Université de Nancy I
UFR S.T.M.I.A

CRIN (URA CNRS 262)
INRIA-Lorraine

COMBINAISON DE RESOLUTIONS DE CONTRAINTES

Thèse de doctorat de l'université de Nancy I
école doctorale IAE+M
département de formation doctorale informatique
présentée par

Christophe Ringeissen

Soutenue publiquement le 21 décembre 1993

devant la commission d'examen

Président	J.-P. Finance
Rapporteurs	H. Ganzinger
	J.-P. Jouannaud
Examinateurs	A. Colmerauer
	H. Kirchner
	P. Lescanne

Remerciements

Je voudrais tout d'abord exprimer ma très profonde gratitude à Hélène Kirchner qui m'a dirigé et conseillé avec une rare clairvoyance durant ces trois dernières années de recherche et sans qui ce travail n'aurait pu voir le jour. Sa compétence, sa permanente disponibilité, sa gentillesse et ses encouragements m'ont été des plus précieux.

Je tiens ensuite à remercier chaleureusement ceux qui ont bien voulu prendre part à ce jury :

Harald Ganzinger qui a accepté de bien vouloir écrire un rapport malgré l'obstacle de la langue et qui apporte au jury ses compétences dans le domaine de la démonstration automatique.

Jean-Pierre Jouannaud qui a bien voulu s'intéresser à ce travail en acceptant la lourde tâche de rapporteur ; ses commentaires fort judicieux m'ont été très utiles dans la préparation de ce document.

Alain Colmerauer qui me fait l'honneur d'examiner cette thèse et qui n'a pas hésité à venir depuis Marseille ; ses travaux précurseurs sont à l'origine de l'engouement suscité par la programmation avec contraintes.

Jean-Pierre Finance qui en tant que directeur du Centre de Recherche en Informatique de Nancy m'a permis de mener cette thèse dans d'excellentes conditions et qui me fait de surcroît l'honneur d'en présider le jury.

Pierre Lescanne qui en tant que rapporteur interne et directeur de l'équipe EURECA s'est prêté à une lecture avisée et intéressée de ce document ; la discussion qui s'en suivit fut très enrichissante.

Ce travail s'est déroulé successivement au sein des équipes EURECA et PROTHEO du CRIN et de l'INRIA-Lorraine. Je profite de l'occasion qui m'est offerte pour remercier celles et ceux de Nancy ou d'ailleurs qui, de près ou de loin, en ont facilité la réalisation.

Je remercie tout particulièrement Claude Kirchner dont les travaux dans le domaine de l'unification sont à l'origine de cette thèse et qui en tant que directeur de l'équipe PROTHEO a toujours soutenu et guidé ma recherche.

Je remercie également Eric Domenjoud et Francis Klay avec lesquels j'ai travaillé pour étendre des résultats de combinaison aux théories non disjointes.

Un grand merci enfin à Alexandre Boudet qui en tant que spécialiste du problème de combinaison s'est consacré à une étude très attentionnée de ce document. Je lui en suis reconnaissant.

Table des matières

Introduction

La mise en équations ou plus généralement la spécification d'un problème relève d'une démarche fondamentale en mathématiques qui est motivée par la connaissance d'une méthode de résolution dans le domaine de calcul adéquat comme par exemple les termes du premier ordre, les booléens, les entiers ou les réels. La diversité des domaines fait de la recherche d'une méthode de résolution, ou *algorithme*, une expérience difficile à sans cesse renouveler. Il s'agit d'un problème *indécidable* en général: on ne peut en effet espérer trouver un algorithme universel mais il est crucial de trouver des classes de problèmes décidables.

Ce constat a ainsi ouvert la voie à une activité de recherche en informatique dont l'objet est la construction d'algorithmes de résolution dans des structures spécifiques et qui s'inscrit dans une perspective très ambitieuse consistant à automatiser le raisonnement (mathématique). Dans ce contexte, la résolution d'équations dans des structures abstraites (de termes), ou *unification*, est une composante fondamentale de nombreux prouveurs de théorèmes élaborés à ce jour. L'intérêt s'est d'abord porté sur l'unification syntaxique, pour laquelle les symboles de fonctions n'ont aucune propriété particulière. Les travaux de G. Plotkin[Plo72], M. Stickel [Sti85] et ceux de J.-P. Jouannaud et H. Kirchner [JK86] pour n'en citer que quelques uns, ont permis d'intégrer des théories équationnelles au processus de déduction afin d'en accroître l'efficacité et l'expressivité grâce à une unification de nature plus sémantique.

Le souci permanent d'améliorer les systèmes de déduction a ensuite conduit à introduire le concept de *contrainte*. Le principe du raisonnement avec contraintes consiste à ne pas calculer les solutions d'un problème mais à garder celui-ci sous la forme d'une contrainte schématisant l'ensemble des instances admissibles. Il suffit alors souvent de tester la *satisfaisabilité* des contraintes avant d'entamer un processus de déduction. La connaissance d'algorithmes de satisfaisabilité revêt donc dans ce cadre une importance capitale.

Le raisonnement avec contraintes est aussi un moyen d'utiliser des domaines de calcul spécifiques et de faire coopérer deux formes de déductions distinctes. L'une dépend fortement d'algorithmes spécialisés disponibles dans le domaine de calcul où s'exprime les contraintes, alors que l'autre est basée sur des mécanismes plus généraux comme la résolution, la simplification ou la superposition. Parmi les exemples de domaines de calcul qui ont été intégrés à la programmation logique avec contraintes [JL87, Col90, DvHS$^+$88], citons notamment les réels, les domaines finis, l'algèbre de Boole, et plus généralement les algèbres primales qui sont des algèbres finies permettant d'exprimer toute fonction définie sur le domaine par un terme. Mais jusqu'à présent, la gestion de plusieurs domaines de calcul dans les systèmes de déduction se limite au cas où les algorithmes de résolution ne

coopèrent pas ou très peu. Ces algorithmes sont le plus souvent utilisés indépendamment l'un de l'autre.

L'objet de cette thèse est l'étude de la *combinaison* des algorithmes de résolution de contraintes afin d'encore augmenter le pouvoir d'expression des systèmes de déduction. Le problème est le suivant: comment construire à partir de deux algorithmes de résolution de contraintes dans deux domaines de calcul, un nouvel algorithme de résolution de contraintes dans la réunion des deux domaines? Une instance de ce problème est obtenue en remplaçant la résolution de contraintes par l'unification.

La situation actuelle

Le problème de combinaison a déjà été abordé dans le cadre de l'unification où le besoin d'outils systématiques pour la construction d'algorithmes [Kir85a] s'est très vite affirmé. Alors qu'un algorithme d'unification modulo l'axiome de commutativité se déduit de celui existant pour l'unification syntaxique, il n'en est pas de même pour l'associativité-commutativité, où l'unification fait appel à la résolution d'équations diophantiennes. Devant la complexité du problème, la recherche d'algorithmes d'unification est devenue au fil des années une activité à part entière ayant pour souci majeur de concevoir des outils pour leur construction quasi-automatique. Deux voies principales pour une telle construction ont été explorées:

- Comment transformer un problème d'unification (sémantique) modulo une théorie en un autre relevant de l'unification syntaxique?

  Le concept de surréduction [Hul80] permet de répondre à cette question dans le cas de théories définies par un système de réécriture convergent.

- Comment construire de façon modulaire un algorithme d'unification dans une théorie équationnelle $E_1 \cup E_2$ en réutilisant les algorithmes existants pour E_1 et E_2?

  C'est ce problème de combinaison d'algorithmes d'unification qui est à l'origine des travaux exposés dans cette thèse.

De nombreux éléments de réponses, toujours plus complets, ont déjà été apportés mais pour le mélange de théories disjointes uniquement. C. Kirchner [Kir85a, Kir89] s'est intéressé au mélange de théories admettant des algorithmes d'unification spécifiques. Mais en général, des restrictions syntaxiques ont d'abord été imposées sur la forme des axiomes. Le cas des axiomes réguliers et non effondrants fut résolu par K. Yelick [Yel87] et a permis ainsi de mélanger symboles libres, commutatifs et associatifs-commutatifs. Puis la méthode fut généralisée aux axiomes réguliers. Enfin, le cas général a été résolu par M. Schmidt-Schauß [SS89] par la présentation d'un algorithme non-déterministe, qui fut ensuite rendu déterministe par A. Boudet [Bou90a]. Mais l'abandon progressif des restrictions syntaxiques s'est accompagné en contre-partie de la spécialisation des algorithmes d'unification à combiner. Il faut en effet pouvoir déterminer les solutions satisfaisant une certaine restriction comme celle de ne pas instancier une variable arbitraire.

Même si le cas général semble avoir été résolu, le problème de la combinaison de l'unification présente encore un large éventail de perspectives. F. Baader et K. Schulz [BS92] ont d'ailleurs montré récemment que la condition sur les algorithmes d'unification avec

restriction pouvait être précisée et qu'il suffisait d'une restriction dite *linéaire* parce que fondée sur un ordre total sur les variables $x_1, \ldots, x_{n_1}, y_1, \ldots, y_{n_2}$ du problème et ce afin d'éviter l'apparition d'un cycle

$$y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1]$$

lors du calcul des solutions. Un autre avantage de ce dernier algorithme, inspiré de celui non-déterministe proposé par M. Schmidt-Schauß, est de s'appliquer aussi bien à la décision de l'unification qu'à l'unification elle même, ce que ne permettaient pas les approches précédentes.

Tous ces algorithmes ont cependant un mécanisme commun s'appuyant sur l'hypothèse que les théories ont des signatures disjointes. Cette condition est primordiale à première vue car elle permet de déterminer sans ambiguïté la théorie d'un symbole et de décomposer un problème hétérogène en deux sous-problèmes *purs*, chacun d'eux formés exclusivement de symboles dans une théorie. L'opération de *purification* consiste à remplacer chaque sous-terme étranger par une nouvelle variable et de rajouter l'équation résolue liant cette variable et le terme abstrait. Ensuite, l'utilisation conjointe des algorithmes d'unification sur les sous-problèmes purs fournit deux *solutions* pures

$$x_1 =^? s_1 \wedge \cdots \wedge x_{n_1} =^? s_{n_1}$$

$$y_1 =^? t_1 \wedge \cdots \wedge y_{n_2} =^? t_{n_2}$$

qu'il faut *combiner* pour en extraire une unique solution, elle, hétérogène. C'est dans cette phase de combinaison des solutions qu'intervient la notion de restriction linéaire.

Derrière l'apparente simplicité de ce mécanisme de combinaison se cachent cependant de sérieuses difficultés:

- Pourquoi un problème pur peut-il être impunément résolu dans sa théorie alors qu'en principe il devrait l'être dans le mélange? Autrement dit, ce mécanisme est-il valide dans l'union des théories?

- Quelle démarche entreprendre lorsqu'une variable se trouve résolue simultanément dans les deux théories ou lorsque un cycle entre les variables empêche la construction d'une forme résolue hétérogène?

La diversité du comportement des théories vis-à-vis de ces problèmes explique l'évolution par paliers à laquelle on a pu assister pour la combinaison de l'unification. En fait, les principes de combinaison, à savoir purification et résolution séparée, s'appliquent avec plus ou moins de facilité suivant les classes de théories considérées.

Ces principes ont déjà été employés par le passé mais sous une forme simplifiée pour un autre problème que l'unification. Nelson et Oppen [NO79] d'une part et Shostak [Sho84] d'autre part se sont intéressés dans les années 1980 à la coopération d'algorithmes de décision dans des théories du premier ordre, avec pour objectif la vérification de programmes mélangeant listes, tableaux et réels. La réutilisation des algorithmes se fait là aussi après avoir transformé une formule hétérogène en deux sous formules pures par l'introduction de nouvelles variables faisant abstraction de sous-termes étrangers. La coopération des algorithmes s'effectue au travers des variables, seuls termes partagés par les deux théories disjointes: on déduit les égalités entre variables puis on les propage dans l'autre théorie.

Les objectifs

La similitude des algorithmes de combinaison laisse présager de l'existence de principes applicables au cadre plus général de la résolution des contraintes. Le but de cette thèse est d'approfondir le champ d'application des principes de combinaison en poursuivant les objectifs suivants:

- Adapter les principes de combinaison à des problèmes spécifiques. Cette voie a été ouverte par T. Nipkow [Nip91] pour la combinaison du filtrage dans les théories régulières. Cette démarche est motivée par le fait que la plupart des théories acceptent des algorithmes spécialisés beaucoup plus efficaces que les algorithmes d'unification, dont l'existence même n'est pas toujours garantie.

- Définir un cadre formel - un langage et son interprétation - dans lequel la résolution de contraintes se fasse par combinaison. Ce problème doit être considéré comme une extension de celui résolu pour l'unification et les contraintes équationnelles. Contrairement au premier point, il existe aussi des structures pour lesquelles on dispose de bien plus qu'un simple algorithme d'unification.

  L'objectif de trouver un cadre général à la combinaison de résolveurs de contraintes est très subjectif puisqu'il dépend fortement de comment est défini un algorithme de résolution et pour quel type de contraintes. Nous nous limiterons aux algorithmes de résolution de contraintes symboliques pour lesquels une solution peut toujours être représentée sous la forme d'une substitution des variables par des termes.

- Etendre la combinaison à des théories à signatures non disjointes.

  Cet objectif est particulièrement ambitieux. Il est effet difficile d'imaginer un algorithme d'unification (ou décision de l'unification) modulo l'associativité-commutativité obtenu à partir de ceux existants respectivement pour l'associativité et la commutativité. Notre but est plus modestement d'établir des conditions sur la forme du mélange qui permettent de toujours utiliser les mêmes principes de combinaison.

- Justifier l'utilité des méthodes de combinaison en programmation logique et déduction automatique, par exemple dans le cas où l'on intègre une structure prédéfinie (comme les booléens) et des opérateurs (éventuellement associatifs-commutatifs) définis par des égalités ou des règles de réécriture.

  Pour cela, nous nous placerons dans le cadre de la surréduction contrainte qui a l'avantage d'illustrer une autre forme de construction systématique de procédures de résolution de contraintes.

Plan du travail

Le plan du document est le suivant.

Après cette introduction, le chapitre 1 a pour but de rappeler les concepts utilisés au cours de cette thèse: algèbre universelle, réécriture, unification, résolution de contraintes

symboliques.

Le chapitre 2 est destiné à faire la synthèse des connaissances acquises dans le domaine de la combinaison d'algorithmes d'unification.

Les problèmes liés à l'approche modulaire sont précisés et l'on montre comment ils ont été résolus en fonction des hypothèses faites sur les théories. Nous mettrons plus particulièrement l'accent sur les deux approches suivies à ce jour. L'une dite *par résolution* conduit à un algorithme déterministe, tandis que l'autre, dite *par décision*, permet d'établir un algorithme non-déterministe pour l'unification et la décision de l'unification. Même si l'approche par résolution est pleinement justifiée pour certaines théories, comme par exemple les théories régulières et non effondrantes, elle ne l'est pas vraiment dans le cas général à cause du caractère non-déterministe de la combinaison des solutions. Ces deux approches relèvent pourtant des mêmes principes pour la résolution des conflits inter-théories:

- Une variable instanciée dans une théorie est considérée comme une constante libre dans l'autre théorie.
- Une solution doit satisfaire une *restriction linéaire* basée sur un ordre $<$, interdisant une variable y de figurer dans s si $x =^? s$ et $x < y$.

La seule différence finalement entre les deux approches se situe dans la construction de la restriction linéaire. Elle est incrémentale dans l'approche par résolution et n'est réactualisée que lors d'un conflit inter-théories, alors qu'elle est fixée initialement dans l'approche par décision pour éviter justement que n'apparaissent ces conflits. Nous montrons dans ce chapitre comment l'approche par décision permet d'améliorer sans difficulté l'algorithme déterministe de combinaison de l'unification, notamment en ce qui concerne sa preuve de correction et de terminaison.

Le chapitre 3 a pour objectif d'étendre la combinaison d'algorithmes d'unification à des mélanges de théories non disjointes, problème ouvert jusqu'alors.

Les preuves des algorithmes de combinaison récents sont toutes basées sur l'existence d'un *système de réécriture combiné* convergent décrivant l'égalité dans le mélange et l'utilisation de techniques de réécriture. L'étude approfondie de ces preuves nous a conduit à introduire la notion de théorie *décomposable* en sous-théories partageant éventuellement des symboles. L'intérêt d'une théorie décomposable est de pouvoir se représenter à l'aide d'un système de réécriture combiné formé de sous-systèmes disjoints, c'est-à-dire n'ayant entre eux aucune paire critique. C'est ce qui permet de s'assurer qu'une équation pure peut se résoudre dans sa théorie sans perte de généralité.

Reste le problème de la combinaison des solutions. Une variable est désormais instanciée dans les deux théories si elle l'est par un terme formé de symboles partagés. L'idée consiste alors à imposer, *a priori*, ce type d'instanciation. Sous l'hypothèse de n'avoir qu'un nombre fini d'instanciations par des termes partagés à effectuer, nous construisons deux algorithmes de combinaison complets pour une théorie décomposable. L'un, itératif, s'applique uniquement à l'unification alors que l'autre, non-déterministe, s'applique aussi à la décision de l'unification. Le mélange de théories avec constantes partagées est un cas particulier des deux approches.

Le chapitre 4 est consacré à la combinaison de certains algorithmes de décision. L'approche suivie au cours de ce chapitre, permet entre autre d'obtenir un nouvel algorithme de combinaison du filtrage s'appliquant aussi au problème de décision du filtrage dans l'union de deux théories, ce en quoi il diffère de celui proposé par T. Nipkow pour les théories régulières.

Le problème de filtrage, bien que pouvant se ramener à celui de l'unification avec constantes, est très spécifique puisqu'il existe des théories dont le filtrage est finitaire (resp. décidable) alors que l'unification est infinitaire (resp. indécidable), et inversement. Une autre motivation pour étudier comment combiner des algorithmes de filtrage s'explique par l'efficacité dont ils font preuve en comparaison de ceux disponibles pour l'unification. Mais contrairement aux apparences, ce problème n'est pas une simple instance de celui abordé au chapitre 2 pour l'unification. Il ne suffit pas en effet de remplacer unification par filtrage dans l'algorithme de combinaison pour obtenir un algorithme (complet) de combinaison du filtrage. Ce comportement s'explique par la perte de spécificité que l'on constate en appliquant sans y prendre garde les principes de combinaison sur une équation de filtrage (ou *filtre-équation*) $s =^? t$ où t est par définition un terme clos:

- La purification du terme s avec variables introduit une nouvelle équation résolue $x =^? s'$, qui n'est plus une filtre-équation.

  Cette étape étant, nous semble-t-il, incontournable pour la combinaison, nous allons plutôt chercher à résoudre des problèmes de filtrage *étendu* composé à la fois de filtre-équations et d'équations résolues.

- La purification d'un terme clos t a pour conséquence de le remplacer par un terme avec variables.

  Le terme t sera préalablement remplacé par une forme canonique appelée *forme réduite par couches*. Elle est mise en correspondance pour la première fois avec la notion de forme normale suivant le système de réécriture combiné, qu'elle remplace avantageusement car elle peut se calculer.

- Deux variables x et y faisant abstraction de deux termes avec variables peuvent s'identifier par $x =^? y$ dans une théorie et engendrer ainsi dans l'autre théorie un problème d'unification $s' =^? s''$ si $x =^? s' \wedge y =^? s''$.

  Pour éviter une telle situation, nous allons introduire une classe de théories dites *partiellement linéaires* avec des propriétés de linéarité permettant de ne pas avoir à considérer ces identifications.

L'algorithme de combinaison du filtrage que nous proposons est complet pour la classe des théories partiellement linéaires qui inclut théories linéaires et théories régulières non effondrantes. Nous montrons ensuite que ce résultat ne peut pas être élargi à d'autres théories sans avoir à engendrer un problème d'unification.

Nous concluons ce chapitre en donnant une version non-déterministe de l'algorithme de Nelson et Oppen s'appuyant là encore sur une phase d'identification des variables similaire à celles nécessaires à l'unification, au filtrage et au problème du mot.

