Deux Fragments Polynomiaux Complets pour le problème de la validité des Formules Booléennes Quantifiées

Florian Letombe

Deux fragments polynomiaux complets pour le problème de la validité des formules booléennes quantifiées

Article Jeune Chercheur JFPC'06

Florian Letombe

Centre de Recherche en Informatique de Lens, CNRS FRE 2499
Faculté des Sciences Jean Perrin
Université d'Artois
F-62307, Lens Cedex, France
letombe@cril.univ-artois.fr

Résumé

Dans cet article nous mettons en évidence deux fragments propositionnels complets duaux $\texttt{OCNF}_<$ et $\texttt{ODNF}_<$ qui constituent respectivement des sous-ensembles des formules propositionnelles CNF et DNF. Chacun de ces fragments permet l'élimination des quantifications en temps polynomial comme une loi interne sur le fragment. Grâce à cette propriété, le problème de la validité des formules booléennes quantifiées peut être décidé en temps polynomial lorsque la matrice de l'instance traitée appartient à l'un des deux fragments considérés. Sur le plan pratique, nous montrons que, contrairement à ce qui se passe pour les instances quantifiées issues d'autres classes polynomiales pour SAT, comme celles des formules Horn CNF ou plus généralement des formules Horn renommables CNF, les prouveurs QBF existants exhibent empiriquement un comportement polynomial sur les instances $\texttt{OCNF}_<$ quantifiées.

Abstract

In this paper, two complete dual propositional fragments $\texttt{OCNF}_<$ and $\texttt{ODNF}_<$, which are respectively subsets of CNF and DNF propositional formulae, are pointed out. Each of those fragments allow quantification's elimination in polynomial time as an internal law in the fragment. From this property, the validity problem for quantified Boolean formulae can be decided in polynomial time when the instance's matrix belongs to any of the two considered fragments. From a practical point of vue, we show that, contrariwise to what happens for quantified instances from other polynomial classes for SAT, like Horn CNF or, more generally, renamable Horn CNF fromulae, state-of-the-art QBF solvers exhibit empirically a polynomial behaviour when dealing with quantified $\texttt{OCNF}_<$ instances.

1 Introduction

$\textsc{val}(\texttt{QPROP}_{PS})$, le problème de validité pour les QBFs a une importance croissante en IA. Cela peut s'expliquer par le fait que, en tant que problème **PSPACE**-complet canonique, de nombreux problèmes d'IA peuvent être réduits à $\textsc{val}(\texttt{QPROP}_{PS})$ de manière polynomiale (cf. [10, 11, 4, 27, 24, 25]) ; de plus, pour différents domaines de l'IA (parmi lesquels la planification, le raisonnement non monotone, l'inférence paraconsistante), une approche basée sur une traduction vers QBF peut se révéler plus « efficace » que des algorithmes dédiés. Par conséquent, de nombreux prouveurs QBF ont été conçus (cf. principalement [5, 28, 12, 29, 15, 22, 30, 3, 14, 26, 2]) et évalués ces dernières années [21, 20].

$\textsc{val}(\texttt{QPROP}_{PS})$ est un problème calculatoirement difficile, aussi bien en théorie qu'en pratique. Les méthodes basées sur la notion de restriction comptent parmi celles permettant d'augmenter l'ensemble d'instances traitables en pratique. L'idée clé est de reconnaître les instances pour lesquelles des algorithmes spécifiques peuvent fournir une solution beaucoup plus efficacement que les prouveurs QBF généraux. Plusieurs re-

strictions traitables de QBF ont déjà été identifiées. [1, 13] montrent que le problème de la validité pour les formules de Krom quantifiées est décidable en temps polynomial. [17] ont prouvé que le problème de la validité des formules Horn CNF quantifiées forment une restriction traitable de QBF. Par conséquent, le problème de la validité des formules Horn renommables CNF quantifiées sont également décidables de façon polynomiale.

Le principal objectif de cet article est de présenter deux nouvelles classes polynomiales complètes pour VAL($\texttt{QPROP}_{PS}$) : $\texttt{OCNF}_{<}$ et $\texttt{ODNF}_{<}$. Ces deux fragments sont respectivement des sous-ensembles des formules propositionnelles CNF et DNF, qui sont des fragments cibles pour la compilation de « connaissances ». L'objectif est également d'approfondir cette étude des nouveaux fragments en comparant le comportement des prouveurs QBF actuels sur de tels fragments complets avec le comportement de ces mêmes prouveurs sur des fragments incomplets mais polynomiaux pour VAL($\texttt{QPROP}_{PS}$).

La suite de cet article est organisée comme suit. Quelques préliminaires formels sur les QBFs sont donnés au paragraphe 2. Les fragments polynomiaux complets et incomplets considérés, parmi lesquels les deux nouvelles classes polynomiales décrites dans ce papier, sont présentés au paragraphe 3. Le paragraphe 4 décrit les résultats expérimentaux obtenus sur certaines des classes étudiées et une analyse des différences expérimentales observées entre les fragments polynomiaux complets et les fragments polynomiaux incomplets. Enfin, le paragraphe 5 conclut le papier et donne quelques perspectives.

2 Formules Booléennes Quantifiées

Définition 1 (Syntaxe des QBFs) *Soit PS un ensemble fini de symboles propositionnels (aussi appelés variables). L'ensemble* $\texttt{QPROP}_{PS}$ *de formules booléennes quantifiées (*QBF*s) sur PS est le plus petit ensemble de mots définis inductivement comme suit*[1] *:*

1. *vrai, faux et chaque variable de PS appartient à* $\texttt{QPROP}_{PS}$ *;*
2. *si* ϕ *et* ψ *appartiennent à* $\texttt{QPROP}_{PS}$*, alors* $(\neg\phi)$*,* $(\phi \wedge \psi)$*,* $(\phi \vee \psi)$*,* $(\phi \Rightarrow \psi)$*,* $(\phi \Leftrightarrow \psi)$*,* $(\phi \oplus \psi)$ *appartiennent à* $\texttt{QPROP}_{PS}$ *;*
3. *si* ϕ *appartient à* $\texttt{QPROP}_{PS}$ *et* x *appartient à PS, alors* $(\forall x.\phi)$ *et* $(\exists x.\phi)$ *appartiennent à* $\texttt{QPROP}_{PS}$*.*

[1] Afin de simplifier la syntaxe, nous nous permettons d'omettre des parenthèses lorsque cela n'a aucun impact sur la sémantique.

Les occurrences de variables propositionnelles x dans une formule Σ de $\texttt{QPROP}_{PS}$ peuvent être partitionnées en trois ensembles : les occurrences *quantifiées*, *liées* et *libres* de x. Les occurrences quantifiées sont celles apparaissant dans une quantification, i.e., juste après un quantificateur $\forall$ ou $\exists$. Dans toute sous-formule $\forall x.\phi$ (resp. $\exists x.\phi$) de Σ, toutes les occurrences de x dans ϕ sont *liées* ; de telles occurrences de x sont dites *dans la portée de la quantification* $\forall x$ (resp. $\exists x$). Enfin, toutes les occurrences restantes de x dans Σ sont libres.

