Extraire des Certificats des Formules Booléennes Quantifiées

Marco Benedetti

Extraire des Certificats des Formules Booléennes Quantifiées *

Marco Benedetti

Université d'Orléans – LIFO
BP 6759
F-45067 Orléans Cedex 2

Marco.Benedetti@univ-orleans.fr

Abstract

En donnant des moyens d'exprimer l'alternation de quantificateurs existentiels et universels, les langages quantifiés permettent de poser formellement des questions dont la réponse est une stratégie. Par exemple, on peut encoder les règles d'un jeu comme un ensemble de contraintes, puis demander l'existence d'une stratégie gagnante (ou au moins non-perdante) pour ce jeu.

Dans le cas de formules booléennes quantifiées, une telle stratégie, si elle existe, est représentée par un *certificat de satisfiabilité*, qui est une représentation compacte d'un des modèles de la formule et qui peut être utilisé pour donner une preuve de satisfiabilité indépendante du solveur. Etant donnée la nature intrinsèque des formules quantifiées, de tels certificats demandent beaucoup de soin pour être extraits efficacement, représentés de façon compacte et pour que l'on puisse aisément leur poser des requêtes. Nous montrons comment résoudre ces problèmes.

1 Introduction

Le terme "certificat" a un sens relativement général, venant à l'origine de la reconnaissance des langages et de la théorie de la complexité. Une fois *vérifié*, un certificat prouve que la chaîne à laquelle il se réfère appartient effectivement à un langage auquel on s'intéresse. Appliqué à la logique, le terme désigne n'importe quel moyen de montrer une preuve de (in-)satisfibilité pour une proposition, autre qu'une approche déductive complète par réfutation. En substance, nous vérifions qu'une formule logique donnée appartient au langage des propositions (in-)satisfiables.

Le *certificat de satisfiabilité* (sat-certificat) le plus naturel pour une formule est une représentation explicite d'un de ses *modèles*. Une formule est en effet satisfiable si et seulement si un de ses modèle s'évalue à vrai. La validité d'un certificat peut être vérifiée par quiconque est compétent avec la *technique d'évaluation* de la logique (les capacités déductives sont ici inutiles), et ceci indépendamment de comment celui-ci a été obtenu.

Dans ce papier, nous nous intéressons aux sat-certificats pour les *formules booléennes quantifiées* (QBFs). De tels certificats n'ont jamais été proposés ou utilisés jusqu'ici pour de nombreuses raisons. Premièrement, la nature intrinsèque des QBFs confère à leurs modèles une structure en forme d'arbre, dont la représentation explicite peut devenir impraticable. Deuxièmement, il existe des arguments théoriques qui rendent peu probables la découverte de procédures de vérification polynômiales (la satisfiabilité des QBFs est PSPACE-complète [11]). Enfin, les solveurs QBF actuels trouvent soit impraticable, soit non direct de collecter toutes les informations nécessaires pour construire un modèle. En conséquence, les modèles des QBFs ne possèdent aucune représentation communément acceptée, et tous les solveurs actuels retournent juste un petit peu plus qu'une réponse sat/unsat.

Malgré ces problèmes, les sat-certificats pour les QBFs sont extrèmement désirables à cause de leur bénéfices potentiels pour les applications et les solveurs. Par exemple, un certificat est un moyen concluant pour décider de réponses en conflit données par différents solveurs sur la même instance. Clairement, cet évènement ne révèle pas de problème plus profond qu'un bug dans l'implantation,

*Ce travail a été publié dans les actes de IJCAI 2005 avec le titre "Extracting Certificates from Quantified Boolean Formulas"

ce qui pourrait sembler être un problème mineur. Ceci arrive toutefois relativement souvent, et, aussi longtemps que l'on considérera les solveurs comme des boîtes noires, une preuve de satisfiabilité est le seul moyen de connaître la vérité. Nous citons [8] :

> *La question de savoir comment vérifier la réponse des solveurs QBF de manière effective est toujours sans réponse* [...] *la question de savoir ce qu'est un bon certificat de satisfiabilité/insatisfiabilité* [...] *est toujours ouverte. Ce point n'est pas seulement une question pour l'évaluation des QBFs, mais aussi pour leur implantation:* [...] *nous avons des problèmes de correction avec 4 solveurs QBF.* [...]

Ainsi, un certificat est bien plus qu'un moyen d'assurer la satisfiabilité : il peut être *inspecté* pour extraire de la sémantique de la formule sous-jacente. Ceci est d'une importance capitale pour les applications, où les certificats ajoutent de l'information importante à une simple réponse sat/unsat. Par exemple, la réponse sat à l'encodage propositionnel (PROP) d'une propriété (ou de sa négation) d'un circuit logique signifie que ce circuit est erroné relativement à cette propriété. Mais il faut un certificat pour trouver un scénario dans lequel ce défaut se montre. Contrairement aux certificats pour les QBFs, ceux de PROP sont faciles à représenter et à vérifier, et donc ils ont de nombreuses applications.

L'importance des certificats grandit avec le champ d'application de la logique. Dans cette perspective, les QBFs sont un cas notable ayant de nombreuses applications. Tout problème pouvant être énoncé comme un jeu fini à deux joueurs peut être modélisé en QBF. Un exemple significatif est le fameux jeu de "Puissance 4". Il est connu que le joueur qui joue en premier gagne toujours. Les règles du jeu et l'existence d'une stratégie gagnante peuvent être encodées comme une instance de QBF [7], dont la réponse doit être sat. Quelle est la stratégie gagnante alors ? Un certificat montrerait cette information : le premier joueur gagnerait juste en inspectant le certificat à chaque étape, quel que soit le coup de son adversaire.

Le point intéressant est que de nombreuses applications réelles peuvent être modélisées comme un jeu à deux joueurs : le model-checking non-borné pour les systèmes à états finis [10], et le conformant planning [9], juste pour citer deux exemples, ont des formulations en QBF naturelles.

Dans la suite de cet article, après une brève introduction aux QBFs et à leurs modèles (Section 2), nous présentons une représentation indépendante du solveur pour les sat-certificats QBF (Section 3). Comme prévu, nous pouvons décrire comment les vérifier (Section 4) *avant* d'aborder la tâche plus complexe de leur extraction (Section 5). Nous concluons en discutant de notre approche et de ses extensions (Section 6).

2 Les QBFs et leurs modèles

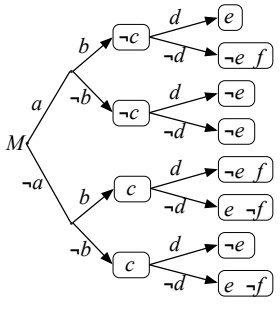
Sans perte de généralité, nous considérons des QBFs en *forme normale conjonctive prénexe* (CNF). Elles consistent en un *préfixe* contenant un nombre arbitraire d'alternations de variables quantifiées existentiellement et universellement, suivi d'une *matrice*, qui est une conjonction de clauses. Par exemple :

$$\begin{array}{l}\forall a\forall b\exists c\forall d\exists e\exists f.(\neg b\vee e\vee f)\wedge(a\vee c\vee f)\wedge(a\vee d\vee e)\wedge\\ \quad(\neg a\vee\neg b\vee\neg d\vee e)\wedge(\neg a\vee b\vee\neg c)\wedge(\neg a\vee\neg c\vee\neg f)\wedge\\ \quad(a\vee\neg d\vee\neg e)\wedge(\neg a\vee d\vee\neg e)\wedge(a\vee\neg e\vee\neg f)\end{array} \quad (1)$$

Etant donnée une QBF F, on note $\widetilde{F}$ sa matrice, $var_\exists(F)$ ($var_\forall(F)$) l'ensemble de ses variables existentiellement (universellement) quantifiées, et par $var_\forall(F,e) \subseteq var_\forall(F)$ l'ensemble des variables universelles qui précèdent (ou *dominent*) $e \in var_\exists(F)$ dans le préfixe (on pose $\delta(e) \doteq |var_\forall(F,e)|$).