Le chapitre 5 est dédié à l'étude de la résolution de contraintes symboliques dans des structures à domaine fini.

Nous commencerons par rappeler les travaux récents de T. Nipkow [Nip90] et W. Büttner [BES+90] concernant la résolution de contraintes dans les algèbres primales.

T. Nipkow a étudié plus particulièrement l'unification en montrant comment étendre les méthodes déjà anciennes concernant les algèbres booléennes. La question fut alors posée de savoir comment combiner un algorithme d'unification dans les algèbres primales à un algorithme d'unification dans une théorie équationnelle. Cette question en amène tout naturellement une autre: existe-t-il une présentation équationnelle à toute algèbre primale? On montre que toute algèbre primale est l'algèbre initiale d'une présentation équationnelle ω-complète, ce qui explique par la même occasion pourquoi il est possible de confondre algèbre de Boole et théorie booléenne dans le contexte de l'unification.

W. Büttner replace les algèbres primales dans le contexte plus général de la résolution de contraintes dans les algèbres finies en donnant un algorithme ayant la propriété de minimiser le nombre de variables introduites dans la solution. L'algorithme que nous proposons s'en inspire fortement mais présente toutefois l'avantage de résoudre symboliquement et uniformément n'importe quel type de contraintes en dissociant le calcul des solutions de celui de la construction d'une solution symbolique. Cela permet ensuite de l'appliquer directement aux pseudo-booléens, une forme de combinaison des booléens et des entiers, et plus généralement à toute autre structure plongeant une ou plusieurs algèbres primales dans les entiers.

Nous disposons donc de résolveurs de contraintes symboliques que l'on va chercher à combiner à la manière de ce qui est fait pour les algorithmes d'unification. Pour cela, nous devons définir un langage de contraintes et une sémantique.

Le chapitre 6 est donc consacré à la définition d'un langage de contraintes combiné pour lequel la résolution symbolique s'effectue par appel à des résolveurs de contraintes dans des sous-structures.

L'objectif est de construire une structure qui soit une extension conservative préservant la validité des contraintes pures. C'est ce qui nous a incité à prendre pour domaine une algèbre de termes quotientée par la théorie équationnelle dont les axiomes sont les égalités valides dans l'une ou l'autre des sous-structures. L'interprétation des prédicats est définie sur ce domaine en fonction de la validité de la formule atomique pure obtenue par normalisation et abstraction des sous-termes étrangers.

Dans le *langage de contraintes combiné* ainsi construit, nous obtenons un résolveur de contraintes symboliques par combinaison des résolveurs de contraintes symboliques avec restriction linéaire.

En guise d'illustration, nous montrons comment adapter pour le combiner le résolveur de contraintes dans les algèbres finies vu au chapitre 5. Pour tenir compte d'une restriction linéaire, nous utilisons un algorithme d'élimination de constante, équivalent à un problème de filtrage sur des termes *éliminants*, que l'on résout grâce à l'algorithme d'unification.

Finalement, nous montrons que le partage des constantes s'applique à la combinaison de structures ayant une intersection des domaines non vide.

Le chapitre 7 présente comment incorporer un résolveur de contraintes dans une structure prédéfinie aux mécanismes de réécriture et de surréduction.

La forme de *spécification basée sur une structure prédéfinie* que nous étudions regroupe des propriétés exprimées équationnellement ou par règles de réécriture avec contraintes

dans la structure prédéfinie. Le choix d'une sémantique opérationnelle s'est naturellement portée sur une relation de réécriture modulo une congruence entre termes hétérogènes qui tient compte de l'égalité dans la structure prédéfinie, des nouveaux symboles et de leurs propriétés équationnelles C. C'est ce qui nous conduit à définir une extension conservative du langage de base qui est un langage combiné.

Le langage de contraintes associé à une spécification basée sur une structure prédéfinie admet alors une procédure de résolution de contraintes fonctionnant sur deux niveaux d'incrémentalité:

- surréduction utilisant la résolution de contraintes dans le langage combiné,
- résolution dans le langage combiné par combinaison d'un résolveur de contraintes du langage de base et d'un algorithme d'unification modulo C.

Mais pour que cette procédure soit correcte et complète, il faut pouvoir vérifier la terminaison et la confluence de la relation de réécriture.

En conclusion, nous résumons les apports et explorons un certain nombre de perspectives de recherche.

1
Notions préliminaires

Nous présentons dans ce chapitre les notions préliminaires aux thèmes abordés au cours de ce document. Nous y introduisons notamment les termes du premier ordre, les Σ-algèbres ainsi que les langages de contraintes. Nous rappelons également divers résultats classiques concernant l'algèbre universelle, la réécriture et l'unification. Pour plus de détails, le lecteur est invité à se reporter à [DJ90a] pour la réécriture et à [JK91] pour l'unification.

Ce chapitre a aussi pour objectif de fixer la terminologie et les notations utilisées par la suite.

1.1 Algèbres Universelles

Deux types de relations seront utilisées dans ce document: les relations d'équivalence et les ordres.

Etant donnée une relation binaire $\rightarrow$ sur un ensemble T dont les éléments sont $t, t', \ldots$, on note $\leftarrow$ la relation inverse de $\rightarrow$, $\xrightarrow{+}$ sa fermeture transitive, $\xrightarrow{*}$ sa fermeture réflexive transitive, $\longleftrightarrow$ sa fermeture symétrique et $\xleftrightarrow{*}$ sa fermeture réflexive, symétrique et transitive. La composition des relations $\rightarrow_1$ et $\rightarrow_2$ est notée $\rightarrow_1 \circ \rightarrow_2$.

Une relation binaire $\sim$ réflexive, symétrique et transitive est une *relation d'équivalence*. Un *ordre* $>$ est une relation binaire irréflexive, antisymétrique et transitive. Un *préordre* $\geq$ est une relation binaire réflexive et transitive.

1.1.1 Algèbres

Soit $\mathcal{S}$ un ensemble d'indices. Un ensemble $\mathcal{S}$-indexé est une famille d'ensembles, un pour chaque $s \in \mathcal{S}$ que l'on notera $(A_s)_{s\in\mathcal{S}}$. Une fonction $\mathcal{S}$-indexée entre deux ensembles $\mathcal{S}$-indexés A et B est une fonction $\alpha : A \rightarrow B$ telle que $\forall a \in A_s,\ \alpha(a) \in B_s$. Un élément a d'un ensemble $\mathcal{S}$-indexé A est *indexé* par $s \in \mathcal{S}$ si $a \in A_s$.

Les éléments de $\mathcal{S}$ seront désormais appelés symboles de sortes.

Définition 1.1 *Une* signature *est un triplet $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ tel que*

- *$\mathcal{S}$ est un ensemble de symboles de sortes.*
- *$\mathcal{F} = \bigcup_{n\geq 0}(\mathcal{F})_n$ est un ensemble de symboles de* fonctions *$\mathcal{S}^* \times \mathcal{S}$-indexé tel que tout symbole $f \in (\mathcal{F})_n$ est indexé par $(s_1, \ldots, s_n, s)$, ce qu'on note $f : s_1, \ldots, s_n \rightarrow s$ et appelle* profil *de f. L'entier n est appelé* arité *de f, notée $|f|$.*

- $\mathcal{P}$ *est un ensemble de symboles de* prédicats $\mathcal{S}^*$*-indexé tel que tout symbole* $p \in \mathcal{P}$ *a un unique index* $(s_1, \ldots, s_n)$*, ce qu'on note* $p : s_1, \ldots, s_n$ *et appelle* profil *de* p*. L'entier* n *est l'arité de* p*, notée* $|p|$*.*

Un symbole de fonction a une arité fixe mais plusieurs profils alors qu'un symbole de prédicat a une arité unique et un seul profil. Nous avons choisi d'introduire immédiatement une information de sorte et la possibilité de surcharger des symboles de fonctions. Cela facilitera ensuite la définition des algèbres à sortes ordonnées.

Les symboles de fonctions d'arité nulle sont notés $a, b, c, \ldots$ et appelés *constantes*, les autres seront notés $f, g, h, \ldots$.

Soit $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ une signature.

Définition 1.2 *Une* $(\mathcal{S}, \mathcal{F})$*-algèbre* $\mathcal{A}$ *est un domaine* $A = (A_s)_{s \in \mathcal{S}}$ *muni d'un ensemble de fonctions* $\mathcal{F}_\mathcal{A}$ *tel que pour tout symbole* $f \in \mathcal{F}$ *correspond une fonction* $f_\mathcal{A} : A^{|f|} \to A$ *pour laquelle si* f *est de profil* $f : s_1, \ldots, s_n \to s$ *et* $(a_1, \ldots, a_n) \in A_{s_1} \times \cdots \times A_{s_n}$ *alors* $f_\mathcal{A}(a_1, \ldots, a_n) \in A_s$*.*

Lorsque $\mathcal{S}$ est un singleton, cet ensemble sera souvent omis et l'on parlera simplement de $\mathcal{F}$-algèbre. Cette remarque est valable pour toutes les notations qui seront définies à l'aide d'un ensemble de sortes $\mathcal{S}$.

Une application qui respecte la structure de $(\mathcal{S}, \mathcal{F})$-algèbre est un homomorphisme.

Définition 1.3 *Etant données deux* $(\mathcal{S}, \mathcal{F})$*-algèbres* $\mathcal{A}$ *et* $\mathcal{B}$*, une application* $\mathcal{S}$*-indexée* θ *de* A *vers* B *telle que*

$$\forall f \in \mathcal{F},\ \forall a_1, \ldots, a_n \in A,\ \theta(f_\mathcal{A}(a_1, \ldots, a_n)) = f_\mathcal{B}(\theta(a_1), \ldots, \theta(a_n))$$

est un homomorphisme de $\mathcal{A}$ *vers* $\mathcal{B}$*.*

Définition 1.4 *Une algèbre* $\mathcal{A}$ *d'une classe de* $(\mathcal{S}, \mathcal{F})$*-algèbres est* libre *sur un ensemble* $\mathcal{S}$*-indexé* $\mathcal{X}$ *si pour toute application* $\mathcal{S}$*-indexée* $\theta : \mathcal{X} \to \mathcal{B}$*, il existe un unique homomorphisme* $\nu : \mathcal{A} \to \mathcal{B}$ *tel que* θ *et* ν *coïncident sur* $\mathcal{X}$*.*

Une algèbre libre est unique à un isomorphisme près si elle existe. On peut donc parler sans ambiguité de la $(\mathcal{S}, \mathcal{F})$-algèbre libre sur $\mathcal{X}$.

Par la suite, les éléments de $\mathcal{X}$ seront appelés variables et notés $x, y, z, \ldots$

1.1.2 Termes

Définition 1.5 *Etant donnés une signature* $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ *et un ensemble* $\mathcal{S}$*-indexé* $\mathcal{X}$ *de symboles de variables, l'ensemble des* termes *(du premier ordre)* $T(\mathcal{S}, \mathcal{F}, \mathcal{X})$ *est le plus petit ensemble contenant* $\mathcal{X}$ *tel que* $f(t_1, \ldots, t_n)$ *est dans* $T(\mathcal{S}, \mathcal{F}, \mathcal{X})$ *si* $f \in \mathcal{F}$ *et* $t_1, \ldots, t_n \in T(\mathcal{S}, \mathcal{F}, \mathcal{X})$*.*

L'ensemble des termes sans variables, appelé termes clos *est noté* $T(\mathcal{S}, \mathcal{F})$*.*

Il est clair que $T(\mathcal{S}, \mathcal{F}, \mathcal{X})$ peut être muni d'une structure de $(\mathcal{S}, \mathcal{F})$-algèbre, et notée $\mathcal{T}(\mathcal{S}, \mathcal{F}, \mathcal{X})$. Dans la classe de toutes les $(\mathcal{S}, \mathcal{F})$-algèbres, $\mathcal{T}(\mathcal{S}, \mathcal{F}, \mathcal{X})$ est l'algèbre libre sur $\mathcal{X}$. Une *assignation* $\mathcal{S}$-indexée $\alpha : \mathcal{X} \to \mathcal{A}$ s' étend de façon unique en un homomorphisme

$\underline{\alpha}$ de $\mathcal{T}(\mathcal{S},\mathcal{F},\mathcal{X})$ vers $\mathcal{A}$. La restriction d'une assignation α à une ensemble de variables $V \subseteq \mathcal{X}$ est notée $\alpha_{|V}$. Cette notation s'étend aux ensembles d'assignations. L'ensemble de toutes les assignations est noté $ASS_A^{\mathcal{X}}$, ou simplement ASS_A lorsque il n'y a pas d'ambiguité.

Définition 1.6 *L'ensemble $\mathcal{V}(t)$ des variables d'un terme t est défini inductivement par:*

- $\mathcal{V}(t) = \emptyset$ *si* $t \in (\mathcal{F})_0$,
- $\mathcal{V}(t) = \{x\}$ *si* $t \in \mathcal{X}$,
- $\mathcal{V}(t) = \bigcup_{i=1}^{n} \mathcal{V}(t_i)$ *si* $t = f(t_1, \ldots, t_n)$.

Un terme peut être vu comme un arbre fini étiqueté.

Arbres, sous-arbres, positions

Définition 1.7 *Soit $\mathbf{N}_+$ l'ensemble des entiers strictement positifs, $\mathbf{N}_+^*$ le monoïde libre engendré par $\mathbf{N}_+$, ϵ le mot vide et . l'opération de concaténation. Pour tous $p, q \in \mathbf{N}_+^*$, p est un préfixe de q, ce que l'on note $p \leq q$, s'il existe $q' \in \mathbf{N}_+^*$ tel que $q = p.q'$. p est un prefixe strict de q, noté $p < q$, si $p \leq q$ et $p \neq q$. Si $p \not\leq q$ et $q \not\leq p$, p et q sont disjointes ou incomparables, ce qu'on note $p \bowtie q$.*

Un arbre sur $\mathcal{F} \cup \mathcal{X}$ est une application t d'une partie non vide $\mathcal{P}os(t)$ de $\mathbf{N}_+^*$ dans $\mathcal{F} \cup \mathcal{X}$ telle que :

1. *$\mathcal{P}os(t)$ est clos par préfixe.*
2. *Pour tout $p \in \mathcal{P}os(t)$ et tout $i \in \mathbf{N}_+$, $p.i \in \mathcal{P}os(t)$ si et seulement si $t(p) = f \in \mathcal{F}$ et $1 \leq i \leq |f|$.*

$\mathcal{P}os(t)$ est appelé ensemble des positions de t, et t est fini si $\mathcal{P}os(t)$ l'est. La taille $|t|$ d'un terme t est dans ce cas le cardinal de $\mathcal{P}os(t)$.

L'ensemble des arbres finis sur $\mathcal{F} \cup \mathcal{X}$ peut être muni naturellement d'une structure de $(\mathcal{S},\mathcal{F})$-algèbre isomorphe à $\mathcal{T}(\mathcal{S},\mathcal{F},\mathcal{X})$. On parlera donc dorénavant indifférement d'arbre ou de terme.

Définition 1.8

- *Pour tout terme t et toute position $p \in \mathcal{P}os(t)$, $t(p)$ est appelé symbole à la position p dans t. $t(\epsilon)$ est également appelé symbole de tête de t.*
- *On appelle sous-terme de t à la position $p \in \mathcal{P}os(t)$, le terme noté $t_{|p}$, et défini par $t_{|p}(q) = t(p.q)$. $t_{|p}$ est un sous-terme strict de t si $p \neq \epsilon$.*
- *Si $t(\epsilon) = f \in \mathcal{F}$, on notera t sous la forme $f(t_{|1}, \ldots, t_{|n})$ où $n = |f|$.*
- *Une position p de t est variable si $t(p) \in \mathcal{X}$. L'ensemble des positions variables de t est noté $\mathcal{VP}pos(t)$ alors que l'ensemble des positions non variables de t est noté $\mathcal{FP}os(t)$.*
  Une position p de t est constante si $t(p) \in (\mathcal{F})_0$. L'ensemble des positions constantes de t est noté $\mathcal{CP}os(t)$.

La notation $t[s]_\omega$ est utilisée pour signifier que t contient s comme sous-terme à la position ω et la notation $t[\omega \hookleftarrow s]$ pour faire remarquer que le sous-terme $t_{|\omega}$ a été remplacé par s dans t.

1.1.3 Substitutions

Une *substitution* est un endomorphisme de $\mathcal{T}(\mathcal{S}, \mathcal{F}, \mathcal{X})$ dont la restriction à $\mathcal{X}$ est l'identité presque partout, c'est-à-dire sauf sur un sous-ensemble fini de $\mathcal{X}$. Les substitutions seront notées par des lettres grecques minuscules, $\sigma, \mu, \gamma, \phi, \ldots$ La notation postfixe, i.e. $t\sigma$, est utilisée pour l'application d'une substitution σ à un terme t. Une substitution bijective est un *renommage*. Une substitution σ est *idempotente* si $\sigma\sigma = \sigma$.

On appelle *domaine* d'une substitution σ l'ensemble $\mathcal{D}om(\sigma) = \{x \in \mathcal{X} \mid x\sigma \neq x\}$ et *codomaine* d'une substitution σ l'ensemble $\mathcal{R}an(\sigma) = \{x\sigma \mid x \in \mathcal{D}om(\sigma)\}$. L'ensemble des variables introduites par une substitution σ est $\mathcal{VR}an(\sigma) = \cup_{x \in \mathcal{D}om(\sigma)} \mathcal{V}(x\sigma)$.

La restriction de σ à un ensemble de variables V, notée $\sigma_{|V}$, est définie par $x\sigma_{|V} = x\sigma$ si $x \in V$ et $x\sigma_{|V} = x$ sinon.

Le préordre de filtrage ou de *subsomption* est défini par $s \leq t$ s'il existe une substitution σ telle que $s\sigma = t$.

1.1.4 Algèbres à sortes ordonnées

Lorsque l'ensemble des sortes $\mathcal{S}$ d'une signature $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ est partiellement ordonné par $\leq$, on dit que Σ est une signature *ordo-sortée*.

Définition 1.9 *Soit $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ une signature pour laquelle $\mathcal{S}$ est un ensemble partiellement ordonné par $\leq$. Une Σ-algèbre* ordo-sortée *est une $(\mathcal{S}, \mathcal{F})$-algèbre telle que le domaine $A = (A_s)_{s \in \mathcal{S}}$ vérifie*

$$\forall s, s' \in \mathcal{S},\ A_s \subseteq A_{s'} \text{ si } s \leq s'.$$

Il faut cette fois construire un ensemble de termes *bien formés*.

Un Σ-*terme* de sorte s est soit une variable de sorte s' avec $s' \leq s$, soit de la forme $f(t_1, \ldots, t_n)$ où $f : s_1, \ldots, s_n \to s'$ est un profil de Σ avec $s' \leq s$, et chaque t_i est un Σ-terme de sorte s_i, pour $i = 1, \ldots, n$. L'ensemble des Σ-termes construits sur un ensemble de variables $\mathcal{X}$ est noté $T(\Sigma, \mathcal{X})$. On écrit en général $t : s$ pour préciser qu'un terme t est de sorte s.

Une signature Σ est *régulière* si tout Σ-terme t possède une plus petite sorte que l'on désigne par $ls(t)$. On restreint ici notre attention aux signatures ordo-sortées régulières.

L'ensemble des termes $T(\Sigma, \mathcal{X})$ peut être muni naturellement d'une structure de Σ-algèbre, notée $\mathcal{T}(\Sigma, \mathcal{X})$, qui est l'algèbre libre sur $\mathcal{X}$ de la classe de toutes les Σ-algèbres ordo-sortées.

Une Σ-substitution est un endomorphisme de $\mathcal{T}(\Sigma, \mathcal{X})$.

1.1.5 Validité

La notion de validité est définie pour une $(\mathcal{S}, \mathcal{F})$-algèbre, et donc a fortiori pour les Σ-algèbres ordo-sortées.