$Var(\Sigma)$ est l'ensemble des variables apparaissant dans Σ. Une variable x de $Var(\Sigma)$ est *libre* si et seulement si x a une occurrence libre dans Σ.

Une formule Σ est dite *polie* si et seulement si chaque occurrence liée d'une variable x de $Var(\Sigma)$ est dans la portée d'un quantificateur unique, et chaque variable libre n'a pas d'occurrence liée. Une formule Σ est dite sous forme *prénexe* si et seulement si $\Sigma = Qx_1(\ldots Qx_n(\phi)\ldots)$ où chaque occurrence de Q vaut soit $\forall$, soit $\exists$, et ϕ ne contient pas d'occurrence quantifiée de variable. ϕ est la *matrice* de Σ et la suite $Qx_1 \ldots Qx_n$ de quantifications est le *préfixe* de Σ. Une formule est dite *fermée* si et seulement si elle ne contient pas de variable libre. Le sous-ensemble de $\texttt{QPROP}_{PS}$ ne contenant que des formules sans quantification est noté $\texttt{PROP}_{PS}$. Un sous-ensemble important de $\texttt{PROP}_{PS}$ est le fragment des CNFs (Formes Normales Conjonctives) contenant les conjonctions (finies) de clauses. Une formule en CNF est souvent vue comme un ensemble de clauses. cnf et DNF sont des cas spécifiques de **formes normales négatives** ou $\texttt{NNF}_{PS}$: une formule est dite sous $\texttt{NNF}_{PS}$ lorsque les seuls connecteurs binaires qu'elle contient sont $\wedge$ et $\vee$ et que la portée de chaque occurrence du connecteur de négation $\neg$ est réduite à un symbole propositionnel.

Définition 2 ($\texttt{NNF}_{PS}$) $\texttt{NNF}_{PS}$ *est le sous-ensemble de* $\texttt{PROP}_{PS}$ *défini inductivement par :*

- $vrai, faux \in \texttt{NNF}_{PS}$,
- *si* $x \in V$*, alors* $x, \neg x \in \texttt{NNF}_{PS}$,
- *si* $\Sigma, \Psi \in \texttt{NNF}_{PS}$*, alors* $(\Sigma \wedge \Psi), (\Sigma \vee \Psi) \in \texttt{NNF}_{PS}$.

Pour chaque formule booléenne quantifiée Σ et chaque variable x, $\Sigma_{x \leftarrow 0}$ (resp. $\Sigma_{x \leftarrow 1}$) désigne la formule conditionnée à *faux* (resp. *vrai*) obtenue en remplaçant chaque occurrence libre de x dans Σ par *faux* (resp. *vrai*).

Définition 3 (Sémantique des QBFs) *Soit I une interprétation sur PS (i.e., une application de PS dans* $BOOL = \{0, 1\}$*). La sémantique d'une formule booléenne quantifiée* Σ *de* $\texttt{QPROP}_{PS}$ *dans I est*

la valeur de vérité $[\![\Sigma]\!](I)$ *de BOOL définie inductivement comme pour les formules propositionnelles, à la différence près que la définition contient deux règles supplémentaires :*

- *si* $\Sigma = \forall x.\phi$,
  alors $[\![\Sigma]\!](I) = min(\{[\![\phi_{x\leftarrow 0}]\!](I), [\![\phi_{x\leftarrow 1}]\!](I)\})$.
- *si* $\Sigma = \exists x.\phi$,
  alors $[\![\Sigma]\!](I) = max(\{[\![\phi_{x\leftarrow 0}]\!](I), [\![\phi_{x\leftarrow 1}]\!](I)\})$.

Une interprétation I est appelée *modèle* de Σ, notée $I \models \Sigma$, si et seulement si $[\![\Sigma]\!](I) = 1$. Si Σ a un modèle, elle est *satisfiable* ; sinon, elle est *contradictoire*. Si toute interprétation I sur PS est un modèle de Σ, Σ est *valide*. Si tout modèle de Σ est un modèle de μ, alors μ est une *conséquence logique* de Σ, notée $\Sigma \models \mu$. Enfin, lorsque $\Sigma \models \mu$ et $\mu \models \Sigma$, Σ et μ sont *équivalentes*, noté $\Sigma \equiv \mu$. La définition 3 montre que tous les connecteurs (y compris les quantifications que l'on peut voir comme des connecteurs unaires) sont vérifonctionnels dans le sens où la sémantique d'une QBF dans une interprétation I ne dépend que de la sémantique de ses sous-formules dans I. La nature vérifonctionnelle des connecteurs suffit ici à énoncer un (méta)théorème de substitution pour les QBFs : remplacer dans une QBF une sous-formule par une formule équivalente préserve l'équivalence avec la QBF de départ.

Comme en logique propositionnelle, on a $\Sigma \models \mu$ si et seulement si la formule $(\Sigma \wedge \neg\mu)$ est incohérente si et seulement si la formule $(\Sigma \Rightarrow \mu)$ est valide. En outre, le problème de la satisfiabilité d'une QBF Σ coïncide avec celui de la validité de sa fermeture existentielle $\exists Var(\Sigma).\Sigma$.

D'autres méta-théorèmes intéressants sont :

Proposition 1 *Soient* Σ, Ψ *des formules de* $\mathtt{QPROP}_{PS}$ *et* x, y *des variables de PS.*

1. $\forall x.\Sigma \equiv \Sigma_{x\leftarrow 0} \wedge \Sigma_{x\leftarrow 1}$.
2. $\exists x.\Sigma \equiv \Sigma_{x\leftarrow 0} \vee \Sigma_{x\leftarrow 1}$.
3. $\forall x.\Sigma \equiv \neg(\exists x.(\neg\Sigma))$.
4. *Si* x *n'est pas libre dans* Σ,
   alors $\forall x.\Sigma \equiv \exists x.\Sigma \equiv \Sigma$.
5. $\forall x.(\Sigma \wedge \Psi) \equiv (\forall x.\Sigma) \wedge (\forall x.\Psi)$.
6. $\exists x.(\Sigma \vee \Psi) \equiv (\exists x.\Sigma) \vee (\exists x.\Psi)$.
7. $\forall x.(\forall y.\Sigma) \equiv \forall y.(\forall x.\Sigma)$.
8. $\exists x.(\exists y.\Sigma) \equiv \exists y.(\exists x.\Sigma)$.
9. *Si* x *n'est pas libre dans* Σ, *alors* $\forall x.(\Sigma \vee \Psi) \equiv \Sigma \vee (\forall x.\Psi)$.
10. *Si* x *n'est pas libre dans* Σ, *alors* $\exists x.(\Sigma \wedge \Psi) \equiv \Sigma \wedge (\exists x.\Psi)$.

