Des classes polynomiales pour l'abduction en logique propositionnelle

Bruno Zanuttini
GREYC, Université de Caen
Boulevard du Maréchal Juin
14032 Caen Cedex
email: `zanutti@info.unicaen.fr`

Résumé

L'abduction consiste à rechercher une explication E d'une observation α étant donnée une connaissance Σ, c'est-à-dire un ensemble de faits E tel que Σ et E impliquent conjointement α. Cette problématique est intimement liée, par exemple, à celles du diagnostic ou de l'explication d'incohérence.

La plupart des problèmes d'abduction sont très difficiles, même si on restreint Σ à des classes de formules pour lequel le problème SAT est polynomial. Dans cet article, nous proposons deux algorithmes génériques pour des restrictions de la problématique générale, puis nous donnons des classes de formules dans lesquelles la connaissance Σ peut être exprimée pour que ces algorithmes deviennent polynomiaux. Les restrictions envisagées concernent principalement la forme de l'observation α et l'ensemble d'hypothèses sur lequel peut être formée l'explication E.

Enfin, nous étudions des cas dans lesquels on peut chercher des explications avec des approximations de Σ dans des classes polynomiales.

1 Introduction

L'abduction [16, 6, 5, 12, 17] consiste à rechercher une explication qui justifie une observation donnée, étant donnée une connaissance générale. Par exemple, un système connaissant son comportement habituel (la connaissance) et détectant une panne (l'observation) peut être amené à rechercher une cause plausible (une explication) de cette panne : cette recherche est appelée *abduction*. Il peut également s'agir d'un système qui, connaissant ses capacités, cherche les opérations à effectuer pour atteindre un but donné ; l'observation est alors le but en question, et l'explication est un ensemble d'opérations suffisant. On s'intéresse principalement, dans tous les cas, aux explications *minimales*, c'est-à-dire dont aucune «sous-explication» n'est correcte. La problématique de l'abduction est également voisine de la problématique du diagnostic [18], du raisonnement par défaut, de la circonscription [12], des ATMS et CMS [13] ou encore de l'explication d'incohérence [19], même si dans ce cas on s'intéresse plutôt à *énumérer* toutes les explications qu'à en rechercher une particulière.

Nous nous intéressons ici à l'abduction en *logique propositionnelle*, c'est-à-dire que les connaissances et les observations sont codées dans ce formalisme. De nombreuses formalisations ont été considérées dans la littérature, différant principalement quant à la forme des observations et des hypothèses sur lesquelles on s'autorise à former une explication (voir [5]). Nous essayons ici d'adopter un point de vue qui généralise au mieux la plupart de ces formalisations. Une explication est alors un ensemble de littéraux formés sur un ensemble de variables donné (les hypothèses), et la connaissance comme l'observation sont des formules propositionnelles. Mais même dans ce formalisme restreint de la logique propositionnelle, l'abduction est très difficile, et le reste si on se restreint encore, par exemple à des connaissances exprimées par des formules de Horn [5]. En fait, très peu de classes polynomiales sont connues. Nous nous intéressons donc ici à des restrictions de la problématique générale qui permettent d'obtenir des algorithmes efficaces. Nous nous intéressons également à la sémantique de l'abduction combinée avec l'*approximation* au sens de Selman et Kautz [15] ; nous cherchons à caractériser les explications que l'on peut découvrir avec des approximations de la connaissance (dans une classe de formules $\mathcal{C}$) en fonction de celles que l'on peut trouver avec la connaissance elle-même ; dans les cas où cette caractérisation est précise, il semble en effet justifié d'étudier la complexité de l'abduction en restreignant la connaissance à des formules de $\mathcal{C}$.

Plus précisément, notre contribution est la suivante. Nous donnons deux algorithmes génériques qui permettent de résoudre le problème de l'abduction, l'un dans le cas général et l'autre en forçant l'ensemble d'hypothèses à être l'ensemble des variables tout entier (sauf celles de l'observation) et l'observation à être une clause (ce qui inclut le cas souvent considéré où c'est un littéral). Puis nous étudions les classes de formules pour lesquelles ces algorithmes deviennent polynomiaux. Nous montrons ainsi que l'on peut obtenir une complexité polynomiale dans le cadre général lorsque l'observation est une clause et que la connaissance est exprimée par une formule positive, négative, 2CNF ou affine. Nous montrons également que si l'ensemble d'hypothèses est l'ensemble tout entier des variables non présentes dans l'observation, et si celle-ci est une clause, l'abduction devient polynomiale avec une connaissance exprimée par une formule de Horn (et même Horn-renommable). Concernant la sémantique de l'approximation, nous montrons principalement que si l'on recherche des explications constituées uniquement de littéraux positifs et que l'observation est positive, on peut aussi bien les rechercher avec une approximation de la connaissance dans la classe des formules de Horn qu'avec la connaissance elle-même.

Le papier est organisé comme suit. Au paragraphe 2, nous présentons les définitions et notations utilisées par la suite et nous formalisons la problématique de l'abduction. Au paragraphe 3, nous étudions la problématique non restreinte (premier algorithme) et au paragraphe 4 la problématique restreinte (deuxième algorithme). Enfin, au paragraphe 5 nous considérons la sémantique de l'abduction et de l'approximation, puis nous concluons (paragraphe 6).

2 Préliminaires

2.1 Définitions et notations

Nous considérons un ensemble dénombrable de variables propositionnelles $x_1, x_2, \ldots$. Une variable x_i est aussi appelée un *littéral positif*, et sa négation $\neg x_i$ un *littéral négatif*. Si V est un ensemble de variables, on note $Lit(V)$ l'ensemble $\{x_i, \neg x_i | x_i \in V\}$ des littéraux formés sur V.

Une *clause* est une disjonction finie de littéraux, et une formule propositionnelle est dite en *Forme Conjonctive Normale* (CNF) si elle est écrite comme une conjonction finie de clauses. Par exemple, la formule

$$\phi = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_3 \vee \neg x_4) \wedge (\neg x_2)$$

est en CNF. Un *terme* est une conjonction finie de littéraux, par exemple $x_1 \wedge \neg x_2 \wedge \neg x_4$, et une formule propositionnelle est dite en *Forme Disjonctive Normale* (DNF) si elle est écrite comme une disjonction finie de termes. On note $\bigwedge E$ la conjonction des éléments (littéraux ou clauses) d'un ensemble E. Si ϕ est une formule propositionnelle, $\overline{\phi}$ dénote sa négation logique, et on note $Var(\phi)$ l'ensemble des variables présentes dans ϕ.