Définition 1.10 *Une paire de termes (l, r) est appelé* égalité, axiome *ou* équation *suivant le contexte, et notée $(l = r)$.*

Définition 1.11 *Une $(\mathcal{S},\mathcal{F})$-algèbre $\mathcal{A}$* valide *une égalité $s=t$, noté $\mathcal{A}\models s=t$ ou plus simplement $s=_{\mathcal{A}} t$ si $\forall\alpha\in ASS_A^{\mathcal{X}}$, $\underline{\alpha}(s)=\underline{\alpha}(t)$ et* satisfait *une égalité $s=t$ s'il existe $\alpha\in ASS_A^{\mathcal{X}}$ telle que $\underline{\alpha}(s)=\underline{\alpha}(t)$. Une $(\mathcal{S},\mathcal{F})$-algèbre $\mathcal{A}$ est un modèle d'un ensemble d'égalités E si elle valide toutes les égalités de E.*

On note $\mathcal{T}h(\mathcal{A})$ l'ensemble des égalités valides dans une $(\mathcal{S},\mathcal{F})$-algèbre $\mathcal{A}$ et $\mathcal{M}od(E)$ la classe des $(\mathcal{S},\mathcal{F})$-algèbres qui sont modèles de E.

1.2 Théories équationnelles

Soit E un ensemble d'égalités de $T(\mathcal{S},\mathcal{F},\mathcal{X})$, appelés dans ce contexte, *axiomes*.

Définition 1.12 *Etant donnée une signature $\Sigma=(\mathcal{S},\mathcal{F},\mathcal{P})$, une* présentation équationnelle *est un couple $(\mathcal{F},E)$ telle que E est un ensemble d'axiomes de $T(\mathcal{S},\mathcal{F},\mathcal{X})$.*

Le problème de validité dans $\mathcal{M}od(E)$ consiste à décider si une égalité $s=t$ est valide dans tout modèle de E. Ce problème peut se ramener à des considérations syntaxiques.

Définition 1.13 *Etant donnée une présentation équationnelle $(\mathcal{F},E)$, on appelle* théorie équationnelle engendrée par $(\mathcal{F},E)$ *ou E-égalité et on note $=_E$ la plus petite congruence sur $T(\mathcal{S},\mathcal{F},\mathcal{X})$ contenant toutes les égalités $(l\sigma=r\sigma)$ où $(l=r)$ est un axiome de E et σ une substitution quelconque.*

Le théorème suivant est le fondement de la logique équationnelle. Il relie le problème sémantique de la validité d'une égalité dans une classe de modèles au problème syntaxique de la E-égalité.

Théorème 1.1 *(Birkhoff [Bir35], Complétude du raisonnement équationnel) $s=t$ est valide dans $\mathcal{M}od(E)$ si et seulement si $s=_E t$.*

La E-égalité peut encore être obtenue par le *remplacement d'égal par égal* décrit ci-après.

Définition 1.14 *Etant donné un ensemble d'axiomes E, on note $\longleftrightarrow_E$ la relation binaire symétrique sur $T(\mathcal{S},\mathcal{F},\mathcal{X})$ définie par $s\longleftrightarrow_E t$ si il existe un axiome $(l=r)$ de E, une position ω de s et une substitution σ tels que $s_{|\omega}=l\sigma$ et $t=s[r\sigma]_\omega$.*

Remarque 1.1 $s=_E t\Leftrightarrow s\overset{*}{\longleftrightarrow}_E t$.

Par abus de langage et de notation, on confond souvent la théorie équationnelle $=_E$, la présentation équationnelle $(\mathcal{F},E)$ et l'ensemble des axiomes E.

L'ensemble des classes de congruence de E dans $T(\mathcal{S},\mathcal{F},\mathcal{X})$ peut être muni naturellement d'une structure de $\mathcal{F}$-algèbre, notée $\mathcal{T}(\mathcal{S},\mathcal{F},\mathcal{X})/=_E$, qui est l'algèbre libre sur $\mathcal{X}$ de la classe des $(\mathcal{S},\mathcal{F})$-algèbres modèles de E.

On donne à présent les caractérisations des principales classes de théories.

Définition 1.15 *Une théorie E est*

- effondrante *s'il existe un terme non variable E-égal à une variable,*
- simple *si aucun terme n'est E-égal à l'un de ses sous-termes stricts,*
- régulière *si pour tout couple* (s,t) *de termes E-égaux,* $\mathcal{V}(s) = \mathcal{V}(t)$*,*
- linéaire *si pour tout couple* (s,t) *de termes E-égaux,* s *et* t *sont linéaires,*
- triviale *si tous ses modèles ont un support réduit à un élément, c'est-à-dire si* $x = y$ *est valide dans* $\mathcal{M}od(E)$*.*

Toutes les théories évoquées dans le reste du document sont supposées non triviales ou consistantes.

Rappelons les définitions des axiomes les plus couramment utilisés:

Abbréviation	Nom	Définition
$A(f)$	Associativité	$f(f(x,y),z) = f(x,f(y,z))$
$C(f)$	Commutativité	$f(x,y) = f(y,x)$
$Dd(f,g)$	Distributivité droite	$f(g(x,y),z) = g(f(x,z),f(y,z))$
$Dg(f,g)$	Distributivité gauche	$f(z,g(x,y)) = g(f(z,x),f(z,y))$
$D(f,g)$	Distributivité	$Dg(f,g) \cup Dd(f,g)$
$I(f)$	Idempotence	$f(x,x) = x$
$InvD(*,e)$	Inverse Droit	$x * i(x) = e$
$InvG(*,e)$	Inverse Gauche	$i(x) * x = e$
$Nd(*,e)$	Neutre droit	$x * e = x$
$Ng(*,e)$	Neutre gauche	$e * x = x$

et les théories mélangeant plusieurs axiomes:

	Nom	Définition
AG	Groupe abélien	$A(*), C(*), InvD(*,e), Ng(*,e)$
BR	Anneau booléen	$A(+), C(+), Nd(+,0), InvG(+,0),$
		$A(*), C(*), Nd(*,1),\ I(*),\ Dd(*,+)$
DA		$D(*,+), A(+)$

Le concept de théorie équationnelle s'étend aux ensembles d'axiomes E de $T(\Sigma, X)$, où Σ désigne une signature ordo-sortée. On parle alors de Σ-axiomes *ordo-sortés* et de présentation équationnelle *ordo-sortée*, notée (Σ, E). La définition d'une *théorie équationnelle engendrée par un ensemble d'axiomes E ordo-sortés* est obtenue à partir de la définition d'une théorie équationnelle en remplaçant les notions de termes et substitutions par celles de Σ-termes et Σ-substitutions.

La généralisation aux Σ-algèbres du théorème de complétude du raisonnement équationnel nécessite quelques précautions comme celle de supposer que toute sorte $s \in \mathcal{S}$ est non vide, c'est-à-dire qu'il existe au moins un terme de cette sorte s.

L'extension du concept de *remplacement d'égal par égal* au cas ordo-sorté n'est pas complètement immédiate.

Définition 1.16 *Etant donné un ensemble de* Σ*-axiomes* E*, on note* $\longleftrightarrow_E$ *la relation binaire symétrique sur* $T(\Sigma, \mathcal{X})$ *définie par* $s \longleftrightarrow_E t$ *s'il existe un* Σ*-axiome* $(l = r)$ *de* E*, une position* ω *de* s *et une substitution* σ *tels que*

- $s_{|\omega} = l\sigma$,
- *il y a une sorte* s *pour laquelle* $s[\omega \hookleftarrow x] \in T(\Sigma, \mathcal{X})$ *et* $x, l\sigma, r\sigma$ *sont des termes de sorte* s,
- $t = s[r\sigma]_\omega$.

Dans notre contexte, les Σ-axiomes de E sont supposés universellement quantifiés sur $\mathcal{X}$.

1.3 Définition et propriétés des systèmes de réécriture

Nous donnons d'abord quelques propriétés abstraites liées aux relations binaires dont nous aurons besoin par la suite. Ces notions seront essentiellement utilisées pour les systèmes de réécriture.

1.3.1 Relations binaires sur un ensemble

Soit $\rightarrow$ une relation binaire sur un ensemble T, avec par exemple $T = T(\mathcal{S}, \mathcal{F}, \mathcal{X})$.

Définition 1.17 *Un élément* t *de* T *est* réductible *par* $\rightarrow$ *s'il existe* t' *dans* T *tel que* $t \rightarrow t'$. *Dans le cas contraire, il est irréductible. On appelle forme normale de* t *tout élément* t' *irréductible tel que* $t \xrightarrow{*} t'$. *Lorsque un terme* t *a une unique forme normale, celle-ci est notée* $t\downarrow$.

Propriété de Church-Rosser

La question que l'on se pose est de savoir si $t \xleftrightarrow{*} t'$. L'idéal serait de calculer une forme normale de chacun des éléments et de tester si elles sont égales. Cela n'est possible que si d'une part une forme normale existe pour tout élément, et si d'autre part elle est unique. Les formes normales existent dès que $\rightarrow$ *termine*, c'est-à-dire qu'il n'existe pas de suite infinie $(t_i)_{i\geq 1}$ d'éléments de T telle que $t_1 \rightarrow t_2 \rightarrow \cdots$. Dans le cas où une forme normale existe, son unicité est assurée par la propriété de Church-Rosser ou par la confluence qui est une propriété équivalente.

Définition 1.18

1. $\rightarrow$ *a la propriété de Church-Rosser si*
$$\xleftrightarrow{*} \subseteq \xrightarrow{*} \circ \xleftarrow{*}$$
2. $\rightarrow$ *est confluente si*
$$\xleftarrow{*} \circ \xrightarrow{*} \subseteq \xrightarrow{*} \circ \xleftarrow{*}$$
3. $\rightarrow$ *est localement confluente si*
$$\leftarrow \circ \rightarrow \subseteq \xrightarrow{*} \circ \xleftarrow{*}$$

4. $\rightarrow$ est convergente si $\rightarrow$ termine et a la propriété de Church-Rosser.

La confluence est une propriété difficile à tester. En pratique, le test de confluence se fait localement grâce au théorème suivant:

Théorème 1.2 *(Newman 1942 [New42]) Si $\rightarrow$ termine, alors les propriétés suivantes sont équivalentes :*

1. $\rightarrow$ a la propriété de Church-Rosser,

2. $\rightarrow$ est confluente,

3. $\rightarrow$ est localement confluente,

4. $\forall t, t' \in T : t \overset{*}{\longleftrightarrow} t' \Leftrightarrow t\downarrow = t'\downarrow$.

Propriété de Church-Rosser modulo une relation d'équivalence

On considère $\rightarrow_R$ et $\longleftrightarrow_E$ deux relations binaires sur T, dont l'une, $\longleftrightarrow_E$, est une relation d'équivalence. Le problème que l'on aborde ici est de déterminer comment décider de $t\ (\longleftrightarrow_R \cup \longleftrightarrow_E)^*\ t'$ en fonction d'une procédure de décision pour $\longleftrightarrow_E$.

On note $\longleftrightarrow_{R\cup E}$ la relation $\longleftrightarrow_R \cup \longleftrightarrow_E$, et $\rightarrow_{R/E}$ la relation $\overset{*}{\longleftrightarrow}_E \circ \rightarrow_R \circ \overset{*}{\longleftrightarrow}_E$ simulant la relation induite par $\rightarrow_R$ sur les classes d'équivalence de $\overset{*}{\longleftrightarrow}_E$.

La première idée serait de calculer comme précédemment des formes normales par la relation $\rightarrow_{R/E}$ puis de tester leur égalité modulo la relation d'équivalence $\overset{*}{\longleftrightarrow}_E$. Il n'est plus possible d'avoir unicité des formes normales mais seulement unicité modulo une classe d'équivalence. La propriété de Church-Rosser doit donc être adaptée en conséquence.

Définition 1.19 *$\rightarrow_{R/E}$ a la propriété de Church-Rosser modulo E si*

$$\overset{*}{\longleftrightarrow}_{R/E} \subseteq \overset{*}{\longrightarrow}_{R/E} \circ \overset{*}{\longleftrightarrow}_E \circ \overset{*}{\longleftarrow}_{R/E}$$

Dans la pratique, on simule la relation $\rightarrow_{R/E}$ par une relation $\rightarrow_S$ plus faible satisfaisant $\rightarrow_R \subseteq \rightarrow_S \subseteq \rightarrow_{R/E}$. On a alors une propriété de Church-Rosser modulo E pour $\rightarrow_S$, ainsi qu'une notion de confluence qui n'implique plus la propriété de Church-Rosser. Pour la récupérer, il faut introduire une nouvelle propriété appelée cohérence qui assure que si t est irréductible par $\rightarrow_S$ et $t \overset{*}{\longleftrightarrow}_E t'$, alors t' est aussi irréductible par $\rightarrow_E$. On note $t\downarrow_S$ une forme normale de t par la relation $\downarrow_S$.

Définition 1.20

1. $\rightarrow_S$ a la propriété de Church-Rosser modulo E si

$$\overset{*}{\longleftrightarrow}_{R/E} \subseteq \overset{*}{\longrightarrow}_S \circ \overset{*}{\longleftrightarrow}_E \circ \overset{*}{\longleftarrow}_S$$

2. $\rightarrow_S$ est confluente modulo E si

$$\overset{*}{\longleftarrow}_S \circ \overset{*}{\longrightarrow}_S \subseteq \overset{*}{\longrightarrow}_S \circ \overset{*}{\longleftrightarrow}_E \circ \overset{*}{\longleftarrow}_S$$

3. $\rightarrow_S$ est cohérente avec $\longleftrightarrow_E$ modulo E si

$$\overset{*}{\longleftarrow}_S \circ \overset{*}{\longleftrightarrow}_E \subseteq \overset{*}{\longrightarrow}_S \circ \overset{*}{\longleftrightarrow}_E \circ \overset{*}{\longleftarrow}_S$$

4. $\rightarrow_S$ *est localement confluente avec* $\rightarrow_R$ *modulo* E *si*

$$\leftarrow_S \circ \rightarrow_R \subseteq \stackrel{*}{\longrightarrow}_S \circ \stackrel{*}{\longleftrightarrow}_E \circ \stackrel{*}{\longleftarrow}_S$$

5. $\rightarrow_S$ *est localement cohérente avec* $\longleftrightarrow_E$ *modulo* E *si*

$$\leftarrow_S \circ \longleftrightarrow_E \subseteq \stackrel{*}{\longrightarrow}_S \circ \stackrel{*}{\longleftrightarrow}_E \circ \stackrel{*}{\longleftarrow}_S$$

6. $\rightarrow_S$ *est convergente modulo* E *si* $\rightarrow_S$ *termine et* $\rightarrow_S$ *a la propriété de Church-Rosser modulo* E.

Théorème 1.3 *(J.-P. Jouannaud & H. Kirchner [JK86]) Si* $\rightarrow_{R/E}$ *termine, alors les propriétés suivantes sont équivalentes :*

1. $\rightarrow_S$ *a la propriété de Church-Rosser modulo* E,
2. $\rightarrow_S$ *est confluente modulo* E *et* $\rightarrow_S$ *est cohérente avec* $\longleftrightarrow_E$ *modulo* E,
3. $\rightarrow_S$ *est localement confluente avec* $\rightarrow_R$ *modulo* E *et localement cohérente avec* $\longleftrightarrow_E$ *modulo* E,
4. $\forall t, t' \in T : t \stackrel{*}{\longleftrightarrow}_{R \cup E} t' \;\Leftrightarrow\; t \downarrow_S \stackrel{*}{\longleftrightarrow}_E t' \downarrow_S$.

Ordres nœthériens

On rappelle qu'un ordre $>$ est une relation binaire irréflexive, antisymétrique et transitive.

Définition 1.21 *Un ordre* $>$ *sur* T *est nœthérien s'il n'existe pas de suite infinie* $(t_i)_{i \geq 1}$ *d'éléments de* T *telle que* $t_1 > t_2 > \ldots$

La construction d'ordres nœthériens peut éventuellement se faire par extension.
L'extension lexicographique permet par exemple de comparer des uplets.

Définition 1.22 *Soient* n *ensembles ordonnés* $(T_i, >_i)_{i=1,\ldots,n}$. *Le* produit cartésien ordonné lexicographiquement $(T_1 \times \cdots \times T_n, >_{lex})$ *est défini par :*

$$(s_1, \ldots, s_n) >^{lex} (t_1, \ldots, t_n)$$

s'il existe i, $1 \leq i \leq n$ *tel que* $s_i >_i t_i$ *et* $\forall j$, $1 \leq j \leq i$, $s_j = t_j$.

Proposition 1.1 *Si* $>_i$ *est nœthérien pour* $i = 1, \ldots, n$ *alors* $>^{lex}$ *l'est aussi.*

Pour comparer des objets de tailles arbitraires, on introduit la notion de *multi-ensemble* sur T qui est une application de T vers $\mathbf{N}$. Un ordre sur T peut être facilement étendu à un ordre sur les multi-ensembles sur T.

Définition 1.23 *Soit* $>$ *un ordre sur* T. *L'*ordre multi-ensemble *est défini par* $L >^{mult} M$ *si* $L \neq M$ *et* $(M(y) > L(y) \Rightarrow \exists x \in T, x > y$ *et* $L(x) > M(x))$.

Proposition 1.2 *Si* $>$ *est nœthérien alors* $>^{mult}$ *l'est aussi.*

1.3.2 Systèmes de réécriture

L'idée centrale de la réécriture est d'imposer une direction dans l'utilisation d'axiomes.

Définition 1.24

- *Une règle de réécriture est une paire de termes orientée, noté $g \to d$, où g est le membre gauche de la règle et d son membre droit.*
- *Un système de réécriture sur les termes est un ensemble de règles de réécriture.*
- *La relation de réécriture $\to_R$ associée à un système de réécriture R est définie par : $t \to_R t'$ s'il existe une position ω dans t, une règle $g \to d$ dans R et une substitution σ telles que $t_{|\omega} = g\sigma$ et $t' = t[d\sigma]_\omega$. Si on veut préciser la position, la règle et la substitution, alors on écrira $t \to_{R,\omega,g\to d,\sigma} t'$.*
- *L'ensemble des variables d'une règle $g \to d$, noté $\mathcal{V}(g \to d)$, est défini par $\mathcal{V}(g) \cup \mathcal{V}(d)$.*
- *Une règle de réécriture est linéaire à gauche si son membre gauche est linéaire. Un système de réécriture est linéaire à gauche si toutes ses règles le sont.*
- *La tête d'une règle est la tête de son membre gauche.*

Par application du théorème 1.2, si la relation de réécriture $\to_R$ est convergente, alors pour décider de l'égalité $t \overset{*}{\longleftrightarrow}_R t'$, il suffit de calculer les formes normales $t \downarrow_R$ et $t' \downarrow_R$ puis de les comparer.

Définition 1.25 *Un système de réécriture R est convergent (resp. est confluent, termine) si la relation de réécriture $\to_R$ est convergente (resp. est confluente, termine).*

1.3.3 Terminaison

La convergence requiert la terminaison. Cette propriété est indécidable en général même pour un système de réécriture réduit à une seule règle linéaire à gauche, comme l'a montré M. Dauchet [Dau89]. On peut néanmoins prouver la terminaison dans certain cas au moyen d'un ordre sur les termes.

Définition 1.26 *Un ordre de réécriture sur les termes est un ordre $>$ stable par contexte et par instanciation: pour tous termes t, t', u et toute substitution σ,*

$$t > t' \Longrightarrow u[t\sigma]_p > u[t'\sigma]_p$$

Un ordre de réduction est un ordre de réécriture nœthérien.

On assure la terminaison de la réécriture en orientant les règles de manière à ce que toute règle $g \to d$ vérifie $g > d$ où $>$ est un ordre de réduction sur les termes.

Théorème 1.4 *[Lan77] Le système de réécriture R termine si et seulement si $\to_R$ est contenu dans un ordre de réduction.*

De nombreux auteurs ont décrit des ordres de réduction sur les termes. Parmi les plus connus, citons l'ordre de Knuth-Bendix ou *kbo* [KB70], les ordres sur les chemins [Pla78, Der82, BP85, JLR82] ou encore les interprétations polynomiales [Lan75, BCL87].

Un ordre de réduction total contient l'ordre sous-terme. Dans le cas contraire, si $t_{|\omega} > t$ pour un terme t et une position ω, alors il existe une chaîne infinie décroissante $t > t[t]_\omega > t[t[t]_\omega]_\omega > \ldots$. On appelle ordre de *simplification*, un ordre de réduction contenant l'ordre sous-terme.

