldots, x_n\}$ est équivalent à une formule de la forme

$$\bigwedge_{1 \leq i \leq n} ((x_i \vee \phi_i^+) \wedge (\neg x_i \vee \phi_i^-))$$

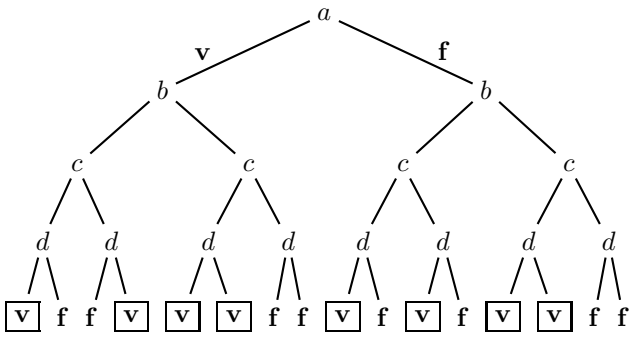

FIG. 1 – 8 modèles pour la formule Booléenne G

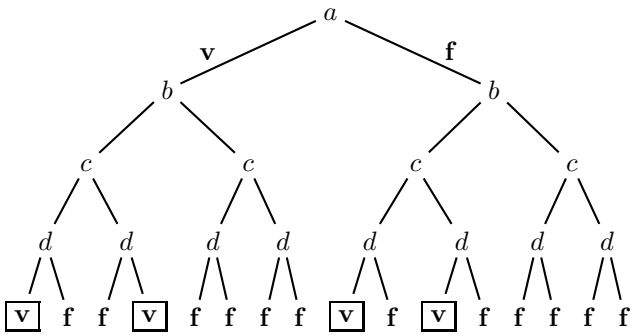

FIG. 2 – 4 modèles pour la matrice de F'

(avec ϕ_i^+ et ϕ_i^- des formules portant sur des variables $x_1, \ldots, x_{i-1}$) de laquelle il est facile de calculer tous les modèles. Par exemple, la formule Booléenne :

$$G = ((c \vee b) \wedge (b \rightarrow ((c \rightarrow d) \wedge (c \vee (a \leftrightarrow \neg d)))))$$

est équivalente à la formule

$$\begin{aligned} G' = \; & (a \vee \top) \wedge (\neg a \vee \top) \wedge \\ & (b \vee \top) \wedge (\neg b \vee \top) \wedge \\ & (c \vee b) \wedge (\neg c \vee \top) \wedge \\ & (d \vee (b \rightarrow \neg(a \rightarrow c))) \wedge (\neg d \vee (b \rightarrow (a \rightarrow c)))). \end{aligned}$$

De cette formule G', il est très facile d'en extraire tous les modèles. La figure 1 montre toutes les interprétations de la formule G ainsi que ses 8 modèles.

Existe-t-il une forme similaire pour les QBF ? La réponse est oui et cette nouvelle forme normale est définie dans la section 4.4.

Comme la formule Booléenne G est équivalente à G', la QBF F est aussi équivalente à une QBF F' similaire à G' par sa forme :

$$\begin{aligned} F' = \forall a \exists b \forall c \exists d ((b \vee \bot) \wedge (\neg b \vee \top) \wedge \\ (d \vee (b \rightarrow \neg(a \rightarrow c))) \wedge (\neg d \vee (b \rightarrow (a \rightarrow c)))). \end{aligned}$$

Dans notre exemple, il est facile de vérifier que la QBF F' préserve l'unique politique valide de F. La figure 2 montre l'ensemble des interprétations de la matrice (non-quantifiée) de F'. Nous pouvons remarquer que les branches correspondant à des modèles pour la matrice de la QBF initiale qui ne font pas parties de la politique valide ne sont pas préservées.

Est-ce difficile de calculer cette nouvelle forme normale avec un algorithme basé sur l'élimination de quantificateurs ? La réponse est non. Elle est calculée par une spécialisation et une restriction de la procédure de décision définie dans la section 4.2. La section 4.5 décrit comment nous pouvons calculer à partir de cette forme normale l'ensemble des politiques valides d'une QBF prénexe. Enfin, dans la section 4.6 nous montrons comment cette forme normale ainsi que l'ensemble des politiques valides pour une QBF en forme normale conjonctive peut être efficacement calculée à partir de la procédure de décision QMRES de Pan et Vardi [13].

4 Algorithmes d'élimination de quantificateurs pour les formules booléennes quantifiées

4.1 Formes partielles de résolution

Nos algorithmes d'élimination de quantificateurs sont tous définis sur des deux nouvelles formes pour formules booléennes quantifiées : la forme partielle de résolution et sa restriction, la forme partielle conjonctive de résolution.