Un n-uplet $m \in \{0, 1\}$ est vu comme une affectation $0/1$ à n variables (dans l'ordre de leurs indices) ; on note $m \models \phi$ si m est un *modèle* de la formule propositionnelle ϕ (m *satisfait* ϕ). Par exemple, $m = 1011$ satisfait la formule ϕ ci-dessus. On note m_i la valeur de vérité que m affecte à x_i. Une formule propositionnelle ϕ *implique* une formule propositionnelle α si tout modèle de ϕ est un modèle de α ; deux formules propositionnelles ϕ et ϕ' sont dites *logiquement équivalentes* ($\phi \equiv \phi'$) si ϕ implique ϕ' et ϕ' implique ϕ. Enfin, si E est un ensemble de littéraux et ϕ est une formule CNF, on note $\text{propagation}_E(\phi)$ la formule obtenue en remplaçant dans ϕ tout littéral $\ell \in E$ par 1, tout littéral $\overline{\ell}$ avec $\ell \in E$ par 0, et en simplifiant les constantes. Par exemple, avec ϕ ci-dessus, on a $\text{propagation}_{\{x_1, \neg x_3\}}(\phi) = (\neg x_4) \wedge (\neg x_2)$. Notons que la formule $\text{propagation}_E(\phi)$ s'obtient en temps linéaire à partir de ϕ.

Une formule CNF ou DNF ϕ est dite *positive* si elle ne contient que des littéraux positifs ; dans ce cas, dès que $m \models \phi$ et m' est supérieur ou égal à m bit à bit, on a $m' \models \phi$. La notion de formule *négative* est duale. Une formule ϕ est dite en 2*CNF* si elle est en CNF et que toutes ses clauses contiennent au plus deux littéraux ; la notion de formule en 2*DNF* est similaire. Une formule est dite *affine* [14, 9] si elle est écrite comme une conjonction d'équations linéaires modulo 2 ; par exemple, la formule $(x_1 \oplus x_2 = 1) \wedge (x_1 \oplus x_3 \oplus x_4 = 0)$ est affine. Enfin, une formule est dite *Horn* si elle est en CNF et si chacune de ses clauses contient au plus un littéral positif, et *Horn-renommable* si on peut obtenir une formule de Horn en remplaçant certaines de ses variables par leur négation dans toute la formule. Toutes ces classes de formules sont polynomiales pour le problème SAT (cf. [14, 10]) ainsi que pour la déduction de clauses, puisqu'elles sont de plus stables par propagation d'un ensemble de littéraux (c'est-à-dire que si ϕ est dans l'une de ces classes et C est une clause, $\phi \bigwedge_{\ell \in C} \overline{\ell}$ est également dans la classe).

2.2 Abduction

Dans ce papier, nous nous intéressons à la problématique de l'*abduction*. Nous donnons tout d'abord la définition d'une *explication* (cf. par exemple [5]).

Définition 1 (explication) *Soient Σ (une connaissance) et α (une observation) deux formules propositionnelles, et $H \subseteq Var(\Sigma) \backslash Var(\alpha)$ (un ensemble d'hypothèses). Une* explication de α sachant Σ sur H *est un ensemble E de littéraux tel que :*
(i) $E \subseteq Lit(H)$
(ii) $\Sigma \wedge \bigwedge E$ soit satisfaisable
(iii) $\Sigma \wedge \bigwedge E$ implique α.
Si de plus il n'existe pas de sous-ensemble propre de E avec les mêmes propriétés, l'explication E est dite minimale.

Notons qu'on interdit aux hypothèses de contenir des variables de $Var(\alpha)$, pour des raisons principalement techniques. Cependant, il faut noter que tous les auteurs n'imposent pas cette restriction (voir par exemple [5], ou [8] pour une application concrète).

Exemple 2 (explication) *Soient $\Sigma = (x_1 \vee x_3 \vee \neg x_5) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_3 \vee x_6) \wedge (\neg x_4 \vee x_5)$, $\alpha = (x_5 \vee x_6)$ et $H = \{x_1, x_2, x_3\}$. Alors $E = \{x_1, \neg x_2\}$ est une explication de α sachant Σ sur H. C'est de plus une explication* minimale*, puisque ni $E' = \{x_1\}$ ni $E'' = \{\neg x_2\}$ ne sont des explications.*

Etant donnée cette notion, le problème d'abduction, dans sa forme générale, est le suivant. Les définitions classiques de ce problème supposent souvent que l'observation α est un littéral [18], voire une variable [17] ; nous nous plaçons ici dans un cadre plus général.

Problème 3 (abduction)
Entrée : Σ, α *deux formules propositionnelles, $H \subseteq Var(\Sigma) \backslash Var(\alpha)$*
Sortie : *Une explication minimale de α sachant Σ sur H, s'il en existe une.*

Nous nous intéressons à la complexité de ce problème selon les restrictions imposées à Σ, α et H. En effet, dans toute sa généralité le problème de décision associé (existe-t-il une explication ?) est Σ_2^P-complet, et NP-complet si on impose que Σ soit Horn [5].

A notre connaissance, les principales restrictions polynomiales de ce problème présentes dans la littérature sont obtenues en imposant que α soit un littéral et que Σ soit Horn définie positive (c'est-à-dire Horn avec *exactement* un littéral positif par clause) ou en 2CNF [18] ; Eshghi [6] donne également une sous-classe des formules de Horn acycliques pour laquelle le problème est polynomial (si l'observation est une variable).

3 Ensemble d'hypothèses quelconque

Nous considérons ici la problématique de l'abduction sans restriction sur l'ensemble H d'hypothèses, sinon que ce soit un sous-ensemble de $Var(\Sigma) \backslash Var(\alpha)$. Nous donnons tout d'abord un algorithme générique pour ce problème, puis étudions des restrictions sur Σ et α qui rendent sa complexité polynomiale.

3.1 Un algorithme générique

L'algorithme que nous proposons, `ExpliquerDansH`, est présenté dans la figure 1. Il prend en entrée une connaissance et une observation propositionnelles quelconques, Σ et α, ainsi qu'un ensemble d'hypothèses $H \subseteq Var(\Sigma) \backslash Var(\alpha)$. S'il existe une explication de α sachant Σ sur H, il fournit une telle explication minimale, sinon il retourne 'pas d'explication'.