A partir des méta-théorèmes donnés en proposition 1 et le théorème de substitution, il est aisé de prouver que toute QBF peut être transformée en temps polynomial en une formule équivalente sous forme prénexe et polie (les variables liées peuvent être renommées sans effet sur l'équivalence). Puisqu'il est possible de permuter deux quantifications successives de la même nature (i.e., universelles ou existentielles) dans une QBF prénexe sans effet sur l'équivalence, pour tout sous-ensemble fini, non vide $S = \{x_1, \ldots, x_n\}$ de PS, nous notons $\forall S.\phi$ (resp. $\exists S.\phi$) pour désigner plus simplement $\forall x_1.(\ldots \forall x_n.\phi \ldots)$ (resp. $\exists x_1.(\ldots \exists x_n.\phi \ldots)$).

Le problème $\textsc{val}(\mathtt{QPROP}_{PS})$ est souvent défini comme suit.

Définition 4 ($\textsc{val}(\mathtt{QPROP}_{PS})$) $\textsc{val}(\mathtt{QPROP}_{PS})$ *est le problème de décision suivant :*

- **Entrée :** *Une formule sous forme prénexe, polie, fermée* Σ *de* $\mathtt{QPROP}_{PS}$ *;*
- **Question :** Σ *est-elle valide ?*

Plus généralement, nous utilisons les notations suivantes.

Définition 5 (Notations) *Soit* $\mathcal{C} \subseteq \mathtt{PROP}_{PS}$.
On note :

- $\mathtt{Q}\mathcal{C}$ *le sous-ensemble de* $\mathtt{QPROP}_{PS}$ *formé des formules prénexes, fermées, polies dont la matrice est dans* $\mathcal{C}$.
- $\textsc{val}(\mathtt{Q}\mathcal{C})$ *est le problème de décision suivant :*
  - **Entrée :** *Une formule* Σ *de* $\mathtt{Q}\mathcal{C}$ *;*
  - **Question :** Σ *est-elle valide ?*

3 Fragments polynomiaux

Dans la suite, un fragment propositionnel est dit traitable pour $\textsc{val}(\mathtt{QPROP}_{PS})$ si et seulement si l'appartenance à ce fragment peut être décidée en temps polynomial et s'il existe un algorithme de décision polynomial pour le problème de validité des formules booléennes quantifiées (fermées, polies, prénexes) dont la matrice appartient à ce fragment.

3.1 Fragments incomplets

Quelques classes polynomiales incomplètes, i.e. les fragments qui ne sont pas suffisamment expressifs pour permettre la représentation de toutes les formules propositionnelles, ont été identifiées jusque là pour les

QBFs. Nous nous focalisons ici sur deux d'entre elles : les formules Horn CNF et les formules Horn CNF quantifiées.

Définition 6 (Formule Horn CNF)
Une **formule Horn** CNF *est une formule* CNF *dont chaque clause contient au plus un littéral positif.*

Exemple 1 (Formule Horn CNF)
La formule suivante est une formule Horn CNF *:*

$$(a \vee \neg b \vee \neg d) \wedge$$
$$(\neg a \vee c) \wedge$$
$$(\neg b \vee \neg c \vee d) \wedge$$
$$(\neg a \vee \neg c \vee \neg d).$$

Clairement, la reconnaissance de formules Horn CNF est polynomiale. De plus, Kleine-Büning *et al.* décrivent une méthode de résolution des formules Horn CNF quantifiées (la Q-unit-résolution) et montrent que le problème de validité de telles formules par Q-unit-résolution peut être décidé en temps $\mathcal{O}(|\Sigma| \times r)$, où r est le nombre d'occurrences positives de variables universelles dans une formule Horn CNF Σ [17].

Définition 7 (Formule Horn renommable CNF)
Une **formule Horn renommable** ϕ *est une formule* CNF *telle qu'il existe une sous-ensemble L_ϕ des littéraux de ϕ et une substitution*[2]

$$\begin{aligned} \sigma : L_{PS} &\rightarrow L_{PS} \\ l &\mapsto l^c \text{ si } l \in L_\phi \\ l &\mapsto l \text{ sinon} \end{aligned}$$

telle que $\hat{\sigma}(\phi)$ est une formule Horn CNF*. $\hat{\sigma}$ est l'extension de σ à NNF_{PS} définie par $\hat{\sigma}(vrai) = vrai$, $\hat{\sigma}(faux) = faux$, $\hat{\sigma}(l) = \sigma(l)$ si l est un littéral et $\hat{\sigma}(\alpha \odot \beta) = \hat{\sigma}(\alpha) \odot \hat{\sigma}(\beta)$ pour $\odot \in \{\wedge, \vee\}$.*

Intuitivement, une formule Horn renommable CNF est une formule qui peut être transformée en une formule Horn CNF par renommage (i.e. passage au complémentaire) de certains de ses littéraux.

Exemple 2 (Formule Horn renommable CNF)
La formule suivante est une formule Horn renommable CNF *:*

$$(a \vee b \vee \neg d) \wedge$$
$$(\neg a \vee \neg c) \wedge$$
$$(b \vee c \vee d) \wedge$$
$$(\neg a \vee c \vee \neg d).$$

Il s'agit de la formule de l'exemple 1 dans laquelle nous avons renommé les variables b et c.

[2] l^c désigne le complémentaire de l, i.e. si $l = x \in$ PS, alors $l^c = \neg x$ et si $l = \neg x \in$ PS, alors $l^c = x$.

Dans [16], J. J. Hébrard propose un algorithme de reconnaissance de formules Horn renommables CNF. Une fois un renommage R calculé (s'il existe) pour la matrice CNF d'une formules quantifiée Σ, il suffit d'utiliser l'algorithme de Kleine-Büning *et al.* [17] sur la QBF Σ de matrice $\hat{\sigma}_R(\Sigma)$ pour déterminer en temps polynomial si la formules de départ est valide ou pas.

3.2 Fragments complets

Considérons tout d'abord les fragments duaux MODS et CI [9].