Définition 3 (les formes partielles de résolution) *Soit q un quantificateur. Une formule booléenne quantifiée $F[x \leftarrow qy((y \vee F_y^+) \wedge (\neg y \vee F^- y))]$ est en forme partielle de résolution (ou FPR) si F_y^+ et F_y^- sont des formules booléennes quantifiées où y n'apparaît pas et x est libre dans F. Une forme partielle de résolution est une forme partielle conjonctive de résolution (ou FPCR) si $F = Q(F' \wedge qy((y \vee F_y^+) \wedge (\neg y \vee F_y^-)))$.*

Exemple 1 *Soit une formule booléenne quantifiée*

$$H = \exists a \forall b \neg((\exists c\ (((a \wedge b) \rightarrow c) \wedge ((a \wedge c) \rightarrow b))) \rightarrow (b \wedge \neg a)).$$

Alors, avec
- $H' = \exists a \forall b \neg(u \rightarrow (b \wedge \neg a))$ *et*
- $H'' = \exists a \forall b (u \wedge \neg(b \wedge \neg a))$,
- $H''' = (\exists c\ ((c \vee \neg(a \wedge b)) \wedge (\neg c \vee (a \rightarrow b))))$,

$H'[u \leftarrow H''']$ est une des formes partielles de résolution de H et $H''[u \leftarrow H''']$ est une des formes partielles conjonctives de résolution de H.

Nous prouvons d'abord que pour chaque formule booléenne quantifiée il existe une formule équivalente sous forme partielle (conjonctive) de résolution. Ce résultat est d'abord mis en évidence pour la forme partielle conjonctive de résolution.

Lemme 1 *Pour chaque formule booléenne quantifiée en forme normale conjonctive il existe une formule en forme partielle de résolution qui lui est équivalente.*

Puisque chaque formule booléenne quantifiée en forme partielle conjonctive de résolution est aussi en forme partielle de résolution, le lemme suivant est une conséquence directe du lemme précédent.

Lemme 2 *Pour chaque formule booléenne quantifiée il existe une formule sous forme partielle de résolution qui lui est équivalente.*

Le lemme technique suivant est le cœur de nos algorithmes d'élimination de quantificateurs.

Lemme 3 *Soit* $F = F'[x \leftarrow \exists y((y \vee F_y^+) \wedge (\neg y \vee F_y^-))]$ *une formule booléenne quantifiée en forme partielle de résolution alors* $F \equiv F'[x \leftarrow (F_y^+ \vee F_y^-)]$.

Soit $F = F'[x \leftarrow \forall y((y \vee F_y^+) \wedge (\neg y \vee F_y^-))]$ *une formule booléenne quantifiée en forme partielle de résolution alors* $F \equiv F'[x \leftarrow (F_y^+ \wedge F_y^-)]$.

Ce lemme est inspiré par l'étape de résolution. Appliqué à une formule booléenne quantifiée en forme partielle (conjonctive) de résolution, c'est en fait un processus d'élimination des quantificateurs.

Certains cas particuliers du lemme 3 sont des généralisations de techniques classiques : soient F une QBF et $F'[x \leftarrow (qy\ ((y \vee F_y^+) \wedge (\neg y \vee F_y^-)))]$ une de ses formes partielles de résolution.

- Si $q = \exists$ et $F_y^+ = \top$ alors
  $F \equiv F'[x \leftarrow (\exists y\ ((y \vee \top) \wedge (\neg y \vee F_y^-)))]$
  $\equiv F'[x \leftarrow (\top \vee F_y^-)] \equiv F'[x \leftarrow \top]$.
- Si $q = \forall$ et $F_y^+ = \top$ alors
  $F \equiv F'[x \leftarrow (\forall y\ ((y \vee \top) \wedge (\neg y \vee F_y^-)))]$
  $\equiv F'[x \leftarrow (\top \wedge F_y^-)] \equiv F'[x \leftarrow F_y^-]$.
- Si $q = \exists$ et $F_y^+ = \bot$ alors
  $F \equiv F'[x \leftarrow (\exists y\ ((y \vee \bot) \wedge (\neg y \vee F_y^-)))]$
  $\equiv F'[x \leftarrow (\bot \vee F_y^-)] \equiv F'[x \leftarrow F_y^-]$.
- Si $q = \forall$ et $F_y = \bot$ alors
  $F \equiv F'[x \leftarrow (\forall y\ ((y \vee \bot) \wedge (\neg y \vee F_y^-)))]$
  $\equiv F'[x \leftarrow (\bot \wedge F_y^-)] \equiv F'[x \leftarrow \bot]$.
  De plus, si F est en forme partielle conjonctive de résolution $F \equiv \bot$.

Les deux premiers cas sont des généralisations de la simplification des littéraux monotones [3] ; les deux derniers sont des généralisations de la propagation [3]. Nous explicitons un peu plus la propagation unitaire : soit $F'' = (\neg y \vee F_y^-)$ donc $F''[y \leftarrow \top] \equiv F_y^-$ donc $F = F'[x \leftarrow (\exists y\ ((y \vee \bot) \wedge (\neg y \vee F_y^-)))] \equiv F'[x \leftarrow F''[y \leftarrow \top]]$.