Nous utilisons la notion de *projection* sur un sous-ensemble de variables. Si $m \in \{0,1\}^n$ est un n-uplet vu comme une affectation aux variables $x_1, \ldots, x_n$, la *projection*

$\Sigma' \leftarrow$ une formule logiquement équivalente à $(\Sigma \wedge \overline{\alpha})_{|H}$;
$\phi \leftarrow$ une formule logiquement équivalente à $\Sigma \wedge \overline{\Sigma'}$;
Si ϕ est insatisfaisable **alors**
  retourner 'pas d'explication' ;
Sinon
  $m \leftarrow$ un modèle de ϕ ;
  $E \leftarrow \{x_i | x_i \in H \text{ et } m_i = 1\} \cup \{\neg x_i | x_i \in H \text{ et } m_i = 0\}$;
  / Minimisation de E : */*
  Pour chaque littéral $\ell \in E$ **faire**
    Si $\Sigma \wedge \bigwedge E \backslash \{\ell\}$ implique α **alors** $E \leftarrow E \backslash \{\ell\}$ **finsi** ;
  Finpour ;
  retourner E ;
Finsi ;

FIG. 1 – Algorithme `ExpliquerDansH`

de m sur $H = \{x_{i_1}, \ldots, x_{i_k}\} \subseteq \{x_1, \ldots, x_n\}$, notée $m_{|H}$, est le k-uplet $m' \in \{0,1\}^k$ tel que pour tout j, $m'_{i_j} = m_{i_j}$. Si ϕ est une formule propositionnelle, on appelle *projection de ϕ sur $H \subseteq Var(\phi)$* toute formule propositionnelle ϕ' telle que l'ensemble des modèles de ϕ' soit exactement l'ensemble $\{m_{|H} | m \models \phi\}$; par abus de langage et à cause de l'équivalence logique, nous dirons que ϕ' est *la* projection de ϕ sur H, et noterons $\phi' \equiv \phi_{|H}$. Nous verrons plus loin comment on peut dans certains cas calculer efficacement la projection d'une formule. Remarquons cependant que dans le cas général, un algorithme polynomial pour calculer la projection d'une formule sur un sous-ensemble de ses variables donnerait un algorithme polynomial pour SAT, puisqu'il suffirait de tester si la projection sur l'ensemble vide est satisfaisable.

L'idée de l'algorithme `ExpliquerDansH` est la suivante. On remarque que si E est une explication de α sachant Σ sur H, alors il n'existe aucun modèle de $\Sigma \wedge \bigwedge E \wedge \overline{\alpha}$. On cherche donc un ensemble $E \subseteq Lit(H)$ tel qu'il n'existe aucune affectation satisfaisant à la fois $\bigwedge E$ et $\Sigma \wedge \overline{\alpha}$; on cherche donc un tel E à partir des affectations qui ne satisfont pas $\Sigma' = (\Sigma \wedge \overline{\alpha})_{|H}$, autrement dit on cherche un modèle m de $\overline{\Sigma'}$; s'il existe, ce modèle affecte une valeur à chaque variable de l'ensemble H, et l'ensemble E des littéraux mis à vrai par m est par construction tel que $\bigwedge E$ implique $\overline{(\Sigma \wedge \overline{\alpha})}$, c'est-à-dire que $\Sigma \wedge \bigwedge E$ implique α. Pour s'assurer que $\Sigma \wedge \bigwedge E$ est satisfaisable, on impose que m soit également un modèle de Σ ; finalement, on cherche donc un modèle E de la formule $\Sigma \wedge \overline{(\Sigma \wedge \overline{\alpha})_{|H}}$. On minimise enfin E en retirant des littéraux tant que E reste une explication. Notons que cette dernière étape peut produire des résultats différents suivant l'ordre dans lequel les littéraux sont retirés, même si tous ces résultats sont bien par construction des explications explications minimales.

Notons que le principe de l'algorithme ressemble à celui proposé par Eshghi [6] : ici, la formule $\overline{\Sigma'}$ a pour rôle d'interdire les ensembles de littéraux sur H qui seraient cohérents avec $\overline{\alpha}$ (autrement dit qui n'impliqueraient pas α), et Eshghi fait jouer le même rôle à une «pseudo-complétion» de la connaissance Σ : cette «pseudo-complétion» impose aux ensembles de littéraux sur H de déclencher un ensemble de règles concluant sur l'observation. Remarquons cependant que cette dernière technique impose que l'observation soit une variable et que la connaissance soit notamment de Horn et acyclique avant même de considérer des questions de complexité, alors qu'ici le principe de l'algorithme n'impose aucune telle restriction.

Exemple 4 (`ExpliquerDansH`) *Nous reprenons le problème de l'exemple 2. La formule* $\Sigma \wedge \overline{\alpha}$ *est* $(x_1 \vee x_3 \vee \neg x_5) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_3 \vee x_6) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5) \wedge (\neg x_6)$*, et sa projection sur* $H = \{x_1, x_2, x_3\}$ *est logiquement équivalente à la formule* $\Sigma' = (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee x_3)$*. On a donc* $\phi = (x_1 \vee x_3 \vee \neg x_5) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_3 \vee x_6) \wedge (\neg x_4 \vee x_5) \wedge \overline{[(x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee x_3)]}$*. Cette formule est satisfaisable, et notamment* $m = 100110$ *en est un modèle ; on pose donc* $E = \{x_1, \neg x_2, \neg x_3\}$ *et la minimisation de cet ensemble nous donne l'explication minimale* $E = \{x_1, \neg x_2\}$*.*

Proposition 5 *Soient* Σ *et* α *deux formules propositionnelles, et* $H \subseteq Var(\Sigma) \backslash Var(\alpha)$*. Si l'algorithme* `ExpliquerDansH` *retourne un ensemble* E *sur l'entrée* (Σ, α, H)*, alors* E *est une explication minimale de* α *sachant* Σ *sur* H*. S'il retourne 'pas d'explication', alors il n'existe pas de tel* E*.*

Preuve : Supposons que l'algorithme retourne un ensemble E. Par construction, on a bien $E \subseteq Lit(H)$. Considérons le modèle m de ϕ dont provient E ; ce modèle nous assure en particulier que $\Sigma \wedge \bigwedge E$ est satisfaisable. Montrons que $\Sigma \wedge \bigwedge E$ implique α. Notons E' l'ensemble $\{x_i | x_i \in H \text{ et } m_i = 1\} \cup \{\neg x_i | x_i \in H \text{ et } m_i = 0\}$, autrement dit l'ensemble E avant minimisation. Comme $m \models \phi$, $\overline{\Sigma'} \wedge \bigwedge E'$ est satisfaisable, et comme E' contient un littéral pour chaque variable de H, $\bigwedge E'$ implique $\overline{\Sigma'}$, donc $\bigwedge E' \wedge \Sigma'$ est insatisfaisable. On en déduit qu'il n'existe pas de modèle de $\bigwedge E'$ qui soit également un modèle de Σ', donc qu'il n'existe pas de modèle de $\bigwedge E'$ qui se prolonge en un modèle de $\Sigma \wedge \overline{\alpha}$; autrement dit, $\bigwedge E'$ implique $(\Sigma \rightarrow \alpha)$, c'est-à-dire que $\Sigma \wedge \bigwedge E'$ implique α. Par construction, cela reste vrai après minimisation de E', donc E est bien une explication correcte.