Définition 8 (MODS et CI)

- *Le langage* MODS *est le sous-ensemble de* DNF *contenant des formules $\Sigma = \gamma_1 \vee \ldots \vee \gamma_k$ telles que, pour chaque terme γ_i ($i \in 1 \ldots k$) et pour chaque variable $x \in Var(\Sigma)$, γ_i contient x ou $\neg x$.*
- *Le langage* CI *est le sous-ensemble de* CNF *contenant des formules $\Sigma = \gamma_1 \wedge \ldots \wedge \gamma_k$ telles que, pour chaque clause γ_i ($i \in 1 \ldots k$) et pour chaque variable $x \in Var(\Sigma)$, γ_i contient x ou $\neg x$.*

Exemple 3 (Formule MODS/CI)
$(a \wedge b \wedge c) \vee (\neg a \wedge \neg b \wedge c) \vee (\neg a \wedge b \wedge c) \vee (\neg a \wedge \neg b \wedge \neg c)$ *est une formule* MODS*.* $(a \vee b \vee c) \wedge (\neg a \vee \neg b \vee c) \wedge (\neg a \vee b \vee c) \wedge (\neg a \vee \neg b \vee \neg c)$ *est une formule* CI*.*

Nous avons également considéré les deux fragments propositionnels $\mathrm{ODNF}_<$ et $\mathrm{OCNF}_<$ qui sont respectivement des sous-ensembles de DNF et de CNF. Soit $<$ un ordre total, strict sur les variables de $PS = \{x_1, \ldots, x_n\}$ s.t. $x_1 < \ldots < x_n$. Nous définissons :

- Le langage $\mathrm{ODNF}_<$ est l' ensemble de toutes les formules DNF $\Sigma = \gamma_1 \vee \ldots \vee \gamma_k$ où chaque terme satisfiable γ_i ($i \in 1 \ldots k$) de Σ est tel que pour tout $j \in 1 \ldots n$, si x_j ou $\neg x_j$ apparaît dans γ_i, alors pour tout l tel que $1 \leq l < j$, x_l ou $\neg x_l$ apparaît dans γ_i.
- Le langage $\mathrm{OCNF}_<$ est l' ensemble de toutes les formules CNF $\Sigma = \gamma_1 \wedge \ldots \wedge \gamma_k$ où chaque clause non tautologique γ_i ($i \in 1 \ldots k$) de Σ est telle que pour tout $j \in 1 \ldots n$, si x_j ou $\neg x_j$ apparaît dans γ_i, alors pour tout l tel que $1 \leq l < j$, x_l ou $\neg x_l$ apparaît dans γ_i.

Exemple 4 (Formule $\mathrm{OCNF}_</\mathrm{ODNF}_<$)
Soit $PS = \{a, b, c\}$ et $<$ tel que $a < b < c$. $\neg a \vee (a \wedge b)$ *est une formule de* $\mathrm{ODNF}_<$*.* $(a \vee b) \wedge (\neg a \vee b \vee c)$ *est une formule de* $\mathrm{OCNF}_<$*.*

Clairement, $\texttt{OCNF}_<$ (resp. $\texttt{QOCNF}_<$) est le fragment dual de $\texttt{ODNF}_<$ (resp. $\texttt{QODNF}_<$) ; en particulier, la négation de toute formule $\texttt{ODNF}_<$ (resp. $\texttt{QODNF}_<$) peut être transformée en temps linéaire en une formule $\texttt{OCNF}_<$ (resp. $\texttt{QOCNF}_<$), et l'inverse est également vrai. De plus, $\texttt{OCNF}_<$ et $\texttt{ODNF}_<$ sont des fragments propositionnels complets puisqu'ils incluent respectivement les fragments complets MODS et CI.

Définition 9 (Préfixe compatible)
Soit $\Sigma = QS_1 \ldots QS_n.\phi$ *dans* $\texttt{QPROP}_{PS}$ *prénexe, polie, fermée où chaque* Q *est un quantificateur et* $\{S_1, \ldots, S_n\}$ *est une partition de* $Var(\phi)$ *qui ne contient pas l'ensemble vide. Le préfixe* $QS_1 \ldots QS_n$ *de* Σ *est dit compatible avec un ordre total, strict* $<$ *sur* $Var(\phi)$ *si et seulement si pour chaque* $x, y \in Var(\phi)$ *tel que* $x < y$*, si* $x \in S_i$ *et* $y \in S_j$*, alors* $j \geq i$*.*

Proposition 2 *Les restrictions de* VAL($\texttt{QODNF}_<$) *et* VAL($\texttt{QOCNF}_<$) *aux instances ayant un préfixe compatible avec* $<$ *sont dans* **P**.

Preuve 1 Puisque la négation d'une formule $\texttt{QOCNF}_<$ peut être transformée en temps linéaire en une formule $\texttt{QODNF}_<$, il suffit de prouver que VAL($\texttt{QODNF}_<$) est dans **P** lorsque l'on restreint $\texttt{QOCNF}_<$ aux instances ayant un préfixe compatible avec $<$.

La preuve est constructive et repose sur l'existence d'un algorithme en temps polynomial SolveODNF$_<$ ci-après (algorithme 1). Ce dernier consiste en l'élimination des quantifications comme une loi interne dans le fragment $\texttt{ODNF}_<$. Dans cet algorithme, toute formule $\texttt{ODNF}_<$ ϕ est considérée, en toute généralité, comme un ensemble de termes et tout terme comme un ensemble de littéraux. Après la première étape de l'algorithme, tout terme dans ϕ est satisfiable (retirer les termes insatisfiables de ϕ peut être réalisé facilement en temps polynomial). La formule $\texttt{ODNF}_<$ résultante est soit l'ensemble vide, (i.e., la disjonction vide équivalente à *faux* ($\{\} \equiv$ *faux*)), soit un ensemble contenant la conjonction vide, équivalente à *vrai* ($\{\{\}\} \equiv$ *vrai*).

Expliquons comment les quantifications peuvent être éliminées et considérons dans un premier temps le cas les quantifications existentielles ; nous tirons avantage des deux équivalences : pour toute paire de QBFs α et β et toute variable $x \in PS$, nous avons $\exists x.(\alpha \vee \beta) \equiv (\exists x.\alpha) \vee (\exists x.\beta)$ (point 6 de la proposition 1), et $\exists x.\gamma$ est équivalent au terme $\gamma \setminus \{x, \neg x\}$ obtenu en retirant x et $\neg x$ de γ lorsque γ est un terme satisfiable (vu comme un ensemble de littéraux) (ce qui est une conséquence directe de la définition du conditionnement et du point 2 de la proposition 1). Soit x la dernière variable de $Var(\Sigma)$ par rapport à $<$. Clairement, si γ est un terme canonique (codant un modèle) sur $X \cup \{x\}$ (avec $x \notin X$), alors $\gamma \setminus \{x, \neg x\}$ est un terme canonique sur X. Donc retirer toute occurrence de x et $\neg x$ dans une telle formule $\texttt{ODNF}_<$ ϕ conduit à une formule $\texttt{ODNF}_<$ équivalente à $\exists x.\phi$.

