Abduction et programmation logique

Noam Zeitoun
zeitoun.noam@gmail.com

Abstract

L'abduction est une forme de raisonnement visant à obtenir des explications pour des observations dans le cadre d'une théorie donnée. Nous résumons comment l'abduction peut être modélisée par la déduction dans le cas où la théorie est un programme logique. Nous utilisons et décrivons un cadre qui permet de résoudre un large éventail de problèmes d'abduction dans différents domaines d'application en variant la sémantique de l'inférence vers l'explication. L'abduction peut également être modélisée par l'argumentation, qui est étroitement liée à la programmation logique. Une correspondance est explorée.

1 Introduction

L'abduction, après la déduction et l'induction, est une forme de raisonnement logique où, étant donné une théorie et certaines observations, le but est de trouver une explication aux observations dans le cadre de la théorie. Il s'agit d'une tâche de raisonnement très intuitive, effectuée par l'esprit humain de manière continue, qui est en quelque sorte liée à ce que l'on pourrait appeler le "sens commun".

Ce travail est motivé par la fascination que l'abduction peut être modélisée par la déduction et par un intérêt pour la programmation logique. Il vise à établir des liens entre trois travaux qui adoptent des approches différentes de l'abduction :

Abduction via la déduction Dans la section 1.1, nous revisitons le raisonnement logique et établissons une intuition pour les trois formes de raisonnement susmentionnées à l'aide de courts exemples afin d'expliquer comment Console et al. [4] réaliser l'abduction par déduction dans la section 3.

Cadre pour les problèmes d'abduction Alors que Eiter et al. [7] s'intéressaient principalement à l'étude de la complexité et à la découverte de correspondances entre différents problèmes d'abduction, ils ont défini un cadre polyvalent pour parler des problèmes d'abduction en tant que programmes logiques sous une sémantique donnée. Dans la section 1.2, la programmation logique (en tant que forme de programmation déclarative) est décrite de manière informelle et très brièvement comparée à la programmation impérative, dont les applications sont beaucoup plus répandues. Une perspective plus formelle sur les programmes logiques est donnée dans la section 2.1, car ils sont la base de la section 4, où ledit cadre généralisant les problèmes d'abduction modélisés par la programmation logique est décrit.

L'abduction dans l'argumentation L'argumentation abstraite est une notion plus récente qui est étroitement liée à la programmation logique. Nous la présenterons dans la section 1.3, et donnerons une définition formelle des cadres d'argumentation dans la section 2.2. Dans la section 5, nous explorerons ensuite comment Booth et al. [2] modéliser l'abduction dans l'argumentation, et soulignerons le lien avec la programmation logique.

1.1 Déduction, induction et abduction

Nous utilisons le travail de Peirce [15] pour mettre en évidence le schéma sous-jacent et les différences entre trois types de raisonnement au moyen de syllogismes[1] :

Deduction

	Règle.	Tous les haricots de ce sac sont blancs.
	Cas.	Ces haricots proviennent de ce sac.
$\therefore$	*Résultat.*	Ces haricots sont blancs.

Induction

	Cas.	Ces haricots proviennent de ce sac.
	Résultat.	Ces haricots sont blancs.
$\therefore$	*Règle.*	Tous les haricots de ce sac sont blancs.

Abduction

	Règle.	Tous les haricots de ce sac sont blancs.
	Résultat.	Ces haricots sont blancs.
$\therefore$	*Cas.*	Ces haricots proviennent de ce sac.

Les trois types de raisonnement sont obtenus en agençant *Règle*, *Cas* et *Résultat* de sorte que deux d'entre eux agissent comme prémisses, concluant le troisième. Trois exemples : (1) Les mathématiques dépendent fortement de la déduction pour dériver des connaissances plus explicites à partir d'axiomes de base. (2) Les physiciens utilisent des expériences et leurs résultats pour dériver les lois de la physique, ce qui peut être considéré comme une induction. (3) Un médecin utilise des connaissances médicales pour expliquer les symptômes de son patient, un diagnostic obtenu par abduction.

L'abduction présente un intérêt particulier dans le domaine de l'intelligence artificielle, lorsqu'il s'agit de générer des explications, des diagnostics et des plans, mais elle a également été appliquée avec succès à la reconnaissance vocale, à la vision et à l'apprentissage [4, 7, Introductions].

Dans le syllogisme ci-dessus montrant l'abduction, la *Règle* correspond à certaines connaissances ou à une théorie, tandis que le *Résultat* représente une observation et le *Cas* correspond à une explication, un diagnostic, un plan et autres choses similaires.

Il est important de noter que l'abduction peut donner lieu à de multiples explications et qu'elle n'est en général pas saine, car certaines explications (ou diagnostics) peuvent s'avérer fausses [7]. Cependant, ceci n'est qu'au niveau méta, ce qui signifie que la théorie peut être ajustée pour conduire à de meilleurs résultats.