4.2 Une procédure de décision pour le problème de validité des formules booléennes quantifiées

L'application répétée du processus d'élimination des quantificateurs du lemme 3 conduit à une procédure de décision pour le problème de validité des formules booléennes quantifiées.

Algorithme 1 (La procédure de décision *rqbf*)

rqbf(*Entrée : une QBF close* F)
Sortie : oui si F *est valide sinon non*
rqbf(f) = (f *est une formule booléenne*) *et* ($f \equiv \top$)
rqbf(f) =
soit $f'[x \leftarrow (qy\ ((y \vee f_y^+) \wedge (\neg y \vee f_y^-)))]$ *une forme partielle de résolution de* f
dans si ($q = \exists$)
alors rqbf($f'[x \leftarrow (f_y^+ \vee f_y^-)]$)
sinon rqbf($f'[x \leftarrow (f_y^+ \wedge f_y^-)]$)

Cette procédure n'a pas pour but d'être efficace mais elle fournit un cadre dans lequel le calcul des politiques valides d'une formule booléenne quantifiée pour être exprimé. Comment la forme partielle de résolution doit être calculée est la seule étape de la procédure de décision qui ne soit pas effective. Le lemme 1 suggère une méthode : calculer la forme normale conjonctive et alors appliquer l'associativité et la distributivité de la conjonction et de la disjonction.

Des lemmes 2 et 3, la procédure de décision *rqbf* est correcte et complète pour les formules booléennes quantifiées closes.

Théorème 1 *Soit* F *une formule booléenne quantifiée close. Alors* *rqbf*(F) = *oui si et seulement si* F *est valide.*

Si la procédure *rqbf* est basée sur la forme partielle conjonctive de résolution au lieu de la forme partielle de résolution, la détection de la non validité d'une formule booléenne quantifiée $QqyF$ (i.e. $QqyF \equiv \bot$) peut être améliorée : soit $Q(F' \wedge qy((y \vee F_y^+) \wedge (\neg y \vee F_y^-)))$ une forme partielle conjonctive de résolution de $QqyF$

- si $q = \exists$ et $F_y^+ \equiv \bot$ et $F_y^- \equiv \bot$ alors
  $Q\exists yF \equiv Q(F' \wedge (F_y^+ \vee F_y^-)) \equiv \bot$,
- si $q = \forall$ et $F_y^+ \equiv \bot$ ou $F_y^- \equiv \bot$ alors
  $Q\forall yF \equiv Q(F' \wedge (F_y^+ \wedge F_y^-)) \equiv \bot$,

Il apparaît aussi clairement par la définition du quantificateur universel que $Q(F \wedge \forall y(y \wedge F')) \equiv \bot$ avec F et F' deux formules booléennes quantifiées.

4.3 Equivalence de politique

Deux formules booléennes quantifiées logiquement équivalentes ne possèdent pas nécessairement le même ensemble de politiques valides. Par exemple $\top \equiv (\exists xx)$ et $x \models (\exists xx)$ mais la politique vide est l'unique politique valide pour $\top$. Ainsi, nous introduisons une nouvelle relation entre les formules booléennes quantifiées qui préserve l'ensemble des politiques et non seulement la validité.

Définition 4 (la relation $\cong$) *Soient F et F' deux formules booléennes quantifiées prénexes et closes. $F \cong F'$ si pour chaque politique π, $\pi \models F$ si et seulement si $\pi \models F'$.*

Clairement, les deux propriétés suivantes sont vérifiées.

- La relation $\cong$ est une relation d'équivalence.
- Soient F et F' deux formules booléennes telles que $F \equiv F'$ et $\{x_1, \ldots, x_n\}$ l'union des ensembles de variables de F et F' alors $q_1 x_1 \ldots q_n x_n F \cong q_1 x_1 \ldots q_n x_n F'$.

Comme corollaire du second point, si F' est une forme normale conjonctive d'une formule booléenne F alors $QF \cong QF'$.

4.4 Calcul des factorisations

En fait la procédure de décision *rqbf* est bien plus qu'une simple procédure de décision pour la validité d'une formule booléenne quantifiée, elle calcule aussi une fonction, appelée factorisation, des variables booléennes vers des paires de formules booléennes. Si la formule booléenne quantifiée est en forme prénexe avec pour lieur $q_1 x_1 \ldots q_n x_n$, chaque variable x_i est associée à une paire de formules booléennes appelées facteurs qui sont uniquement composés à partir des variables $x_1, \ldots, x_{i-1}$.