Théorème 1.5 *[Der82] Soit $\mathcal{F}$ un ensemble fini de symboles de fonctions. Un système de réécriture R termine s'il existe un ordre de simplification $>$ tel que pour toute règle $g \to d$ de R, $g > d$.*

Les ordres de simplifications peuvent être construits à partir d'un ordre sur les symboles de fonctions $\mathcal{F}$ appelé *précédence*. Des exemples sont décrits ci-après.

Définition 1.27 *Soit $>_\mathcal{F}$ une précédence sur $\mathcal{F}$. L'*ordre multi-ensemble sur les chemins *$>_{rpo}$ (RPO) est défini sur les termes clos par $s = f(s_1, .., s_n) >_{rpo} t = g(t_1, \ldots, t_m)$ si l'une des conditions suivantes est satisfaite.*

1. *$f = g$ et $\{s_1, \ldots, s_n\} >_{rpo}^{mult} \{t_1, \ldots, t_m\}$*
2. *$f >_\mathcal{F} g$ et $\forall j \in \{1, \ldots, m\}, s >_{rpo} t_j$*
3. *$\exists i \in \{1, \ldots, n\}$ tel que $s_i >_{rpo} t$ ou $s_i \sim t$*
   avec $\sim$ l'equivalence à une permutation près des sous-termes.

Proposition 1.3 *[Der82] L'ordre multi-ensemble sur les chemins est un ordre de simplification.*

Lorsque la précédence $>_\mathcal{F}$ est totale (sur $\mathcal{F}$), $>_{rpo}$ l'est aussi.

Cette définition s'étend aux termes avec variables à la condition de ne pas pouvoir comparer deux variables ou une variable et un symbole de fonction.

L'ordre suivant compare les sous-termes lexicographiquement.

Définition 1.28 *Soit $>_\mathcal{F}$ une précédence sur $\mathcal{F}$. L'*ordre lexicographique sur les chemins *$>_{lpo}$ (LPO) est défini sur les termes clos par $s = f(s_1, .., s_n) >_{lpo} t = g(t_1, \ldots, t_m)$ si l'une des conditions suivantes est satisfaite:*

1. *$f = g$ et $(s_1, \ldots, s_n) >_{lpo}^{lex} (t_1, \ldots, t_m)$ et $\forall j \in \{1, \ldots, m\}, s >_{lpo} t_j$*
2. *$f >_\mathcal{F} g$ et $\forall j \in \{1, \ldots, m\}, s >_{lpo} t_j$*
3. *$\exists i \in \{1, \ldots, n\}$ tel que $s_i >_{lpo} t$ ou $s_i = t$.*

Proposition 1.4 *[KL80] L'ordre lexicographique sur les chemins est un ordre de simplification.*

Lorsque la précédence $>_\mathcal{F}$ est totale (sur $\mathcal{F}$), $>_{lpo}$ l'est aussi.

L'ordre lexicographique sur les chemins s'étend aux termes avec variables de la même manière que l'ordre multi-ensemble.

Au cours de ce document, nous n'utiliserons que les ordres de simplification totaux sur les termes clos, l'orientation d'un couple de termes clos ne pouvant pas dans ce cas échouer.

Pour plus de détails concernant la terminaison, nous renvoyons le lecteur à [Der87].

1.4 Langages de contraintes

Après avoir abordé le problème de validité, on s'intéresse à celui de la satisfaisabilité.

Définition 1.29 *Soit $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ une signature. Une* Σ-structure *$\mathcal{A}$ est une $(\mathcal{S}, \mathcal{F})$-algèbre $\mathcal{A}$ de domaine $A = \bigcup_{s\in\mathcal{S}} A_s$ munie d'un ensemble de relations $\mathcal{P}_\mathcal{A}$ tel que pour tout $p \in \mathcal{P}$ de profil $p : s_1, \ldots, s_n$ correspond une relation $p_\mathcal{A}$ de $A_{s_1} \times \cdots \times A_{s_n}$.*

Une signature Σ est toujours supposée contenir le prédicat d'égalité $=$, celui-ci est interprété comme l'identité dans $\mathcal{A}$.

La définition d'un langage de contraintes donnée ci-dessous est celle qu'on trouve dans [KKR90] et [Smo89]. Les contraintes sont des formules du premier ordre construites sur les termes $T(\mathcal{S}, \mathcal{F}, \mathcal{X})$, les prédicats $\mathcal{P}$ et les connecteurs logiques $\top, \wedge, \neg, \exists$.

Définition 1.30 *Soit $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P})$ une signature. Un* langage de contraintes *$CL_\mathcal{K}[\Sigma, \mathcal{X}]$ est donné par:*

- *Un ensemble $\mathcal{X}$ de variables $\mathcal{S}$-indexé et infini dénombrable.*
- *Un ensemble de* contraintes *$C[\Sigma, \mathcal{X}]$ défini comme étant le plus petit ensemble tel que*
  - $\top \in C[\Sigma, \mathcal{X}]$,
  - $p^?(t_1, \ldots, t_n) \in C[\Sigma, \mathcal{X}]$ *si* $p \in \mathcal{P}$ *et* $t_1, \ldots, t_n \in T(\mathcal{S}, \mathcal{F}, \mathcal{X})$,
  - $c \wedge c' \in C[\Sigma, \mathcal{X}]$ *si* $c, c' \in C[\Sigma, \mathcal{X}]$.
  - $\neg c \in C[\Sigma, \mathcal{X}]$ *si* $c \in C[\Sigma, \mathcal{X}]$,
  - $\exists x : c \in C[\Sigma, \mathcal{X}]$ *si* $c \in C[\Sigma, \mathcal{X}]$.
- *L'ensemble $\mathcal{V}(c)$ des variables d'une contrainte c est défini par:*
  - $\mathcal{V}(\top) = \emptyset$,
  - $\mathcal{V}(p^?(t_1, \ldots, t_n)) = \bigcup_{i=1}^n \mathcal{V}(t_i)$,
  - $\mathcal{V}(c \wedge c') = \mathcal{V}(c) \cup \mathcal{V}(c')$,
  - $\mathcal{V}(\neg c) = \mathcal{V}(c)$,
  - $\mathcal{V}(\exists x : c) = \mathcal{V}(c) \backslash \{x\}$.
- *Une Σ-structure $\mathcal{K}$.*
- *Une application $Sol_\mathcal{K}$ qui associe à chaque terme ou contrainte c l'ensemble des assignations $Sol_\mathcal{K}(c)$ défini comme suit:*
  - $Sol_\mathcal{K}(\top) = \{\alpha \in ASS_K^\mathcal{X}\}$,
  - $Sol_\mathcal{K}(t) = \{\alpha \mid \underline{\alpha}(t) \in K\}$,
  - $Sol_\mathcal{K}(p^?(t_1, \ldots, t_j)) = \{\alpha \in ASS_K^\mathcal{X} \mid (\underline{\alpha}(t_1), \ldots, \underline{\alpha}(t_j)) \in p_\mathcal{K}\}$,
  - $Sol_\mathcal{K}(c \wedge c') = Sol_\mathcal{K}(c) \cap Sol_\mathcal{K}(c')$,
  - $Sol_\mathcal{K}(\neg c) = ASS_K^\mathcal{X} \backslash Sol_\mathcal{K}(c)$,
  - $Sol_\mathcal{K}(\exists x : c) = \{\alpha \in ASS_K^\mathcal{X} \mid \exists \beta \in ASS_K^\mathcal{X},\ \alpha_{|\mathcal{X}\backslash\{x\}} = \beta_{|\mathcal{X}\backslash\{x\}}$ *et* $\beta \in Sol_\mathcal{K}(c)\}$.

Une assignation de $Sol_{\mathcal{K}}(c)$ est une solution *de c dans $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$. Une contrainte c est* satisfaisable *dans $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$ si $Sol_{\mathcal{K}}(c) \neq \emptyset$. Une contrainte c est dite* valide *dans $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$, noté $\mathcal{K} \models c$, si $Sol_{\mathcal{K}}(c) = ASS_{\mathcal{K}}^{\mathcal{X}}$. Deux contraintes c et c' sont V-*équivalentes *si $Sol_{\mathcal{K}}(c)_{|V} = Sol_{\mathcal{K}}(c')_{|V}$ ou tout simplement* équivalentes *lorsque $V = \mathcal{X}$.*

Lorsque κ est une classe de Σ-structures $\mathcal{K}$, on note $CL_{\kappa}[\Sigma, \mathcal{X}]$ la classe des langages de contraintes $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$. Une contrainte est valide *dans $CL_{\kappa}[\Sigma, \mathcal{X}]$ si elle est valide dans tous les langages de contraintes $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$ de la classe.*

Les abbréviations suivantes sont utilisées pour définir les connecteurs logiques $\vee, \forall, \bot$:

- $c \vee c' = \neg(\neg c \wedge \neg c')$,
- $\forall x : c = \neg(\exists x : \neg c)$,
- $\bot = \neg(\top)$.

Une contrainte *équationnelle* est une contrainte formée avec le seul prédicat d'égalité.

Au cours de ce document, nous allons principalement nous intéresser aux contraintes c sans quantification et aux contraintes existentiellement quantifiées

$$\exists x_1, \ldots, x_n : c = \exists x_1 : (\ldots (\exists x_n : c) \ldots)$$

abrégées en $\exists \vec{x} : c$ où c est une contrainte sans quantification et $\vec{x} = \{x_1, \ldots, x_n\}$. Les contraintes universellement quantifiées sont abrégées de même en $\forall \vec{y} : c$.

> **Nota:** *La quantification d'une contrainte est toujours mentionnée explicitement. Ainsi, c dénote systématiquement une contrainte sans quantification dans $\exists \vec{x} : c$ ou $\forall \vec{y} : c$.*

La résolution de contraintes sans quantification est dite *décidable* s'il existe un algorithme permettant de décider si une contrainte arbitraire $\exists \mathcal{V}(c) : c$ est valide dans le langage.

1.4.1 Règles de transformation

Une contrainte $\exists \vec{x} : c$ est équivalente à une disjonction de conjonction de contraintes atomiques existentiellement quantifiées

$$\bigvee_{j \in J} \exists \vec{x_j} : c_{j,1} \wedge \cdots \wedge c_{j,n_j}$$

grâce aux égalités valides dans l'algèbre des contraintes (cf. figure 1.1).

Un algorithme de résolution de contraintes symboliques *à base de règles* est un ensemble de règles de transformation muni d'un contrôle sur leurs applications. Une règle

$$\exists \vec{x} : c \mapsto\!\!\!\!\!\rightarrow \exists \vec{x'} : c'$$

Pour tous termes s, t et toutes contraintes a, b, c:

$$
\begin{array}{rcl}
(s =^? t) & = & (t =^? s) \\
(a \wedge b) \wedge c & = & a \wedge (b \wedge c) \\
a \wedge b & = & b \wedge a \\
a \wedge \top & = & a \\
a \wedge \bot & = & \bot \\
a \wedge a & = & a \\
(a \vee b) \vee c & = & a \vee (b \vee c) \\
a \vee b & = & b \vee a \\
a \vee \bot & = & a \\
a \vee \top & = & \top \\
a \vee a & = & a \\
a \wedge (b \vee c) & = & (a \wedge b) \vee (a \wedge c) \\
\exists x : (a \vee b) & = & (\exists x : a) \vee (\exists x : b) \\
(\exists x : a) \wedge (\exists y : b) & = & \exists x, y : a \wedge b \text{ si } x \notin \mathcal{V}(b), y \notin \mathcal{V}(a)
\end{array}
$$

Figure 1.1. Egalités entre contraintes

transforme une conjonction $\exists \vec{x} : c$ en une disjonction de conjonctions

$$\bigvee_{j \in J} \exists \vec{x'_j} : c'_j.$$

Nous remplacerons parfois cette disjonction par un ensemble fini de règles *non-déterministes*

$$(\exists \vec{x} : c \mapsto\!\!\!\!\!\rightarrow \exists \vec{x'_j} : c'_j)_{j \in J}$$

pour $|J| > 1$, chacune d'entre elles transformant la conjonction $\exists \vec{x} : c$ en une autre conjonction $\exists \vec{x'_j} : c'_j$.

Une règle non-déterministe (nommée regle^+ lors de sa définition) est appliquée suivant une stratégie "don't choose" alors qu'une règle standard est appliquée suivant une stratégie "don't care".

Une règle $\exists \vec{x} : c \mapsto\!\!\!\!\!\rightarrow \exists \vec{x'} : c'$ est

- *correcte* si $Sol_{\mathcal{K}}(\exists \vec{x'} : c') \subseteq Sol_{\mathcal{K}}(\exists \vec{x} : c)$,
- *complète* si $Sol_{\mathcal{K}}(\exists \vec{x} : c) \subseteq Sol_{\mathcal{K}}(\exists \vec{x'} : c')$.

Une règle non-déterministe $(\exists \vec{x} : c \mapsto\!\!\!\!\!\rightarrow \exists \vec{x'_j} : c'_j)_{j \in J}$ est

- *correcte* si $\bigcup_{j \in J} Sol_{\mathcal{K}}(\exists \vec{x'_j} : c'_j) \subseteq Sol_{\mathcal{K}}(\exists \vec{x} : c)$,

- *complète* si $Sol_{\mathcal{K}}(\exists\vec{x} : c) \subseteq \bigcup_{j\in J} Sol_{\mathcal{K}}(\exists\vec{x'_j} : c'_j)$.

Une règle correcte et complète est *adéquate*. Nous dirons aussi qu'elle *préserve* l'ensemble des solutions.

L'ensemble des *nouvelles variables* introduites par une règle $\exists\vec{x} : c \mapsto\!\!\!\!\rightarrow \exists\vec{x'} : c'$ est $\mathcal{V}(c')\backslash\mathcal{V}(c)$. Ces variables sont toujours supposées existentiellement quantifiées, i.e.

$$\mathcal{V}(c')\backslash\mathcal{V}(c) = \vec{x'}\backslash\vec{x}$$

afin de ne pas *capturer* leurs solutions. Par conséquent, $\mathcal{V}(\exists\vec{x} : c) = \mathcal{V}(\exists\vec{x'} : c')$.

Un ensemble de règles de transformation peut être vu comme un système de réécriture dans le langage des contraintes. L'application d'une règle $\exists\vec{x} : c \mapsto\!\!\!\!\rightarrow \exists\vec{x'} : c'$, notée le plus souvent

$$\frac{\exists\vec{x} : c}{\exists\vec{x'} : c'}$$

par souci de lisibilité, consiste à remplacer une sous-contrainte égale modulo les égalités de la figure 1.1 au "numérateur" par la contrainte au "dénominateur". Nous n'utiliserons par la suite qu'une règle de transformation ayant une contrainte existentiellement quantifiée au "numérateur": il s'agit justement de la règle qui consiste à éliminer la quantification lorsque c'est possible. Toutes les autres règles transforment une contrainte non quantifiée par une autre pouvant être éventuellement existentiellement quantifiée s'il y a introduction de nouvelles variables.

1.4.2 Solutions symboliques

Une solution symbolique (ou unificateur lorsque la contrainte est une équation $s=^? t$) est une substitution permettant de schématiser un ensemble de solutions.

Une $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$-substitution σ est une substitution *bien formée* vérifiant

$$\forall(x : s) \in \mathcal{D}om(\sigma),\ \forall\alpha \in Sol_{\mathcal{K}}(x\sigma),\ \underline{\alpha}(x\sigma) \in K_s.$$

L'ensemble des $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$-substitutions est noté $SUBST_{\mathcal{K}}$.

Définition 1.31 *L'ensemble des* solutions symboliques *d'une $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$-contrainte c, noté $SS_{\mathcal{K}}(c)$, est défini par:*

- $SS_{\mathcal{K}}(\top) = SUBST_{\mathcal{K}}$,
- $SS_{\mathcal{K}}(p^?(t_1, \ldots, t_n)) = \{\sigma \in SUBST_{\mathcal{K}} \mid \mathcal{K} \models p(t_1\sigma, \ldots, t_n\sigma)\}$,
- $SS_{\mathcal{K}}(c \wedge c') = SS_{\mathcal{K}}(c) \cap SS_{\mathcal{K}}(c')$,
- $SS_{\mathcal{K}}(\neg c) = SUBST_{\mathcal{K}}\backslash SS_{\mathcal{K}}(c)$,
- $SS_{\mathcal{K}}(\exists x : c) = \{\sigma \in SUBST_{\mathcal{K}} \mid \exists\phi \in SUBST_{\mathcal{K}},\ \sigma_{|\mathcal{X}\backslash\{x\}} = \phi_{|\mathcal{X}\backslash\{x\}}\ et\ \phi \in SS_{\mathcal{K}}(c)\}$.

A partir de maintenant et pour ne pas compliquer la notation, nous nous autoriserons à ne plus toujours préciser le langage de contraintes mais simplement la Σ-structure qui le définit. Dans ce qui suit, toutes les substitutions sont bien formées.

Une substitution ϕ est une *$\mathcal{K}$-instance* sur $V \subseteq \mathcal{X}$ d'une substitution σ, ce que l'on note $\sigma \leq^V_{\mathcal{K}} \phi$, s'il existe une substitution μ telle que $\forall x \in V,\ x\phi =_{\mathcal{K}} x\sigma\mu$.

Définition 1.32 *Un ensemble de substitutions est un* ensemble complet de solutions *(symboliques) d'une $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$-contrainte c, noté $CSS_{\mathcal{K}}(c)$, si*

1. *(Protection des variables)* $\forall \sigma \in CSS_{\mathcal{K}}(c),\ \mathcal{D}om(\sigma) \cap \mathcal{VR}an(\sigma) = \emptyset$.
2. *(Correction)* $CSS_{\mathcal{K}}(c) \subseteq SS_{\mathcal{K}}(c)$.
3. *(Complétude)* $\forall \phi \in SS_{\mathcal{K}}(c), \exists \sigma \in CSS_{\mathcal{K}}(c),\ \sigma \leq_{\mathcal{K}}^{\mathcal{V}(c)} \phi$.

Lorsque deux substitutions de $CSS_{\mathcal{K}}(c)$ sont incomparables par $\leq_{\mathcal{K}}^{\mathcal{V}(c)}$, l'ensemble complet des solutions $CSS_{\mathcal{K}}(c)$ est minimal. *Si $CSS_{\mathcal{K}}(c)$ est réduit à un singleton alors cette solution est dite* principale, *et notée $mgs_{\mathcal{K}}(c)$.*

L'ensemble $SS_{\mathcal{K}}(c)$ des solutions symboliques d'une $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$-contrainte c est un $CSS_{\mathcal{K}}(c)$.

Etant donnée une substitution idempotente $\sigma = \{x_i \mapsto t_i\}_{i \in I}$, on note $\hat{\sigma}$ la contrainte équationnelle $\bigwedge_{i \in I} x_i =^? t_i$ qui est dite en *forme résolue* car σ est un $CSS_{\mathcal{K}}(\hat{\sigma})$. Cette remarque est valable pour n'importe quelle Σ-structure $\mathcal{K}$ puisque σ est bien formée. Lorsque I est vide alors $\hat{\sigma}$ dénote $\bot$. A contrario, $\hat{\sigma}$ dénote $\top$ si σ est la substitution identité.

Nous allons nous restreindre à certains langages de contraintes, à savoir ceux dont l'existence d'une solution coïncide avec l'existence d'une solution symbolique.

Définition 1.33 *Un langage de contraintes $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$, ou plus simplement la Σ-structure $\mathcal{K}$, est* engendré par les termes *si*

$$\forall k \in K,\ \exists t \in T(\mathcal{S}, \mathcal{F}),\ \exists \alpha \in Sol_{\mathcal{K}}(t),\ k = \underline{\alpha}(t).$$

Corollaire 1.1 *Si $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$ est un langage de contraintes engendré par les termes et c une contrainte de ce langage alors $Sol_{\mathcal{K}}(c) = \emptyset$ si et seulement si $SS_{\mathcal{K}}(c) = \emptyset$.*

On suppose dorénavant qu'un langage de contraintes est engendré par les termes.