1.2 Programmation logique

La programmation logique est considérée comme une approche très puissante de la programmation déclarative. Plutôt qu'un algorithme pour résoudre un problème, le problème lui-même (et sa solution, dans une certaine mesure) est décrit. Comme un programme logique ne définit aucun type de flux de contrôle, il n'est pas exécuté directement, comme les programmes impératifs, mais placé dans un solveur (qui est aussi un programme). De la même manière que les programmes impératifs sont d'abord écrits, puis (dans certains cas) interprétés par un interprète, un programme logique est écrit, puis les solutions sont obtenues par un solveur.

[1] Le lecteur intéressé peut se référer à [5] resp. [14] afin d'en savoir plus sur les racines philosophiques de l'abduction ou de la logique de Peirce en général.

Prenons par exemple le problème des cliques : Étant donné un graphe, sa solution est l'ensemble de tous les sous-graphes entièrement connectés. L'écriture d'un programme impératif (que ce soit en Java, en C#, en Go ou dans un langage similaire) pour résoudre le problème Clique n'est pas simple, bien qu'il s'agisse d'une tâche raisonnable pour un étudiant en informatique de deuxième semestre. L'étudiant doit prendre en compte le flux de contrôle du programme, garder la trace des solutions déjà obtenues, et ainsi de suite. En utilisant un langage non hébergé, elle pourrait même avoir à s'occuper de l'allocation et de l'affectation de la mémoire. En revanche, avec la programmation logique, la tâche du programmeur consiste simplement à formaliser le problème de manière à ce que le résolveur le comprenne, ce qui s'avère normalement être la tâche la plus facile.

Lorsque l'on regarde pour la première fois la structure des programmes logiques (voir section 2), la structure des règles, essentiellement des implications, semble étonnamment similaire à la nature déductive de la logique mathématique. Cela fait de la programmation logique un véhicule intéressant pour encoder des théories et des connaissances sous forme de règles. Ce qui n'est pas si évident, c'est la façon dont ces programmes peuvent être transformés de manière à ce que l'abduction soit modélisée de manière déductive, comme nous le verrons dans la section 3.

Pour une introduction et un aperçu de la programmation par ensembles de réponses, voir [8].

1.3 Argumentation abstraite

L'argumentation abstraite introduite par Dung [6] est un domaine actif de la recherche en intelligence artificielle. Bien que ce domaine ait été inspiré par la capacité humaine à comprendre de nouveaux concepts, à effectuer des raisonnements scientifiques, à exprimer et à clarifier ses opinions dans sa vie quotidienne par le biais de l'argumentation et du dialogue, la notion centrale de cadre d'argumentation (voir la définition 3) est mieux comprise comme un ensemble d'arguments qui peuvent entrer en conflit les uns avec les autres. Cela rend l'argumentation abstraite intéressante pour les problèmes où la résolution des conflits est essentielle.[2]

En exploitant la simplicité des AF, de nombreuses sémantiques ont été présentées qui définissent comment calculer les extensions à partir d'un AF et donc ce qui constitue un (sous-)ensemble souhaitable d'arguments. Ces sémantiques sont en quelque sorte concurrentes et inspirent de nouvelles lignes de recherche. L'identification de classes de sémantiques, c'est-à-dire l'utilisation d'un accord par paire sur les solutions entre sémantiques et autres relations, est également une recherche active.

De même, Dung [6] soutient que "de nombreuses approches majeures du raisonnement nonmonotonique en IA et en programmation logique sont en fait différentes formes d'[argumentation abstraite]". [6, Introduction, p. 325]. Grâce à ce lien, nous examinerons comment l'abduction de la programmation logique s'inscrit dans le paradigme des cadres d'argumentation tel qu'il est défini par Booth et al. [2].

2 Préliminaires

Nous reviendrons brièvement sur les programmes logiques, car ils sont fondamentaux pour [4] et [7].

[2]Pour un exposé introductif, voir [1].

2.1 Programmes logiques

Définition 1 (voir [17, Slide 15]). *Une* rule *est une paire ordonnée de la forme*

$$a_1 \vee \ldots \vee a_m \leftarrow b_1 \wedge \ldots \wedge b_k \wedge \text{not } b_{k+1} \wedge \ldots \wedge \text{not } b_n$$

où $a_1, \ldots, a_m, b_1, \ldots, b_n$ *sont des littéraux,* not *est* négation comme échec *(ou* négation par défaut*),* $a_1 \vee \ldots \vee a_m$ *est la* head *de* r *et* $b_1, \ldots, b_k, \text{not } b_{k+1}, \ldots, \text{not } b_n$ *est la* body *de* r.