Réciproquement, si l'algorithme retourne 'pas d'explication', montrons que pour tout $E \subseteq Lit(H)$ avec $\Sigma \wedge \bigwedge E$ satisfaisable, $\Sigma \wedge \bigwedge E$ n'implique pas α. En effet, pour tout modèle m de $\Sigma \wedge \bigwedge E$, on a $m \not\models \overline{\Sigma'}$ (sinon m serait un modèle de ϕ), donc $m \models \Sigma'$, donc $m_{|H} \models \Sigma'$. En particulier, $\bigwedge E \wedge \Sigma'$ est satisfaisable, donc il existe un modèle M de $\bigwedge E \wedge \Sigma \wedge \overline{\alpha}$, donc $\bigwedge E \wedge \Sigma$ n'implique pas α. □

Nous nous intéressons au paragraphe suivant à des classes de formules pour lesquelles cet algorithme est polynomial. Remarquons pour l'instant que, de façon générale, il faut et il suffit que les classes de Σ et de α permettent d'effectuer efficacement les calculs suivants : celui de la projection de $\Sigma \wedge \overline{\alpha}$ sur H, le test de satisfaisabilité de ϕ et le calcul d'un modèle le cas échéant, et enfin la vérification que $\Sigma \wedge \bigwedge E \backslash \{\ell\}$ implique α. Les autres étapes de l'algorithme sont des calculs purement syntaxiques, qui s'effectuent tous en temps linéaire.

3.2 Classes polynomiales

Nous nous intéressons aux restrictions sur Σ et α qui rendent la complexité de l'algorithme `ExpliquerDansH` polynomiale. Comme les formules conjonctives (c'est-à-dire CNF ou affines) sont les formules le plus couramment utilisées pour modéliser des bases de connaissances propositionnelles, nous nous intéressons aux connaissances Σ sous cette forme. Les classes les plus intéressantes étant également celles pour laquelle la déduction de clauses est efficace, nous exhibons dans les trois lemmes suivants certaines de ces classes pour lesquelles on peut calculer efficacement la projection sur un sous-ensemble des variables.

Par souci de complétude, nous présentons le résultat pour les formules positives, négatives et 2CNF dans notre cadre ; cependant, il est connu que le problème de l'abduction est polynomial si α est une clause et si Σ appartient à l'une de ces trois classes, car alors Σ peut être convertie efficacement en l'ensemble de ses impliqués premiers (pour plus de précisions voir [12, Paragraphes 3.3 et 4.2]). A l'inverse, le résultat n'était pas connu pour les formules affines.

Lemme 6 *Soient ϕ une formule propositionnelle positive sur l'ensemble de variables V, et $H \subseteq V$. Alors on a $\phi_{|H} \equiv propagation_{V\setminus H}(\phi)$; si ϕ est négative, on a $\phi_{|H} \equiv propagation_{\{\overline{\ell}, \ell \in V\setminus H\}}(\phi)$).*

Preuve : Par symétrie, nous montrons le résultat seulement pour les formules positives. Soit m satisfaisant $\phi_{|H}$. Alors il existe $M \models \phi$ avec $M_{|H} = m$. Comme ϕ est positive, M reste un modèle de ϕ si on y met toutes les variables de $V\setminus H$ à 1, donc $m \models \text{propagation}_{V\setminus H}(\phi)$. La réciproque est triviale. $\square$

Nous nous intéressons maintenant à la projection des formules 2CNF. Pour ϕ une telle formule et H un sous-ensemble de ses variables, notons $\mathcal{C}l_2(H,\phi)$ l'ensemble des clauses de longueur 2 formées sur H et impliquées par ϕ. Notons que le nombre de clauses de longueur 2 formées avec des variables de H est quadratique en le nombre de variables de H, et donc que $\mathcal{C}l_2(H,\phi)$ peut être calculé efficacement en énumérant ces clauses et en testant pour chacune si elle est impliquée par ϕ (en temps linéaire).

Lemme 7 *Soient ϕ une formule 2CNF sur l'ensemble de variables V, et $H \subseteq V$. Alors on a $\phi_{|H} \equiv \bigwedge \mathcal{C}l_2(H,\phi)$.*

Preuve : Soit tout d'abord m satisfaisant $\phi_{|H}$. Alors il existe $M \models \phi$ avec $M_{|H} = m$; par définition de l'implication, on a donc $M \models C$ pour toute clause $C \in \mathcal{C}l_2(H,\phi)$, donc $M \models \bigwedge \mathcal{C}l_2(H,\phi)$, et comme $\bigwedge \mathcal{C}l_2(H,\phi)$ est formée sur H, $m \models \bigwedge \mathcal{C}l_2(H,\phi)$. Réciproquement, si $m \not\models \phi_{|H}$, alors tout M avec $M_{|H} = m$ vérifie $M \not\models \phi$. Donc la clause $C_m = \bigvee_{m_i=0} x_i \vee \bigvee_{m_i=1} \neg x_i$ est impliquée par ϕ. Puisque ϕ est en 2CNF, il existe une clause C de longueur 2 impliquée par ϕ et impliquant C_m ; comme $Var(C_m) \subseteq H$, on a $C \in \mathcal{C}l_2(H,\phi)$, et comme $m \not\models C_m$ par construction, on a $m \not\models C$ et finalement $m \not\models \bigwedge \mathcal{C}l_2(H,\phi)$. $\square$

Nous nous intéressons enfin aux formules affines. Soient ϕ une telle formule et H un sous-ensemble de ses variables. Soit ϕ_t une formule affine logiquement équivalente à ϕ, obtenue en triangulant par un pivot de Gauss la matrice des coefficients de ses équations, où les variables sont ordonnées de sorte que le coefficient de toute variable de H soit placée plus à droite dans la matrice que toute variable de $V\setminus H$ (voir par exemple [3]). On note alors $\mathcal{E}q_{\phi_t}(H,\phi)$ l'ensemble des équations de ϕ_t qui ne contiennent que des variables de H. Notons que le calcul de $\mathcal{E}q_{\phi_t}(H,\phi)$ consiste essentiellement en une application du pivot de Gauss à la matrice des coefficients de ϕ, et peut donc être effectué efficacement.

Exemple 8 ($\mathcal{E}q_{\phi_t}(H,\phi)$) *Soit ϕ la formule affine $(x_1 \oplus x_2 \oplus x_3 \oplus x_4 = 0) \wedge (x_2 \oplus x_3 = 1) \wedge (x_1 \oplus x_3 = 1) \wedge (x_1 \oplus x_4 = 1)$, et soit $H = \{x_1, x_2\}$. Un ordre convenable des variables* *est alors (x_3, x_4, x_1, x_2). La matrice des coefficients de ϕ selon cet ordre est la suivante :*

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{array}\right)$$

En la triangulant avec le pivot de Gauss, on obtient la matrice :

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right)$$

ce qui correspond à la formule affine $\phi_t = (x_3 \oplus x_4 \oplus x_1 \oplus x_2 = 0) \wedge (x_4 \oplus x_1 = 1) \wedge (x_1 \oplus x_2 = 0)$. L'ensemble $\mathcal{E}q_{\phi_t}(H, \phi)$ est donc réduit à $\{(x_1 \oplus x_2 = 0)\}$.