Définition 5 (Ensemble des factorisations) *L'ensemble $\mathcal{F}(Q)$ des factorisations d'une formule booléenne quantifiée est défini inductivement par :*

$\mathcal{F}($ *Entrée* : *un lieur*)
  Sortie : *l'ensemble des factorisations*
$\mathcal{F}(Q) = \mathcal{F}(Q, \emptyset)$
$\mathcal{F}($ *Entrées* : *un lieur*,
  un ensemble de variables)
  Sortie : *l'ensemble des factorisations*
$\mathcal{F}(\varepsilon, V) = \emptyset$
$\mathcal{F}(qyQ, V)$
$= \{\{y \mapsto (\psi^+(y), \psi^-(y))\} \cup \psi \mid (\psi^+(y), \psi^-(y)) \in \mathbf{PROP}^2_V, \psi \in \mathcal{F}(Q, V \cup \{y\})\}$

Dans la définition précédente, pour $q_1 x_1 q_2 x_2 \ldots q_n x_n$ un lieur , $\psi^+(x_1)$ et $\psi^-(x_1)$ peuvent seulement être $\bot$ ou $\top$ et $\psi^+(x_2)$ et $\psi^-(x_2)$ peuvent toujours être réduit à $\bot$, $\top$, x_1 ou $\neg x_1$.

L'algorithme fm calcule pour une formule booléenne quantifiée close et prénexe une factorisation associée.

Algorithme 2 (Calcul d'une factorisation)

$fm($ *Entrées* : *le lieur d'une QBF close et prénexe F,*
  la matrice de cette formule F)
  Sortie : *une factorisation si F est valide*
  sinon non
$fm(Q, f) = fm(Q, f, \emptyset)$
$fm($ *Entrées* : *un lieur*,
  une formule booléenne,
  une factorisation)
  Sortie : *une factorisation si F est valide*
  sinon non
$fm(\varepsilon, f, \psi) = \psi$ *si* $(f \equiv \top)$ *sinon non*
$fm(Q\exists y, f, \psi) =$
  soit $Q(\phi_y \wedge \exists y((y \vee \phi_y^+) \wedge (\neg y \vee \phi_y^-)))$, *FCPR de* $Q\exists yf$
  dans $fm(Q, (\phi_y \wedge (\phi_y^+ \vee \phi_y^-)), \{y \mapsto (\phi_y^+, \phi_y^-)\} \cup \psi)$
$fm(Q\forall y, f, \psi) =$
  soit $Q(\phi_y \wedge \forall y((y \vee \phi_y^+) \wedge (\neg y \vee \phi_y^-)))$, *FCPR de* $Q\forall yf$
  dans $fm(Q, (\phi_y \wedge (\phi_y^+ \wedge \phi_y^-)), \{y \mapsto (\phi_y^+, \phi_y^-)\} \cup \psi)$

Étant donné que dans la définition précédente seulement une forme conjonctive partielle de résolution est requise, la fonction de factorisation n'est pas unique.

Exemple 2 *Soit $\exists a \forall b \exists c\, G$ une QBF avec*

$$G = ((a \wedge (b \rightarrow c)) \vee (\neg a \wedge b \wedge c)).$$

Nous avons

$$\exists a \forall b \exists c\, G \equiv \exists a \forall b((a \vee b) \wedge \exists c(c \vee (a \wedge (\neg b \vee \neg a) \wedge \neg b))).$$

Donc

$fm(\exists a \forall b \exists c, G)$
$= fm(\exists a \forall b \exists c, G, \emptyset)$
$= fm(\exists a \forall b, (a \vee b), \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\})$
$= fm(\exists a, a, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top), b \mapsto (a, \top)\})$
$= \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top), b \mapsto (a, \top), a \mapsto (\bot, \top)\}$

La *forme conjonctive de résolution* est une formule booléenne quantifiée construite à partir d'une factorisation et de son lieur associé.

Définition 6 (Forme conjonctive de résolution) *Soit QF une formule booléenne quantifiée close et prénexe et $\phi = fm(Q, F)$ sa factorisation. Si QF est valide alors $Q\, crf(Q, \psi)$, avec*

$$crf(Q, \psi) = (\bigwedge_{\exists y \in Q} ((y \vee \psi^+(y)) \wedge (\neg y \vee \psi^-(y)))),$$

est la forme conjonctive de résolution de QF sinon c'est $\bot$.

Il est important de noté que dans la définition précédente seulement les facteurs des variables quantifiées existentiellement sont utilisées et que F n'est pas équivalente à $crf(Q, \psi)$.

L'exemple ci-dessous illustre le fait que les politiques valides d'une formule booléenne quantifiée peuvent être obtenues de la forme normale conjonctive de résolution.