Certaines classes de règles plus concrètes présentent un intérêt : Une règle est une fact *si* $n = 0$ *et* $m \geq 1$ *;* basic *si* $n = k$ *et* $m \geq 1$ *;* non-disjonctif *si* $m = 1$ *;* normal *si elle est non-disjonctive et ne contient pas de négation forte* $\neg$ *;* Horn *si elle est normale et basique ;* ground *si tous ses littéraux sont ground.*

Définition 2. *Un programme logique (généralement noté LP) est un ensemble fini de règles. De manière analogue à la classification d'une règle, un programme est basique, normal, Horn etc. si toutes ses règles le sont.*

Lors de l'évaluation de programmes logiques, il n'existe pas de sémantique canonique, surtout si l'on considère les informations incomplètes et la négation par défaut. Deux approches possibles sont décrites dans la section 4, car l'idée centrale du cadre de [7] est de varier la sémantique de l'inférence.

2.2 Cadres d'argumentation

Dans la section 1.3, nous nous baserons sur la notion de cadre d'argumentation (AF).

Définition 3 (originalement [6, Definition 2], repris de [2, Definition 1]). *Étant donné un ensemble infini dénombrable* $\mathcal{U}$ *appelé* univers d'arguments*, un* cadre d'argumentation (AF) *est une paire* $F = (A, \rightsquigarrow)$ *où* A *est un sous-ensemble fini de* $\mathcal{U}$ *et* $\rightsquigarrow$ *une relation binaire sur* A*. Si* $a \rightsquigarrow b$ *on dit que* a attacks b*.* $\mathcal{F}$ *désigne l'ensemble de tous les cadres d'argumentation.*

3 L'enlèvement par déduction

D'après l'introduction ci-dessus, l'abduction et la déduction semblent être distinctes et il n'est pas évident de savoir comment l'abduction peut être mise en œuvre ou programmée. Il est cependant possible de déduire les conclusions de l'abduction de manière déductive.

Le "résultat fondamental" de [4] est "mettant en évidence le pont entre l'abduction et la déduction à travers la sémantique de complétion". Nous allons utiliser cette section du document pour nous familiariser avec leur résultat. Afin de ressembler au théorème principal, nous importons la complétion de Clark [3] ainsi qu'une définition des problèmes d'abduction au niveau méta et au niveau objet.

Considérons l'exemple concret ([13, p. 371] tel que cité dans [4, p. 663]) d'un problème d'interprétation simple représenté par la théorie de domaine suivante LP_1 :

$$\begin{aligned} T_1 = \{ \quad & l'herbe\_est\_humide \leftarrow pluie\_de\_la\_dernière\_nuit, \\ & l'herbe\_est\_humide \leftarrow depuis\_l'arrosage, \\ & l'herbe\_est\_froide\_et\_brillante \leftarrow l'herbe\_est\_mouillée, \\ & les\_chaussures\_sont\_mouillées \leftarrow l'herbe\_est\_mouillée \quad \} \end{aligned}$$

Maintenant, supposons que nous voulons obtenir une explication pour notre manifestation $M_1 \equiv l'herbe\_est\_froide\_et\_brillante$, c'est-à-dire que nous nous demandons : "Pourquoi l'herbe pourrait-elle être brillante ?" Intuitivement, nous allons regarder la théorie et essayer de suivre les implications du conséquent à l'antécédent (resp. de gauche à droite), en partant de l'observation et en menant aux deux atomes *pluie_de_la_dernière_nuit* et *depuis_l'arrosage*. En effet, dans ce cas, les deux atomes coïncident avec les deux explications minimales de M_1 :

$$S_1 = \{pluie\_de\_la\_dernière\_nuit\}$$
$$S_2 = \{depuis\_l'arrosage\}$$

Plus généralement, nous remarquons tout d'abord que les seules explications (ou hypothèses) "sensible" de notre observation sont les atomes qui sont atteints en dernier lorsque l'on suit intuitivement les implications du conséquent à l'antécédent : L'ensemble des atomes qui n'apparaissent dans la tête d'aucune règle de la théorie sont appelés *atomes abductibles* (terminologie de [9] tel que cité dans [4, p. 664]). Deuxièmement, pour atteindre une définition formelle de l'intuition, l'achèvement de Clark [3], une transformation qui permet (encore une fois, intuitivement) de suivre les implications de la théorie en sens inverse est appliquée.

La complétion des prédicats non abductibles d'une théorie LP, notée LP_C est formellement définie dans [4] comme suit : La complétion LP_C est un ensemble d'équivalences $\{p_i \leftrightarrow D_i | i = 1, \ldots, n\}$, où $p_1, \ldots p_n$ sont tous les atomes non abdicables dans LP et $D_i \equiv Q_{i1} \vee \ldots \vee Q_{im}$ dans le cas où $\{Q_{ij} \rightarrow p_i | j = 1, \ldots, m\}$ est l'ensemble des clauses de LP ayant p_i comme tête. L'application de cette transformation à notre exemple conduit à :

$$\begin{aligned} LP_{1,C} = \{ & l'herbe\_est\_humide \leftrightarrow \quad pluie\_de\_la\_dernière\_nuit \vee depuis\_l'arrosage, \\ & l'herbe\_est\_froide\_et\_brillante \leftrightarrow \quad l'herbe\_est\_humide, \\ & les\_chaussures\_sont\_mouillées \leftrightarrow \quad l'herbe\_est\_humide \quad \} \end{aligned}$$