Lemme 9 *Soient ϕ une formule affine sur l'ensemble de variables V, et $H \subseteq V$. Soit ϕ_t la triangulation de ϕ sur un ordre des variables de V tel que toute variable de H apparaisse après toute variable de $V \backslash H$. Alors on a $\phi_{|H} \equiv \bigwedge \mathcal{E}q_{\phi_t}(H, \phi)$.*

Preuve : Soit tout d'abord m satisfaisant $\phi_{|H}$. Comme toute équation de $\bigwedge \mathcal{E}q_{\phi_t}(H, \phi)$ est une équation de ϕ_t formée sur H et $\phi_t \equiv \phi$, on a $m \models \bigwedge \mathcal{E}q_{\phi_t}(H, \phi)$. Réciproquement, si m satisfait $\bigwedge \mathcal{E}q_{\phi_t}(H, \phi)$, alors comme ϕ_t est triangulaire avec un ordre des variables mettant H à droite, on peut prolonger m en un modèle de ϕ_t avec la méthode de résolution classique des systèmes d'équations linéaires (cf. [3] par exemple). Or $\phi_t \equiv \phi$, donc $m \models \phi_{|H}$. □

Nous pouvons maintenant donner des restrictions sur Σ et α qui permettent de rendre le problème de l'abduction polynomial.

Proposition 10 *La recherche d'une explication minimale E de α sachant Σ sur $H \subseteq Var(\Sigma) \backslash Var(\alpha)$ est polynomiale si α est une clause et Σ est une formule CNF positive, négative, 2CNF ou affine.*

Preuve : Nous montrons le résultat pour les formules positives ; la preuve est similaire pour les autres classes. Soit donc Σ positive ; nous détaillons les calculs de l'algorithme `ExpliquerDansH`. Comme α est une clause, $\overline{\alpha}$ est une conjonction de clauses unitaires, donc $\Sigma \wedge \overline{\alpha} \equiv \text{propagation}_{\{\overline{\ell} | \ell \in \alpha\}}(\Sigma) \wedge \overline{\alpha}$ et $(\Sigma \wedge \overline{\alpha})_{|H} \equiv (\text{propagation}_{\{\overline{\ell} | \ell \in \alpha\}}(\Sigma))_{|H}$ puisque H et $Var(\alpha)$ sont disjoints ; or la formule $\text{propagation}_{\{\overline{\ell} | \ell \in \alpha\}}(\Sigma)$ est toujours en CNF et positive, donc on peut calculer sa projection Σ' sur H efficacement avec le lemme 6. Notons $\Sigma' = C_1 \wedge \cdots \wedge C_k$, où chaque C_i est une clause. Alors la formule ϕ est $\Sigma \wedge \overline{[C_1 \wedge \cdots \wedge C_k]} \equiv \Sigma \wedge [\overline{C_1} \vee \cdots \vee \overline{C_k}] \equiv (\Sigma \wedge \overline{C_1}) \vee \cdots \vee (\Sigma \wedge \overline{C_k})$. Cette formule est satisfaisable si et seulement si l'une des $\Sigma \wedge \overline{C_i}$ est satisfaisable ; or comme chaque C_i est une clause, chaque $\overline{C_i}$ est une conjonction de clauses unitaires, et comme Σ est positive, le test de satisfaisabilité est efficace ainsi que la recherche d'un modèle. Enfin, la déduction de clauses est efficace avec les formules positives, ce qui permet de minimiser E efficacement, et on conclut avec les remarques du paragraphe 3.1. □

Proposition 11 *La recherche d'une explication minimale E de α sachant Σ sur $H \subseteq Var(\Sigma) \backslash Var(\alpha)$ est polynomiale si α est en 2DNF et Σ est en 2CNF.*

Preuve : Puisque α est en 2DNF, $\overline{\alpha}$ est en 2CNF, donc $\Sigma \wedge \overline{\alpha}$ également ; on peut donc calculer efficacement sa projection sur H avec le lemme 7 ; cette projection est également en 2CNF, et on a le résultat comme pour la proposition 10. □

$\Sigma' \leftarrow$ une formule première logiquement équivalente à Σ ;
$\phi \leftarrow$ propagation$_{\{\overline{\ell}|\ell\in\alpha\}}(\Sigma')$;
Si toute clause de ϕ est une clause de Σ' **alors**
  retourner 'pas d'explication' ;
Sinon
  $C \leftarrow$ une clause de ϕ différente de toutes celles de Σ' ;
  $E \leftarrow \{\overline{\ell}|\ell \in C\}$;
  / Minimisation de E :*/*
  Pour chaque littéral $\ell \in E$ **faire**
    Si $\Sigma \wedge \bigwedge E\backslash\{\ell\}$ implique α **alors** $E \leftarrow E\backslash\{\ell\}$ **finsi** ;
  Finpour ;
  retourner E ;
Finsi ;

FIG. 2 – Algorithme `ExpliquerDansV`

4 Restriction sur l'ensemble d'hypothèses et l'observation

Nous considérons désormais une restriction sur l'ensemble d'hypothèses H ; nous n'autorisons plus un ensemble $H \subseteq Var(\Sigma)\backslash Var(\alpha)$ quelconque, mais nous considérons $H = Var(\Sigma)\backslash Var(\alpha)$. Autrement dit, nous nous autorisons à chercher des explications quelconques (seulement disjointes de $Lit(Var(\alpha))$), ce qui correspond par exemple au cas où nous n'avons pas d'idée *a priori* sur les hypothèses plausibles pour expliquer une observation. Notons que c'est bien une *restriction* du problème général, puisque le choix de H n'est plus libre (voir [5]). Pour des raisons principalement techniques, nous ne considérons également plus que des observations α sous forme clausale et des connaissances Σ sous forme CNF. Comme au paragraphe 3, nous donnons tout d'abord un algorithme générique, puis étudions des restrictions sur Σ et α qui rendent sa complexité polynomiale. Ces restrictions beaucoup plus importantes nous permettent d'obtenir une complexité polynomiale pour une connaissance exprimée sous forme Horn, et même Horn-renommable.

4.1 Un algorithme générique

L'algorithme que nous proposons, `ExpliquerDansV`, est présenté figure 2. Il prend en entrée une connaissance propositionnelle quelconque Σ et une observation α (une clause), mais ne prend pas d'ensemble d'hypothèses H ; celui-ci est implicitement tout l'ensemble $V = Var(\Sigma)\backslash Var(\alpha)$. S'il existe une explication de α sachant Σ sur V, l'algorithme fournit une telle explication minimale, sinon il retourne 'pas d'explication'.

Nous avons besoin de la notion de formule CNF *première*. Une formule CNF Σ est dite *première* si toutes ses clauses sont minimales, en ce sens qu'en retirant n'importe quel littéral d'une de ses clauses, on obtient une clause qui n'est plus impliquée par Σ.