La notion de solution symbolique et d'ensemble complet de solutions est motivée par le fait suivant qui stipule qu'un ensemble de complet de solutions schématise l'ensemble des solutions.

Remarque 1.2 *Soit c une $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$-contrainte. On a*

- $\forall \sigma \in CSS_{\mathcal{K}}(c),\ Sol_{\mathcal{K}}(\hat{\sigma}) \subseteq Sol_{\mathcal{K}}(c)$ *(Correction),*
- $\forall \alpha \in Sol_{\mathcal{K}}(c), \exists \sigma \in CSS_{\mathcal{K}}(c),\ \alpha \in Sol_{\mathcal{K}}(\hat{\sigma})$ *(Complétude).*

Définition 1.34 *Un langage de contraintes est de* type

- unitaire *si toute contrainte admet un ensemble complet de solutions contenant au plus une solution,*
- finitaire *si toute contrainte c admet un ensemble complet fini de solutions,*
- infinitaire *si pour toute contrainte, il existe un ensemble complet minimal de solutions et s'il existe une contrainte dont tout ensemble complet de solutions est infini,*

- nullaire *s'il existe une contrainte n'admettant pas d'ensemble complet minimal de solutions.*

Définition 1.35 *Un* résolveur de contraintes symbolique *est un algorithme permettant de calculer un ensemble complet de solutions pour une contrainte arbitraire du langage.*

Remarque 1.3 *Si le langage de contraintes $CL_{\mathcal{K}}[\Sigma, \mathcal{X}]$ est finitaire alors toute contrainte c est équivalente à la contrainte équationnelle*

$$\bigvee_{\sigma \in CSS_{\mathcal{K}}(c)} \exists \mathcal{V}(\hat{\sigma}) \backslash \mathcal{V}(c) : \hat{\sigma}.$$

En général, la connaissance du type d'un langage de contraintes est une conséquence de l'élaboration d'un résolveur de contraintes symbolique. Par abus de terminologie, nous dirons que la résolution de contraintes est d'un certain type (finitaire ou unitaire) si le langage est de ce type et si de surcroît un résolveur symbolique est connu.

1.5 Unification dans une théorie équationnelle

L'*unification* est la résolution de contraintes équationnelles $\exists \vec{x} : c$ sans négation dans un langage dont la structure est $\mathcal{T}(\mathcal{F}, \mathcal{X}) / =_E$.

L'unification modulo une théorie E étant indécidable en général, chaque théorie est un cas particulier qu'il s'agit de traiter de manière adéquate. Des algorithmes d'unification pour bon nombre de théories intéressantes en pratique ont néanmoins été décrits dans la littérature. On pourra citer entre autres la plus simple d'entre elles, la théorie vide, pour laquelle des algorithmes d'unification ont été proposés par de nombreux auteurs dont Robinson [Rob65], A. Martelli & U. Montanari [MM82], J. Corbin & M. Bidoit [CB83]; la théorie AC (M. Stickel [Sti75], Livesey & Siekmann [LS76]); la théorie AG des groupes abéliens (D. Lankford, G. Butler & B. Brady [LBB84]) et bien d'autres encore. Des algorithmes moins spécifiques ont également été présentés pour certaines classes de théories comme par exemple les théories *commutatives* ou *monoïdales* décrites par F. Baader [Baa89] et W. Nutt [Nut90], dans lesquelles l'unification se ramène à la résolution d'équations dans un semi-anneau. Cette classe inclut en particulier les monoïdes commutatifs et les groupes abéliens.

Il existe aussi des procédures complètes d'énumération des solutions d'un problème d'unification modulo une théorie quelconque. De telles procédures ont été décrites par J. Gallier & W. Snyder [GS89] et D. Dougherty & P. Johann [DJ90b]. Ces procédures ne terminent pas en général, mais sont néanmoins des procédures de semi-décision pour l'unification.

La surréduction introduite par M. Fay [Fay79] est elle aussi une méthode générale d'unification modulo une théorie engendrée par un système de réécriture convergent. Cette méthode de résolution a le même inconvénient de ne pas terminer en général.

L'objectif de cette section est de rappeler les concepts fondamentaux liés à l'unification en les replaçant dans le contexte plus général de la résolution de contraintes.

Un *problème d'unification* est une contrainte équationnelle $\exists \vec{x} : \Gamma$ où Γ représente une conjonction d'équations, ou *système* d'équations, notées $s =^? t$. Une solution de $\exists \vec{x} : \Gamma$ est une solution symbolique de $\exists \vec{x} : \Gamma$ modulo E. L'ensemble des solutions symboliques de

$\exists\vec{x} : \Gamma$, ou *unificateurs*, est noté $SU_E(\exists\vec{x} : \Gamma)$. L'ensemble des solutions de $\exists\vec{x} : \Gamma$ est engendré par instanciation des éléments d'un ensemble complet de solutions symboliques ou *ensemble complet d'unificateurs* de $\exists\vec{x} : \Gamma$, noté $CSU_E(\exists\vec{x} : \Gamma)$. La dimension de tout ensemble complet minimal d'unificateurs permet de définir le type de l'unification dans E, ou par abus de langage le type de E.

Exemple 1.1

- *La théorie vide ($\emptyset$) est unitaire.*
- *la théorie présentée par les axiomes*

$$\begin{aligned}(x+y)+z &= x+(y+z)\\ x+y &= y+x\\ x+0 &= x\end{aligned}$$

  est unitaire si les seuls symboles de la signature sont $+$ et 0, finitaire sinon.
- *La théorie présentée par $C(+)$ est finitaire mais non unitaire.*
- *La théorie présentée par $A(+)$ est infinitaire. Par exemple, si a est une constante, $a+x$ et $x+a$ n'admettent qu'un ensemble complet d'unificateurs et celui-ci est infini.*
- *La théorie présentée par les axiomes $1 * x = x$, $f(x * y) = f(y)$ est nullaire comme l'ont montré F. Fages et G. Huet [FH83].*

Nous parlerons plus précisement d'unification *élémentaire* lorsque les problèmes équationnels ne contiennent que des symboles de fonctions de la signature de E. Cette distinction est nécessaire car le type de l'unification est, en général, différent de celui de l'unification avec constantes libres [BHSS90], où sont autorisés des symboles d'arité nulle ne figurant pas dans la signature de E.

L'approche à base de règles de l'unification [JK91] a permis une certaine standardisation dans la construction des algorithmes. Le processus de résolution est vu comme la transformation progressive d'un problème d'unification en un autre équivalent jusqu'à ce qu'une forme résolue soit atteinte. Ces transformations peuvent être *déterministes* ou *non-déterministes*. Dans le premier cas, on obtient une seul système d'équations alors que dans le second, plusieurs règles sont à appliquer en parallèle pour obtenir finalement une disjonction de systèmes d'équations.

Un algorithme d'unification à base de règles présente l'avantage de séparer complètement les règles de transformation correctes et complètes de leur utilisation, ou *contrôle*. La preuve d'un tel algorithme s'effectue en quatre étapes:

- *Correction*. Il s'agit de prouver que les règles de transformation sont correctes et complètes.
- *Complétude*. On vérifie ensuite que les formes normales (par rapport aux règles de transformation) sont les formes résolues.
- *Terminaison*. Il faut trouver et choisir un contrôle ou des contrôles pour lesquels l'application des règles de transformation termine.
- *Equité*. Les formes normales sont atteintes pour le contrôle choisi.

La preuve de *Complétude* est souvent routinière. Dans l'absolu, une forme résolue est un problème d'unification pour lequel il est possible d'extraire directement un unificateur.

Définition 1.36 *(Forme résolue) Un problème d'unification en* forme résolue *est* $\bot$, $\top$ *ou*

$$\exists \vec{z} : x_1 =^? t_1 \wedge \cdots x_n =^? t_n$$

si

- $\vec{z} \cap \{x_1, \ldots, x_n\} = \emptyset$,
- $\forall 1 \leq i, j \leq n, x_i \neq x_j$,
- $\forall 1 \leq i, j \leq n, x_i \notin \mathcal{V}(t_j)$,

La substitution $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ est un unificateur principal du problème $\exists \vec{z} : x_1 =^? t_1 \wedge \cdots x_n =^? t_n$ en forme résolue.

Remarque 1.4 *Si P est un problème d'unification en forme résolue alors l'application de la règle* EQE *(cf. figure 1.2) termine et transforme $\exists \vec{x} : P$ en une forme résolue équivalente.*

La stratégie employée systématiquement ici consiste à mettre de côté la quantification existentielle jusqu'à ce qu'une forme résolue soit atteinte pour le problème d'unification sans quantification. La règle d'élimination des quantificateurs existentiels transforme ensuite le problème d'unification avec quantification en une forme résolue.

La principale difficulté dans la preuve d'un algorithme d'unification à base de règles est sûrement l'aspect *Terminaison*. Pour résoudre ce problème spécifique, on cherchera d'abord la plus grande classe de contrôles qui terminent. Un exemple d'algorithme d'unification terminant pour tout contrôle est donné en figure 1.3 pour le cas de la théorie vide. La notion de forme résolue utilisée dans ce contexte ne correspond pas directement à une substitution.

Définition 1.37 *(Forme séquentiellement résolue) Un problème d'unification en* forme séquentiellement résolue *est* $\bot$, $\top$ *ou*

$$x_1 =^? t_1 \wedge \cdots \wedge x_n =^? t_n$$

si

- $\forall 1 \leq i \leq j \leq n, x_i \neq x_j$,
- $\forall 1 \leq i \leq j \leq n, x_i \notin \mathcal{V}(t_j)$,
- $\forall 1 \leq i \leq n, t_i \in \mathcal{X} \Rightarrow \forall 1 \leq j \leq n, x_i \notin \mathcal{V}(t_j)$.

Remarque 1.5 *Si P est un problème d'unification en forme séquentiellement résolue, alors l'application des règles* EQE *et* Remplacement *(cf. figure 1.2) termine et transforme $\exists \vec{x} : P$ en une forme résolue équivalente.*

La règle de remplacement est en elle même un obstacle à la terminaison pour tout contrôle. C'est pourquoi elle n'est utilisée qu'en post-traitement, lorsque par exemple plus aucune autre règle de la figure 1.3 ne s'applique.

EQE

$$\frac{\exists x : (\exists \vec{x'} : P \wedge x =^? t)}{\exists \vec{x'} : P} \quad \textbf{si } x \notin \mathcal{V}(P)$$

Remplacement

$$\frac{P \wedge x =^? t}{P\{x \mapsto t\} \wedge x =^? t} \quad \textbf{si } x \in \mathcal{V}(P) \text{ et } t \notin \mathcal{X}$$

Figure 1.2. Ensemble de règles appliquées en post-traitement

Proposition 1.5 *L'application des règles $\mathcal{RV}$ à un problème d'unification P termine pour tout contrôle et retourne un problème d'unification équivalent à P dans la théorie vide en forme séquentiellement résolue.*

La théorie vide ($\emptyset$) a la particularité de retourner au plus un unificateur principal: il n'y a pas de règle non-déterministe.

La finalité des algorithmes d'unification est d'être intégrée à des mécanismes de déduction comme celui que nous allons décrire succintement.

1.6 Complétion de systèmes de réécriture

Lorsqu'un système de réécriture n'est pas convergent, il est parfois possible d'obtenir la convergence en ajoutant des règles. On dit alors qu'on complète le système de réécriture. La première procédure de complétion fut décrite par Knuth & Bendix [KB70] en 1970. Ce processus de complétion est basé sur le calcul de paires critiques et le test de confluence locale. Intuitivement, une paire critique correspond à deux réécritures possibles d'un même terme.

Définition 1.38

- *Un terme t se superpose (ou se chevauche) sur un terme t' à une position $\omega \in \mathcal{P}os(t')$ s'il existe une substitution σ telle que $t\sigma = t'_{|\omega}\sigma$.*
- *Soient $g \to d$ et $l \to r$ deux règles avec des ensembles disjoints de variables, tel que l se superpose sur g à une position non variable ω de g avec l'unificateur principal σ. Le terme* superposé *$g\sigma$ produit une* paire critique *(p, q) définie par $p = (g[\omega \hookleftarrow r])\sigma$ et $q = d\sigma$.*
- *Une paire critique (p, q) est* joignable *si $p \xrightarrow{*}_R \circ \xleftarrow{*}_R q$.*

On notera que le calcul d'une paire critique requiert l'unification dans la théorie vide.

Théorème 1.6 *[Hue80] Un système de réécriture R est localement confluent si et seulement si toute paire critique de R est joignable.*

Ce théorème donne un moyen implicite de compléter un système R, qui consiste à engendrer les paires critiques et à tester si elles sont confluentes. Si ce n'est pas le cas, la

Suppression
$$\frac{P \wedge s =^? s}{P}$$

Decomposition
$$\frac{P \wedge f(s_1, \ldots, s_n) =^? f(t_1, \ldots, t_n)}{P \wedge s_1 =^? t_1 \wedge \cdots \wedge s_n =^? t_n}$$

VarRep
$$\frac{P \wedge x =^? y}{P\{x \mapsto y\} \wedge x =^? y} \quad \textbf{si } x, y \in \mathcal{V}(P) \text{ et } x \neq y$$

Fusion
$$\frac{P \wedge x =^? s \wedge x =^? t}{P \wedge x =^? s \wedge s =^? t} \quad \textbf{si } 0 < |s| \leq |t|$$

Conflit
$$\frac{P \wedge f(s_1, \ldots, s_n) =^? g(t_1, \ldots, t_m)}{\bot} \quad \textbf{si } f \neq g$$

Cycle
$$\frac{P \wedge x_1 =^? t_1[x_2]_{p_1} \wedge \cdots x_n =^? t_n[x_1]_{p_n}}{\bot} \quad \textbf{si } p_i \neq \epsilon \text{ pour un } i \in \{1, \ldots, n\}$$

Figure 1.3. Ensemble de règles $\mathcal{RV}$ pour l'unification dans la théorie vide

paire critique est orientée en une nouvelle règle de réécriture quand c'est possible. Il faut ensuite calculer les paires critiques entre la nouvelle règles et les règles déjà présentes dans le système. Cette procédure de complétion peut être décrite par un ensemble de règles d'inférence suivant un formalisme introduit par Bachmair, Dershowitz et Hsiang [BDH86]. Mais elle peut s'avérer incapable d'orienter une paire critique. C'est le cas par exemple de $(x + y, y + x)$ qu'aucun ordre de réduction ne permet d'orienter. L'idée est alors de partager l'ensemble des axiomes en d'une part un système de réécriture R et d'autre part un système d'axiomes E. On raisonne alors modulo la congruence engendrée par E, notée $=_E$. D'après le théorème 1.3, il s'agit d'utiliser une relation de réécriture qui simule la relation de réécriture $\rightarrow_{R/E}$ sur les classes de congruence. Nous n'allons détailler que celle étudiée par J.-P. Jouannaud et H. Kirchner et proposée à l'origine par Peterson et Stickel [PS81].

Définition 1.39 *La relation de réécriture $\rightarrow_{R,E}$ associée à un ensemble de règles R et un ensemble d'axiomes E est définie par : $t \rightarrow_{R,E} t'$ s'il existe une position ω dans t, une règle $g \rightarrow d$ dans R et une substitution σ telles que $t_{|\omega} =_E g\sigma$ et $t' = t[d\sigma]_\omega$. Si on veut préciser la position, la règle et la substitution, alors on écrit $t \rightarrow_{R,E,\omega,g\rightarrow d,\sigma} t'$.*

Les tests de confluence et de cohérence locale modulo E, comme dans le cas précédent, sont basés sur des calculs de paires critiques. On a alors à la fois des paires critiques de confluence, issues de superpositions entre des membres gauches de règles de réécriture, et

des paires critiques de cohérence, issues de superpositions entre le membre gauche d'une règle et un membre d'axiome.

Définition 1.40

- *Un terme t se superpose (ou se chevauche) modulo E sur un terme t' à une position $\omega \in \mathcal{P}os(t')$ s'il existe une substitution σ telle que $t\sigma =_E t'_{|\omega}\sigma$.*
- *Soient $g \to d$ et $l \to r$ deux règles avec des ensembles disjoints de variables, tel que l se superpose sur g à une position non variable ω de g. L'ensemble*

$$\{((g[\omega \hookleftarrow r])\sigma, d\sigma) \mid \sigma \in CSU_E(g_{|\omega} =^? l)\}$$

  est appelé ensemble complet de paires critiques de confluence de $l \to r$ sur $g \to d$ à la position ω.
- *Soient $(g = d) \in E$ et $l \to r$ une règle avec des ensembles disjoints de variables, tel que l se superpose sur g à une position non variable $\omega \neq \epsilon$ de g. L'ensemble*

$$\{((g[\omega \hookleftarrow r])\sigma, d\sigma) \mid \sigma \in CSU_E(g_{|\omega} =^? l)\}$$

  est appelé ensemble complet de paires critiques de cohérence de $l \to r$ sur $g \to d$ à la position ω.
- *On note $CP(R, E)$ la réunion de tous les ensembles complets de paires critiques modulo E d'une règle de R sur une règle de R ou un axiome de E.*
- *Une paire critique (p, q) est* joignable *modulo E si $p \xrightarrow{*}_{R,E} \circ \xleftrightarrow{*}_E \xleftarrow{*}_{R,E} q$.*

Théorème 1.7 *Si $\to_{R/E}$ termine, alors $\to_{R,E}$ a la propriété de Church-Rosser modulo E si et seulement si toutes les paires critiques de $CP(R, E)$ sont joignables.*

La procédure de complétion d'un système de réécriture pour la réécriture $\to_{R,E}$ est alors identique à celle décrite dans le cas où E est vide. Les seules différences résident d'une part dans le test d'égalité qui se fait modulo E, et d'autre part dans le calcul des paires critiques qui nécessite l'unification modulo E. Pour plus de détails sur cette procédure de complétion, le lecteur est convié à se reporter à [JK86, Kir85b].

1.7 La modularité et son vocabulaire

Le principe de modularité est omniprésent en informatique. Il consiste à décomposer un problème en sous-problèmes plus faciles à résoudre. Dans le cadre de la déduction automatique, on dit qu'une propriété est *modulaire* si le fait qu'elle soit vraie dans deux théories T_1 et T_2 implique qu'elle le soit encore dans l'union des théories $T_1 \cup T_2$. On parle d'union disjointe lorsque T_1 et T_2 sont formées sur des signatures disjointes. La notion de théorie doit être ici prise au sens large. Il peut s'agir d'une théorie équationnelle présentée par un système de réécriture. On s'est ainsi beaucoup intéressé à la modularité de propriétés de systèmes de réécriture comme la confluence locale, la terminaison et la convergence. Rappelons les principaux résultats acquis pour l'union disjointe de deux systèmes de réécriture $R_1 \cup R_2$.

La confluence [Toy87] est une propriété modulaire mais la terminaison ne l'est pas [Toy86]. Des résultats positifs concernant la terminaison sont obtenus en rajoutant des hypothèses supplémentaires sur la forme des règles. Une règle est *effondrante* si son membre droit est une variable. Une règle est *duplicante* si une variable a plus d'occurrences dans le membre droit que dans le membre gauche. M. Rusinowitch [Rus87] a montré que la terminaison est modulaire si les systèmes R_1 et R_2 ne contiennent pas de règle duplicante ou si elles ne contiennent pas de règle effondrante. A. Middeldorp [Mid89] a ensuite montré qu'il suffisait que l'un des systèmes R_i ne contienne ni règle effondrante ni règle duplicante pour que la terminaison soit modulaire. Il a été montré récemment par Kurihara et Ohuchi [KO90] que la terminaison est modulaire dans la classe des systèmes de réécriture pour lesquels la terminaison est prouvée par un ordre de simplification.

A. Middeldorp et Y. Toyama [MT91] se sont intéressés à la modularité de certaines propriétés lorsque les système de réécriture partagent des symboles de fonctions qui sont des constructeurs, c'est-à-dire des symboles n'apparaissant pas en tête des membres gauches des règles. La convergence est, par exemple, une propriété modulaire pour des systèmes de réécriture respectant la discipline de constructeurs.