Focalisons nous à présent sur le cas des quantificateurs universels. Soit ϕ une formule $\texttt{ODNF}_<$, vue comme un ensemble de termes satisfiables sur $X \cup \{x\}$, où $x \notin X$ et x est la dernière variable de $Var(\Sigma)$ par rapport à $<$. Tout terme de ϕ contenant x ou $\neg x$ est canonique sur $X \cup \{x\}$. À présent, les termes γ de ϕ peuvent être partitionnés en trois ensembles (interprétés disjonctivement) : W, S et S'. W est l'ensemble de tous les termes γ tels que $x \notin Var(\gamma)$. W peut être vu comme une formule $\texttt{ODNF}_<$ sur X. S est l'ensemble de tous les termes γ de ϕ contenant x ou $\neg x$ tels que $switch(\gamma, x)$ appartient à ϕ également, où $switch(\gamma, x)$ est le terme canonique sur $X \cup \{x\}$ qui coïncide avec γ pour toute variable de X mais qui contient $\neg x$ lorsque γ contient le littéral x, alors que $switch(\gamma, x)$ contient x lorsque γ contient $\neg x$. Il est évident que γ appartient à S si et seulement si $switch(\gamma, x)$ appartient à S. Enfin, S' est l'ensemble des termes restants de ϕ (i.e. tout terme $\gamma \in S'$ contenant x ou $\neg x$ mais qui n'appartient pas à S). Par construction, nous avons $\forall x.\phi \equiv \forall x.(W \vee S \vee S') \equiv W \vee \forall x.(S \vee S')$ (du point 9 de la proposition 1). Nous observons à présent que $S = \{\gamma_1, \ldots, \gamma_k\}$ (interprété disjonctivement) est indépendant de x [19] dans le sens où il est équivalent à la disjonction $\psi = (\gamma_1 \setminus \{x, \neg x\}) \vee \ldots \vee (\gamma_k \setminus \{x, \neg x\})$ qui est une formule $\texttt{ODNF}_<$ sur X également (pour tout γ_i ($i \in 1 \ldots k$), nous avons $\gamma_i \vee switch(\gamma_i, x) \equiv \gamma_i \setminus \{x, \neg x\}$). Clairement, ψ peut être calculé en temps polynomial dans la taille de ϕ. Puisque ψ est indépendant de x, nous avons $\forall x.\phi \equiv W \vee \psi \vee (\forall x.S')$. Enfin, il nous reste à montrer que $\forall x.S'$ est insatisfiable (et ainsi que $\forall x.\phi \equiv W \vee \psi$). Soit γ un terme canonique quelconque sur X vu comme une interprétation sur X ; par définition, γ satisfait $\forall x.S'$ si et seulement si à la fois l'extension à l'affectation de x à 0 et l'extension à l'affectation de x à 1 sont des modèles de S' ; mais cela est impossible par la définition de S' (si l'une de ces extensions appartient à S', cela ne peut être le cas de la seconde également puisqu'elle est obtenue en inversant x dans la première). Par conséquent, $(\forall x.S')$ est insatisfiable. □

SolveODNF$_<$ s'exécute en temps $\mathcal{O}(n \times |\phi|^2)$. Cet algorithme pourrait être facilement affiné, en retournant 0 dès que l'ensemble vide a été obtenu.

Illustrons à présent la manière dont SolveODNF$_<$ fonctionne sur deux exemples simples (où $a < b$) :

Exemple 5

Algorithm 1: Algorithme polynomial pour la restriction de VAL(QODNF$_<$) aux instances ayant un préfixe compatible

procédure SOLVEODNF$_<$
Données: Une formule $Q_1x_1 \dots Q_nx_n.\phi$
où $\phi \in$ ODNF$_<$ et $x_1 < \dots < x_n$
Résultat: 1 if $Q_1x_1 \dots Q_nx_n.\phi$ est valide, 0 sinon
début
  Retirer les termes insatisfiables de ϕ ;
  pour i **de** n **à** *1* **faire**
    si $Q_i = \forall$ **alors**
      $\phi' \leftarrow \{\gamma \in \phi \mid switch(\gamma, x_i) \in \phi$ ou $x_i \notin Var(\gamma)\}$;
    sinon
      $\phi' \leftarrow \phi$;
    $\phi \leftarrow \{\gamma \setminus \{x_i, \neg x_i\} \mid \gamma \in \phi'\}$;
  si $\phi = \{\}$ **alors**
    retourner 0 ;
  sinon
    retourner 1 ;
fin

1. $\forall a \exists b.\{\{\neg a\},\{a,\neg b\}\} \rightarrow \forall a.\{\{\neg a\},\{a\}\} \rightarrow \{\{\}\} \equiv vrai$.
2. $\exists a \forall b.\{\{\neg a, b\},\{a, b\}\} \rightarrow \exists a.\{\} \rightarrow \{\} \equiv faux$.

Une conséquence immédiate est que l'on obtient la traitabilité de VAL(QMODS) et VAL(QCI) :

Corollaire 1 VAL(QMODS) *et* VAL(QCI) *sont dans* **P**.

Preuve 2 Directe à partir du fait que toute formule de QMODS (resp. QCI) est une formule de QODNF$_<$ (resp. QOCNF$_<$) ayant un préfixe compatible avec $<$, quel que soit l'ordre strict, total $<$. □

4 Expérimentations

D'un point de vue pratique, il est important de déterminer comment les prouveurs QBF existants se comportent face à des instances de fragments traitables. Spécifiquement, lorsque les prouveurs ne résolvent pas « facilement » des instances de classes traitables, déterminer les raisons de cet échec peut conduire à une avancée intéressante dans l'amélioration des prouveurs actuels. Afin de répondre à cette question, nous avons effectué des expérimentations sur des instances de QCI et QOCNF$_<$ dont le préfixe est compatible avec $<$, qui sont des sous-ensembles de QCNF, la classe de formules que la plupart des prouveurs QBF actuels traitent.

De tels résultats expérimentaux existent déjà sur des instances dont la matrice est réduite à des fragments incomplets traitables tels que les formules Horn et Horn renommables [7]. Ces résultats montrent les difficultés des prouveurs actuels à résoudre de tels types d'instances et sont rappelés au paragraphe suivant.

4.1 Cadre expérimental

Nous avons généré des benchmarks de QOCNF$_<$ munis d'un préfixe compatible de la manière suivante. Étant donné un nombre de variables n, tout entier dans $\{1, \dots, n\}$ identifie une variable. Nous avons pris l'ordre naturel des entiers pour $<$. Afin de générer une clause de la matrice d'une formule QOCNF$_<$, la taille s d'une clause est d'abord choisie aléatoirement dans $\{1, \dots, n\}$ selon une distribution uniforme, puis, le signe de chacune des s variables (de 1 à s) de la clause est choisi selon une distribution uniforme.