Le principe de l'algorithme est globalement similaire à celui de l'algorithme précédent et peut être résumé comme suit. Intuitivement, nous cherchons un ensemble de littéraux E tel que $\Sigma \wedge \bigwedge E \wedge \overline{\alpha}$ soit insatisfaisable, ce qui assure que $\Sigma \wedge \bigwedge E$ implique α. Pour cela, nous propageons les littéraux dont l'opposé est présent dans α (qui est une clause) dans

Σ. La formule obtenue correspond intuitivement à la formule $\Sigma \wedge \overline{\alpha}$ (cf. $(\Sigma \wedge \overline{\alpha})_{|H}$ dans l'algorithme `ExpliquerDansH`) ; il suffit alors de prendre pour E un ensemble de littéraux qui falsifie cette formule, par exemple un ensemble E qui en falsifie une quelconque clause C ; nous prenons donc l'ensemble des littéraux dont l'opposé est présent dans C, puis nous le minimisons comme au paragraphe 3. La nécessité de rendre initialement Σ première est un point technique justifié dans la preuve de la proposition 13.

Remarquons que le principe de cet algorithme consiste essentiellement en la recherche d'un impliqué premier de $\Sigma \wedge \overline{\alpha}$ qui ne soit pas un impliqué de Σ ; en cela la technique présente de nombreux points communs avec celles synthétisées dans [12].

Exemple 12 (`ExpliquerDansV`) *Nous continuons avec le problème de l'exemple 2 (où l'ensemble d'hypothèses H n'est plus pris en compte). En minimisant les clauses de Σ, on obtient par exemple la formule $\Sigma' = (x_1 \vee \neg x_5) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (x_2 \vee \neg x_3 \vee x_6) \wedge (\neg x_4 \vee x_5)$. La propagation de $\{\neg x_5, \neg x_6\}$ dans Σ' donne la formule $\phi = (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (x_2 \vee \neg x_3) \wedge (\neg x_4)$. La clause $(\neg x_4)$ est différente de toutes celles de Σ', donc l'ensemble $E = \{x_4\}$ est une explication de α sachant Σ sur $\{x_1, x_2, x_3, x_4\}$. La minimisation laisse E inchangé.*

Proposition 13 *Soient Σ une formule CNF et α une clause. Si l'algorithme* `Expliquer-DansV` *retourne un ensemble E sur l'entrée (Σ, α), alors E est une explication minimale de α sachant Σ sur $Var(\Sigma) \backslash Var(\alpha)$. S'il retourne 'pas d'explication', alors il n'existe pas de tel E.*

Preuve : Supposons tout d'abord que l'algorithme retourne un ensemble E, et soit C la clause de ϕ dont provient E. Alors ϕ implique C, donc $\Sigma' \wedge \overline{\alpha}$ implique C, donc $\Sigma' \wedge \overline{C}$ implique α, c'est-à-dire que $\Sigma' \wedge \bigwedge E$ implique α avec $\Sigma' \equiv \Sigma$. D'autre part, puisque C est différente de toutes les clauses de Σ' et que la propagation ne peut que retirer des clauses ou diminuer leur taille, il existe une clause de Σ' dont C est une sous-clause propre. Comme Σ' est première, Σ' n'implique pas C, donc $\Sigma \wedge \bigwedge E \equiv \Sigma' \wedge \overline{C}$ est satisfaisable. Pour terminer, par construction E est minimal.

Réciproquement, si l'algorithme retourne 'pas d'explication', alors toutes les clauses de ϕ sont des clauses de Σ'. On en déduit que toute clause C de Σ' contenant un littéral $\ell \in \alpha$ contient également un $\overline{\ell'}$ avec $\ell' \in \alpha$; en effet, C doit être retirée par la propagation, sinon elle donnerait lieu par retrait de ℓ à une clause différente de toutes celles de Σ'. Donc tout modèle de Σ' (c'est-à-dire de Σ) mettant des littéraux de α à vrai reste un modèle si on les y met tous à faux, donc il n'existe pas d'explication de α sur $Var(\Sigma) \backslash Var(\alpha)$. □

Pour que cet algorithme soit de complexité polynomiale, nous remarquons qu'il faut et suffit que la recherche d'une CNF première équivalente à Σ et la vérification que $\Sigma \wedge \bigwedge E \backslash \{\ell\}$ implique α soient polynomiaux dans la classe de formules dans laquelle est prise Σ. Encore une fois, les autres opérations sont purement syntaxiques et linéaires.

4.2 Classes polynomiales

Nous prouvons maintenant que l'algorithme `ExpliquerDansV` est polynomial pour les formules Horn-renommables. Notons que cette classe de formules inclut bien sûr les formules de Horn et anti-Horn (avec au plus un littéral *négatif* par clause).

Proposition 14 *La recherche d'une explication minimale E de α sachant Σ sur $V = Var(\Sigma)\backslash Var(\alpha)$ est polynomiale si α est une clause et Σ est une formule CNF Horn-renommable.*

Preuve : Puisque Σ est Horn-renommable, on peut effectuer efficacement le calcul de Σ' (cf. [10, 7]) ainsi que la vérification que $\Sigma \wedge \bigwedge E\backslash\{\ell\}$ implique α. $\square$

5 Abduction et approximation

L'approximation d'une formule (CNF) ϕ dans une classe de formules $\mathcal{C}$ consiste à calculer une formule ϕ' de $\mathcal{C}$ aussi proche que possible de ϕ. Nous considérons ici les deux notions sémantiques de proximité entre formules proposées dans [15] : deux formules sont considérées comme proches si leurs *ensembles de modèles* sont proches. Plus précisément, une formule $\Sigma_{\mathcal{C}-LUB}$ est appelée une *approximation faible minimale dans* $\mathcal{C}$ d'une formule Σ si $\Sigma_{\mathcal{C}-LUB}$ est une formule de $\mathcal{C}$, si son ensemble de modèles inclut celui de Σ (approximation faible) et si cela n'est vrai pour aucune autre formule de $\mathcal{C}$ dont l'ensemble de modèles est inclus dans celui de $\Sigma_{\mathcal{C}-LUB}$. La notion d'*approximation forte maximale dans* $\mathcal{C}$ est définie de façon duale.

Selman et Kautz montrent que l'on peut faire du raisonnement déductif avec les approximations $\Sigma_{\mathcal{C}-LUB}$ et $\Sigma_{\mathcal{C}-GLB}$ d'une formule Σ : ils caractérisent les résultats obtenus avec les approximations en fonction du résultat qu'on aurait obtenu avec Σ elle-même. Dans de nombreux cas, on déduit aisément ce dernier des premiers. Il est donc intéressant d'avoir des approximations de Σ dans des classes de formules pour lesquelles la déduction est efficace.