Au cours de ce document, il sera traité de la modularité de l'unification, du problème du mot, du filtrage et de la résolution de contraintes en général. Le problème de modularité posé est alors de déterminer si l'existence d'algorithmes permettant de résoudre des contraintes dans E_1 et E_2 implique l'existence d'un algorithme permettant de résoudre le même type de contraintes mais dans l'union $E_1 \cup E_2$. Cela conduit indirectement à se demander comment combiner deux algorithmes existants pour en construire un troisième de façon systématique pour le *mélange* $E_1 \cup E_2$ des théories E_1 et E_2.

La combinaison d'algorithmes d'unification dans le mélange de deux théories disjointes a fait l'objet de beaucoup d'études et résultats que nous rappelons au chapitre 2. Nous montrons de notre côté comment les étendre à des signatures non disjointes au chapitre 3.

Mais tous les problèmes liés à la modularité ou à la combinaison d'algorithmes utilisent un même vocabulaire et des notions fondamentales que nous allons définir de la façon la plus générale possible. Il suffira ensuite de les spécialiser en fonction des problèmes traités.

Les contraintes et termes considérés dans le contexte de la combinaison sont formés sur l'union de deux signatures $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1, \mathcal{P}_1)$ et $\Sigma_2 = (\mathcal{S}_2, \mathcal{F}_2, \mathcal{P}_2)$, notée $\Sigma = \Sigma_1 \cup \Sigma_2$ et définie par $\Sigma = (\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{P}_1 \cup \mathcal{P}_2)$.

Cette signature permet de construire des contraintes et termes sur un ensemble $\mathcal{S}_1 \cup \mathcal{S}_2$-indexé de variables $\mathcal{X} = \mathcal{X}_{\mathcal{S}_1} \cup \mathcal{X}_{\mathcal{S}_2}$.

Une contrainte de $C[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$ est dite *hétérogène* ou *mixte*.

Plus précisement, une contrainte de $C[\Sigma_i, \mathcal{X}_{\mathcal{S}_i}]$ est dite *i-pure* pour $i = 1, 2$.

La même terminologie est utilisée pour les termes: $T(\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})$ est l'ensemble des termes *hétérogènes* alors que $T(\mathcal{S}_i, \mathcal{F}_i, \mathcal{X}_{\mathcal{S}_i})$ est l'ensemble des termes *i-purs*. Un *i-terme* est un terme tel que $t(\epsilon) \in \mathcal{F}_i$.

Dans l'avenir, nous parlerons en général de contraintes (ou termes) hétérogènes par opposition aux contraintes (ou termes) pures.

Le point commun aux approches modulaires que nous allons développer est de décomposer une contrainte hétérogène en plusieurs contraintes pures en vue de réutiliser les algorithmes de chaque structure composante. On définit pour cela la notion de sous-terme étranger qu'on appelle encore sous-terme étranger maximal ou immédiat dans la littérature [Bou90a, Nip91].

Définition 1.41

- *L'ensemble des positions étrangères d'un terme t est*

  $$AlienPos(t) = \{\omega \neq \epsilon \mid t(\omega) \notin \mathcal{F}_i \cup \mathcal{X} \text{ et } \forall \omega' \in \mathcal{P}os(t),\ \omega' < \omega \Rightarrow t(\omega') \in \mathcal{F}_i,\ i = 1, 2\}.$$

- *L'ensemble des sous-termes étrangers d'un terme t est* $AST(t) = \{t_{|\omega} \mid \omega \in AlienPos(t)\}$.
- *La hauteur de théories $ht(t)$ d'un terme t est nulle si $AST(t) = \emptyset$, sinon* $ht(t) = 1 + \max_{s \in AST(t)} ht(s)$.
- *L'ensemble des positions de rupture d'un terme t est*

  $$RPos(t) = \{\epsilon\} \cup \{p.n \mid n \in \mathbf{N},\ t(p) \notin \mathcal{F}_i,\ t(p.n) \in \mathcal{F}_i,\ i = 1, 2\}.$$

Le remplacement des sous-termes étrangers par des variables transforme une contrainte hétérogène en une contrainte pure pouvant être résolue grâce à un algorithme disponible dans l'une des structures du mélange.

Définition 1.42 *Une* abstraction par variables *est une application bijective π d'un ensemble de termes non variables vers un ensemble de variables. On dit qu'un terme t est* abstrait *dans une contrainte s'il est remplacé par la variable $\pi(t)$.*

> **Nota:** *Lorsque des termes d'une contrainte sont abstraits, ils le seront toujours par des variables n'apparaissant pas déjà dans la contrainte.*

On choisit donc π en fonction de la contrainte où figurent les termes à abstraire.

2
Combinaison d'algorithmes d'unification

Ce chapitre a pour objectif de présenter les principes de combinaison utilisées dans cette thèse. Ils sont à la base de l'extension de la combinaison des algorithmes d'unification aux mélanges de théories non disjointes que nous verrons au chapitre 3.

Nous rappelons tout d'abord les résultats acquis au sujet de la combinaison d'algorithmes d'unification [Kir85a, Tid86, Her87, Yel87, SS89, Bou90a, BS92]. Nous commencerons par le cas des théories simples pour lequel les problèmes liés à la combinaison se résolvent de façon évidente, à la différence du cas général. Deux approches ont été successivement étudiées pour résoudre le cas général des théories disjointes. L'une l'a été par A. Boudet et a permis d'obtenir un algorithme pseudo-déterministe dont la principale difficulté est la preuve de terminaison. L'autre approche suivie par M. Schmidt-Schauß puis par F. Baader et K. Schulz a abouti à un algorithme complètement non-déterministe dont l'avantage est de terminer trivialement. La présentation des deux approches est faite dans le souci de pouvoir les comparer et surtout de mettre en évidence qu'elles relèvent toutes les deux des mêmes principes. Cette comparaison nous a conduit à les présenter un peu différemment de leurs auteurs. Par souci de complétude, nous incluons les preuves de certains résultats utiles pour les chapitres suivants.

Les deux approches utilisent en partie la connaissance d'un algorithme d'élimination de constante, problème qui n'a pas encore été étudié de façon systématique même s'il est fortement relié au problème de combinaison des solutions. Nous présenterons l'état des connaissances ainsi que des procédures basées sur des méthodes de surréduction.

2.1 Le mélange de théories simples et disjointes

Nous allons commencer par le cas de mélange le plus simple, afin d'exposer les problèmes et les notions qui seront détaillés au cours du chapitre. L'algorithme de combinaison qui en résulte est sûrement celui qui s'apparente le plus à celui, exemplaire, connu pour la théorie vide et rappelé au chapitre 1.

Une théorie simple [Kir85a] est une théorie ayant de bonnes propriétés vis-à-vis des cycles.

Proposition 2.1 *Une théorie E est* simple *si et seulement si il n'existe pas de solution au problème de cycle $x =^?_E t[x]_\omega$ avec $\omega \neq \epsilon$.*

La théorie vide est simple. De nombreuses théories intéressantes d'un point vue pratique le sont aussi.

Exemple 2.1 *Les théories C, AC sont simples.*

L'hypothèse de simplicité d'une théorie impose de fortes restrictions syntaxiques sur la forme des axiomes qui seront du plus grand intérêt pour la combinaison.

Proposition 2.2 *Une théorie simple est régulière et non effondrante.*

Preuve : Si E est effondrante alors il existe une E-égalité $x =_E t[x]$ et l'identité est solution de $x =^?_E t[x]$. Si E n'est pas régulière alors il existe une E-égalité $l =_E r[x]$ avec $x \notin \mathcal{V}(l)$ et $\{x \mapsto l\}$ est solution de $x =^?_E r[x]$. □

La réciproque est fausse puisque la théorie close $\{a = f(a)\}$ n'est pas simple alors qu'elle est régulière et non effondrante.

Le problème de déterminer si une théorie arbitraire appartient à la classe des théories simples est indécidable [BHSS90].

2.1.1 Abstraction

Si l'on veut adopter une approche modulaire, il faut pouvoir projeter un terme, une égalité ou une équation du mélange de théories $E_1 \cup E_2$ vers un terme, une égalité ou une équation dans une théorie E_i composant le mélange.

La projection d'un terme hétérogène vers un terme dit "pur" s'effectue par abstraction des sous-termes étrangers. Chaque sous-terme étranger est remplacé, ou *abstrait*, par une variable de telle manière qu'une même variable serve à abstraire deux sous-termes étrangers équivalents modulo $E_1 \cup E_2$. Cette opération fait l'objet de la définition suivante.

Définition 2.1 *Une* abstraction par variables simple *est une application bijective π d'un ensemble des représentants de $(T(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})\backslash\mathcal{X})/=_{E_1 \cup E_2}$ vers un sous-ensemble de $\mathcal{X}$.*

La i-abstraction *t^{π_i} d'un terme t est définie par*

- $x^{\pi_i} = x$ *si* $x \in \mathcal{X}$,
- $f(s_1, \ldots, s_p)^{\pi_i} = f(s_1^{\pi_i}, \ldots, s_p^{\pi_i})$ *si* $f \in \mathcal{F}_i$,
- $f(s_1, \ldots, s_p)^{\pi_i} = \pi(f(s_1, \ldots, s_p))$ *si* $f \notin \mathcal{F}_i$.

La i-abstraction *σ^{π_i} d'une substitution σ est définie par $\sigma^{\pi_i} = \{x \mapsto (x\sigma)^{\pi_i} \mid x \in \mathcal{D}om(\sigma)\}$.*

Remarque 2.1 *N'importe quel représentant de $(\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})\backslash\mathcal{X})/=_{E_1 \cup E_2}$ est un terme non variable car $E_1 \cup E_2$ est une théorie régulière et non effondrante.*

Remarque 2.2 *La i-abstraction t^{π_i} d'un terme t est un terme i-pur.*

Un problème d'unification Γ est lui aussi transformé en une conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes respectivement 1-pur et 2-pur. Cette transformation, appelée *purification* consiste elle aussi à remplacer un sous-terme étranger t par une nouvelle variable x et à rajouter l'équation $x =^? t$ correspondante. En appliquant jusqu'à saturation cette "règle" de transformation, on obtient la conjonction $\Gamma_1 \wedge \Gamma_2$ annoncée.

Une équation i-pure doit encore être résolue dans le mélange de théories $E_1 \cup E_2$, la purification n'étant qu'une opération syntaxique. Pour cela, l'objectif essentiel reste bien évidemment de pouvoir réutiliser les algorithmes d'unification dans les théories composantes E_1 et E_2.

2.1.2 Résolution dans une composante

Une méthode correcte pour résoudre une équation i-pure consiste à la résoudre dans sa théorie E_i. Une E_i-solution d'une équation i-pure est en effet une $E_1 \cup E_2$-solution puisque $E_i \subseteq E_1 \cup E_2$. La principale difficulté est de montrer la complétude de cette méthode, c'est-à-dire de mettre en évidence que toute $E_1 \cup E_2$-solution est l'instance d'une E_i-solution. Pour se faire, nous allons voir comment se projette dans une composante un pas de remplacement d'égal par égal. Nous utiliserons pour l'instant uniquement le fait qu'une théorie simple est régulière et non effondrante.

Lemme 2.1 *Soient E_1 et E_2 des théories non effondrantes.*
Si $s \longleftrightarrow_{E_1 \cup E_2} t$ alors $s^{\pi_i} \overset{*}{\longleftrightarrow}_{E_1 \cup E_2} t^{\pi_i}$.

Preuve : Supposons que s est un i-terme. Si le pas de $\longleftrightarrow_{E_1 \cup E_2}$ s'effectue dans un sous-terme étranger de s, alors $s^{\pi_i} = t^{\pi_i}$ car le sous-terme obtenu par remplacement est encore étranger pour t puisque E_1 et E_2 sont non effondrantes. Si le pas de $\longleftrightarrow_{E_1 \cup E_2}$ s'effectue à une position inférieure à toute position de $AlienPos(s)$, alors c'est un pas de $\longleftrightarrow_{E_i}$, où les sous-termes étrangers de s figurent dans la partie instance. On peut donc appliquer le même pas de $\longleftrightarrow_{E_i}$ entre s^{π_i} et t^{π_i} où t est un i-terme puisque E_i est non effondrant. Nous avons donc aussi $s^{\pi_j} = t^{\pi_j} \in \mathcal{X}$ pour $j \neq i$. □

Un remplacement d'égal par égal d'un terme i-pur s'effectue nécessairement par un axiome de E_i pour obtenir un terme i-pur.

Lemme 2.2 *Soient E_1 et E_2 des théories régulières et non effondrantes.*
Si $s \in T(\mathcal{F}_i, \mathcal{X})$ et $s \longleftrightarrow_{E_1 \cup E_2} t$ alors $t \in T(\mathcal{F}_i, \mathcal{X})$ et $s \longleftrightarrow_{E_i} t$.

Preuve : Puisque E_j est non effondrante, il ne peut pas s'agir d'un E_j-remplacement d'égal par égal. Ensuite, aucun sous-terme étranger pour t ne peut être introduit par un pas de $\longleftrightarrow_{E_i}$ puisque la théorie E_i est régulière. Donc t est i-pur. □

Des lemmes 2.1 et 2.2, il découle qu'à toute preuve équationnelle dans le mélange correspond une preuve équationnelle dans une composante sur les termes abstraits.

Proposition 2.3 *Soient E_1 et E_2 des théories régulières et non effondrantes. Pour tous les termes s et t, on a $s =_{E_1 \cup E_2} t \Leftrightarrow s^{\pi_i} =_{E_1 \cup E_2} t^{\pi_i} \Leftrightarrow s^{\pi_i} =_{E_i} t^{\pi_i}$.*

Cette proposition permet en particulier de montrer la complétude de la résolution dans une composante qu'on exprime par le lemme suivant:

Lemme 2.3 *Soient E_1 et E_2 des théories régulières et non effondrantes. Si s et t sont deux terme i-purs et σ une substitution alors*

$$s\sigma =_{E_1 \cup E_2} t\sigma \Leftrightarrow s\sigma^{\pi_i} =_{E_i} t\sigma^{\pi_i}.$$

Preuve : Les termes s et t étant i-purs, on a $(s\sigma)^{\pi_i} = s\sigma^{\pi_i}$ et $(t\sigma)^{\pi_i} = t\sigma^{\pi_i}$. □

La substitution σ est une instance de σ^{π_i} puisque $\sigma =_{E_1 \cup E_2} \sigma^{\pi_i}\pi^{-1}$.

Une fois les problèmes Γ_1 et Γ_2 résolus dans leurs théories respectives, il faut encore s'assurer qu'à partir de deux solutions issues de chaque composante, on peut en construire une dans le mélange. Nous allons voir que dans le mélange de théories simples, soit une conjonction de formes résolues est une forme résolue, soit cette conjonction n'a pas de solution dans le mélange parce qu'elle contient un conflit de théories ou un cycle.

2.1.3 Conflit de théories

Si une variable est instanciée dans les deux théories alors le problème n'a pas de solution d'après la proposition suivante:

Proposition 2.4 *Si E_1 et E_2 sont des théories non effondrantes, alors il n'existe pas d'égalité $s =_{E_1 \cup E_2} t$ avec $s(\epsilon) \in \mathcal{F}_1$ et $t(\epsilon) \in \mathcal{F}_2$.*

Preuve : Si $s =_{E_1 \cup E_2} t$ alors $s^{\pi_1} =_{E_1} t^{\pi_1}$ avec $t^{\pi_1} \in \mathcal{X}$ puisque $t(\epsilon) \in \mathcal{F}_2$. La théorie E_1 est donc effondrante, ce qui est en contradiction avec l'hypothèse. □

2.1.4 Résolution de cycle

Si le problème contient un cycle, alors il n'a pas de solution puisque l'union (disjointe) de théories simples est encore une théorie simple.

Proposition 2.5 *Si E_1 et E_2 sont des théories simples alors $E_1 \cup E_2$ est une théorie simple.*

Preuve : On prouve par récurrence sur la hauteur de théories du terme t qu'une équation $x =^?_{E_1 \cup E_2} t[x]_\omega$ n'a pas de solution. Supposons l'existence d'une substitution σ telle que $x\sigma =_{E_1 \cup E_2} t\sigma[x\sigma]_\omega$.

- Si la hauteur de théories de t est nulle, alors t est un terme i-pur et $x\sigma =_{E_1 \cup E_2} t\sigma[x\sigma]$ implique $x\sigma^{\pi_i} =_{E_i} t\sigma^{\pi_i}[x\sigma^{\pi_i}]$, ce qui est absurde par hypothèse sur E_i.
- On peut supposer sans perte de généralité que $x\sigma(\epsilon), t(\epsilon) \in \mathcal{F}_i$ d'après la proposition 2.4. Ensuite, grâce à la proposition 2.3, nous avons $x\sigma^{\pi_i} =_{E_i} (t\sigma[x\sigma]_\omega)^{\pi_i}$. S'il n'existe pas de position étrangère entre ϵ et ω alors $x\sigma^{\pi_i} =_{E_i} (t\sigma)^{\pi_i}[x\sigma^{\pi_i}]$, ce qui est absurde puisque E_i est simple.

  S'il existe une position étrangère entre ϵ et ω alors une variable y fait abstraction d'un terme ayant $x\sigma$ pour sous-terme strict. Cette variable doit aussi faire abstraction d'un sous-terme étranger de $x\sigma$ puisque E_i est régulière et non effondrante. Par conséquent, il existe une solution à une équation $y =^? u[y]$, où u est un terme dont la hauteur de théories est strictement inférieure à celle de t, ce qui est absurde par hypothèse de récurrence.

□

En conséquence, un cycle composé obtenu par la conjonction de formes résolues dans chaque composante n'a pas de solution. Nous verrons qu'il n'est pas nécessaire de supposer les théories simples pour qu'un cycle composé n'ait pas de solution. Les théories régulières et non effondrantes en sont un autre exemple. De plus, nous n'avons utilisé pour le lemme 2.3 que les propriétés de régularité et de non effondrement des théories simples. L'algorithme que nous allons présenter est de ce fait encore valable pour les théories régulières et non effondrantes. Il est dû à A. Boudet [Bou90a].

2.1.5 Algorithme à base de règles

L'algorithme s'apparente beaucoup à celui donné pour l'unification dans la théorie vide avec forme séquentiellement résolue, c'est-à-dire sans remplacement. En voici un bref descriptif:

- les règles VA et IE transforment le problème en une conjonction de problèmes purs,
- la règle E_i-Res résout le sous-problème i-pur dans la théorie E_i,
- la règle Conflit est similaire à celle dans la théorie vide, à ceci près qu'elle s'applique à des symboles disjoints appartenant chacun à une théorie différente,
- la règle Cycle est identique à celle dans la théorie vide.

Rappelons que les transformations ne portent que sur la partie non quantifiée du problème d'unification. C'est pourquoi on ne mentionne que les quantifications existentielles sur les variables rajoutées.

Structure de données

La structure de données est celle prise par A. Boudet [BJSS90, Bou90b, Bou90a]. Un problème d'unification $\exists\vec{x} : P$ se décompose en

- P_H le sous-problème de P composé d'équations hétérogènes,
- P_V le sous-problème de P composé d'équations impropres, c'est-à-dire d'équations entre variables,
- P_i le sous-problème de P $(i = 1, 2)$ composé d'équations i-pures.

La plus petite relation d'équivalence sur $\mathcal{X}$ contenant P_V est notée $=_V$.

> **Nota:** *Lorsqu'on utilise $P, Q, \ldots$ pour dénoter un problème d'unification, il est fait référence à cette structure de données.*

On note Γ le problème d'unification initial et $\Gamma_1 \wedge \Gamma_2$ la conjonction de problèmes purs respectivement 1-pur et 2-pur obtenue par application répétée des règles VA et IE.