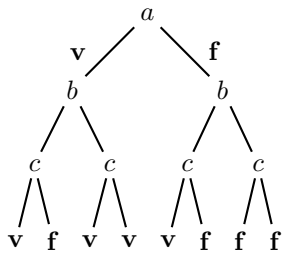

FIG. 3 – Arbre des interprétations pour la formule G

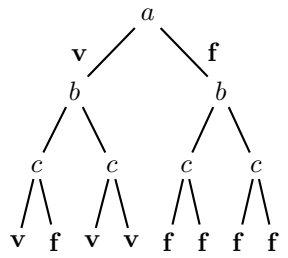

FIG. 4 – Arbre des interprétations pour la formule $crf(\exists a \forall b \exists c, fm(\exists a \forall b \exists c, G))$

Exemple 3 (Suite de l'exemple 2) *Nous notons* $\psi = fm(\exists a \forall b \exists c, G)$. *Alors*

$$\begin{array}{ll} & crf(\exists a \forall b \exists c, fm(\exists a \forall b \exists c, G)) \\ = & ((a \vee \psi^+(a)) \wedge (\neg a \vee \psi^-(a))) \wedge \\ & ((c \vee \psi^+(c)) \wedge (\neg c \vee \psi^-(c))) \\ = & a \wedge (c \vee (a \wedge (\neg b \vee \neg a) \wedge \neg b)). \end{array}$$

L'arbre de la figure 3 représente l'ensemble des interprétations de la formule G tandis que l'arbre de la figure 4 représente l'ensemble des interprétations de la formule $crf(\exists a \forall b \exists c, fm(\exists a \forall b \exists c, G))$.

Maintenant, nous observons que l'ensemble des politiques valides de $\exists a \forall b \exists c\, crf(\exists a \forall b \exists c, fm(\exists a \forall b \exists c, G))$ qui est

$$\{a; \{b \mapsto c, \neg b \mapsto c\}, \\ a; \{b \mapsto c, \neg b \mapsto \neg c\}\}$$

est aussi l'ensemble des politiques valides de $\exists a \forall b \exists c\, G$.

Comme démontré dans l'exemple suivant, la forme conjonctive de résolution ne préserve pas nécessairement les politiques.

Exemple 4 *Puisque* $\exists a \exists b (b \rightarrow a) \equiv \exists a \exists b (a \rightarrow b)$, *les deux factorisations* $\psi = \{a \mapsto (\top, \top), b \mapsto (\top, a)\}$ *et* $\psi' = \{a \mapsto (\top, \top), b \mapsto (\neg a, \top)\}$ *peuvent être le résultat de* $fm(\exists a \exists b, (b \rightarrow a))$. *Or* $crf(\exists a \exists b, \psi) = (\neg b \vee a)$ *et* $crf(\exists a \exists b, \psi') = (\neg a \vee b)$. *Donc,* $\neg a; b \models \exists a \exists b crf(\exists a \exists b, \phi')$ *mais* $\neg a; b \not\models \exists a \exists b (b \rightarrow a)$.

Une restriction sur les factorisations doit être imposée pour préserver les politiques dans la forme conjonctive de résolution.

Définition 7 *Soit $qxQF$ une formule booléenne quantifiée close et prénexe et $\beta \in \{\bot, \top\}$. Un algorithme fm préserve les politiques si*

$$\begin{array}{ll} & crf(qxQ, fm(qxQ, F))[x \leftarrow \beta] \\ \equiv & crf(Q, fm(Q, F[x \leftarrow \beta])). \end{array}$$

Par les propriétés sur la relation d'équivalences $\cong$ évoquées dans la section 4.3, tout algorithme fm qui ne réalise pas de renommage de variable préserve les politiques. Sous cette restriction d'un algorithme fm qui préserve les politiques, une politique est valide pour une formule booléenne quantifiée si et seulement si elle est valide pour sa forme conjonctive de résolution.

Théorème 2 *Soit F' une forme conjonctive de résolution pour une formule booléenne quantifiée F obtenue par un algorithme fm qui préserve les politiques alors $F \cong F'$.*

4.5 Calcul de l'ensemble des politiques valides d'une formule booléenne quantifiée prénexe

Un algorithme fm qui préserve les politiques calcule d'une formule booléenne quantifiée close et prénexe une formule booléenne quantifiée close et prénexe $\cong$-équivalente. De cette formule booléenne quantifiée, l'algorithme Π suivant calcule l'ensemble des politiques valides de la formule initiale.

Algorithme 3 (Calcul des politiques valides d'une QBF)

$\Pi($*Entrée* : *le lieur d'une QBF close et prénexe F, la matrice de la QBF F*)
Sortie : *l'ensemble des politiques valides de F*
$\Pi(Q, f) =$
soit $\psi = fm(Q, f)$
dans si $(\psi = non)$ *alors* $\emptyset$ *sinon* $\Pi(Q, \emptyset, \psi)$

$\Pi($*Entrées* : *un lieur, une interprétation, une factorisation*)
Sortie : *un ensemble de politiques*
$\Pi(\varepsilon, v, \emptyset) = \{\lambda\}$
$\Pi(\exists y Q, v, \psi) =$
$\{y; \pi \mid \pi \in \Pi(Q, v \cup \{y\}, \psi)$ *et* $v \models \psi^-(y)\}$
$\cup \quad \{\neg y; \pi \mid \pi \in \Pi(Q, v \cup \{\neg y\}, \psi)$ *et* $v \models \psi^+(y)\}$
$\Pi(\forall y Q, v, \psi) =$
$\{\{y \mapsto \pi, \neg y \mapsto \pi'\} \mid \pi \in \Pi(Q, v \cup \{y\}, \psi), \pi' \in \Pi(Q, v \cup \{\neg y\}, \psi)\}$

La correction et la complétude de l'algorithme Π pour le problème de recherche associé aux formules booléennes quantifiées est une conséquence du théorème 2.