A l'inverse, dans la littérature il n'y a à notre connaissance pas de caractérisation des résultats obtenus par l'*abduction* avec les approximations de Σ en fonction du résultat obtenu avec Σ. Ce paragraphe va dans ce sens, principalement dans le cas des formules de Horn et des approximations faibles minimales.

Nous commençons avec quelques résultats généraux, sans hypothèse sur la classe de l'approximation ; ces résultats découlent directement des définitions. Remarquons que dans le cas général, les caractérisations sont moins précises que pour la déduction.

Proposition 15 *Soient Σ et α deux formules propositionnelles, et $\mathcal{C}$ une classe de formules. Soit $\Sigma_{\mathcal{C}-LUB}$ une approximation faible minimale (resp. $\Sigma_{\mathcal{C}-GLB}$ une approximation forte maximale) de Σ dans $\mathcal{C}$, et soit $E \subseteq Lit(Var(\Sigma)\backslash Var(\alpha))$. Alors :*
(i) Si $\Sigma_{\mathcal{C}-GLB} \wedge \bigwedge E$ est satisfaisable, alors $\Sigma \wedge \bigwedge E$ est satisfaisable
(ii) Si $\Sigma_{\mathcal{C}-GLB} \wedge \bigwedge E$ n'implique pas α, alors $\Sigma \wedge \bigwedge E$ n'implique pas α
(iii) Si $\Sigma_{\mathcal{C}-LUB} \wedge \bigwedge E$ implique α, alors $\Sigma \wedge \bigwedge E$ implique α
(iv) Si $\Sigma_{\mathcal{C}-LUB} \wedge \bigwedge E$ n'est pas satisfaisable, alors $\Sigma \wedge \bigwedge E$ n'est pas satisfaisable.

Preuve : Pour (i) et (ii), il suffit d'utiliser le fait que si $m \models \Sigma_{\mathcal{C}-GLB}$, alors $m \models \Sigma$; en effet, pour (i) il existe par hypothèse un tel m avec de plus $m \models \bigwedge E$, et pour (ii) un tel m avec de plus $m \models \bigwedge E$ et $m \models \overline{\alpha}$. Pour (iii) et (iv), le raisonnement est dual. $\square$

Nous considérons désormais le cas des formules de Horn, avec lesquelles la caractérisation est précise. A l'inverse, il ne semble pas aisé d'obtenir une telle caractérisation avec les formules positives, 2CNF ou affines.

Pour m et m' deux n-uplets, notons $m'' = m \wedge m'$ ($\wedge$ bit à bit) le n-uplet tel que pour tout i, $m''_i = m_i \wedge m'_i$. Nous utiliserons le résultat suivant.

Lemme 16 ([11]) *Si Σ est une formule de Horn et si m, m' sont deux de ses modèles, alors $m \wedge m'$ est également un de ses modèles.*

On en déduit que l'approximation faible Horn minimale d'une formule Σ est unique (à équivalence logique près), et que son ensemble de modèles est la clôture de celui de Σ par $\wedge$ bit à bit (cf. [4]). Cette propriété nous permet de relier les explications sachant l'approximation faible Horn minimale de la connaissance Σ à celles sachant Σ elle-même. Nous avons cependant besoin de nous restreindre à des observations positives. Le lemme suivant montre qu'alors les explications obtenues avec une connaissance de Horn sont constituées uniquement de littéraux positifs, ce qui généralise le résultat connu dans le cas où α est une variable [13, Corollaire 4].

Lemme 17 *Soient Σ une formule de Horn, α une formule positive non tautologique et $E \subseteq Lit(Var(\Sigma) \backslash Var(\alpha))$ une explication minimale de α sachant Σ. Alors E est constituée uniquement de littéraux positifs.*

Preuve : Supposons que E contient $\neg x_i$ pour un certain $x_i \in Var(\Sigma) \backslash Var(\alpha)$. Alors il existe un modèle m de Σ avec $m \models \alpha$ et $m \models \bigwedge E$, donc en particulier $m_i = 0$; on sait également, puisque α est positive et non tautologique, que $m_j = 1$ pour au moins une variable $x_j \in Var(\alpha)$. Nous montrons que $\Sigma \wedge \bigwedge E \backslash \{\neg x_i\}$ implique α, donc que E n'est pas minimale ; pour cela, nous montrons qu'il n'existe pas de modèle m' de Σ avec $m' \not\models \alpha$ et $m' \models \bigwedge E \backslash \{\neg x_i\}$. En effet, s'il existe un tel m', alors $m'' = m \wedge m'$ (bit à bit) doit être un modèle de Σ par le lemme 16. Or m'' est par construction un modèle de $\bigwedge E$, puisque m et m' sont égaux sur les variables de $E \backslash \{\neg x_i\}$ et que $m_i = 0$, donc $m''_i = 0$; mais m'' n'est pas un modèle de α, puisque $m' \not\models \alpha$, m'' affecte 0 à toutes les variables de α auxquelles m' affecte 0, et α est positive. Donc m'' contredit le fait que $\Sigma \wedge \bigwedge E$ implique α. □

En utilisant le lemme 17, nous pouvons alors montrer que faire de l'abduction avec l'approximation faible Horn minimale de Σ revient au même qu'avec Σ elle-même lorsqu'on ne considère que les observations et explications positives.

Proposition 18 *Soient Σ une formule propositionnelle, α une formule propositionnelle positive, et $H \subseteq Var(\Sigma) \backslash Var(\alpha)$. Si E est une explication minimale de α sachant $\Sigma_{Horn-LUB}$ sur H, alors E est une explication minimale de α sachant Σ sur H. Réciproquement, si E est une explication minimale constituée uniquement de littéraux* positifs *de α sachant Σ sur H, alors E est une explication minimale de α sachant $\Sigma_{Horn-LUB}$ sur H.*

Preuve : Soit E une explication minimale de α sachant $\Sigma_{Horn-LUB}$. On sait par la proposition 15 que $\Sigma \wedge \bigwedge E$ implique α. Puisque $\Sigma_{Horn-LUB} \wedge \bigwedge E$ est satisfaisable, il existe un modèle m de $\Sigma_{Horn-LUB}$ avec $m \models \bigwedge E$, donc il existe un modèle m' de Σ avec m' supérieur ou égal à m bit à bit (par la remarque qui suit le lemme 16) ; comme E est constituée uniquement de littéraux positifs (lemme 17), m' met également E à vrai, donc $\Sigma \wedge \bigwedge E$ est satisfaisable. Reste à montrer que E est minimale sachant Σ. Dans le cas contraire, il existe $E' \subset E$, $E' \neq E$ avec $\Sigma \wedge \bigwedge E'$ satisfaisable, donc $\Sigma_{Horn-LUB} \wedge \bigwedge E$ satisfaisable par la proposition 15, et tel que $\Sigma \wedge \bigwedge E'$ implique α mais $\Sigma_{Horn-LUB} \wedge \bigwedge E'$ n'implique pas α (sinon E ne serait pas minimale sachant $\Sigma_{Horn-LUB}$). Il existe donc un modèle m de $\Sigma_{Horn-LUB}$ satisfaisant $\bigwedge E'$ mais pas α, donc un modèle m' de Σ satisfaisant $\bigwedge E'$ mais pas α (toujours par la remarque après le lemme 16) ; donc $\Sigma \wedge \bigwedge E'$ n'implique pas α, contradiction.