Etant donnée une matrice $\widetilde{F}$, la formule $\widetilde{F} * l$ est la CNF obtenue en affectant le littéral l, c'est à dire en retirant de F chaque littéral $\neg l$ et chaque clause contenant l. Cette notation est aisément étendue à un ensemble de littéraux. Une matrice $\widetilde{F}$ est satisfaite par un ensemble de littéraux M (on écrit $M \models \widetilde{F}$) si $\widetilde{F} * M$ est la formule vide.

L'alternation des quantificateurs dans le préfixe nous guide pour étendre cette notion de satisfiabilité des matrices aux QBFs. Par exemple, le problème de satisfaction (1) demande si pour chaque combinaison (cohérente) possible de littéraux sur a et b, il existe un moyen de choisir un littéral sur c tel que, pour chaque affectation possible de d, deux littéraux e et f existent tels que le résultat satisfait la matrice. Donc, un modèle pour une QBF est un ensemble de fonctions $|var_\exists(F)|$, chacune spécifiant le littéral qui doit être choisi (si c'est le cas) pour la variable existentielle e, en fonction des variables universelles qui la dominent.

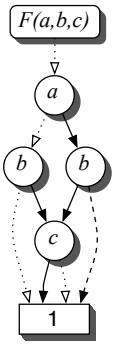


Afin de représenter toutes ces fonctions d'un coup, nous pouvons utiliser un arbre étiqueté, juste comme celui qui est à gauche qui représente un modèle pour la formule (1). Si nous appelons *hypothèse universelle* tout ensemble cohérent de littéraux universels (ou de façon équivalente, toute affectation $[u_1 = \psi_1, \ldots, u_n{=}\psi_n]$ des variables universelles avec $\psi_i \in \mathfrak{B} = \{0,1\}$), nous pouvons dire informellement qu'une structure d'arbre comme celle ci-dessus est un modèle pour une QBF avec matrice $\widetilde{F}$ si et seulement si pour chaque hypothèse universelle U, l'ensemble des littéraux existentiels collectés le long des branches de U satisfait $\widetilde{F} * U$ (voir [6] pour plus de détails).

3 Représentation des certificats

Un modèle d'une QBF peut être représenté *explicitement* en employant des structures de données telles que les arbres ou les tables de vérité. Ou bien, on peut exiger plus de *compacité*, au prix du coût de manipuler une représentation *implicite*[1] qui requiert du calcul pour produire des valeurs.

Un certificat idéal doit être compact (facile à manipuler) *et* explicite (facile à vérifier et à questionner). Un compromis qui a du succès est obtenu en employant des *diagrammes de décisions binaires* [5]. Nous en considérons la version *réduite ordonnée* (ROBDDs, ou juste BDDs par la suite) avec *arcs complémentés*. Un BDD $\mathcal{E}$ représentant une fonction totale $F(u_1, u_2, \ldots, u_n)$ depuis $\mathfrak{B}^n$ vers $\mathfrak{B}$ est un graphe dirigé acyclique ayant une racine (étiquetée par F) et un puits (étiqueté par "1"). Chaque noeud interne est étiqueté par une variable de $U = \{u_1, u_2, \ldots u_n\}$, et a toujours deux enfants, l'un attaché à l'arc *then*, l'autre à l'arc *else*. L'arc *else* peut être ou non complémenté. Un chemin unique depuis la racine vers le puits est décrit en affectant une valeur à chaque variable de U : l'arc *then* est choisi pour les variables affectées à "1", sinon l'arc *else* est suivi. La fonction F représentée par $\mathcal{E}$ s'évalue à 1 sur $\langle \psi_1, \psi_2, \ldots, \psi_n \rangle \in \mathfrak{B}^n$ si et seulement si un nombre pair d'arcs complémentés est franchi le long du chemin défini par $\psi_1, \psi_2, \ldots, \psi_n$. Par exemple, considérons le BDD ci-contre, où les arcs pleins désignent les arcs *then*



et les arcs en tirets (en pointillés) représentent les arcs *else* réguliers (complémentés). Il représente une fonction binaire $F(a, b, c)$ de trois variables a, b et c. On peut lire, par exemple, que $F(0, 1, 1) = 1$ et $F(1, 1, 1) = 0$. La fonction représentée peut être écrite comme $F = b \wedge (a \vee c) \wedge (\neg a \vee \neg c)$. Dans une interprétation théorie des ensembles, ce BDD représente l'ensemble ayant F comme fonction caractéristique. Dans notre cas, c'est l'ensemble $\{\langle 0, 1, 1\rangle, \langle 1, 1, 0\rangle\}$ où F s'évalue à 1.

Les BDDs que nous utilisons sont *ordonnés* et *réduits*: le même ordre sur les variables est suivi sur chacun des chemins et il n'existe pas deux noeuds représentant le même ensemble, donc chaque fonction a exactement une représentation *canonique*. De plus, la version avec arcs complémentés est telle que l'ensemble $\overline{S}$ est décrit avec les même noeuds que S (désigné par un arc complémenté).

La technique de représentation des ensembles par des BDD est considérée comme *symbolic* dans le sens qu'elle évite l'énumération explicite des éléments de l'ensemble et lui substitue une technique plus abstraite de calcul de la fonction caractéristique basée sur les diagrammes. Une telle représentation peut être exponentiellement plus concise qu'une représentation explicite (voir [12]), et toutes les opérations sur les ensembles/fonctions qu'elles représentent (union/disjonction, intersection/conjonction, etc.) peuvent être effectuées en manipulant le BDD associé [5]. Avec un petit abus de notation, nous considérons les BDDs comme s'ils étaient les ensembles qu'ils représentent. Par exemple, $x \in \mathcal{E}$ est un élément dans le sous-ensemble de $\mathfrak{B}^n$ qui est $\mathcal{E}$.

Quand nous manipulons des collections de BDDs, la canonicité est préservée sur l'ensemble des noeuds. Ceci permet de partager des informations structurelles entre les diagrammes. Un BDD dans un tel ensemble de diagrammes interconnectés — appelé une forêt — est identifié par un arc (complémenté) pointant sur sa racine.

Définition 3.1 (sat-certificat pour QBF , validité) *Un sat-certi-ficat pour une QBF F avec $var_\exists(F) = \{e_1, \ldots, e_m\}$, $var_\forall(F) = \{u_1, \ldots, u_n\}$, et $\delta_i = \delta(e_i)$ est une forêt de BDDs contenant deux racines $\langle \mathcal{E}_i^+, \mathcal{E}_i^- \rangle$ pour chaque i dans $[1, m]$. A la fois $\mathcal{E}_i^+$ et $\mathcal{E}_i^-$ sont définis sur $var_\forall(F, e_i)$=$\{u_1, \ldots, u_{\delta_i}\}$. Le certificat*

$$\mathcal{C}(F) = [\langle \mathcal{E}_1^+, \mathcal{E}_1^- \rangle, \langle \mathcal{E}_2^+, \mathcal{E}_2^- \rangle, \ldots, \langle \mathcal{E}_m^+, \mathcal{E}_m^- \rangle]$$

est cohérent *quand $\forall i \in [1, m]$ il est vrai que $\mathcal{E}_i^+ \cap \mathcal{E}_i^- = \emptyset$. Il est* valide *pour F quand pour tout $\langle \psi_1, \ldots, \psi_n \rangle \in \mathfrak{B}^n$ la formule $\widetilde{F}_{[u_1=\psi_1,\ldots,u_n=\psi_n]}$ est satisfaite par $\{e_i = s^{(i)}(\psi_1, \ldots, \psi_{\delta_i}), i \in [1, m]\}$, où les fonctions $s^{(i)} : \mathfrak{B}^{\delta_i} \to \mathfrak{B}$ sont définies par*

$$s^{(i)}(\psi_1, \ldots, \psi_{\delta_i}) = \begin{cases} 1 & \text{if } \langle \psi_1, \ldots, \psi_{\delta_i} \rangle \in \mathcal{E}_i^+ \\ 0 & \text{if } \langle \psi_1, \ldots, \psi_{\delta_i} \rangle \in \mathcal{E}_i^- \\ undef. & \text{sinon} \end{cases}$$

En résumé, un sat-certificat est une représentation compacte mais explicite des dépendances qui doivent exister entre les variables existentielles (dépendantes) et universelles (indépendantes) de façon à satisfaire la matrice quelles que soient les hypothèses universelles.