Lemme 4 *Soit QF une formule booléenne quantifiée close et prénexe. Alors*

$\Pi(Q, F) =$
$\{\pi \mid \pi$ *est une politique de* QF *telle que* $\pi \models QF\}$.

Exemple 5 (Suite de l'exemple 3) *Nous rappelons que*

$$fm(\exists a \forall b \exists c, G) = \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top), b \mapsto (a, \top), a \mapsto (\bot, \top)\}$$

avec $G = ((a \wedge (b \rightarrow c)) \vee (\neg a \wedge b \wedge c))$. *Alors*

$$\Pi(\exists c, \{a, b\}, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\}) = \{c\}$$

et

$$\Pi(\exists c, \{a, \neg b\}, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\}) = \{c, \neg c\}$$

Donc

$$\begin{array}{l}\Pi(\forall b \exists c, \{a\}, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\}) = \\ \quad \{\{b \mapsto \pi, \neg b \mapsto \pi'\} \mid \\ \quad\ \pi \in \Pi(\exists c, \{a, b\}, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\}), \\ \quad\ \pi' \in \Pi(\exists c, \{a, \neg b\}, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\})\} \\ = \quad \{\{b \mapsto c, \neg b \mapsto c\}, \{b \mapsto c, \neg b \mapsto \neg c\}\end{array}$$

Donc

$$\begin{array}{l}\Pi(\exists a \forall b \exists c, \epsilon, fm(\exists a \forall b \exists c, G)) = \\ \{a; \pi \mid \pi \in \Pi(\forall b \exists c, \{a\}, \{c \mapsto (a \wedge (\neg b \vee \neg a) \wedge \neg b, \top)\})\} \\ = \{a; \{b \mapsto c, \neg b \mapsto c\}, a; \{b \mapsto c, \neg b \mapsto \neg c\}\end{array}$$

qui est l'ensemble des politiques valides de $\exists a \forall b \exists c\, G$.

4.6 Calcul de l'ensemble des politiques valides pour une formule booléenne quantifiée en forme normale conjonctive

Aucune procédure effective n'a été décrite pour calculer à partir d'une formule booléenne quantifiée close et prénexe une formule équivalente sous forme partielle conjonctive de résolution. En fait, comme suggéré par le lemme 1, C'est facile si la formule booléenne quantifiée $QqyF$ est en forme normale conjonctive : c'est une partition des clauses suivant un simple test d'appartenance sur y i.e.

$$F \equiv \bigwedge_{1 \leq i \leq n} C_i \wedge \bigwedge_{1 \leq i \leq n_y^+} (y \vee C_i^{y^+}) \wedge \bigwedge_{1 \leq i \leq n_y^-} (\neg y \vee C_i^{y^-})$$

avec $C_1, \ldots, C_n, C_1^{y^+}, \ldots, C_{n_y^+}^{y^+}, C_1^{y^-}, \ldots, C_{n_y^-}^{y^-}$ des clauses sans occurrence de y. Dans ce qui suit $\bigwedge_{1 \leq i \leq n} C_i$ est noté ϕ_y, et $\bigwedge_{1 \leq i \leq n_y^+} C_i^{y^+}$ est à nouveau noté ϕ_y^+, $\bigwedge_{1 \leq i \leq n_y^-} C_i^{y^-}$ est noté ϕ_y^- et le triplet $(\phi_y, \phi_y^+, \phi_y^-)$ représente la partition de F selon y.

La fonction *distribute* calcule à partir de deux formules booléennes en forme normale conjonctive une nouvelle formule booléenne en forme normale conjonctive sans tautologies par distributivité[1].

Maintenant, l'algorithme fm^{CNF} est définie comme une spécialisation de l'algorithme fm pour la forme normale conjonctive.