De plus, nous exigeons que la théorie LP soit *hierarchique*, ce qui signifie qu'il est possible d'attribuer un rang à chaque atome de sorte que, pour chaque règle de la théorie, le rang des atomes de la tête est strictement supérieur au rang associé à tout atome du corps de la règle. En pratique, cela signifie qu'un graphe orienté où les sommets correspondent à des atomes et les arêtes à la relation corps/tête d'une règle est acyclique. Cette hypothèse est nécessaire pour montrer la terminaison d'une procédure traversant les règles de la même manière que ci-dessus. Remarquez que les atomes abductibles seront du rang le plus bas, et que l'ordre strict implique la terminaison. [4, cf. section 3, p. 667, 669]

Maintenant que nous avons une intuition de ce en quoi consiste un problème d'abduction, une idée de ce que l'on peut attendre comme explication, et une idée approximative d'une transformation qui aidera à générer l'explication, nous poursuivons avec une définition d'un problème d'abduction. Ici, nous utilisons la définition plus générale du cadre postulé dans [7].

Définition 4 ([7, Definition 1, p. 140])**.** *Soit V un ensemble d'atomes propositionnels. Un problème d'abduction par programmation logique (LPAP) $\mathscr{P}$ sur V consiste en un tuple $\langle H, M, LP, \models \rangle$ où $H \subseteq V$ est un ensemble fini d'hypothèses, $M \subseteq V \cup \{\neg v | v \in V\}$ est un ensemble fini de manifestations, LP est un programme logique propositionnel sur V et $\models$ est un opérateur d'inférence.*

Console et al. [4] exigent en outre que LP soit hiérarchique.

Dès lors, la question suivante est de savoir comment caractériser les explications. Une définition au niveau méta est présentée en premier lieu, et une définition au niveau objet en second lieu.

Définition 5. *Soit ème* $= \langle H, M, LP, \models \rangle$ *un LPAP, et soit* $S \subseteq H$*, alors* S *est une* solution *(ou* m-explanation*) de (resp. pour)* $\mathscr{P}$ *si* $LP \cup S \models M$*. L'ensemble des solutions de* $\mathscr{P}$ *est noté* $\mathrm{Sol}(\mathscr{P})$*.*

Remarque 1. *Deux commentaires sur les différences entre la définition d'une m-explication en [4, Definition 3, p. 671] et une solution dans [7, Definition 2, p. 140] pour plus de clarté : Console et al. [4] limitent l'opérateur de référence à* $\vdash_{NF}$*, la dérivation SLDNF, c'est-à-dire la procédure d'évaluation de requête de [3], tandis que Eiter et al. [7] s'intéressent à un cadre plus général pour les relations d'inférence arbitraires (développé dans la section 4). La notation diffère également légèrement, puisque les observations, appelées manifestations, sont notées* Ψ *et les solutions, appelées m-explications, sont notées* E *dans [4].*

Définition 6 (adapté de [4, Definition 2, p. 669])**.** *Soit* $\mathscr{P} = \langle H, M, LP, \vdash_{NF} \rangle$ *être un LPAP utilisant la dérivation SLDNF et* LP_C *la complétion (de Clark) des prédicats non-abductibles dans* LP*. La* formule d'explication *pour* $\mathscr{P}$ *est la formule la plus spécifique* F *dans le langage des atomes abductibles telle que :*

$$LP_C, M \models F$$

où F *est plus spécifique que* F' *si* $\models F \rightarrow F'$*.*

Avec ceci, une procédure pour calculer une formule d'explication F est décrite dans [4], qui produit F à partir de LP_C en substituant les atomes non abductibles p_i avec leur contrepartie D_i de la complétion jusqu'à ce que seuls les atomes abductibles soient présents. Ils soutiennent que cette procédure s'arrête au motif que LP est supposé être hiérarchique. Que la procédure esquissée aboutit effectivement à F est montré dans [4, Theorem 1].

Une correspondance entre la définition au niveau de l'objet et au niveau méta des explications des problèmes d'abduction, ressemble à la connexion entre la déduction au niveau de l'objet et l'abduction au niveau méta.

Théorème 1 (adapté de [4, Theorem 2, p. 671])**.** *Soit ème* $= \langle H, M, LP, \vdash_{NF} \rangle$ *être une LPAP utilisant une dérivation SLDNF ayant* F *comme formule d'explication. Soit* S *un ensemble d'atomes abductibles et* I *une interprétation telle que pour chaque atome abductible* α

$$I \models \alpha \text{ si } \alpha in S$$

Alors S *est une solution (une m-explication) si* $S \models F$*.*

La preuve du théorème 1 est omise par souci de concision.