Réciproquement, soit E une explication minimale positive de α sachant Σ. On sait alors par la proposition 15 que $\Sigma_{Horn-LUB} \wedge \bigwedge E$ est satisfaisable. On sait également que $\Sigma_{Horn-LUB} \wedge \bigwedge E$ implique α par le même raisonnement qu'avec E' au paragraphe précédent. La première partie du théorème montre enfin par minimalité de E pour Σ qu'il ne peut pas exister un $E' \subset E$, $E' \neq E$ avec E' une explication sachant $\Sigma_{Horn-LUB}$, donc que E est minimale sachant $\Sigma_{Horn-LUB}$. □

6 Conclusion

Nous avons donné deux algorithmes génériques pour l'abduction. En étudiant les classes de formules pour lesquelles ces algorithmes deviennent polynomiaux, nous avons montré que l'on peut rechercher des explications efficacement lorsque l'observation est une clause et que la connaissance est exprimée par une formule positive, négative, 2CNF ou affine, ou que l'observation est en 2DNF et la connaissance en 2CNF ; nous avons également montré que l'on peut rechercher efficacement une explication avec des formules Horn-renommables (en particulier Horn) si l'on se restreint au cas où l'ensemble d'hypothèses est l'ensemble des variables tout entier (sauf celles de l'observation) et où l'observation est une clause ; notons que dans les deux cas, nous généralisons le cadre souvent considéré où l'observation est un unique littéral. Enfin, les deux algorithmes proposés peuvent donner lieu à d'autres cas polynomiaux, notamment si un problème pratique comprend certaines hypothèses spécifiques (par exemple si on sait que la connaissance est une CNF première, si on sait calculer des projections, etc.).

Nous avons également présenté quelques résultats concernant l'abduction avec une approximation de la connaissance ; nous avons montré que les explications trouvées avec l'approximation faible Horn minimale d'une formule étaient exactement celles que l'on aurait trouvées avec la formule elle-même, du moment que l'observation est une formule positive et que l'on se restreint aux explications constituées uniquement de littéraux positifs ; l'explication d'une observation positive sachant une formule de Horn ne peut en effet être que de cette forme. Ce résultat est également vrai par dualité pour les formules anti-Horn avec des observations et des explications négatives. A notre connaissance, c'est le premier essai de compréhension de la sémantique abduction-approximation.

Pour terminer, nous remarquons que ces résultats se généralisent aux formules *régulières* sur des domaines totalement ordonnés [2], avec les adaptations évidentes de la problématique : les formules régulières sont les formules «CNF» dont les littéraux sont de la forme $x \geq a$ (littéraux positifs) pour x une variable et a une valeur de son domaine, ou $x < a$ (littéraux négatifs). Avec les transformations de [1], on voit que les propositions 10 et 14 se généralisent respectivement aux formules 2CNF généralisées et Horn(-renommables) généralisées.

Remerciements

Je tiens à remercier Jean-Jacques Hébrard pour son aide à la rédaction de ce papier. Je tiens également à remercier deux relecteurs anonymes pour leurs remarques pertinentes et particulièrement constructives.

Références

[1] B. Beckert, R. Hähnle et F. Manyà, Transformations between signed and classical clause logic. *Proc. 29th International Symposium on Multiple-Valued Logics (ISMVL'99)*, 248–255, 1999.

[2] B. Beckert, R. Hähnle et F. Manyà, The 2-SAT problem of regular signed CNF formulas. *Proc. 30th International Symposium on Multiple-Valued Logics (ISMVL'00)*, 331–336, 2000.

[3] C.W. Curtis, *Linear algebra. An introductory approach.* Springer-Verlag, 1984.

[4] R. Dechter et J. Pearl, Structure identification in relational data. *Artificial Intelligence*, vol. 58 : 237–270, 1992.

[5] T. Eiter et G. Gottlob, The complexity of logic-based abduction. *Journal of the ACM*, vol. 42 : 3–42, 1995.

[6] K. Eshghi, A tractable class of abduction problems. *Proc. 13th International Joint Conference on Artificial Intelligence (IJCAI'93)*, 3–8, 1993.

[7] P.L. Hammer et A. Kogan, Horn functions and their DNFs. *Information Processing Letters*, vol. 44 : 23–29, 1992.

[8] J. Hobbs, M. Stickel, D. Appelt et P. Martin, Interpretation as abduction. *Artificial Intelligence*, vol. 63 : 69–142, 1993.

[9] D. Kavvadias et M. Sideri, The inverse satisfiability problem. *SIAM J. Comput.*, vol. 28 : 152–163, 1998.

[10] H.R. Lewis, Renaming a set of clauses as a Horn set. *Journal of the ACM*, vol. 25 : 134–135, 1978.

[11] J. McKinsey, The decision problem for some classes of sentences without quantifiers. *J. Symbolic Logic*, vol. 8 : 61-77, 1943.

[12] P. Marquis, Consequence finding algorithms. *Handbook of DRUMS*, vol. 5 : 41–145, 2000.

[13] R. Reiter et J. de Kleer, Foundations of assumption-based truth maintenance systems : preliminary report. *Proc. 6th National Conference on Artificial Intelligence (AAAI'87)*, 183–188, 1987.

[14] T.J. Schaefer, The complexity of satisfiability problems. *Proc. 10th Annual ACM Symposium on Theory Of Computing (STOC'78)*, 216–226, 1978.

[15] B. Selman et H. Kautz, Knowledge compilation and theory approximation. *Journal of the ACM*, vol. 43 : 193–224, 1996.

[16] B. Selman et H. Levesque, Abductive and default reasoning : a computational core. *Proc. 8th National Conference on Artificial Intelligence (AAAI'90)*, 343–348, 1990.

[17] A. del Val, On some tractable classes in deduction and abduction. *Artificial Intelligence*, vol. 116 : 297–313, 2000.

[18] A. del Val, The complexity of restricted consequence finding and abduction. *Proc. 17th National Conference on Artificial Intelligence (AAAI'00)*, 337–342, 2000.

[19] G. Verfaillie et L. Lobjois, Problèmes incohérents : expliquer l'incohérence, restaurer la cohérence. *Actes des 5ièmes Journées Nationales sur la Résolution Pratique de Problèmes NP-Complets (JNPC'99)*, 111–120, 1999.