Algorithme 4 (Calcul d'une factorisation pour une QBF en CNF)

fm^{CNF}(*Entrées :* *le lieur d'une QBF en CNF F, la matrice de F*)
Sortie : *une factorisation si F est valide sinon non*

$fm^{CNF}(Q, f) = fm^{CNF}(Q, f, \emptyset)$

fm^{CNF}(*Entrées :* *un lieur, une formule booléenne CNF, une factorisation*)
Sortie : *une factorisation si F est valide sinon non*

$fm^{CNF}(\varepsilon, f, \psi) = \psi$ *si* $(f \equiv \top)$ *sinon non*

$fm^{CNF}(Qqy, f, \psi) =$
soit $(\phi_y, \phi_y^+, \phi_y^-)$ *une partition de f selon y.*
dans *si* $q = \exists$
alors $fm^{CNF}(\ Q, \phi_y \wedge distribute(\phi_y^+, \phi_y^-), \{y \mapsto (\phi_y^+, \phi_y^-)\} \cup \psi)$
sinon $fm^{CNF}(\ Q, \phi_y \wedge \phi_y^+ \wedge \phi_y^-, \{y \mapsto (\phi_y^+, \phi_y^-)\} \cup \psi)$

Étant donné que l'algorithme fm^{CNF} manipule la matrice seulement par équivalence, l'algorithme fm^{CNF} préserve les politiques.

Ce nouvel algorithme fm^{CNF} (qui est en fait une fonction) peut être inséré à la place de l'algorithme fm dans l'algorithme Π pour obtenir l'algorithme Π^{CNF} (qui devient aussi une fonction) qui calcule pour une formule booléenne quantifiée sous forme normale conjonctive l'ensemble des politiques valides. Tandis que la complexité de l'algorithme Π réside dans le calcul de la forme partielle de résolution, la complexité de l'algorithme Π^{CNF} réside dans le calcul de la distributivité.

L'algorithme fm^{CNF} reprend la procédure de décision QMRES de Pan et Vardi [13] qui est une Multi-resolution pour les formules booléennes quantifiées, étendant l'approche Multi-resolution basée sur les ZBDD de [4] pour le problème SAT. Le formalisme des ZBDD [12] (pour zero-suppressed binary decision diagrams) offre une représentation compacte pour des

[1] Si $(A \vee B) \equiv \top$ alors $distribute(A, B) = \top$.

ensembles de clauses et est doté des opérateurs booléens. La procédure QMRES a montré son efficacité sur des problèmes structurés, ce que sont des problèmes qui réclament le calcul des politiques. Nous développons une application similaire en C++ aussi basée sur les ZBDD. Une version jouet a été implantée en Sicstus Prolog[2] qui démontre que le surcoût du calcul des politiques par rapport au calcul de validité est très faible.

4.7 Détection des équivalences logiques

L'étape de résolution du lemme 3 n'est pas l'unique moyen de réaliser une élimination de quantificateurs : les équivalences logiques entre les variables booléennes peuvent aussi aider ([1] l'introduit dans le cas de formule en forme normale conjonctive). Le lemme suivant démontre comment le quantificateur le plus interne liant les deux variables logiquement équivalentes peut être éliminé.

Lemme 5 *Soit* $F = Qq_1yQ'q_2zQ''(F' \wedge (y \leftrightarrow z))$ *une formule booléenne quantifiée close et prénexe.*
- *Si* $q_2 = \exists$ *alors* $F \equiv Qq_1yQ'Q''F'[z \leftarrow y]$.
- *Si* $q_2 = \forall$ *alors* $F \equiv \bot$.

Pour calculer la politique, le lien entre la variable existentielle z et son équivalente y doit être conservé dans une classe d'équivalence (noté $equ(y)$).

Bien sûr, le lemme 5 à sa contrepartie avec $F = Qq_1yQ'q_2zQ''(F' \wedge (y \leftrightarrow \neg z))$. De manière similaire, une classe opposée (noté $opp(y)$) doit être maintenue pour conserver le lien entre les deux variables opposées.

Exemple 6 *Soit* $H = \forall a \exists b \exists c((a \leftrightarrow \neg b) \wedge (c \leftrightarrow \neg b))$. *Par une première application du lemme 5,* $H \equiv \forall a \exists b(a \leftrightarrow \neg b)$ *avec* $opp(b) = \{c\}$ *et* $equ(b) = \emptyset$. *Par une seconde application,* $H = \forall a \top$ *avec* $opp(a) = \{b\}$ *et* $equ(a) = \{c\}$. *Ainsi l'unique politique valide pour* H *est* $\{a \mapsto \neg b; c, \neg a \mapsto b; \neg c\}$.

5 Comparaisons et discussion

Dans cet article nous avons présenté des algorithmes d'élimination de quantificateurs pour le problème de recherche associé à une formule booléenne quantifiée. Nous n'avons pas trouvé dans la littérature de travaux portant sur le calcul des politiques par un algorithme d'élimination de quantificateurs.

La partie "décision" de nos algorithmes est en lien avec des procédures de décision basées sur la résolution ou l'élimination de quantificateurs. Büning et al. [2] étend le principe de résolution pour les formules booléennes [6] aux formules booléennes quantifiées. Leur procédure de décision, la Q-resolution, considère uniquement des formules booléennes quantifiées en forme normale conjonctive à l'instar de l'algorithme Π^{CNF}. La Q-resolution génère des résolvantes (des clauses booléennes) au lieu d'éliminer les quantificateurs et nécessite de les garder toutes pour être complet. Biere [1] étend la Q-resolution à ceci près qu'elle est appliquée uniquement sur le quantificateur existentiel le plus interne. Il applique l'expansion pour éliminer le quantificateur universel. Cette procédure de décision, nommée Quantor, est très efficace sur des classes de problèmes structurés.