Lemme 3.1 *Si $\mathcal{C}(F)$ est valide pour une QBF F, alors F est satisfiable. Toute QBF satisfiable a au moins un certificat valide.*

Proof sketch. Une QBF est satisfiable ssi elle a au moins un modèle, c.a.d. ssi l'on trouve au moins une structure arborescente comme celle décrite en Section 2 telle que, pour chaque affectation $U = [u_1 = \psi_1, \ldots, u_n = \psi_n]$ des variables universelles, l'ensemble des littéraux existentiels collectés le long du chemin U satisfait $\widetilde{F} * U$. Si l'on a un certificat cohérent C pour F, on insère le littéral e_i dans le label du noeud atteint en suivant le chemin $u_1 = \psi_1, \ldots, u_{\delta_i} =$

[1] Dans [6] apparaissent des formules propositionnelles et des QBFs avec variables libres. Une représentation *implicite* n'est pas envisagée car les auteurs s'intéressent à caractériser des classes de modèles et de formules.

ψ_{δ_i} ssi $\langle \psi_1, \ldots, \psi_{\delta_i} \rangle \in \mathcal{E}_i^+$ (et, de façon duale, $\neg e_i$ apparaît dans l'étiquelle ssi $\langle \psi_1, \ldots, \psi_{\delta_i} \rangle \in \mathcal{E}_i^-$). Par construction, si le certificat est valide relativement à la notion de validité donnée en Definition 3.1, la structure obtenue est un modèle. □

Un sat-certificat valide pour (1) est donné en Figure 1.

4 Vérification de certificat

La première chose que l'on veut faire avec un certificat cohérent $\mathcal{C}$ pour F est de vérifier sa validité. On vérifie que l'on satisfait toujours la matrice en choisissant les valeurs de vérité pour les variables existentielles en suivant les suggestions du certificat.

Une technique facile mais hélas impraticable serait de vérifier que $\mathcal{M}$ produit une affectation satisfiable sous toutes les hypothèses universelles possibles. Heureusement, la nature symbolique du certificat nous aide à réaliser une vérification plus efficace, clause par clause et basée sur le BDD. Nous allons utiliser le ou exclusif "$\otimes$" pour construire des littéraux à partir des variables ($\varphi \otimes v$ signifie v quand $\varphi = 0$, et $\neg v$ quand $\varphi = 1$).

Lemme 4.1 *L'algorithme* `checkValidity` *répond* TRUE *sur* $\langle F, \mathcal{C} \rangle$ *si et seulement si* $\mathcal{C}$ *est un certificat valide pour* F.

Considérons, par exemple, la clause $\neg u_1 \vee e_1 \vee u_2 \vee \neg e_2 \vee e_3$ avec le préfixe $\forall u_1 \exists e_1 \forall u_2 \exists e_2 \exists e_3$. Les seules hypothèses universelles pertinentes pour cette clause sont celles qui affectent à la fois $u_1 = 1$ et $u_2 = 0$: toutes les autres satisfont immédiatement les clauses via un des littéraux universels. Donc, il reste à vérifier que, sous l'affectation $[u_1 = 1, u_2 = 0]$, au moins un des trois littéraux restants est vrai, c.a.d. que chaque hypothèse universelle contenant $u_1 = 1$, $u_2 = 0$ arrive au "1" par au moins un chemin parmi $\mathcal{E}_1^+$ (pour e_1), $\mathcal{E}_2^-$ (pour $\neg e_2$), et $\mathcal{E}_3^+$ (pour e_3). Ceci est une vérification en deux étapes: Premièrement, nous collectons les hypothèses universelles $\mathcal{E} = \mathcal{E}_1^+ \cup \mathcal{E}_2^- \cup \mathcal{E}_3^+$ avec lesquelles la clause est satisfaite par un littéral existentiel. Ensuite, nous vérifions que toutes les hypothèses $\overline{\mathcal{E}}$ (dans lesquelles aucun littéral existentiel ne satisfait la

Fonction `checkValidity` (*QBF F, certificate* $\mathcal{C}$)

Let $var_\exists(F)$ be $\{e_1, \ldots, e_m\}$;
Let $var_\forall(F)$ be $\{u_1, \ldots, u_n\}$;
Let $\mathcal{C}$ be $[\langle \mathcal{E}_1^+, \mathcal{E}_1^- \rangle, \langle \mathcal{E}_2^+, \mathcal{E}_2^- \rangle \ldots, \langle \mathcal{E}_m^+, \mathcal{E}_m^- \rangle]$;
forall $\Gamma \in F$ **do**
  Let Γ be $\psi_1 \otimes u_{i_1} \vee \cdots \vee \psi_h \otimes u_{i_h} \vee$
    $\phi_1 \otimes e_{j_1} \vee \cdots \vee \phi_k \otimes e_{j_k}$;
  $\mathcal{E} \leftarrow (\cup_{\phi_i = 0} \mathcal{E}_{j_i}^+) \cup (\cup_{\phi_i = 1} \mathcal{E}_{j_i}^-)$;
  if $\overline{\mathcal{E}}_{[u_{i_1} = \overline{\psi_1}, \ldots, u_{i_h} = \overline{\psi_h}]} \neq \emptyset$ **then return** FALSE;
return TRUE;

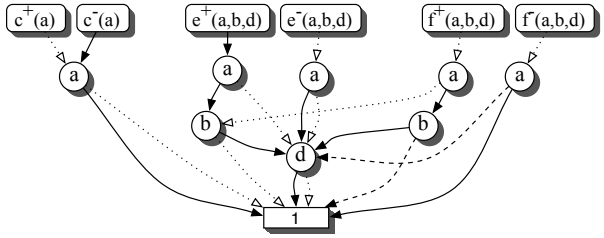


Figure 1: Un sat-certificat BDD pour les QBF (1).

clause) affecte soit $u_1 = 1$, ou $u_2 = 0$, ou les deux, et donc que la clause est satisfaite par un littéral universel.

Le sens d'une vérification succès est double: on est assuré que la formule est sat, et que le certificat code un modèle. Réciproquement, la vérification échoue soit si le certificat est invalide, soit si la formule est unsat(on ne peut pas dire dans quel cas on est). Le test de validité est un problème coNP-complet [6].

5 Extraction de certificat

Pour que l'extraction de certificat soit symbolique comme notre cetificat l'est, nous devons travailler sur une représentation en BDD du problème. Une telle représentation est décrite (Section 5.1), suivie d'une description des relations avec les certificats (Section 5.2). Ensuite, une procédure en deux temps permettant de (a) évaluer l'instance et (b) construire un certificat basé sur l'étape d'évaluation est présentée (Section 5.4).

5.1 Produire des formules symboliques par skolémisation

Le théorème de Skolem montre comment transformer toute formule de *logique du premier ordre* (FOL) F en une formule *skolemisée* $Sk(F)$ ayant deux propriétés : (1) $Sk(F)$ ne contient aucun quantificateur existentiel, et (2) $Sk(F)$ et F sont equisatisfiables. Les quantificateurs existentiels sont supprimés en remplaçant les variables qu'ils touchent par des *fonctions de Skolem* dont les domaines de définition sont choisis de façon à préserver la satisfiabilité. Dans la forme *extérieure* de la skolemisation, la fonction remplaçant $e \in var_\exists(F)$ dépend des variables universelles $var_\forall(F, e)$ qui couvrent e (pour les formules prenexes : toutes les variables universelles à gauche de e dans le préfixe).

Les solveurs basés sur la skolémisation remplacent la formule originale F par l'instance equisatisfiable $Sk(F)$. Cette instante n'est plus propositionnelle. Cependant, il est possible de capturer sa sémantique sans dépasser le pouvoir d'expression de la logique propositionnelle en gérant de manière explicite les valeurs de vérité des (interprétations des) fonctions de Skolem à chaque point de leur domaine de définition, comme montré dans [2].