VA

$$\frac{P \wedge s =^? t}{\exists x : P \wedge s =^? t[x]_\omega \wedge x =^? t_{|\omega}} \quad \textbf{si} \begin{cases} \omega \in AlienPos(t) \\ x \text{ est une nouvelle variable} \end{cases}$$

IE

$$\frac{P \wedge s =^? t}{\exists x : P \wedge x =^? s \wedge x =^? t} \quad \textbf{si} \begin{cases} s \in T(\mathcal{F}_1, \mathcal{X}) \backslash \mathcal{X},\ t \in T(\mathcal{F}_2, \mathcal{X}) \backslash \mathcal{X} \\ x \text{ est une nouvelle variable} \end{cases}$$

E_i-Res$^+$

$$\frac{P_i}{\exists \vec{x_i} : \hat{\sigma}_i} \quad \textbf{si } \sigma_i \in CSU_{E_i}(P_i) \text{ et } \vec{x_i} = \mathcal{V}(\hat{\sigma}_i) \backslash \mathcal{V}(P_i).$$

VarRep

$$\frac{P \wedge x =^? y}{P\{x \mapsto y\} \wedge x =^? y} \quad \textbf{si } x, y \in \mathcal{V}(P)$$

Conflit

$$\frac{P \wedge x =^? s \wedge x =^? t}{\bot} \quad \textbf{si } s(\epsilon) \in \mathcal{F}_1, t(\epsilon) \in \mathcal{F}_2$$

Cycle

$$\frac{P \wedge x_1 =^? t_1[x_2]_{p_1} \wedge \cdots x_n =^? t_n[x_1]_{p_n}}{\bot} \quad \textbf{si } p_i \neq \epsilon \text{ pour un } i \in \{1, \ldots, n\}$$

Figure 2.1. Ensemble de règles $\mathcal{RS}$ pour l'unification dans le mélange de théories simples et disjointes

Lemme 2.4 *L'application à un problème d'unification* Γ *des règles* VA *et* IE *avec un contrôle arbitraire termine et retourne un problème d'unification* $\exists \vec{x} : \Gamma_1 \wedge \Gamma_2$ *équivalent à* Γ.

Les règles VA et IE font décroître respectivement

- le multi-ensemble des hauteurs de théories des sous-termes étrangers des termes de P_H, noté $TH_{mul}(P)$,
- le nombre d'équations impures de P_H, noté $IE(P)$.

Une forme normale pour $\{\mathsf{VA}, \mathsf{IE}\}$ est un problème d'unification n'ayant pas de sous-problème hétérogène P_H. En considérant simplement les nouvelles variables comme existentiellement quantifiées, VA et IE préservent de façon évidente l'ensemble des solutions.

La preuve de terminaison donnée ci-dessous du système de règles $\mathcal{RS}$ est un peu plus simple que celle donnée dans la thèse d'A. Boudet [Bou90a]. Elle utilise les variables du problème $\Gamma_1 \wedge \Gamma_2$ associé au problème initial Γ.

Forme résolue

Les règles $\mathcal{RS}$ de la figure 2.1 transforment un problème en un autre qui lui est équivalent d'après les propositions 2.4, 2.5 et 2.4. Il s'agit encore de vérifier qu'une forme normale est une forme séquentiellement résolue (et réciproquement).

Lemme 2.5 *([Bou90a]) Si les règles de $\mathcal{RS}$ appliquées avec un contrôle arbitraire à un problème Γ terminent, alors elles retournent une disjonction équivalente de problèmes d'unification $\exists \vec{x} : P$ où les P sont en forme séquentiellement résolue.*

Terminaison

La preuve de terminaison est basée sur le fait que la résolution dans une théorie ne peut engendrer de nouvelles variables qui puissent être partagées avec l'autre théorie.

Définition 2.2 *Une variable est* introduite *par un E_i-unificateur σ_i d'un sous-problème i-pur P_i si c'est une variable de $\mathcal{V}(\hat{\sigma}_i)\backslash\mathcal{V}(P_i)$. Un ensemble de variables $W \subseteq \mathcal{X}$ est* protégé *par unification des sous-problèmes purs de P si*

$$\forall \sigma_i \in CSU_{E_i}(P_i),\ x\sigma_i \in W \text{ si } x\sigma_i \in \mathcal{X} \text{ et } x \in W$$

et si les variables introduites par σ_i sont en dehors de l'ensemble des variables du problème P, de W et celles introduites respectivement par σ_1 et σ_2 sont distinctes entre elles.

Lemme 2.6 *Il est toujours possible de protéger un ensemble W de variables par unification des sous-problèmes purs de P.*

Preuve : Il existe un renommage μ tel que

$$\forall x, y \in \mathcal{V}(P_i),\ (x\sigma_i)\mu = (y\sigma_i)\mu \in W \text{ si } x\sigma_i = y\sigma_i = w\sigma_i \in \mathcal{X}, w \in W.$$

La substitution γ_i définie par $\gamma_i =^{\mathcal{V}(P_i)} \sigma_i\mu$ est solution de P_i et satisfait la condition requise. Elle vérifie aussi $\sigma_i =^{\mathcal{V}(P_i)} \gamma_i\mu^{-1}$ puisque μ est un renommage. L'ensemble des substitutions γ_i définit donc un $CSU_{E_i}(P_i)$. □

Les hypothèses prises ci-dessous sont valables pour tous les algorithmes à base de règles détaillés par la suite.

Hypothèse 2.1

1. *Les variables introduites par purification sont incluses dans $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)$.*
2. *$\mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ est protégé par unification des sous-problèmes purs de P.*

Ces hypothèses permettent de s'assurer d'une part que les classes de nouvelles variables introduites par E_i-unification et apparaissant dans P_i sont des nouvelles classes de variables et d'autre part que les nouvelles classes de variables apparaissant dans P_1 et P_2 sont disjointes.

Lemme 2.7 *Les propriétés suivantes:*

1. $(\mathcal{V}(P_i)\backslash\mathcal{V}(\Gamma_1 \wedge \Gamma_2))/=_V \ \cap\ \mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V \ = \emptyset$
2. $(\mathcal{V}(P_1)\backslash\mathcal{V}(\Gamma_1 \wedge \Gamma_2))/=_V \ \cap\ (\mathcal{V}(P_2)\backslash\mathcal{V}(\Gamma_1 \wedge \Gamma_2))/=_V \ = \emptyset$

*sont conservées par application de E_i-*Res*.*

Preuve : Avant application de E_i-Res, on a $\mathcal{V}(P_i) \subseteq \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ et les propriétés énoncées sont donc vraies initialement.

Supposons maintenant que ces propriétés soient vraies pour un problème P et que celui-ci soit transformé en Q par E_i-Res.

1. Si une équation impropre $x =^? y$, $x \in \mathcal{V}(P_i) \backslash \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, $y \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, est engendrée par E_i-Res, alors $x \in \mathcal{V}(Q_V)$ mais $x \notin \mathcal{V}(Q_i)$ car par hypothèse, $x\sigma_i = y$ si $\sigma_i \in CSU_{E_i}(P_i)$ et non pas $y\sigma_i = x$.
2. Une variable introduite par E_i-Res n'est pas élément de $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ et n'apparaît pas ailleurs dans P. Si cette variable est dans $\mathcal{V}(Q_i)$ tout en étant égale modulo $=_V$ à une variable de $\mathcal{V}(P_i)$, alors cette dernière est nécessairement dans $\mathcal{V}(P_i) \backslash \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, sinon cette variable est dans $\mathcal{V}(Q_V)$ en utilisant le même argument que pour le premier point.

□

L'application de E_i-Res ne produit pas de nouvelles classes de variables qui puissent créer un conflit inter-théories.

Lemme 2.8 *La propriété*

$$\mathcal{V}(P_1)/=_V \ \bigcap \ \mathcal{V}(P_2)/=_V \ \subseteq \mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$$

est conservée par application de E_i-Res, VA *et* IE.

Preuve : Pour E_i-Res, il s'agit d'une conséquence immédiate de la proposition précédente.

La nouvelle variable introduite par VA et IE figure à la fois dans P_1 et P_2 mais aussi par hypothèse dans $\Gamma_1 \wedge \Gamma_2$. □

Lemme 2.9 VarRep *ne s'applique qu'à des variables de* $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)$.

Preuve : Supposons que la règle E_i-Res engendre une nouvelle équation $x =^? y$ avec $x \in \mathcal{V}(P_i) \backslash \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, c'est-à-dire qu'elle transforme un problème P en un problème $Q \wedge x =^? y$. Nous allons montrer que $x \notin \mathcal{V}(Q)$. La variable x n'est déjà plus élément de $\mathcal{V}(Q_i)$ par application de E_i-Res.

Si $x \notin \mathcal{V}(Q_V)$ alors on montre par l'absurde que $x \notin \mathcal{V}(Q_H \wedge Q_j)$ pour $j \neq i$:

- Si $x \in \mathcal{V}(Q_H)$ alors $x \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ puisque $\mathcal{V}(Q_H) \subseteq \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$.
- Si $x \in \mathcal{V}(Q_j) = \mathcal{V}(P_j)$ alors les lemmes 2.8 et 2.7 impliquent que $x \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$.

Si $x \in \mathcal{V}(Q_V)$ alors un problème tel que x est résolue et n'apparaît nulle part ailleurs a d'abord été engendré puis x a forcément été réintroduite par E_1-Res ou E_2-Res, ce qui est contraire à l'hypothèse 2.1.

En définitive, $x \notin \mathcal{V}(Q)$ et VarRep ne s'applique pas avec $x =^? y$. □

Mesures de complexité:

A chaque problème d'unification, on associe une mesure de complexité strictement décroissante pour chaque règle de $\mathcal{RS}$ suivant un ordre nœthérien.

- $TH_{mul}(P)$ le multi-ensemble des hauteurs de théories des sous-termes étrangers des termes de P_H,
- $IE(P)$ le nombre d'équations impures de P_H,
- $PVR(P)$ le nombre d'applications potentielles de VarRep (entre variables de $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)$),
- $USP(P)$ le nombre de sous-problèmes purs de P qui ne sont pas en formes résolues,
- $NVC(P) = |\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/ =_V |$ où $=_V$ est la plus petite relation d'équivalence sur $\mathcal{X}$ contenant P_V.

Lemme 2.10 E_i-Res *fait décroître* $NVC(P)$ *ou ne modifie ni* $NVC(P)$ *ni* $PVR(P)$. VarRep *ne modifie pas* $NVC(P)$ *et fait décroître* $PVR(P)$.

Preuve : La règle E_i-Res a deux comportements:

1. E_i-Res engendre une équation impropre $x =^? y$, $x, y \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ et $NVC(P)$ décroît,
2. Si toutes les équations impropres $x =^? y$ engendrées par E_i-Res vérifient $x \notin \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ ou $y \notin \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, alors $NVC(P)$ n'est pas modifié, de même que $PVR(P)$ d'après la proposition 2.9, mais $USP(P)$ décroît.

la règle VarRep ne modifie pas le nombre de classes d'équivalence de $=_V$ mais diminue strictement le nombre de représentants d'une classe dans $P_H \wedge P_1 \wedge P_2$. □

La mesure de complexité $NVC(P)$ précède $PVR(P)$ dans l'ordre lexicographique comparant les mesures de complexité.

Proposition 2.6 *Le système de règles* $\mathcal{RS}$ *termine pour tout contrôle.*

Preuve : La preuve est résumée par le tableau suivant:

regles	$TH_{mul}(P)$	$IE(P)$	$NVC(P)$	$PVR(P)$	$USP(P)$
VA	$\downarrow$				
IE	$=$	$\downarrow$			
E_i-Res(1)	$=$	$=$	$\downarrow$		
E_i-Res(2)	$=$	$=$	$=$	$=$	$\downarrow$
VarRep	$=$	$=$	$=$	$\downarrow$	

Les règles Conflit et Cycle ne sont pas reproduites ici car elles mènent directement à des formes normales. □

Théorème 2.1 *Si* E_1 *et* E_2 *sont deux théories disjointes et simples alors la* $E_1 \cup E_2$*-unification est finitaire si et seulement si la* E_i*-unification* ($i = 1, 2$) *est finitaire.*

On notera qu'il s'agit de l'unification élémentaire, le cas général d'unification avec symboles libres correspondant au mélange de théories quelconques que nous allons maintenant aborder.

2.2 Le mélange de théories disjointes

Dans la section précédente, de fortes conditions étaient imposées sur les théories pouvant être combinées. On traite à présent du cas général sans restriction sur la forme des axiomes. De nouveaux problèmes surgissent:

- un sous-terme étranger ne reste pas toujours étranger par remplacement d'égal par égal,
- un terme i-pur peut se réécrire en un j-terme, $i \neq j$, par un axiome effondrant,
- un nouveau sous-terme étranger peut être introduit par remplacement d'égal par égal.

Les deux derniers points peuvent être réglés en imposant une certaine direction dans l'utilisation des axiomes.

2.2.1 Système de réécriture combiné

L'idée d'introduire un système de réécriture pour décrire le comportement de l'égalité dans le mélange de théories est dûe à A. Boudet et J.-P. Jouannaud. Cette idée a ensuite été reprise par F. Baader et K. Schulz, montrant ainsi son intérêt pour le problème de décision dans le mélange de théories [BS92]. La définition du *système de réécriture combiné* que nous donnons est plus directe et ne nécessite pas l'utilisation de la complétion sans échec sur les axiomes E_1 et E_2. Nous allons l'introduire comme étant l'ensemble des instances hétérogènes et closes des E_1-égalités et E_2-égalités.

Soit $E_i^{>} = \{l\sigma \to r\sigma \mid l =_{E_i} r,\ \mathcal{R}an(\sigma) \subseteq \mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X}), l\sigma > r\sigma\}$ pour $i = 1, 2$.

Pour orienter toutes les instances closes des E_i-égalités, un ordre de réduction total sur $T(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})$ est utilisé.

Hypothèse 2.2 *$>$ est un ordre de simplification total sur $T(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})$ tel que $>_{|\mathcal{X}}$ est nœthérien.*

Il n'y a aucune difficulté à satisfaire cette hypothèse, un ordre LPO ou RPO fait très bien l'affaire. Le théorème de Dershowitz 1.5 ne permet pourtant pas de conclure que $>$ est un ordre de réduction car $\mathcal{F} \cup \mathcal{X}$ est infini.

Proposition 2.7 *Un ordre de simplification sur $T(\mathcal{F} \cup \mathcal{X})$ est nœthérien si $>_{|\mathcal{X}}$ est nœthérien.*

Preuve : La preuve est basée sur le théorème de Kruskal [Kru60] et la notion de pré-bel-ordre.

La relation $\geq$ définie sur $\mathcal{F} \cup \mathcal{X}$ par l'égalité sur $\mathcal{F}$ et $>_{|\mathcal{X}}$ est une relation de pré-bel-ordre. En vertu du théorème de Kruskal, la relation de plongement $\geq_{emb}$ induite par $\geq$ et définie comme étant la plus petite relation reflexive transitive et stable par contexte incluant

$$\begin{array}{rcll} f(s_1, \ldots, s_n) & \geq_{emb} & s_i & 1 \leq i \leq n \\ x & \geq_{emb} & y & x, y \in \mathcal{X},\ x > y \end{array}$$

est une relation de pré-bel-ordre sur $T(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})$. L'ordre de simplification $>$ contient le pré-bel-ordre $\geq_{emb}$. Par conséquent, s'il existe une chaîne infinie décroîssante $t_1 > t_2 > \ldots$, alors d'après la définition d'un pré-bel-ordre, il existe deux termes t_j, t_k tels que $j < k$, $t_j \leq_{emb} t_k$ et donc $t_j < t_k$, ce qui est absurde. □

Le fait que $>$ soit un ordre de simplification permet de valider la proposition suivante:

Lemme 2.11 *Si $l \to r \in E_1^> \cup E_2^>$ et $l(\epsilon) \in \mathcal{F}_i$ alors $l \to r \in E_i^>$.*

Preuve : Considérons une règle $l\sigma \to r\sigma \in E_j^>$ telle que $l\sigma(\epsilon) \in \mathcal{F}_i$ pour $j \neq i$. Dans ce cas, l est nécessairement une variable x avec $x \in \mathcal{V}(r)$ puisque E_j est consistante. Par conséquent, $l\sigma$ est sous-terme strict de $r\sigma$ et $r\sigma > l\sigma$ puisque $>$ est un ordre de simplification, ce qui est absurde. □

L'hypothèse d'un ordre de simplification total sur les termes clos est donc essentielle pour déterminer la théorie d'une règle en fonction de sa tête. Toutes les paires critiques entre $E_1^>$ et $E_2^>$ sont donc joignables. Les paires critiques entre règles de $E_i^>$ sont elles aussi joignables puisque par définition $E_i^>$ représente les instances (closes) de toutes les E_i-égalités et pas seulement des axiomes de E_i. Il en résulte la convergence du système de réécriture combiné.

Proposition 2.8 *Les systèmes de réécriture $E_1^>, E_2^>$ et $E_1^> \cup E_2^>$ sont convergents.*

Preuve : Terminaison: Soient s et t deux termes de $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})$. Si $s \to_{E_1^> \cup E_2^>} t$ alors $s \to_{E_1^>} t$ ou $s \to_{E_2^>} t$. Dans les deux cas, $s > t$ où $>$ est un ordre nœthérien sur $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})$.

Confluence: Puisque $\to_{E_1^> \cup E_2^>}$ termine, il suffit de montrer que $\to_{E_1^> \cup E_2^>}$ est localement confluent. Deux cas non triviaux sont à considérer:

Supposons l'existence d'un pic $s \leftarrow_{E_i^>} t \to_{E_i^>} u$, où (s, u) est l'instance d'une paire critique (l, r), qui est aussi une E_i-égalité. Ainsi $s = l\psi, u = r\psi, l =_{E_i} r$ et donc $s \to u$ ou $u \to s$ est une règle de $E_i^>$.

Supposons l'existence d'un pic $s \leftarrow_{E_i^>} t \to_{E_j^>} u$ avec, sans perte de généralité, la règle de $E_i^>$ appliquée à la position ϵ. Donc $t = g\psi$ et $s = d\psi$ avec $g =_{E_i} d$ et $g, d \in \mathcal{T}(\mathcal{F}_i, \mathcal{X})$. La règle de $E_j^>$ s'applique nécessairement à une position dont le symbole est dans $\mathcal{F}_j$. Il existe donc une variable $x \in \mathcal{V}(g)$ à une position p, une substitution ψ' et une position q telle que $u = g\psi[r\psi']_{p \cdot q}$ avec $x\psi_{|q} = l\psi'$ et $l\psi' \to r\psi' \in E_j^>$. Soit σ la substitution définie par $y\sigma = y\psi$ pour $y \neq x$ et $x\sigma = x\psi[r\psi']_q$. Alors

$$s \xrightarrow{*}_{l\psi' \to r\psi'} s\sigma \longleftrightarrow_{\epsilon, g\sigma \longleftrightarrow d\sigma} u\sigma \xleftarrow{*}_{r\psi' \leftarrow l\psi'} u.$$

□

Le fait de supposer les instances des variables de $\mathcal{V}(r) \backslash \mathcal{V}(l)$ normalisées assure que la réduction d'un terme s'effectue sans l'introduction de nouveaux sous-termes étrangers réductibles.

On note R le système de réécriture inclus dans $E_1^> \cup E_2^>$ défini par

$$R = \bigcup_{i=1}^{2} \{l\sigma \to r\sigma \mid l =_{E_i} r,\ \mathcal{R}an(\sigma) \subseteq \mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X}),\ l\sigma > r\sigma,$$

$$x\sigma \text{ est } E_1^> \cup E_2^>\text{-normalisée pour } x \in \mathcal{V}(r)\backslash\mathcal{V}(l)\}.$$

Corollaire 2.1 $s =_{E_1 \cup E_2} t \iff s \downarrow_R = s \downarrow_{E_1^> \cup E_2^>} = t \downarrow_{E_1^> \cup E_2^>} = t \downarrow_R$.

Nous supposerons par défaut que les variables de $\mathcal{X}$ sont inférieures aux autres termes suivant l'ordre $>$. Cela permet de ne pas introduire de nouveaux sous-termes étrangers et de s'assurer ainsi que la forme normale d'un terme i-pur est i-pur.

2.2.2 Abstraction

Le terme $t \downarrow_R$ est maintenant choisi comme représentant canonique de la classe de t modulo $E_1 \cup E_2$, ce qui conduit à une nouvelle définition de l'abstraction par variables qui devient une bijection entre termes normalisés et un ensemble de variables. Le domaine de l'abstraction par variables ne contient pas de variables puisqu'il s'agit de termes qui ne sont abstraits dans aucune théorie.