Pour nos expérimentations, nous avons fixé n à 100, 200, 300, 400, 500, 1000 variables, ce qui a produit de gros benchmarks, comparativement aux possibilités des prouveurs actuels. Le nombre de clauses a été arbitrairement fixé au quadruple du nombre de variables ($m = 4 \times n$). Le préfixe est généré tel que la compatibilité avec $<$ soit assurée. Le nombre d'alternances de quantificateurs a été fixé à 12 : 11 ensembles de la même taille (n *div* 11) et au moins un avec les variables restantes (n *mod* 11). Notons que ce nombre d'alternances est également relativement élevé par rapport aux benchmarks standard de QBFs.

Nous avons utilisé trois prouveurs QBF pour ces expérimentations : un prouveur QBF développé dans notre laboratoire, appelé OpenQBF, qui a participé aux trois évaluations QBF et qui a obtenu des résultats moyens lors de ces évaluations, mais qui fait partie du peu de prouveurs corrects des deux dernières évaluations QBF. Nous avons également utilisé Semprop (version 010604) et Qube-Rel 1.3, deux prouveurs QBF disponibles publiquement. Les prouveurs ont fonctionné sur un PIV 3GHz muni de 1.5 Go de RAM sous Linux. Le prouveur Java a tourné sur la dernière Sun Java VM (1.5.0_06) en utilisant les paramètres par défaut.

4.2 Résultats sur des fragments complets

Le tableau 1 résume le comportement des trois prouveurs QBF sur des instances QCI et QOCNF$_<$. Notons que les temps d'exécution sont exprimés en secondes, pour résoudre 100 instances avec les mêmes paramètres. La colonne *taille* présente l'espace total utilisé par les benchmarks générés (pour un total supérieur à 3 Go pour ces expérimentations !). La colonne *#Vrai* présente le pourcentage d'instances positives de QBF. Nous pouvons remarquer que toutes les instances de QCI étaient positives. Cela peut s'expliquer par le fait

	CI					$\texttt{OCNF}_<$ (préfixe compatible)				
#vars	taille	OpenQBF	Semprop	Qube	#Vrai	taille	OpenQBF	Semprop	Qube	#Vrai
100	14	36	10	2	100	8	28	2	1	11
200	62	71	71	7	100	30	44	12	4	12
300	144	124	240	16	100	69	73	38	8	13
400	260	209	554	28	100	125	104	94	14	19
500	411	316	1062	43	100	198	145	165	21	10
1000	1680	1217	8638	174	100	824	532	1601	82	13

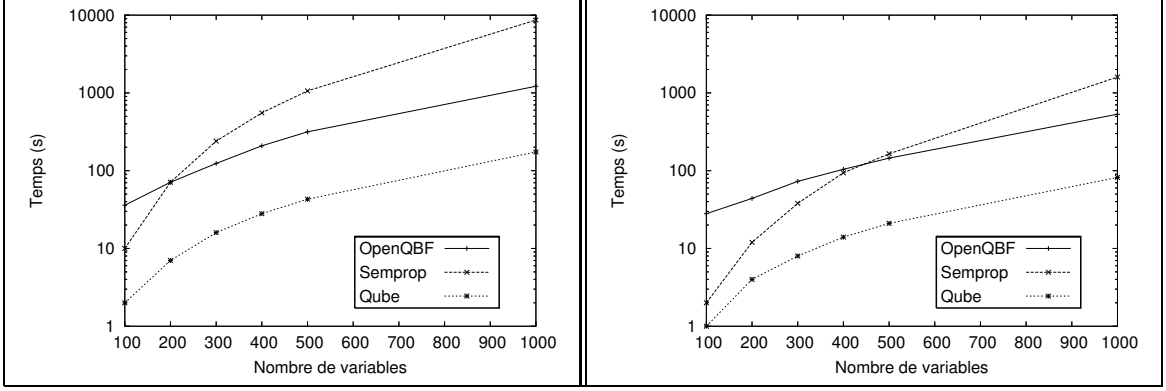

Table 1: Temps d'exécution de trois prouveurs QBF sur des instances de deux classes polynomiales de QBF. Temps CPU cumulés en secondes pour résoudre 100 instances (taille de ces 100 instances donnée en Mo) pour un nombre donné de variables.

que chaque clause possède environ la moitié de ses littéraux quantifiés existentiellement (puisque par construction, à peu près la moitié des variables sont quantifiées existentiellement et que les clauses contiennent toutes les variables).

Notons que tous les prouveurs se comportent mieux sur des benchmarks $\texttt{QOCNF}_<$ que sur des benchmarks QCI. Cela est probablement dû à la différence de taille des benchmarks : les benchmarks QCI sont deux fois plus gros que les benchmarks $\texttt{OCNF}_<$. Précisons également qu'il est généralement plus facile de résoudre des instances négatives que des instances positives et que la plupart des instances dans $\texttt{QOCNF}_<$ sont négatives.

Les trois prouveurs ont été capables de résoudre l'ensemble des benchmarks (environ 86 secondes par instance dans le pire des cas, même si les temps d'exécution pour ce cas s'étendent sur plus d'un ordre de grandeur). Notons que ces benchmarks sont généralement gros comparés aux benchmarks QBF usuels, ce qui explique pourquoi un prouveur généralement robuste tel que Semprop ne se comporte pas bien.

Ces expérimentations montrent que les prouveurs actuels n'ont pas de problèmes pour résoudre des instances de QCI et $\texttt{QOCNF}_<$. Ce résultat contraste avec nos résultats expérimentaux précédents sur d'autres fragments traitables (mais incomplets) pour le problème de la validité (Horn CNF quantifiées et Horn renommables CNF quantifiées) [6] qui ont été confirmés lors de la dernière évaluation de prouveurs QBF [23].

4.3 Résultats sur des fragments incomplets

Il est intéressant de noter le comportement des prouveurs QBF actuels sur des instances de formules dont la matrice est constituée de clauses Horn CNF ou Horn renommables CNF.

Les matrices de ces formules, en tant qu'instances SAT, peuvent être facilement résolues par n'importe quel prouveur SAT actuel : bien que ces prouveurs n'utilisent pas d'algorithmes dédiés à de tels fragments polynomiaux pour SAT, ces instances « faciles en théorie » le sont aussi en pratique. A contrario, nous avons observé que les prouveurs QBF actuels sont peu performants sur les instances QBF correspondantes. En effet, nous avons soumis des instances QBF Horn renommables à l'évaluation des prouveurs QBF 2005 et voici les résultats obtenus par les prouveurs participant à cette évaluation :

Notons que pour chaque prouveur indiqué dans la première colonne, nous indiquons dans la colonne #Vrai (resp. #Faux, Total) le nombre d'instances positives (resp. d'instances négatives, total d'instances) que celui-ci a résolues sur les 81 (resp. 82, 163) proposées à l'évaluation.