Le prix à payer est une (possible) explosion exponentielle de la taille du problème. les BDD s'avèrent alors précieux pour garder sous contrôle ce problème d'explosion de l'espace : on manipule en réalité une *formule symbolique*, c'est-à-dire une représentation compacte en BDD de l'instance propositionnelle représentant la *définition* de l'ensemble des termes de Skolem de $Sk(F)$.

Notons par $\Psi|_k$ le préfixe de longueur k bits de $\Psi \in \mathfrak{B}^n, n \geq k$. Cette notion est étendue aux ensembles : $\mathcal{I}|_k = \{\Psi|_k . \Psi \in \mathcal{I}\}$.

Définition 5.1 (Formule symbolique) *Une formule symbolique $\mathcal{F}$ est une représentation en BDD d'une instance en CNF. Elle consiste en un* préfixe symbolique $[e_1]_{\delta_1} \dots [e_m]_{\delta_m}$ *sur les variables* $var(\mathcal{F}) = \{e_1, \dots, e_m\}$, *avec* $0 \leq \delta_1 \leq \dots \leq \delta_m$, *suivi d'une* matrice symbolique $\widetilde{\mathcal{F}}$, *c'est-à-dire une conjonction de* clauses symboliques. *Une clause symbolique $\Gamma_\mathcal{I}$ est composée d'un ensemble consistant $\Gamma = [\varphi_1 \otimes e_{i_1}, \dots, \varphi_h \otimes e_{i_h}]$ de littéraux sur $var(\mathcal{F})$, et d'un sous-ensemble (en BDD) $\mathcal{I}$ de $\mathfrak{B}^{\delta(\Gamma)}$, $\delta(\Gamma) \doteq \max_{l \in \Gamma} \delta(l)$. La CNF représentée par $\mathcal{F}$ est appelée* expansion propositionnelle *de $\mathcal{F}$. Elle a pour variables $\{s^{(i)}_\Phi, i \in [1,m], \Phi \in \mathfrak{B}^{\delta(e_i)}\}$, et est définie par $Prop(\mathcal{F}) \doteq \wedge_{\Gamma_\mathcal{I} \in \mathcal{F}} Prop(\Gamma_\mathcal{I})$, où*

$$Prop(\Gamma_\mathcal{I}) \doteq \bigwedge_{\Psi \in \mathcal{I}} \varphi_1 \otimes s^{(1)}_{\Psi|_{\delta_1}} \vee \dots \vee \varphi_m \otimes s^{(m)}_{\Psi|_{\delta_m}}$$

Les formules symboliques héritent de la sémantique de leurs expansions propositionnelles (que l'on appelle aussi *ground*-équivalent). Notamment, un ensemble consistant $\mathcal{M} = \{[l_1]_{\mathcal{I}_1}, \dots, [l_m]_{\mathcal{I}_m}\}$ de littéraux symboliques satisfait $\mathcal{F}$ ($\mathcal{M} \models \mathcal{F}$) ssi son expansion $Prop(\mathcal{M}) = \cup_{[l]_\mathcal{J} \in \mathcal{M}} Prop([l]_\mathcal{J})$ satisfait $Prop(\mathcal{F})$.

Définition 5.2 (Skolémisation symbolique) *La skolémisation symbolique $SymbSk(F)$ d'une QBF F avec $var_\exists(F) = \{e_1, \dots, e_m\}$ est une formule symbolique dont le préfixe est $[e_1]_{\delta(e_i)} \cdots [e_m]_{\delta(e_m)}$, ayant une clause symbolique $[l_1, \dots, l_h]_\mathcal{I}$ pour chaque clause $\lambda \in F$, où $\{l_1, \dots, l_h\}$ sont les littéraux existentiels de Λ, $\{\varphi_1 \otimes e_{i_1}, \dots, \varphi_h \otimes e_{i_h}\}$ sont les litteraux universels de λ, et $\mathcal{I} = \{\langle \psi_i, \dots, \psi_k \rangle \in \mathfrak{B}^k \mid \forall j . \psi_{i_j} \neq \varphi_j\}$, $k = \delta(\Lambda)$.*

Par exemple, la skolémisation symbolique de (1) est donnée en Figure 2. Pour l' essentiel, une skolémisation symbolique $\mathcal{F} = SymbSk(F)$ est une représentation compacte d'une instance purement existentielle $Prop(\mathcal{F})$ vérifiant la propriété clé.

Théorème 5.1 *Pour toute QBF F, on a $Prop(SymbSk(F)) \overset{sat}{\equiv} F$.*

Proof sketch. En appliquant la skolémisation extérieure telle que décrite dans [2], on transforme une instance QBF F en une formule equisatisfiable purement universelle : on substitue toutes les variables existentielles v dominées par $\{u_1, u_2, \dots, u_n\} \subseteq var_\forall(F)$ avec une *fonction de Skolem* $s^v(u_1, u_2, \dots, u_n)$. Par exemple, l'instance (1) avec la matrice $\widetilde{N}$, est equisatisfiable à

$$\forall a \forall b \forall d.\ \widetilde{N}[s^c(a,b)/c,\ s^e(a,b,d)/e,\ s^f(a,b,d)/f] \quad (2)$$

Donc, on encode la définition des termes de Skolem introduits sous forme propositionnelle, ce qui renvoie une propriété remarquable : ils sont tous fonctions de $\mathfrak{B}^n$ dans $\mathfrak{B}$ (pour $n \geq 0$ donné), et donc, ils sont entièrement spécifiés par 2^n paramètres booléens, et ont une représentation directe en CNF. Par exemple, si on note $\{s^c_{\alpha\beta}, \langle \alpha, \beta \rangle \in \mathfrak{B}^2\}$ les quatre paramètres booléens représentant les valeurs de vérité de s^c sur $\langle x, y \rangle$, pour $s^c(a,b)$, on obtient la *skolémisation propositionnelle* $Sk(s^c) \equiv s^c(a,b)$ suivante :

$$(a \vee b \vee s^c_{00}) \wedge (a \vee \neg b \vee s^c_{01}) \wedge (\neg a \vee b \vee s^c_{10}) \wedge (\neg a \vee \neg b \vee s^c_{11})$$

En remplaçant chaque $v \in var_\exists(F)$ par $Sk(s^v)$ on obtient $Sk(F)$, où $\exists \underline{s}^c$ signifie $\exists s^c_{00} \exists s^c_{01} \exists s^c_{10} \exists s^c_{11}$, et de même pour les autres fonctions. Cette formule peut être facilement écrite—clause par clause— en CNF. En distribuant les connecteurs, supprimant les clauses ayant des littéraux complémentaires, et éliminant les littéraux sur les quanificateurs universels, on obtient $2^{\delta(\Gamma)-m}$ clauses d'une clause QBF Γ ayant m littéraux universels. Par exemple, on obtient 2^{3-1}=4 clauses à partir de la clause $a \vee c \vee f$ de (1):

$$(s^c_{00} \vee s^f_{000}) \wedge (s^c_{00} \vee s^f_{001}) \wedge (s^c_{01} \vee s^f_{010}) \wedge (s^c_{01} \vee s^f_{011})$$

Toutes les clauses venant d'une clause QBF donnée mentionnent les mêmes fonction de Skolem. Les seules différences entre elles sont les index inscrits en indice. Cela permet de les écrire de manière compacte, en représentant les noms de fonction séparément des index. Par exemple, les quatres clauses ci-dessus peuvent être plus succintement représentées par$\Gamma_\mathcal{I} = [c,f]_{\{000,001,010,011\}}$, où $\mathcal{I} \subseteq \mathfrak{B}^{\delta(\Gamma)}$ contient un élément par clause, et où la i-ème composante de chaque $\Psi \in \mathcal{I}$ fait référence à la variable universelle u_i: Une fois $\Psi \in \mathcal{I}$ sélectionnée, chaque littéral l obtient son propre indice en *projetant* Ψ sur le sous-espace décrit par les $\delta(l)$ premières composantes notées $\Psi|_{\delta(l)}$. Par exemple, étant donné $\Psi = 010$, on a $\Psi|_{\delta(c)} = \Psi|_2$=01 et $\Psi|_{\delta(f)}$=$\Psi|_3$=010, et donc $[c,f]_{\{010\}}$=$s^c_{01} \vee s^f_{010}$. Cette propriété nous permet de récupérer le sens fondamental d'une clause factorisée à travers la fonction $Prop$. La représentation factorisée deviens *symbolique* dès que l'on représente et que l'on manipule des ensembles d'index via les BDD sur $var_\forall(F)$, obtenant ce faisant la représentation $SymbSk(F)$ (de taille linéaire). □

Par exemple, l'expansion propositionnelle de la formule de la figure 2 mène à une instance CNF équivalente à (1).