En outre, Console et al. [4] décrivent comment les définitions ci-dessus peuvent être étendues pour tenir compte de la dépendance entre les atomes abductibles (connaissances supplémentaires sur les abductibles). Ils tiennent compte de deux types de dépendance :

- *taxonomie* ou *relations d'abstraction* entre atomes abductibles, c'est-à-dire implications de la forme

  $$\alpha \rightarrow \beta$$

  (où α et β sont des atomes abductibles) ;

- *contraintes* entre atomes abductibles sous forme de négations, c'est-à-dire de la forme
$$\neg(\alpha_1 \wedge \ldots \wedge \alpha_n)$$
où $\alpha_i, \ldots, \alpha_n$ sont des atomes abductibles.

Là encore, les transformations au niveau de l'objet sont décrites [4, Section 4.1], et la correspondance avec le méta-niveau [4, Section 4.2] est montrée.

Enfin, ils fournissent une extension au cas du premier ordre qui découle d'une "théorie de l'égalité incluse dans la complétion". [4, Section 5.1, p. 684].

4 Un cadre pour les problèmes d'abduction

Eiter et al. [7] ont formulé un cadre général pour les problèmes d'abduction permettant de varier la sémantique de l'inférence et ont comparé la complexité des sémantiques communes.

La flexibilité de leur cadre est évidente dans la définition 4, qui a déjà été utilisée dans la section précédente, car elle permet un opérateur d'inférence arbitraire. Ils justifient cela en expliquant que différentes sémantiques peuvent être nécessaires en fonction du domaine d'application.

Brièvement, ils pensent que $\models_{wf}$, $\models^b_{st}$, $\models^c_{st}$ sont intéressants, définis comme suit :

Définition 7 (voir [11, Section 2])**.** *Étant donné un programme logique LP et une interprétation I, la* transformation de Gelfond-Lifschitz *de LP par rapport à I, notée* LP^I *est définie comme suit :*

$$\begin{aligned} LP^I = \{ & a_1 \vee \ldots \vee a_m \leftarrow b_1, \ldots, b_k \mid \\ & a_1 \vee \ldots \vee a_m \leftarrow b_1, \ldots, b_k, \text{not } b_{k+1}, \ldots, \text{not } b_n \in LP \\ & \textit{ et } \{b_{k+1}, \ldots, b_n\} \cap M = \emptyset\} \end{aligned}$$

Notez que si LP est non-disjonctif, alors LP^I est un programme de Horn.

Définition 8. *Tout programme de Horn P a un moindre modèle* $\mathrm{lm}(P)$*. Étant donné un programme logique LP et une interprétation I, I est appelé un* modèle stable *si* $I = \mathrm{lm}(LP^I)$*. La collection de tous les modèles stables de LP est dénotée par* $\mathrm{STM}(LP) = \{I | I = \mathrm{lm}(LP^I)\}$.

Définition 9 ([7, p. 137])**.** Raisonnement courageux *(ou* Raisonnement crédule*) déduit qu'un littéral Q est vrai dans LP (noté* $LP \models^b_{st} Q$*) si Q est vrai par rapport à M pour* ***some*** $M \in \mathrm{STM}(LP)$.

Définition 10 ([7, p. 137])**.** Raisonnement prudent *(ou* Raisonnement sceptique*) déduit qu'un littéral Q est vrai dans LP (noté* $LP \models^b_{st} Q$*) si (1) Q est vrai par rapport à M pour* ***all*** $M \in \mathrm{STM}(LP)$ *et (2)* $\mathrm{STM}(LP) \neq \emptyset$.

La sémantique bien fondée décrite par Gelder et al. [10] représente une progression de l'achèvement de Clark [10, Section 1.1], discuté dans la section 3. Elle implique des modèles à trois valeurs (en ajoutant $\perp$ comme "troisième valeur de vérité inconnue" à true et false) et ce qu'on appelle *ensembles non fondés*. Une définition est omise dans ce travail par souci de brièveté et pour éviter toute complexité. Le point le plus important à retenir est que la sémantique bien fondée infère qu'un littéral Q est vrai dans LP, noté $LP \models_{wf} Q$, s'il est vrai dans le modèle bien fondé, et qu'il existe au plus un tel modèle (contrairement à la sémantique stable, où il peut y avoir plusieurs modèles stables).

Il est intéressant de noter ici que le choix d'un modèle unique ($\vdash_{NF}$ comme dans Prolog, et $\models_{wf}$) par rapport à une sémantique à modèles multiples (modèles stables dans la programmation par ensembles de réponses, et $\models^b_{st}$, $\models^c_{st}$ basés sur des modèles multiples) pour l'inférence est à l'origine de ce qui est parfois appelé le *Great Logic Programming Schism.* [16, Section 3, p. 13].

Dans [7], trois problèmes de décision importants sont mis en évidence : Étant donné un $\mathscr{P}LPAP = \langle H, M, LP, \models \rangle$,

1. does there exist a solution for $\mathscr{P}$? (*cohérence*),
2. est une hypothèse donnée $h \in H$ pertinente pour $\mathscr{P}$, c'est-à-dire, est-ce que h contribue à une certaine solution de $\mathscr{P}$? (*pertinence*),
3. est une hypothèse donnée h pertinente pour $\mathscr{P}$, c'est-à-dire, h est-elle contenue dans toutes les solutions de $\mathscr{P}$? (*nécessité*).