Prouveur	#Vrai /81	#Faux /82	Total /163
ssolve	78	77	155
Semprop	54	55	**109**
yquaffle	40	35	75
GRL	7	59	66
qbfbdd	42	15	57
QchaffLearn	2	53	55
walkqsat	0	39	39
sKizzo 0.4	0	33	33
sKizzo 0.5	0	32	32
OpenQBF	0	20	**20**
quantor	0	10	10
QMRes	0	0	0

Table 2: Nombre d'instances résolues par les prouveurs QBF participant à l'évaluation comparative 2005 sur des instances QBF Horn renommables.

On peut observer que seuls deux prouveurs sont capables de résoudre plus de 50% des instances considérées dans le temps imparti. De notre point de vue, les performances des prouveurs actuels sur ces instances montrent que les approches actuelles, issues pour la plupart de généralisations de techniques pour SAT, ne sont pas adaptées à la résolution d'instances de VAL($\texttt{QPROP}_{PS}$). En effet, ceux-ci ne sont même pas capables de résoudre systématiquement des instances a priori « faciles ».

En conséquence, nous pensons que le développement de nouveaux algorithmes pour des fragments polynomiaux de VAL($\texttt{QPROP}_{PS}$) peut aussi aider au développement d'algorithmes généraux plus performants.

4.4 Discussion

Les résultats expérimentaux présentés précédemment révèlent diverses lacunes des prouveurs QBF actuels. Ils laissent également présager de possibilités d'orientation diverses pour la conception d'algorithmes spécifiques pour VAL($\texttt{QPROP}_{PS}$) que nous développons dans ce paragraphe.

Dans un premier temps, nous sommes en droit de nous questionner sur l'utilité de reconnaître des sous-formules appartenant aux fragments polynomiaux complets découverts. En effet, les prouveurs QBF actuels qui ne sont pas munis d'algorithmes spécifiques pour résoudre des instances de formules de ce type se comportent très convenablement en pratique. De plus, ces résultats sont obtenus sur des formules dont les paramètres sont relativement conséquents (au plus 1000 variables et 4000 clauses). Enfin, en dépit de ce dernier point, les instances générées sont de très grande taille, comparées aux formules Horn CNF et Horn renommables CNF munies de paramètres identiques. Par exemple, il nous a fallu plus de 30 fois plus d'espace mémoire pour générer les 100 formules du fragment CI de 400 variables et 1600 clauses par rapport aux 100 formules Horn renommables CNF comportant le même nombre de variables et de clauses (260 Mo contre 7.9 Mo).

Ces expérimentations confirment, au contraire, l'importance de l'existence d'algorithmes spécifiques pour la reconnaissance et la résolution de classes polynomiales incomplètes. Les résultats obtenus avec les mêmes prouveurs QBF (OpenQBF et Semprop) sur des instances de formules ayant des configurations similaires sont significatifs. Clairement, les instances de formules Horn CNF renommables soumises à l'évaluation comparatives de prouveurs QBF 2005 comportaient au plus 400 variables et 2360 clauses et nombre d'entre elles n'ont pas été résolues par les prouveurs OpenQBF et Semprop (respectivement 54 et 143 instances) dans le temps imparti. En revanche, l'ensemble des instances CI et $\texttt{OCNF}_{<}$ générées ont été résolues facilement par ces mêmes prouveurs. Ces résultats nous encouragent donc à poursuivre notre étude en nous axant plus particulièrement sur des fragments incomplets pour lesquelles le problème VAL($\texttt{QPROP}_{PS}$) serait polynomial.

5 Conclusion et perspectives

Dans cet article, nous avons présenté deux nouveaux fragments propositionnels complets pour lesquels le problème de la validité des QBFs est polynomial en temps : les fragments $\texttt{ODNF}_{<}$ et $\texttt{OCNF}_{<}$.

Nous avons montré qu'en pratique, les prouveurs QBF actuels parvenaient à résoudre ce type d'instances « facilement ». Ce résultat contraste avec ce qui a pu être observé lors de la dernière évaluation comparative de prouveurs QBF qui a montré les difficultés des prouveurs à résoudre des instances de formules polynomiales incomplètes (instances à matrices Horn et Horn renommables).

Cette tendance, qui s'est confirmée pour les deux fragments considérés, nous encourage dans la poursuite des travaux sur les classes polynomiales incomplètes. En particulier, on est en droit de se demander s'il n'existerait pas d'autres classes de QBFs incomplètes plus (ou au moins autant) expressives que les formules Horn renommables, pour lesquelles le problème de la validité serait polynomial. Cela nous permettrait d'étendre davantage le panorama de complexité de VAL($\texttt{QPROP}_{PS}$). Plus particulièrement, il serait intéressant de déterminer comment la notion de modèles

K-booléens de [18], dans le cas où K est une classe de fonctions booléennes qui peut être encodée en temps polynomial, pourrait être exploitée pour donner naissance à de nouvelles restrictions de VAL($\texttt{QPROP}_{PS}$) qui sont calculatoirement plus faciles (selon les hypothèses habituelles de la théorie de la complexité).

Dans le même ordre d'idées, une perspective est d'élargir notre investigation à l'étude de restrictions de VAL($\texttt{QPROP}_{PS}$) basées sur les quantifications. Une première approche esquissée dans [8] a concerné les restrictions sur le nombre d'alternances des quantifications des fragments des impliqués et des impliquants premiers. À l'instar de ce que nous avons fait pour ces derniers, il s'agit d'étudier des restrictions portant conjointement sur la nature du préfixe et de la matrice des instances. En effet, d'un point de vue théorique, ne considérer que des restrictions sur le préfixe (i.e., sans aucune exigence sur la matrice) ne semble pas aussi intéressant. Alors que le fait de limiter le nombre d'alternances de quantificateurs dans le préfixe provoque immédiatement une diminution de la complexité de VAL($\texttt{QPROP}_{PS}$) de **PSPACE** à un niveau donné dans la hiérarchie polynomiale, cela ne conduit pas à la traitabilité (sous les hypothèses habituelles de la théorie de la complexité) ; ainsi, dans le cas limite où aucune alternance de quantificateurs n'apparaît dans le préfixe, VAL($\texttt{QPROP}_{PS}$) se réduit à `SAT` ou à `UNSAT` (dépendant de la nature des quantifications des variables), et aucun de ces problèmes ne peut vraisemblablement appartenir à **P** si aucune supposition sur la matrice n'est faite. Bien entendu, cela ne signifie pas que la façon dont les quantifications sont traitées n'a pas d'impact sur l'efficacité des prouveurs `QBF`. Par exemple, une `QBF` Σ de la forme $\forall\{x_1,\ldots,x_{n-1}\}\exists\{x_n\}.\phi$ où ϕ est une formule `CNF` peut être résolue en temps quadratique (construire une représentation `CNF` de $\exists\{x_n\}.\phi$ en utilisant la résolution pour oublier x_n dans ϕ, réduire ensuite toutes les clauses résultantes en y supprimant tout littéral construit à partir de $\{x_1,\ldots,x_{n-1}\}$; Σ est valide si et seulement si la formule `CNF` résultante ne contient pas la clause vide). Pourtant, les prouveurs `QBF` qui traitent les quantificateurs du plus externe au plus imbriqué peuvent requérir un temps exponentiel pour résoudre des formules de ce type. Une étude plus approfondie de l'interaction entre les restrictions basées sur le préfixe et celles basées sur la matrice constitue donc une perspective de recherche de ce travail.