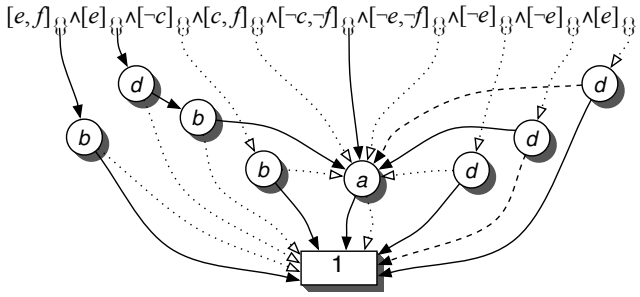


Figure 2: Skolémisation symbolique propositionnelle de (1).

5.2 Des modèles symboliques aux certificats

Le lien entre modèle symbolique et certificat est tellement fort que l'on passe très facilement de l'un à l'autre. Soient $\mathfrak{i}(\mathcal{M}, l) \doteq \cup_{[l]_\mathcal{I} \in \mathcal{M}} \mathcal{I}$ les index sur le littéral l mentionné dans un modèle symbolique $\mathcal{M}$.

Théorème 5.2 *Si $\mathcal{M} \models SymbSk(F)$, alors le certificat*

$$\mathcal{C}(\mathcal{M}) \doteq [\langle \mathcal{C}^+_{e_1}, \mathcal{C}^-_{e_1} \rangle, \langle \mathcal{C}^+_{e_2}, \mathcal{C}^-_{e_2} \rangle, \ldots, \langle \mathcal{C}^+_{e_m}, \mathcal{C}^-_{e_m} \rangle]$$

avec $\mathcal{C}^+_e = \mathfrak{i}(\mathcal{M}, e)$, et $\mathcal{C}^-_e = \mathfrak{i}(\mathcal{M}, \neg e)$ est valide pour F.

Preuve. La variable propositionnelle s^i_Ψ de $Prop(\mathcal{F})$, $\mathcal{F} = SymbSk(F)$ représente, pour tout $i \in [1, m]$ et pour tout $\Psi = \langle \psi_1, \ldots, \psi_k \rangle \in \mathfrak{B}^k$, $k = \delta(e_i)$, la valeur que la fonction de Skolem $s^{(i)}(u_1, \ldots, u_k)$ (introduite pour skolémiser $e_i \in var_\exists(F)$) prend en considérant l'hypothèse universelle $[u_1 = \psi_1, \ldots, u_k = \psi_k]$. Par construction, $\mathcal{F}$ représente la définition des termes de Skolem, c'est-à-dire toutes les contraintes mutuelles sur l'interprétation des termes auxquelles on doit satisfaire pour toujours satisfaire la matrice. Un modèle $\mathcal{M}$ pour $\mathcal{F}$ est une manière de satisfaire toutes ces contraintes en une seule fois: $s^{(i)}(\psi_1, \ldots, \psi_k)$ doit s'évaluer à 1 si $s^{(i)}_\Psi \in Prop(\mathcal{M})$, à 0 si $\neg s^{(i)}_\Psi \in Prop(\mathcal{M})$, et n'est pas contraint sinon. En comparant les fonctions résultantes

$$s^e(\psi_1, \ldots, \psi_k) = \begin{cases} 1 & \text{if } \langle \psi_1, \ldots, \psi_k \rangle \in \mathcal{C}^+_e \\ 0 & \text{if } \langle \psi_1, \ldots, \psi_k \rangle \in \mathcal{C}^-_e \\ indfini & \text{sinon} \end{cases}$$

avec la définition 3.1 (certificat valide), le théorème est démontré.

5.3 Evaluation des formules symboliques

Nous nous concentrons sur les procédures *solution-based*, car elles peuvent être étendues pour extraire des modèles *symboliquement*. On considère un solveur manipulant des clauses symboliques à travers des règles construites pour donner sur $\Gamma_\mathcal{I}$ le même résultat que donneraient les règles de base appliquées séparément sur chaque clause de $Prop(\Gamma_\mathcal{I})$.

- *Résolution*: Etant données deux clauses, l'une contenant une variable e et l'autre $\neg e$, on construit une conséquence nécessaire—appelée clause *résolvante*—composée de tous les littéraux des deux clauses originales, à l'exception de e et $\neg e$. Pour que la version symbolique produise au moins un résolvant par étape, on résout seulement sur les variables ayant la même profondeur universelle que toutes les clauses dans lesquelles elles apparaissent. Le résolvant de $\Gamma_\mathcal{I}$ et $\Gamma'_{\mathcal{I}'}$ sur e, avec $e \in \Gamma$, $\neg e \in \Gamma'$, $\delta(\Gamma) = \delta(\Gamma') = \delta(e)$, est[2]

$$(\Gamma \setminus \{e\} \cup \Gamma' \setminus \{\neg e\})_{\mathcal{I} \cap \mathcal{I}'} \tag{3}$$

  Si soit $\mathcal{I} \cap \mathcal{I}' = \emptyset$, soit Γ et Γ' partagent d'autres couples de littéraux complémentaires autres que e et $\neg e$, le résolvant représente un ensemble vide de clauses et est supprimé.

- *Subsomption*: $\Gamma_\mathcal{I}$ est subsumée par $\Gamma'_{\mathcal{I}'}$ quand toutes les clauses de base de $\Gamma_\mathcal{I}$ sont subsumées par les clauses de $\Gamma'_{\mathcal{I}'}$, c'est-à-dire quand $\Gamma' \subseteq \Gamma$ et $\mathcal{I} \subseteq \mathcal{I}'$. Sous cette condition, $\Gamma_\mathcal{I}$ peut être enlevée. Dans le cas d'une subsomption *partielle* (($\Gamma' \subseteq \Gamma$, mais $\mathcal{I} \not\subseteq \mathcal{I}'$, la clause subsumée est remplacée par $\Gamma_{\mathcal{I} \cap \overline{\mathcal{I}'}}$.

- *Substitution*: On écrit $[l/a]_\mathcal{J}$, $\delta(l) \leq \delta(a)$, pour noter la substitution de $l_{\Psi|_{\delta(\phi)}}$ pour a_Ψ pour tout $\Psi \in \mathcal{J}$:

$$\Gamma_\mathcal{I}[l/a]_\mathcal{J} = (\Gamma[l/a])_{\mathcal{I} \cap \mathcal{J}} \wedge \Gamma_{\mathcal{I} \cap \overline{\mathcal{J}}} \tag{4}$$

Les règles ci-dessus décrivent les capacités primitives de manipulation du solveur. Celui-ci les arrange dans les transformation de plus haut niveau suivantes, qui préservent aussi la satisfiabilité :

Affectation. Quand il est découvert que$[l]_\mathcal{J}$ est une conséquence de $\mathcal{F}$ (c'est-à-dire : pour tout $\Psi \in \mathcal{J}$, on a $Prop(\mathcal{F}) \vdash l_\Psi$), la formule est simplifiée en $\mathcal{F} * [l]_\mathcal{J}$ en *affectant* $[l]_\mathcal{J}$, c'est-à-dire en résolvant avec $[\neg l]_\mathcal{J}$ et en subsumant avec $[l]_\mathcal{J}$.

Equivalence Si le solveur découvre une preuve que $[a] \overset{\mathcal{J}}{\leftrightarrow} [l]$ (c'est-à-dire : $\forall \Psi \in \mathcal{J}$, $Prop(\mathcal{F}) \vdash a_{\Psi|_{\delta(a)}} \leftrightarrow l_\Psi$) on peut simplifier $\mathcal{F}$ en appliquant la substitution $[l/a]_\mathcal{J}$.