Ils continuent ensuite à prouver la complexité de la vérification de la solution ($S \in \mathrm{Sol}(\mathscr{P})$), de la vérification de la cohérence ($\mathrm{Sol}(\mathscr{P}) \neq \emptyset$) et de la pertinence $\preceq$, -nécessité de certains $h \in H$ pour les LPAP et les LPAP disjonctifs (LP disjonctif, voir Définition 2) où la préférence $\preceq$ est soit la minimalité par rapport à l'inclusion ($\subseteq$), la cardinalité ($\leq$) ou aucune préférence ($=$) le long des trois opérateurs d'inférence $\models_{wf}$, $\models^b_{st}$, $\models^c_{st}$. Les résultats montrent que ces problèmes sont complets dans la partie inférieure de la hiérarchie polynomiale ($\Delta^P_2[O(\log(n)]$ à Σ^P_4), et permettent quelques transformations intéressantes entre des problèmes d'abduction de sémantique différente.

5 L'abduction dans l'argumentation

Dans cette section, nous allons examiner de plus près le travail de Booth et al. [2], qui, en élargissant la définition d'un cadre d'argumentation (AF), introduit les cadres d'argumentation abductifs (AAF).

Tout d'abord, étant donné une AF $F = (A, \rightsquigarrow)$, nous devons reconnaître comment l'abduction peut être exprimée : Une observation correspond à un ensemble d'arguments $X \subseteq A$, "qui chacun supporte individuellement l'observation" [2, p. 118] et le but est de trouver une explication, un cadre d'argumentation G qui soutient X.

Remarque 2. *Dans le travail de Booth et al. [2] il n'y a pas de distinction claire entre* Rule *(théorie),* Résultat *(observation) et* Cas *(explication) (en référence aux syllogismes de la section 1.1). La théorie (qui semble être enchevêtrée avec l'observation) et l'explication sont toutes deux des AF alors que l'observation est un ensemble d'arguments, mais le lien n'est pas clairement expliqué. De plus, la signification de "soutient l'observation" n'est pas claire. Ils définissent un AF soutenant une observation, mais pas un argument soutenant une observation.*

Par analogie avec $\models^c_{st}$ et $\models^b_{st}$, Booth et al. [2] définit le soutien sceptique et crédule d'une observation, sur la base de la sémantique complète.

Définition 11 ([2, Definition 2])**.** *Soit $F = (A, \rightsquigarrow)$. Une extension de F est un ensemble $E - ensembledeA$. Une extension E est sans conflit si, pour chaque $a, b \in E$, il existe que ab. Un argument a dans$A$ est défendu par E si pour tout b tel que $b \rightsquigarrow a$ il existe un c dans$E$ tel que $c \rightsquigarrow b$. Étant donné une extension E, nous définissons* $\mathrm{Def}_F(E)$ *par* $\mathrm{Def}_F(E) = \{a \in A \mid E \text{ defends } a\}$. *Une extension E est admissible si E est sans conflit et $E \subseteq \mathrm{Def}_F(E)$, et complète si E est sans conflit et $E = \mathrm{Def}_F(E)$. L'ensemble des extensions complètes de F sera dénoté par* $\mathrm{Co}(F)$. *De plus, l'extension fondée (dénotée par* $\mathrm{Gr}(F)$*) est l'unique $\subseteq$ extension complète minimale de F.*

Définition 12. *Étant donné une AF $F = (A, \rightsquigarrow)$, une observation $X \subseteq A$ est soutenue avec scepticisme (resp. crédulité) si pour tout (resp. certains) $E \in \mathrm{Co}(F)$ il est vrai que $x \in E$ pour tout $x \in X$.*

Or il se peut qu'une AF F ne soutienne pas de manière sceptique ou crédule une observation X. Dans ce cas, l'objectif est de trouver une nouvelle AF G qui soutienne à son tour X. Nous arrivons à la définition d'un cadre d'argumentation abductif.

Définition 13. *Un* abductif cadre d'argumentation *est une paire $M = (F, I)$ où F est un cadre d'argumentation et $I \subseteq \mathcal{F}$ un ensemble de cadres d'argumentation appelé* abducible *tel que $F \in I$.*

Comme prévu, un AAF $M = (F, I)$ explique de manière sceptique (resp. crédule) une observation X si tout $G \in I$ soutient de manière sceptique (resp. crédule) X. Nous continuons à mettre en évidence le lien entre F et $G \in I$ à travers la programmation logique.