Dans [5] il est fait état que la notion de politique est trop exigeante. En effet, dans le cas d'un jeu à deux joueurs, $J_\forall$ et $J_\exists$ jouant alternativement en valuant les variables d'une formule qui doit être valide pour que le joueur $J_\exists$ soit sûr de gagner, si l'adversaire a au moins une manière de gagner alors aucune politique valide n'existe et le joueur $J_\exists$ ne peut être aidé en rien. Pour remédier à cette absence de réponse [5] introduit la notion de politique partielle : si le joueur $J_\forall$ ne joue pas les coups qui interdisent la victoire du joueur $J_\exists$ celui-ci pourra obtenir ceux à jouer pour vaincre. Nous souhaitons nous pencher sur le calcul des politiques partielles des formules booléennes quantifiées pour des formules non valides par des algorithmes d'élimination de quantificateurs. Nous nous intéressons aussi dans ce cadre à l'application de meta-heuristiques comme celle effectuée dans [8] pour la recherche locale. Nous nous focalisons sur l'emploi de l'algorithmique génétique [11] et de recherche par colonie des fourmis [7]. Dans le cas où un processus de calcul meta-heuristique (algorithme génétique ou colonie de fourmis) s'arrête sans avoir trouvé la solution (soit parce que l'on n'a pas attendu assez longtemps, soit parce qu'il n'y en a pas), l'état ultime de la structure de donnée offre une solution approchée qui dans le cas du calcul des politiques peut s'apparenter au calcul des politiques partielles.

Enfin nous désirons calculer directement les formes partielles de résolution pour ne plus calculer la forme normale conjonctive qui, pensons-nous, n'est pas nécessairement la forme la mieux adaptée pour résoudre les problèmes ni de satisfiabilité, ni de validité.

[2] `http : ://www.info.univ-angers.fr/pub/stephan/Research/QBF/models_prolog.html`

Références

[1] A. Biere. Resolve and expand. In *SAT*, 2004.

[2] H. K. Büning, M. Karpinski, and A. Flögel. Resolution for quantified boolean formulas. *Information and Computation*, 117(1) :12–18, 1995.

[3] M. Cadoli, M. Schaerf, A. Giovanardi, and M. Giovanardi. An algorithm to evaluate quantified boolean formulae and its experimental evaluation. *Journal of Automated Reasoning*, 28(2) :101–142, 2002.

[4] P. Chatalic and L. Simon. Multi-resolution on compressed sets of clauses. In *International Conference on Tools with Artificial Intelligence*, 2000.

[5] S. Coste-Marquis, H. Fargier, J. Lang, D. Le Berre, and P. Marquis. Résolution de formules booléennes quantifiées : problèmes et algorithmes. In *Actes du 13e Congrès AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle (RFIA-02)*, Angers, France, 2002.

[6] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3) :201–215, July 1960.

[7] M. Dorigo, E. Bonabeau, and G. Theraulaz. Ant algorithms and stimergy. *Future Generation Computer Systems*, 16 :851–871, 2000.

[8] I.P. Gent, H.H. Hoos, A.G.D. Rowley, and K. Smyth. Unsing stochastic local search to solve quantified boolean formulae. In *Principles and Pratice of Contraints Programming*, 2003.

[9] E. Giunchiglia, M. Narizzano, and A. Tacchella. Backjumping for quantified boolean logic satisfiability. *Artificial Intelligence*, 145 :99–120, 2003.

[10] E. Giunchiglia, M. Narizzano, and A. Tacchella. Qube++ : an efficient qbf solver. In *Formal Methods in Computer-Aided Design*, 2004.

[11] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, 1996.

[12] S. Minato. *Binary Decision Diagrams and Applications to VLSI CAD*. Kluwer, 1996.

[13] G. Pan and M.Y. Vardi. Symbolic decision procedures for qbf. In *International Conference on Principles and Practice of Constraint Programming*, 2004.

[14] D.A. Plaisted, A. Biere, and Y. Zhu. A staisfiability procedure for quantified boolean formulae. *Discrete Applied Mathematics*, 130 :291–328, 2003.

[15] J. Rintanen. Improvements to the evaluation of quantified boolean formulae. In *IJCAI*, pages 1192–1197, 1999.

[16] J. Rintanen. Partial implicit unfolding in the davis-putnam procedure for quantified boolean formulae. In *Workshop on Theory and Applications of QBF, Int. Join Conference on Automated Reasoning*, Sienna, Italia, 2001.

[17] J.A. Robinson. A machine-oriented logic based on the resolution principle. *JACM*, 12(1) :23–41, 1965.