Suppression des redondances. La subsomption est utilisée pour éliminer les clauses subsumées. Cela peut réduire de manière drastique la charge du solveur tout en préservant l'équivalence logique.

Elimination de variable. Les clauses contenant une variable d sont remplacées par l'ensemble des résolvants de toutes les clauses contenant d contre chaque clause

[2]Pour simplifier la notation, on omet la projection des ensembles d'index des clauses dérivées sur les sous-espaces : chaque fois que $\Gamma_\mathcal{I}$ est dérivé, la clause insérée dans la formule est en réalité $\Gamma_{\mathcal{I}|_{\delta(\Gamma)}}$.

contenant $\neg = gd$ (Figure 3). L'élimination symbolique exclut une variable symbolique, et donc toutes les variables de bases concernées en même temps.

Selon nous, une procédure d'évaluation solution-based est un algorithme produisant une suite d'instantiations des règles exposées ci-dessus, et garantit de terminer avec la formule vide sur les instances sat , et avec une contradiction (clause vide) sur les formules unsat . Chaque étape est une réduction en chaînage arrière d'un problème $\mathcal{F}'$ vers un problème $\mathcal{F}$. Donc, on n'extrait *pas* directement un modèle de l'instance originale. Plutôt, l'équivalence de satisfaisabilité garantit que, à chaque pas, si toutefois $\mathcal{F}$ a un modèle, alors on peut aisément dériver un modèle pour $\mathcal{F}'$

Cet inconvénient apparent devient soudain un avantage : il permet en effet de *découpler* l'évaluation de la reconstruction du modèle, avec pratiquement aucune surcharge pour la première et une sémantique claire pour la dernière. Les deux maillons de la chaîne sont reliés grâce à un *rapport d'inférence*, produit par le solveur, et lu par la suite par un *reconstructeur de modèle*.

Définition 5.3 (rapport et trace d'inférence.) *Un* rapport d'inférence *est une liste d'entrées, chacune décrivant une instantiation de l'une des transformations préservant la satisfiabilité présentées ci-dessus. Il contient :*

- $\langle \texttt{assign}, l, \mathcal{J} \rangle$*, pour une assignation* $[l]_{\mathcal{J}}$.
- $\langle \texttt{subst}, a, l, \mathcal{J}, \mathcal{G} \rangle$*, pour une substitution* $[l/a]_{\mathcal{J}}$*, où* $\mathcal{G}$ *est l'ensemble des clauses contenant* $\{a, \neg l\}$ *ou* $\{\neg a, l\}$.
- $\langle \texttt{elim}, d, \mathcal{G}^+, \mathcal{G}^- \rangle$*, pour l'élimination de d,où* $\mathcal{G}^+$ *et* $\mathcal{G}^-$ *sont les ensembles de clauses contenant* d *et* $\neg d$.

Notons par $\mathcal{I}(\mathcal{F}, \mathbf{op})$ *la formule obtenue en applicant à* $\mathcal{F}$ *la transformation décrite par l'entrée* $\mathbf{op}$*. Un rapport d'inférence* $\mathcal{L} = [\mathbf{op}_1, \mathbf{op}_2, \ldots, \mathbf{op}_t]$ *induit une* trace d'inférence $[\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_t]$ *où* $F_i = \mathcal{I}(\mathcal{F}_{i-1}, \mathbf{op}_i)$*, avec* $\mathcal{F}_0 = \mathcal{F}$*. Un rapport tel que* $\mathcal{F}_t$ *est vide est appelé* Rapport-sat *pour* $\mathcal{F}$.

Clairement, il existe un rappost-sat pour $\mathcal{F}$ ssi $\mathcal{F}$ est sat. Par exemple, le lecteur peut vérifier que la figure (4) décrit un rapport-sat pour (1).

5.4 Reconstruction inductive de modèle

Une fois qu'un rapport-sat est connu, le reconstructeur entre en scène. Il se fie au solveur sur le fait que le rapport soit bien un rapport-sat, et il l'analyse *dans l'ordre inverse*, en raisonnant par induction sur le nombre d'entrées:

Cas de base. A la fin de la trace d'inférence, on trouve la formule vide $\mathcal{F}_t$, satisfaite par un modèle vide $\mathcal{M}_t$.

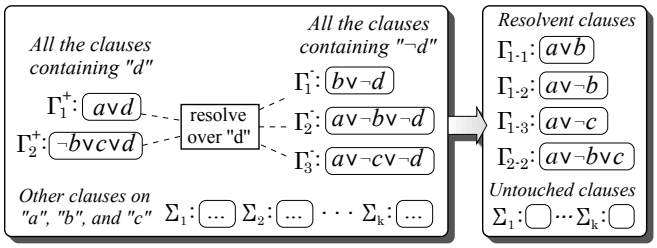


Figure 3: On passe de F' à F en éliminant la variable d.

Cas inductif. Etant donné un modèle $\mathcal{M}_i$ pour $\mathcal{F}_i$, le reconstructeur calcule un modèle $\mathcal{M}_{i-1}{=}\mathcal{R}(\mathcal{M}_i, \mathbf{op}_i)$ pour $\mathcal{F}_{i-1}$ en raisonnant sur la manière dont $\mathbf{op}_i$ a transformé $\mathcal{F}_{i-1}$ en $\mathcal{F}_i$.

Ceci contuit à un modèle $\mathcal{M}_0$ pour $\mathcal{F}$, et donc à un certificat pour F une fois que la fonction $\mathcal{R}$ a été convenablement définie. Notons $\mathfrak{i}(\mathcal{F}) \doteq \cup_{\Gamma_{\mathcal{I}} \in \mathcal{F}} \mathcal{I}$ l'ensemble d'index mentionné dans $\mathcal{F}$.

Théorème 5.3 *La procédure inductive de reconstruction de modèle définie par la fonction suivante* $\mathcal{R}$ *calcule, quand elle est appliquée à n'importe quel rapport-*sat *pour* $\mathcal{F}$*, un modèle* $\mathcal{M}_0$ *pour* $\mathcal{F}_0 = \mathcal{F}$.

$\mathbf{op}$	$\mathcal{R}(\mathcal{M}, \mathbf{op})$
$\langle \texttt{assign}, l, \mathcal{I} \rangle$	$\mathcal{M} \cup \{[l]_{\mathcal{I}}\}$
$\langle \texttt{subst}, e, l, \mathcal{I}, \mathcal{G} \rangle$	$ext^{\mathsf{s}}(\mathcal{M} \cup \{[e]_{\mathcal{I} \cap \mathfrak{i}(\mathcal{M}, l)}, [\neg e]_{\mathcal{I} \cap \mathfrak{i}(\mathcal{M}, \neg l)}\})$
$\langle \texttt{elim}, e, \mathcal{G}^+, \mathcal{G}^- \rangle$	$\mathcal{M}^{\mathsf{e}} \cup \{[e]_{\mathfrak{i}(\mathcal{G}^+ * \mathcal{M}^{\mathsf{e}})}, [\neg e]_{\mathfrak{i}(\mathcal{G}^- * \mathcal{M}^{\mathsf{e}})}\}$

où $ext^{\mathsf{s}}(\mathcal{M}) \doteq \mathcal{M} \cup [e]_{\mathfrak{i}(\mathcal{G} * \mathcal{M})}$*, et* $\mathcal{M}^{\mathsf{e}} \doteq ext^{\mathsf{e}}(\mathcal{M})$*, avec*

$$ext^{\mathsf{e}}(\mathcal{M}) = \begin{cases} ext(\mathcal{M} \cup [v]_{\mathcal{I}}) \text{ if } \exists \langle \Gamma_{\mathcal{I}}, \Gamma'_{\mathcal{I}'} \rangle \in \mathcal{G}^+ * \mathcal{M} \times \mathcal{G}^- * \mathcal{M} \\ \qquad\qquad \text{avec } e \neq v, v \in \Gamma \cap \overline{\Gamma'},\ \mathcal{I} \cap \mathcal{I}' \neq \emptyset \\ \mathcal{M}, \text{ sinon} \end{cases}$$

Preuve. Pour chaque $\mathbf{op}$, on montre que $\mathcal{R}(\mathcal{M}, \mathbf{op}) \models \mathcal{F}'$, fonctionnant sous l'hypothèse d'induction que l'on connait un modèle $\mathcal{M}$ pour $\mathcal{F} = \mathcal{I}(\mathcal{F}', \mathbf{op})$ (c'est-à-dire $\mathcal{F} * \mathcal{M}$ est la formule vide).