Pour la correspondance entre les AF et les programmes logiques, Booth et al. [2] s'appuie sur les résultats de Wu et al. [18], selon lesquels un programme logique LP sur V peut être transformé en un AF F tel que les conséquences de P sous la sémantique stable partielle[3] peuvent être calculés en examinant les extensions complètes de F. L'idée est qu'un argument consiste en une conclusion $C \in V$, un ensemble de règles $R \subseteq LP$ utilisées pour dériver C et un ensemble $N \subseteq V$ d'atomes qui doivent être sous-vivables pour que l'argument soit acceptable (citation de [2], éditée pour la même notation que dans les sections 3 et 4). Un tel triple (C, R, N) représente un *argument instancié* qui est dit *généré*[4] de LP et l'ensemble des arguments générés par LP est noté A_{LP}. La relation d'attaque générée par LP, notée $\rightsquigarrow_{LP}$, est définie par $(C, R, N) \rightsquigarrow_{LP} (C', R', N')$. Avec ceci, nous avons établi le lien entre les AFs et la programmation logique : Pour un programme logique LP sur V, un atome $C \in V$ est une conséquence sceptique (resp. crédule) de LP si un certain $(C, R, N) \in A_{LP}$ est sceptiquement (resp. crédule) accepté dans $(A_{LP}, \rightsquigarrow_{LP})$.

Dans le cadre instancié par Booth et al. [2] la définition d'une hypothèse par rapport à un programme logique LP sur V diffère des hypothèses selon la définition 4 (et donc du cadre défini par Eiter et al. [7]) : Nous examinons maintenant les paires (Δ^+, Δ^-) où Δ^+ et Δ^- sont des ensembles d'abductibles (qui à leur tour sont dans V). Δ^+ signifie abdubles ajoutés et complète les abdubles retirés notés Δ^-. Ainsi, une hypothèse explique sceptiquement (resp. créduleusement) une requête $Q \in V$ si Q est une conséquence sceptique (resp. créduleuse) de Q. Avec ceci, nous pouvons expliquer la connexion entre F et $G \in I$ pour un AAF $F = (G, I)$.

Définition 14 ([2, Definition 16] édité pour les mêmes notations que dans les sections 3 et 4)**.** *Étant donné un programme logique LP sur V et d'une hypothèse (Δ^+, Δ^-), nous désignons par $F_{(\Delta^+, \Delta^-)}$ l'AF $(A_{(LP \cup \Delta^+) \setminus \Delta^-}, \rightsquigarrow_{(LP \cup \Delta^+) \setminus \Delta^-})$. L'AAF* généré par *$LP$ est noté M_{LP} et défini par $M_{LP} = ((A_{LP}, \rightsquigarrow_{LP}), I_{LP})$, où $I_{LP} = \{F_{(\Delta^+, \Delta^-)} \mid \Delta^+, \Delta^- \subseteq V, \Delta^+ \cap \Delta^- = \emptyset\}$.*

Booth et al. [2] concluent que leur modèle d'abduction dans l'argumentation est une abstraction de la programmation de la logique abductive.

[3] Note que selon **?**], cette sémantique coïncide avec la sémantique bien fondée qui fait partie de l'étude de [7] et est très brièvement mentionnée dans la section 4.

[4] Une définition formelle d'un programme logique *générant* un argument instancié est omise pour des raisons de brièveté. Voir [2, Définition 13]

6 Conclusion

Si la principale motivation de ce travail était la fascination pour l'abduction, c'est-à-dire la manière dont un ordinateur peut tenter de raisonner de manière sensée, l'objectif était également d'approfondir la compréhension de la programmation logique et de comprendre comment elle peut mettre en œuvre l'abduction. Cela s'inscrit dans une perspective plus large de combinaison de perspectives symboliques (représentation des connaissances par la programmation logique) et non symboliques (réseaux neuronaux) sur l'intelligence artificielle, que nous considérons comme un domaine de recherche important pour les années à venir.

6.1 Résumé

En ce qui concerne les trois sections précédentes, les principales conclusions sont les suivantes : (1) l'abduction à partir de programmes logiques, bien que ce problème paraisse intangible au premier abord, peut être transposée à la déduction de manière directe, (2) les complexités des problèmes de décision importants découlant de l'abduction résident dans la hiérarchie polynomiale, avec des correspondances intéressantes entre elles, et (3) sans surprise, l'argumentation peut généraliser l'abduction avec une forte connexion à la programmation logique.

Nous nous sommes efforcés d'introduire l'abduction, l'induction et la déduction par l'exemple, afin de juxtaposer l'abduction et la déduction dans la section 3. Le lien entre la section 3 et la section 4 est explicite dans la mesure où nous avons utilisé le cadre décrit dans cette dernière pour notre définition et notre raisonnement dans la première. De plus, le lien entre la section 4 et la section 5 est double : (1) La notion d'acceptation crédule et sceptique d'un argument, et la vérité d'un atome sous la sémantique crédule et sceptique sont remarquablement familières. (2) Le lien étroit entre les programmes logiques et les cadres d'argumentation (qui a même été mis en évidence dans le travail initial de Dung [6]) invite à raisonner sur les AAF dans le cadre défini par Eiter et al. [7].