Annexe

La figure 1 décrit le graphe d'inclusion des fragments propositionnels complets considérés dans cet article.

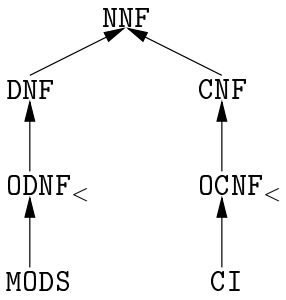

Figure 1: Graphe d'inclusion des fragments propositionnels complets considérés. Un arc $L_1 \rightarrow L_2$ signifie que L_1 est un sous-ensemble propre de L_2.

Remerciements

Merci aux relecteurs anonymes pour leurs commentaires. Ce travail a bénéficié du soutien de l'IUT de Lens et de l'Université d'Artois.

References

[1] B. Aspvall, M. Plass, and R. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8:121–123, 1979. Erratum: Information Processing Letters 14(4): 195 (1982).

[2] G. Audemard and L. Saïs. SAT based BDD solver for Quantified Boolean Formulas. In *ICTAI'04*, pages 82–89, 2004.

[3] M. Benedetti. sKizzo: a QBF Decision Procedure based on Propositional Skolemization and Symbolic Reasoning. Technical Report 04–11–03, ITC-Irst, Institute for Scientific and Technological Research, 2004.

[4] P. Besnard, T. Schaub, H. Tompits, and S. Woltran. Paraconsistent Reasoning via Quantified Boolean Formulas, I: Axiomatising Signed Systems. In *JELIA'02*, pages 320–331, 2002.

[5] M. Cadoli, A. Giovanardi, and M. Schaerf. An algorithm to evaluate Quantified Boolean Formulae. In *AAAI'98*, pages 262–267, 1998.

[6] S. Coste-Marquis, D. Le Berre, and F. Letombe. A branching heuristics for quantified renamable Horn formulas. In *SAT'05*, volume 3569 of *LNCS*, pages 393–399, 2005.

[7] S. Coste-Marquis, D. Le Berre, F. Letombe, and P. Marquis. Fragments propositionnels pour la compilation de connaissances

et formules booléennes quantifiées. In *RFIA'06*, 2005. (electronic proceedings: http://www.afrif.asso.fr/archive/rfia2006).

[8] S. Coste-Marquis, D. Le Berre, F. Letombe, and P. Marquis. Propositional fragments for knowledge compilation and quantified boolean formulae. In *AAAI'05*, pages 288–293, 2005.

[9] S. Coste-Marquis, D. Le Berre, F. Letombe, and P. Marquis. Complexity results for quantified boolean formulae based on complete propositional languages. *Journal on Satisfiability, Boolean Modeling and Computation*, 1:61–88, 2006.

[10] U. Egly, T. Eiter, H. Tompits, and S. Woltran. Solving advanced reasoning tasks using Quantified Boolean Formulas. In *AAAI'00*, pages 417–422, 2000.

[11] H. Fargier, J. Lang, and P. Marquis. Propositional logic and one-stage decision making. In *KR'00*, pages 445–456, 2000.

[12] R. Feldmann, B. Monien, and S. Schamberger. A distributed algorithm to evaluate quantified boolean formula. In *AAAI'00*, pages 285–290, 2000.

[13] I. P. Gent and A. G. D. Rowley. Solving 2-CNF Quantified Boolean Formulae using variable assignment and propagation. In *QBF workshop at SAT'02*, pages 17–25, 2002.

[14] M. GhasemZadeh, V. Klotz, and C. Meinel. Zqsat: a qsat solver based on zero-suppressed binary decision diagrams. Technical report, FB-IV Informatik, University of Trier, 2004.

[15] E. Giunchiglia, M. Narizzano, and A. Tacchella. Backjumping for Quantified Boolean Logic satisfiability. In *IJCAI'01*, pages 275–281, 2001.

[16] J. J. Hébrard. A Linear Algorithm for Renaming a Set of Clauses as a Horn Set. In *Theoretical Computer Science*, volume 124, pages 343–350, 1994.

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

[18] H. Kleine-Büning, K. Subramani, and X. Zhao. On boolean models for quantified boolean Horn formulas. In *SAT'03*, volume 2919 of *LNCS*, pages 93–104, 2003.

[19] J. Lang, P. Liberatore, and P. Marquis. Propositional independence - formula-variable independence and forgetting. *Journal of Artificial Intelligence Research*, 18:391–443, 2003.

[20] D. Le Berre, M. Narizzano, L. Simon, and A. Tacchella. The second QBF solvers evaluation. In *SAT'04*, volume 3542 of *LNCS*, pages 376–392, 2004.

[21] D. Le Berre, L. Simon, and A. Tacchella. Challenges in the QBF arena: the SAT'03 evaluation of QBF solvers. In *SAT'03*, volume 2919 of *LNCS*, pages 468–485, 2003.

[22] R. Letz. Lemma and model caching in decision procedures for Quantified Boolean Formulas. In *Tableaux'02*, pages 160–175, 2002.

[23] M. Narizzano, L. Pulina, and A. Tacchella. The third QBF solvers comparative evaluation. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:143–162, 2006.

[24] G. Pan, U. Sattler, and M.Y. Vardi. BDD-based decision procedures for K. In *CADE'02*, pages 16–30, 2002.

[25] G. Pan and M.Y. Vardi. Optimizing a BDD-based modal solver. In *CADE'03*, pages 75–89, 2003.

[26] G. Pan and M.Y. Vardi. Symbolic decision procedures for QBF. In *CP'04*, pages 453–467, 2004.

[27] J. Rintanen. Constructing conditional plans by a theorem-prover. *Journal of Artificial Intelligence Research*, 10:323–352, 1999.

[28] J. Rintanen. Improvements to the Evaluation of Quantified Boolean Formulae. In *IJCAI'99*, pages 1192–1197, 1999.

[29] J. Rintanen. Partial implicit unfolding in the Davis-Putnam procedure for Quantified Boolean Formulae. In *QBF workshop at IJCAR'01*, pages 84–93, 2001.

[30] L. Zhang and S. Malik. Towards a symetric treatment of satisfaction and conflicts in quantified boolean formula evaluation. In *CP'02*, pages 200–215, Ithaca, NY, USA, Sept. 2002.