Affectation. $\mathcal{F} * \mathcal{M} = (\mathcal{F}' * [l]_{\mathcal{I}}) * \mathcal{M} = \mathcal{F}' * (\{[l]_{\mathcal{I}}\} \sqcup \mathcal{M})$ est vide, donc $\{[l]_{\mathcal{I}}\} \cup \mathcal{M}$ satisfait $\mathcal{F}'$.

Equivalence. Soit $\mathcal{F}' = \mathcal{F}'_0 \cup \mathcal{F}'_1 \cup \mathcal{F}'_{2+} \cup \mathcal{F}'_{2-}$, où aucune clause de $\mathcal{F}'_0$ ne mentionne e, toutes les clauses de $\mathcal{F}'_1$ mentionnent e mais pas l, $\neg l$; toutes les clauses de $\mathcal{F}'_{2+}$ contiennent e et l (ou $\neg e$ et $\neg l$), et les clauses de $\mathcal{F}'_{2-}$ contiennent e et $\neg l$ (ou $\neg e$ et l). $\mathcal{M}$ satisfait $\mathcal{F}{=}\mathcal{F}'[l/e]_{\mathcal{I}}{=}\mathcal{F}_0 \cup (\mathcal{F}'_1 \cup \mathcal{F}'_{2+})[l/e]_{\mathcal{I}}$, donc, il satisfait $\mathcal{F}'_0 = \mathcal{F}_0$ sans aucune modification. Il satisfait aussi $\mathcal{F}'_1 \cup \mathcal{F}'_{2+}$ à condition que l'on répercute sur e les assignations récupérées jusque là sur l en ajoutant $[e]_{\mathcal{I} \cap \mathfrak{i}(\mathcal{M}, l)}$ et $[\neg e]_{\mathcal{I} \cap \mathfrak{i}(\mathcal{M}, \neg l)}$. En général, le modèle résultant $\mathcal{M}^+ = \mathcal{M} \cup \{[e]_{\mathcal{I} \cap \mathfrak{i}(\mathcal{M}, l)}, [\neg e]_{\mathcal{I} \cap \mathfrak{i}(\mathcal{M}, \neg l)}\}$ satisfait toutes les clauses de $\mathcal{F}'$ mais seulement quelques

#1	$\langle$`substitute`$, f, \neg e, \{010, 011\}, \{[e, f]_{\{010,011,110,111\}},$ $[\neg e, \neg f]_{\{000,001,010,011\}}\}\rangle$
#2	$\langle$`assign`$, \neg e, \{001, 011, 100, 110\}\rangle$
#3	$\langle$`resolve`$, f, \{[f]_{\{110\}}, [c, f]_{\{000,001\}}, [e, f]_{\{111\}}\},$ $\{[\neg c, \neg f]_{\{100,101,110,111\}}, [\neg e, \neg f]_{\{000\}}\}\rangle$
#4	$\langle$`assign`$, e, \{111\}\rangle$
#5	$\langle$`resolve`$, c, \{[c]_{\{00,01\}}\}, \{[\neg c]_{\{10,11\}}\}\rangle$

Figure 4: Un rapport-satqui résout le QBF (1).

unes dans $\mathcal{F}'_{2-}$: par construction, de telles clauses n'ont pu passer à $\mathcal{F}$ car $\mathcal{F}'_{2-}[l/e]_{\mathcal{I}} = \mathcal{G}[l/e]_{\mathcal{I}}$ est vide (clauses satisfaites par les littéraux complémentaires $e/\neg l$). Pour satisfaire l'ensemble (pouvant être) non-vide restant de clauses $\mathcal{G} * \mathcal{M}^+$,on étend $\mathcal{M}^+$ en appliquant la fonction *ext*$^{\mathsf{s}}$: elle ajoute une valeur de vérité arbitraire à tous (et seulement ceux-là) les index pour e mentionnés dans $\mathcal{G} * \mathcal{M}^+$, c'est-à-dire elle ajoute $[e]_{\mathrm{i}(\mathcal{G}*\mathcal{M}^+)}$.

Elimination de variable. On se concentre sur la preuve intuitive du cas de base (voir la figure 3). L'extention au cas symbolique est une affaire de notation. Supposons que le modèle M pour F satisfasse au moins une clause de chaque couple $\langle \Gamma_i^+, \Gamma_j^- \rangle$, $i \in [1, n], j \in [1, m]$. Ceci implique qu'il satisfait aussi tout l'ensemble $G_d^+ = \wedge_i \Gamma_i^+$, ou tout l'ensemble $G_d^- = \wedge_i \Gamma_i^-$, ou les deux. Donc, on est libre de choisir un littéral sur d de manière à satisfaire F': $M \cup \{a\}$ satisfait F' quand $G_d^+ * M$ est non-vide, $M \cup \{\neg a\}$ quand $G_d^- * M$ est non-vide, tandis que M lui-même suffit quand ces deux ensembles sont vides. Maintenant, on montre que tout modèle M pour F, soit satisfait au moins une clause de chaque couple $\langle \Gamma_i^+, \Gamma_j^- \rangle$, soit peut être étendu à un modèle *ext*$^{\mathsf{e}}(M)$ qui vérifie cette propriété. Le premier cas survient (au moins) quand aucune clause resolvante n'a été satisfaite par les littéraux complémentaires pendant l'élimination de variable. Dans ce cas, F contient exactement $m * n$ résolvants, un pour chaque couple de $G_d^+ \times G_d^-$. Tout modèle d'un résolvant est valide pour au moins une de ses clauses, d'où la thèse. Maintenant, supposons que le couple $\langle \Gamma_i^+, \Gamma_j^- \rangle$ n'est pas satisfait par M. Il en découle que M en modélise pas le résolvant de Γ_i^+ et Γ_j^-, et que donc un tel resolvant n'est pas passé dans F: il était satisfait par les littéraux complémentaires sur une variable encore non assignée $v \neq$. On peut ajouter arbitrairement un littéral sur v au modèle en construction pour satisfaire soit Γ_i^+, soit Γ_j^- (aucun conflit ne survient : v ne serait pas apparue non-assignée si elle avait été impliquée dans une étape d'inférence antérieure—par rapport au point actuel du solveur—). Cette *extention* de M à *ext*$^{\mathsf{e}}(M)$ est répétée jusqu'à ce qu'il ne reste aucun couple non-satisfait. □

Exemples : $M' = \{a, \neg b, c\}$ est un modèle pour $F' = (\neg a \vee c) \wedge (a \vee b \vee \neg c) \wedge (\neg a \vee \neg b)$ because $M = \{a, \neg b\}$ is a model for $F = F' * c = (a \vee b) \wedge (\neg a \vee \neg b)$.

$M' = \{\neg a, b, c\}$ est un modèle pour $F' = (\neg a \vee c) \wedge (a \vee b \vee \neg c) \wedge (\neg a \vee \neg b)$ car $M = \{b, c\}$ est un modèle pour $F'[\neg c/a] = c \wedge (b \vee \neg c) \wedge (c \vee \neg b)$. Dans la figure 3, l'assignation $M_1 = \{a, b, c\}$ satisfait F et aussi F' : d est laissé non-assignée. Le modèle $M_2 = \{a, \neg b\}$ satisfait G_d^+ mais pas G_d^-, donc $M' = M_2 \cup \{\neg d\}$ est construit de manière à satisfaire F'. Le couple $\langle \Gamma_2^+, \Gamma_1^- \rangle$ est intouché par $M_3 = \{a\}$, donc aucune valeur de vérité pour d n'aide. Mais, le résolvant de Γ_2^+ et Γ_1^-était satisfait par les littéraux complémentaires sur b, donc on construit *ext*$^{\mathsf{e}}(M_3) = M_3 \cup \{b\}$, et alors on satisfait F' avec M'=*ext*$^{\mathsf{e}}(M_3) \cup \{\neg d\}$=$\{a, b, \neg d\}$.