6.2 Questions ouvertes

La question de savoir comment les programmes logiques générant des AAFs (le résultat de Wu et al. [18] comme discuté dans la section 5) peuvent être intégrés dans le cadre de Eiter et al. [7] reste ouverte : Comment des hypothèses de la forme (Δ^+, Δ^-) (avec $\Delta^+, \Delta^- \subseteq V$) d'un côté peuvent-elles être mises en correspondance avec des hypothèses de la forme $H \subseteq V$ de l'autre côté, où V est l'ensemble des atomes ?

Par ailleurs, les dialogues d'explication détaillés dans la [2, Section 4] n'ont pas été abordés dans ce travail. Ils constituent une variante intéressante de la preuve dialogique utilisant des coups hypothétiques, mais le lien avec la programmation logique n'est pas direct.

References

[1] P. Baroni. An Introduction to Abstract Argumentation. `http://www.epcl-study.eu/content/downloads/slides/baroni_2013_abstract_argumentation.pdf`.

[2] R. Booth, D. M. Gabbay, S. Kaci, T. Rienstra, and L. W. N. van der Torre. Abduction and dialogical proof in argumentation and logic programming. In

T. Schaub, G. Friedrich, and B. O'Sullivan, editors, *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 117–122. IOS Press, 2014. ISBN 978-1-61499-418-3. doi: 10.3233/978-1-61499-419-0-117. URL http://dx.doi.org/10.3233/978-1-61499-419-0-117.

[3] K. L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, 1977.*, Advances in Data Base Theory, pages 293–322, New York, 1977. Plemum Press. ISBN 0-306-40060-X.

[4] L. Console, D. T. Dupré, and P. Torasso. On the relationship between abduction and deduction. *J. Log. Comput.*, 1(5):661–690, 1991. doi: 10.1093/logcom/1.5.661. URL http://dx.doi.org/10.1093/logcom/1.5.661.

[5] I. Douven. Abduction. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition, 2016. https://plato.stanford.edu/archives/win2016/entries/abduction/.

[6] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995. doi: 10.1016/0004-3702(94)00041-X. URL https://doi.org/10.1016/0004-3702(94)00041-X.

[7] T. Eiter, G. Gottlob, and N. Leone. Abduction from logic programs: Semantics and complexity. *Theor. Comput. Sci.*, 189(1-2):129–177, 1997. doi: 10.1016/S0304-3975(96)00179-X. URL http://dx.doi.org/10.1016/S0304-3975(96)00179-X.

[8] T. Eiter, G. Ianni, and T. Krennwallner. Answer set programming: A primer. In Tessaris et al. [16], pages 40–110. ISBN 978-3-642-03753-5. doi: 10.1007/978-3-642-03754-2_2. URL http://dx.doi.org/10.1007/978-3-642-03754-2_2.

[9] K. Eshghi. Abductive planning with event calculus. In Kowalski and Bowen [12], pages 562–579. ISBN 0-262-61056-6.

[10] A. V. Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *J. ACM*, 38(3):620–650, 1991. doi: 10.1145/116825.116838. URL http://doi.acm.org/10.1145/116825.116838.

[11] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In Kowalski and Bowen [12], pages 1070–1080. ISBN 0-262-61056-6.

[12] R. A. Kowalski and K. A. Bowen, editors. *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, August 15-19, 1988 (2 Volumes)*, 1988. MIT Press. ISBN 0-262-61056-6.

[13] J. Pearl. Embracing causality in formal reasoning. In K. D. Forbus and H. E. Shrobe, editors, *Proceedings of the 6th National Conference on Artificial Intelligence. Seattle, WA, July 1987.*, pages 369–373. Morgan Kaufmann, 1987. URL http://www.aaai.org/Library/AAAI/1987/aaai87-066.php.

[14] C. Peirce and J. Buchler. *Philosophical Writings of Peirce*. Dover books on philosophy. Dover Publications, 1940. ISBN 9780486202174. URL https://books.google.at/books?id=Uq1kDQAAQBAJ.

[15] C. S. Peirce. Illustrations of the Logic of Science VI: Deduction, Induction, and Hypothesis. *The Popular Science Monthly*, 13, 1878.

[16] S. Tessaris, E. Franconi, T. Eiter, C. Gutierrez, S. Handschuh, M. Rousset, and R. A. Schmidt, editors. *Reasoning Web. Semantic Technologies for Information Systems, 5th International Summer School 2009, Brixen-Bressanone, Italy, August 30 - September 4, 2009, Tutorial Lectures*, volume 5689 of *Lecture Notes in Computer Science*, 2009. Springer. ISBN 978-3-642-03753-5. doi: 10.1007/978-3-642-03754-2. URL http://dx.doi.org/10.1007/978-3-642-03754-2.

[17] H. Tompits. Introduction to Knowledge Based Systems: Answer Set Programming. Lecture at Vienna University of Technology, 2016.

[18] Y. Wu, M. Caminada, and D. M. Gabbay. Complete extensions in argumentation coincide with 3-valued stable models in logic programming. *Studia Logica*, 93(2-3):383–403, 2009. doi: 10.1007/s11225-009-9210-5. URL https://doi.org/10.1007/s11225-009-9210-5.