En guise d'exemple complet, le lecteur peut vérifier en exécutant la reconstruction inductive de modèle selon les règles données dans le théorème 5.3 sur le rapport-satde la figure 4 pour le $SymbSk(F)$ de la figure 1.

6 Discussions et conclusions

Nous avons présenté une solution à une question ouverte sur les QBF, à savoir le problème de représentation, de vérification et d'extraction de leurs certificats-sat. L'importance d'une telle solution est double : elle donne des moyens d'exprimer des preuves de satisfiabilité indépendantes du solveur, et elle autorise l'extraction d'informations précieuses à partir des formules certifiées. Cette première caractéristique peut être utilisée, par exemple, pour tester effectivement les décisions des solveurs QBF. La dernière est précieuses pour les applications dans lesquelles un certificat est nécessaire pour illustrer un scénario donné dans lesquels les problèmes encodés en QBF révèlent leur satisfiabilité. Par exemple, une réponse satsuffit pour savoir qu'il existe au moins une stratégie gagnante dans un jeu QBF à deux joueurs, mais on a besoin d'un certificat pour exhiber une telle stratégie.

Les certificats à eux seuls ne révèlent aucune sémantique auto-contenue, étant donné que la signification de chaque variable est une information tenue par les "encodeurs". Pour permettre aux possesseurs de la sémantique d'interroger leur propre modèles QBF, nous avons implémenté une suite logicielle solveur/vérificateur [4, 1] pour produire, vérifier, enregistrer dans des fichiers (aux formats ouverts), et questionner les certificats.

Cette implémentation aide à faire la lumière sur d'autres questions ouvertes sur les QBF. Par exemple, *La certification QBF est-elle inappliquable ?* Définissons la certification de F comme étant inapplicable quand on peut résoudre F sans qu'il soit possible de mémoriser et/ou vérifier aucun certificat $\mathcal{C}(F)$. Notre approche suggère que l'inapplicabilité n'est pas un problème. Par exemple, les certificats de la table 1 sont bien dans les capacités de traitement des machines actuelles, tandis que les formules qu'ils certifient sont difficiles pour les solveurs actuels [3]. Une

[3]la famille des "additionneurs" appartient à un ensemble de bench-

autre question cruciale que l'on peut se poser est la suivante : *Etant donné un certificat $\mathcal{C}$ pour F, de combien en pratique le trio $\langle F, \mathcal{C}, \text{verify}\rangle$ améliore le duo $\langle F, \text{decide}\rangle$ comme moyen de prouver que F est* sat*?*

Des métriques telles que le produit temps · espace doivent être utilisées pour comparer précisément les deux stratégies. Nos résultats permettent déjà de suggérer que la marge d'amélioration est grande, comme indiquée en table 1.

Nous observons un phénomène surprenant: le temps nécessaire pour reconstruire un modèle peut dépasser le temps mis pour résoudre l'instance elle-même. Ceci est relativement inhabituel si l'on compare, par exemple, avec le raisonnement SAT basé sur la recherche, où un modèle est extrait sans surcoût par rapport à la décision de satisfiabilité. Réciproquement, nous n'avons pas encore observé un phénomène que nous attendions: `checkValidity` devrait opérer en temps polynômial si nous employions un oracle BDD en temps constant. La non-polynômialité de la vérification vient de la taille des forêts de BDDs, qui devrait croître exponentiellement pour certaines classes paramétriques d'instances. Ce phénomène défavorable *ne se montre pas* dans la Table 1: Les certificats croissent polynômilalement avec la taille de l'instance. Ces effets et des bornes supérieures sur la taille des certificats doivent être plus étudiés.

Nous avons montré comment extraire des certificats en utilisant une classe particulière de solveurs QBF (ceux basés sur la skolémisation). Toutefois, la représentation basée sur les BDDs est indépendante du solveur. En conséquence, non seulement la représentation même, mais aussi la technique pour construire des certificats peut être amenée à être adaptée à d'autres familles de solveurs. Les ingredients essentiels sont les mêmes: la reconstruction inductive de modèle détachée de l'évaluation de l'instance, avec un log d'inférence au milieu. Ce qui change est le type d'information enregistrée dans le log (et son interprétation).

Nous sommes en train d'étendre notre technique vers des solveurs basés sur (1) la q-résolution non symbolique, (2) le raisonnement SAT et (3) le raisonnement symbolique:non symbolique avec branchement à la DPLL. Le but ultime est de construire un reconstructeur de modèle QBF capable d'extraire des certificats en interprétant des logs d'inférences QBF *génériques*[4], quelle que soit la stratégie adoptée pour résoudre l'instance (incluant des stratégies hybrides telles que celles qui ont montré leur force dans le solveur QBF sKizzo [3]).

instance	$\forall$	$\exists$	T_s	T_r	T_v	$\lvert\mathcal{L}\rvert$	$\lvert\mathcal{C}\rvert$
adder-2	27	37	0.1	0.1	0.1	38	$3.7 \cdot 10^2$
adder-4	106	174	0.2	0.1	0.1	165	$7.1 \cdot 10^3$
adder-6	237	411	0.6	2.3	0.1	384	$5.6 \cdot 10^4$
adder-8	420	748	4.6	35.6	0.4	695	$1.1 \cdot 10^5$
adder-10	755	1185	40.2	537.2	4.6	1098	$4.1 \cdot 10^6$

Table 1: Une famille d'encodage QBF avec alternation $\forall\exists\forall\exists$. Nous mentionnons: le nombre de variables existentielles ($\exists$) et universelles ($\forall$), le temps mis pour résoudre/reconstruire/vérifier (T_s,T_r,T_v), la taille du log ($|\mathcal{L}|$, nombre de pas) et celle du certificat ($|\mathcal{C}|$, nombre de noeuds).

Acknowledgments

Merci à Amedeo Cesta et Gigina Aiello pour leurs suggestions sur l'amélioration du plan de l'article, et à Marco Cadoli pour des discussions utiles sur des points techniques.

References

[1] M. Benedetti. sKizzo: a QBF Decision Procedure based on Propositional Skolemization and Symbolic Reasoning. Technical Report 04-11-03, ITC-irst, 2004.

[2] M. Benedetti. Evaluating QBFs via Symbolic Skolemization. In *Proc. of the 11th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR04)*, number 3452 in LNCS. Springer, 2005.

[3] M. Benedetti. sKizzo: a Suite to Evaluate and Certify QBFs. In *Proc. of 20th International Conference on Automated Deduction (CADE05)*, 2005.

[4] M. Benedetti. The `http://sKizzo.info` web site. 2005.

[5] R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transaction on Computing*, C-35(8):677–691, 1986.

[6] H. K. Büning and X. Zhao. On Models for Quantified Boolean Formulas. In *Proc. of SAT'04*, 2004.

[7] I. Gent and A. Rowley. Encoding Connect-4 using Quantified Boolean Formulae. Technical Report 68-2003, APES Research Group, 2003.

[8] D. Le Berre, M. Narizzano, L. Simon, and A. Tacchella. Second QBF solvers evaluation, avaliable on-line at `www.qbflib.org`. 2004.

[9] J. Rintanen. Construction Conditional Plans by a Theorem-prover. *Journal of A. I. Research*, pages 323–352, 1999.

[10] J. Rintanen. Partial implicit unfolding in the davis-putnam procedure for quantified boolean formulae. In *Proceedings of the International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR'01)*, 2001.

[11] L. J. Stockmeyer and A. R. Meyer. Word Problems Requiring Exponential Time. In *In 5th Annual ACM Symposium on the Theory of Computing*, 1973.

[12] Ingo Wegener. *Branching Programs and Binary Decision Diagrams*. Monographs on Discrete Mathematics and Applications. SIAM, 2000.

marks en rapport avec la vérification d' équivalence d'implémentations partielles de circuite, et est considérée comme étant extrêmenent ambitieux [8]

[4]Un format standard de log pour chaque règle d'inférence est requis dans ce cqs.