Définition 2.3 *Une* abstraction par variables *est une bijection π entre l'ensemble des termes R-normalisés $T \downarrow_R = \{u \downarrow_R | u \in T(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})$ et $u \downarrow_R \in T(\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{X})\backslash\mathcal{X}\}$ et un sous-ensemble de variables de $\mathcal{X}$.*

Le terme t^{π_i}, appelé i-abstraction du terme t, est défini de façon inductive comme suit:

- *si $t = x \in \mathcal{X}$ alors $t^{\pi_i} = x$,*
- *si $t = f(s_1, \ldots, s_p)$ et $f \in \mathcal{F}_i$ alors $t^{\pi_i} = f(s_1^{\pi_i}, \ldots, s_p^{\pi_i})$*
  sinon si $t \downarrow_R \notin \mathcal{X}$ alors $t^{\pi_i} = \pi(t \downarrow_R)$ sinon $t^{\pi_i} = t \downarrow_R$.

Il est intéressant de noter que les variables sont R-normalisées, $x \downarrow_R = x$ pour $x \in \mathcal{X}$. Nous aurions donc pu définir l'abstraction par variables comme l'identité sur $\mathcal{X}$, c'est-à-dire $\pi(x \downarrow_R) = x$ et définir ainsi la i-abstraction sans avoir à considérer différemment les variables. C'est l'option choisie par F. Baader et K. Schulz [BS92].

2.2.3 Résolution dans une composante

L'objectif essentiel reste de pouvoir résoudre un problème i-pur obtenu par purification en utilisant l'algorithme de E_i-unification. La règle correspondante est bien évidemment correcte mais sa complétude mérite qu'on s'y attarde. On suit la même démarche que précédemment. Le premier point est de déterminer comment se projette un pas de réécriture du mélange.

Proposition 2.9 *Soit s un i-terme dont les sous-termes étrangers sont R-normalisés. Si $s \to_R t$ alors*

- *$s \to_{E_i^>} t$ où t est soit R-normalisé, soit un i-terme dont les sous-termes étrangers sont R-normalisés,*

- $s^{\pi_i} =_{E_i} t^{\pi_i}$.

Preuve : La règle de R s'applique nécessairement à une position où figure un symbole de $\mathcal{F}_i$ puisque les sous-termes étrangers sont R-normalisés. Il s'agit donc d'une règle de $E_i^{>}$.

Si t est une variable, alors elle est R-normalisée. Si t est un j-terme, $j \neq i$, alors c'est un sous-terme étranger de s, donc R-normalisé. Sinon, t est un i-terme dont tous les sous-termes étrangers sont R-normalisés puisque les sous-termes étrangers éventuellemment introduits sont irréductibles.

Les sous-termes étrangers figurant dans la partie "instance" de la règle $E_i^{>}$ appliquée, il existe donc une E_i-égalité entre les i-abstractions de s et t. □

Proposition 2.10 *Si s est un i-terme dont les sous-termes étrangers sont R-normalisés alors $s^{\pi_i} =_{E_i} (s \downarrow_R)^{\pi_i}$.*

Preuve : Par la proposition 2.9 et induction nœthérienne sur $\rightarrow_R$. □

Corollaire 2.2 *Si s et t sont des i-termes dont les sous-termes étrangers sont R-normalisés alors*

$$s =_{E_1 \cup E_2} t \Longleftrightarrow s^{\pi_i} =_{E_i} t^{\pi_i}.$$

Preuve : Si $s =_{E_1 \cup E_2} t$ alors $s^{\pi_i} =_{E_i} (s \downarrow_R)^{\pi_i} = (t \downarrow_R)^{\pi_i} =_{E_i} t^{\pi_i}$. □

La théorie $E_1 \cup E_2$ est donc en particulier une extension conservative de E_1 et de E_2.

Corollaire 2.3 *Si s et t sont des termes i-purs alors $s =_{E_1 \cup E_2} t \Longleftrightarrow s =_{E_i} t$.*

Lemme 2.12 *Si s est un terme i-pur et σ une substitution R-normalisée alors*

- $(s\sigma) \downarrow_R = (s\sigma) \downarrow_{E_i^{>}}$,
- $((s\sigma) \downarrow_R)^{\pi_i} =_{E_i} s\sigma^{\pi_i}$.

Preuve : Par la proposition 2.9 et induction nœthérienne sur $\rightarrow_R$. Un sous-terme étranger de $s\sigma$ est R-normalisé car s est pur et σ R-normalisée. □

Si s est une variable alors $s\sigma$ est R-normalisé et l'on a même plus besoin d'utiliser la proposition 2.9.

Ce lemme permet de montrer la complétude de la règle E_i-Res qu'on exprime par ce qui suit.

Lemme 2.13 *Si s, t sont des termes i-purs et σ une substitution R-normalisée alors*

$$s\sigma =_{E_1 \cup E_2} t\sigma \Leftrightarrow s\sigma^{\pi_i} =_{E_i} t\sigma^{\pi_i}.$$

Preuve : Si $(s\sigma) \downarrow_R = (t\sigma) \downarrow_R$ alors $((s\sigma) \downarrow_R)^{\pi_i} = ((t\sigma) \downarrow_R)^{\pi_i}$ et

$$s\sigma^{\pi_i} = (s\sigma)^{\pi_i} =_{E_i} ((s\sigma) \downarrow_R)^{\pi_i} = ((t\sigma) \downarrow_R)^{\pi_i} =_{E_i} (t\sigma)^{\pi_i} = t\sigma^{\pi_i}.$$

□

2.2.4 Combinaison des solutions

A ce stade, un problème hétérogène Γ a été séparé en deux sous-problèmes purs Γ_1 et Γ_2 qui ont été résolus dans leur théorie respective. Deux formes résolues sont calculées, une par théorie, et il va falloir en extraire une solution pour le mélange de théories. Nous avons vu que dans le cas de théories simples, soit la conjonction des formes résolues est une forme résolue, soit cette conjonction contient un conflit de théories ou un cycle et elle n'a pas de solution. Il en va autrement dans le cas général. Un conflit ou un cycle inter-théories peut désormais posséder des solutions. Pour les déterminer, deux approches ont été étudiées; elles seront présentées puis comparées dans les sections suivantes.

La première approche, dûe à A. Boudet, consiste à reprendre l'algorithme d'unification dans le mélange de théories simples et d'y modifier les règles Conflit et Cycle qui ne sont plus correctes. Il s'agit donc de rajouter de nouvelles règles de combinaison à la règle E_i-Res.

La seconde approche a été introduite par M. Schmidt-Schauß [SS89] puis reprise par F. Baader et K. Schulz [BS92]. L'idée consiste cette fois à éviter *a priori* d'engendrer des conflits ou des cycles inter-théories au cours de la résolution. Pour ce faire, il faut modifier la résolution dans chaque théorie pour ne calculer que les solutions qui puissent être combinées, c'est-à-dire celles dont la conjonction définit une forme (séquentiellement) résolue. L'avantage de cette seconde approche est de pouvoir s'appliquer à la fois au problème de résolution d'équations et surtout au problème de décision de l'unification puisque les solutions n'ont pas à être explicitement engendrées.

Les avantages et inconvénients des approches par décision et résolution seront plus longuement discutés en section 2.5.

2.3 Approche par résolution

La i-abstraction d'une solution σ de P doit satisfaire certaines restrictions dans la théorie E_i que nous allons coder par des prédicats définis *localement* dans chaque théorie E_i à la manière de ce qui est fait dans [Bou90a].

Nous supposons ici l'existence d'un algorithme d'unification dans chaque théorie composante.

2.3.1 Conflit de théories

Une variable x peut apparaitre résolue dans deux théories, i.e. $x =^?_{E_1} t_1 \wedge x =^?_{E_2} t_2$, mais ne peut être instanciée par une solution R-normalisée σ que dans l'une seule des théories, disons E_i. La j-abstraction du terme $(x\sigma)^{\pi_j}$ pour $j \neq i$ est une variable telle que $(x\sigma)^{\pi_j} =_{E_j} (t_j\sigma)^{\pi_j}$. Il va donc falloir s'intéresser en priorité aux E_j-solutions instanciant x par une variable et c'est pourquoi A. Boudet introduit un prédicat unaire M_{E_i} avec une variable en argument pour la "marquer". La sémantique de ce prédicat est définie de la façon suivante:

Définition 2.4 *Une substitution σ_i est* solution *d'un problème de E_i-unification $P_i \wedge M_{E_i}(x)$ si σ_i est une solution de P_i telle que $x\sigma_i \in \mathcal{X}$.*

Une équation $x =^?_{E_i} t \wedge M_{E_i}(x)$ n'est plus en forme résolue puisque $\{x \mapsto t\}$ n'est pas solution de $x =^?_{E_i} t \wedge M_{E_i}(x)$.

Définition 2.5 *Le problème d'unification $P_i = \hat{\sigma}_i \wedge M_{E_i}(x_1) \wedge \cdots \wedge M_{E_i}(x_n)$ est en* forme résolue *si $\hat{\sigma}_i$ est en forme résolue et*

$$\forall k \in \{1, \ldots, n\},\ x_k\sigma \in \mathcal{X}.$$

La forme résolue $\hat{\sigma}_i$ est notée $tsf(P_i)$.

Le singleton $\{\sigma_i\}$ est un $CSU_{E_i}(P_i)$.

L'introduction du prédicat de marquage M_{E_i} est motivée par la proposition suivante, décrivant comment résoudre un conflit de théories.

Proposition 2.11 *Soit P un problème $x=^?t_1 \wedge x=^?t_2$ avec $t_1 \in T(\mathcal{F}_1, \mathcal{X})\backslash\mathcal{X}$ et $t_2 \in T(\mathcal{F}_2, \mathcal{X})\backslash\mathcal{X}$. Si σ est une solution R-normalisée de P alors il existe un indice $i \in \{1, 2\}$ tel que σ^{π_i} est solution de $P_i \wedge M_{E_i}(x)$.*

Preuve : Si σ est une substitution R-normalisée alors il existe $j \in \{1, 2\}$ tel que $x\sigma(\epsilon) \in \mathcal{F}_j \cup \mathcal{X}$ et donc $x\sigma^{\pi_i} \in \mathcal{X}$ pour $i \neq j$. □

On pourra remarquer qu'il n'a pas été nécessaire de supposer la variable résolue dans chaque théorie pour prouver la proposition.

2.3.2 Résolution de cycles

Comme nous l'avons déjà évoqué pour le mélange de théories simples, il suffit de s'intéresser aux cycles composés de formes résolues de chaque théorie composante.

Définition 2.6 *Un* cycle composé *est un cycle*

$$y_1=^?t_1[x_1] \wedge x_1=^?s_1[y_2] \wedge \cdots y_n=^?t_n[x_n] \wedge x_n=^?s_n[y_1]$$

tel que

- $t_1, \ldots, t_n \in T(\mathcal{F}_1, \mathcal{X})\backslash\mathcal{X}$,
- $s_1, \ldots, s_n \in T(\mathcal{F}_2, \mathcal{X})\backslash\mathcal{X}$,
- $x_1, \ldots, x_n, y_1, \ldots, y_n$ *sont des variables distinctes.*

Si l'on considère le cycle composé $y=^?t_1[x] \wedge x=^?t_2[y]$ et σ une solution R-normalisée instanciant y dans E_1 et x dans E_2, alors le terme $x\sigma$ (resp. la variable $(x\sigma)^{\pi_1}$) ne peut pas être sous-terme strict de $y\sigma$ (resp. $(y\sigma)^{\pi_1}$) et $y\sigma$ (resp. la variable $(y\sigma)^{\pi_2}$) sous-terme strict de $x\sigma$ (resp. $(x\sigma)^{\pi_2}$). C'est ce qui motive l'introduction d'un prédicat binaire avec des variables en arguments et la sémantique suivante:

Définition 2.7 *Une substitution σ_i est* solution *d'un problème de E_i-unification $P_i \wedge Arc_{E_i}(x, y)$ si $x\sigma_i \in \mathcal{X}$ et $x\sigma_i \notin \mathcal{V}(y\sigma_i)$.*

Définition 2.8 *Le problème d'unification*

$$P_i = \hat{\sigma}_i \wedge M_{E_i}(z_1) \wedge \cdots \wedge M_{E_i}(z_q) \wedge Arc_{E_i}(x_1, y_1) \wedge \cdots \wedge Arc_{E_i}(x_n, y_n)$$

est en forme résolue *si* $\hat{\sigma}_i \wedge M_{E_i}(z_1) \wedge \cdots \wedge M_{E_i}(z_q)$ *est en forme résolue et si*

$$\forall k \in \{1, \ldots, n\},\ x_k\sigma_i \in \mathcal{X} \text{ et } x_k\sigma_i \notin \mathcal{V}(y_k\sigma_i).$$

La forme résolue $\hat{\sigma}_i$ *est notée* $tsf(P_i)$.

Le singleton $\{\sigma_i\}$ est un $CSU_{E_i}(P_i)$.
Un problème $y =^?_{E_i} t[x] \wedge Arc_{E_i}(x, y)$ n'est plus en forme résolue puisque $x \in \mathcal{V}(t[x])$.
La résolution d'un cycle composé est basée sur la proposition suivante:

Proposition 2.12 *Si* $P_1 \wedge P_2$ *est un cycle composé*

$$y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1]$$

et σ *une solution R-normalisée de* $P_1 \wedge P_2$*, alors*

- σ^{π_1} *est solution de* $P_1 \wedge M_{E_1}(y_m)$ *pour un* $m \in \{1, \ldots, n\}$,
- *ou* σ^{π_2} *est solution de* $P_2 \wedge M_{E_2}(x_m)$ *pour un* $m \in \{1, \ldots, n\}$,
- *ou* σ^{π_1} *est solution de* $P_1 \wedge Arc_{E_1}(x_m, y_m)$ *pour un* $m \in \{1, \ldots, n\}$,
- *ou* σ^{π_2} *est solution de* $P_2 \wedge Arc_{E_2}(y_m, x_m)$ *pour un* $m \in \{2, \ldots, n\}$,
- *ou* σ^{π_2} *est solution de* $P_2 \wedge Arc_{E_2}(y_1, x_n)$.

Preuve : Si

$$\forall m \in \{1, \ldots, n\},\ x_m\sigma(\epsilon) \in \mathcal{F}_2,$$
$$\forall m \in \{1, \ldots, n\},\ y_m\sigma(\epsilon) \in \mathcal{F}_1,$$

et

$$\forall m \in \{1, \ldots, n\},\ x_m\sigma^{\pi_1} \in \mathcal{V}(y_m\sigma),$$
$$\forall m \in \{2, \ldots, n\},\ y_m\sigma^{\pi_2} \in \mathcal{V}(x_{m-1}\sigma),$$
$$y_1\sigma^{\pi_2} \in \mathcal{V}(x_n\sigma),$$

alors $y_1\sigma$ est sous-terme strict de $x_n\sigma$ et $x_n\sigma$ est sous-terme strict de $y_1\sigma$, ce qui est absurde. □

2.3.3 Unification avec restriction

Il faut pouvoir résoudre les problèmes d'unification spécifiques introduits lors d'un conflit ou d'un cycle inter-théories.

Restriction variable

Un problème contenant des prédicats M_{E_i} et Arc_{E_i} est appelé problème d'unification avec restriction variable.

Définition 2.9 *Un* problème de E_i-unification avec restriction variable *est un problème de E_i-unification*

$$P_i = P_i' \wedge M_{E_i}(z_1) \wedge \cdots \wedge M_{E_i}(z_q) \wedge Arc_{E_i}(x_1, y_1) \wedge \cdots \wedge Arc_{E_i}(x_n, y_n),$$

où P_i' ne contient pas de prédicat M_{E_i}, ni de prédicat Arc_{E_i}. On note $\mathcal{M}(P_i)$ l'ensemble des variables $\{z_1, \ldots, z_q, x_1, \ldots, x_n\}$.

Pour résoudre un problème d'unification avec restriction variable, il ne suffit pas d'empêcher l'instanciation des variables marquées $\mathcal{M}(P_i)$. Les variables de $\mathcal{M}(P_i)$ ne pourront être considérées comme constantes (libres) qu'après avoir été identifiées.

Identification de variables

La définition que nous prenons impose l'idempotence des substitutions instanciant des variables par des variables.

Définition 2.10 *([SS89]) Soient V et W deux ensembles de variables. Une identification ξ de V sur W est une substitution idempotente telle que $\mathcal{D}om(\xi) \subseteq V$ et $\mathcal{R}an(\xi) \subseteq W$. L'ensemble des identifications de V sur W est noté ID_V^W. L'ensemble des identifications de V sur lui-même est noté ID_V.*

Il suffit d'une partition de V en au plus $|W|$ parties pour construire une identification.

Proposition 2.13 *Si deux substitutions σ et ϕ sont deux substitutions de ID_V^W comparables par $\leq^V$ alors σ et ϕ sont définies à un renommage près.*

Restriction constante

Les variables marquées dans une théorie puis identifiées sont ensuite considérées comme des constantes libres dans l'une ou l'autre des théories mais pas dans les deux théories simultanément. Les contraintes équationnelles à résoudre ne sont donc plus simplement des problèmes d'unification.

Définition 2.11 *([Bür90]) Un problème de décision de E-*unification avec constantes *est une contrainte équationnelle*

$$\forall \vec{y} : (\exists \vec{x} : \Gamma)$$

où $\Gamma = (\bigwedge_{k \in K} s_k =^? t_k)$ est un problème de E-unification et $\vec{x} \oplus \vec{y} = \mathcal{V}(\Gamma)$. On parle plus précisement de problème du mot *lorsqu'il n'y a pas de variable existentiellement quantifiée et de problème de décision du* filtrage *lorsque $\bigcup_{k \in K} \mathcal{V}(t_k) \subseteq \vec{y}$.*

La E-unification consiste à trouver les assignations des variables existentiellement quantifiées dans le problème de décision (équivalent à $\top$ ou $\bot$)

$$\exists \mathcal{V}(\Gamma) : \Gamma.$$

De même, la E-unification avec constantes définie ci-après consiste à trouver les assignations des variables existentiellement quantifiées dans le problème de décision (sans variable libre)

$$\forall \vec{y} : (\exists \vec{x} : \Gamma).$$

Définition 2.12 *Un* problème de E-unification avec constantes *est un couple* (Γ, C) *formé*

- *d'un problème de E-unification* Γ*,*
- *et d'un ensemble* $C \subseteq \mathcal{X}$ *de variables* skolémisées*.*

Une substitution σ *est* solution *d'un problème de E-unification avec constantes* (Γ, C) *si* σ *est solution de* Γ *et si* $\forall c \in C,\ c\sigma = c$*. L'ensemble (resp. un ensemble complet) de E-solutions au problème* (Γ, C) *est noté* $SU_E(\Gamma, C)$ *(resp.* $CSU_E(\Gamma, C)$*).*

Il est important de noter que:

- Toutes les constantes libres apparaissant dans un problème de E-unification sont dorénavant vus comme des variables skolémisées même si le contexte C (les variables universellement quantifiées) n'est pas toujours explicite.
- Nous n'avons pas défini la notion d'unification avec constantes sur un problème d'unification existentiellement quantifié car nous chercherons toujours à expliciter les solutions de toutes les variables avant de tenir compte de la quantification existentielle en éliminant celles introduites par le calcul.

L'unification avec constantes suffit pour résoudre un problème d'unification n'incluant que des prédicats M_{E_i} mais ne l'est plus lorsque il contient aussi des prédicats Arc_{E_i}.

Définition 2.13 *([BS92]) Un* problème de E-unification avec restriction constante *est un triplet* (Γ, C, η) *formé*

- *d'un problème de E-unification avec constantes* (Γ, C)*,*
- *et d'une relation* η *de* $C \times \mathcal{V}(\Gamma)$*.*

Pour tout $c \in C$*, on note* $\eta(c)$ *l'ensemble des variables* $\{v \mid (c, v) \in \eta\}$*.*

Une substitution σ *est* solution *d'un problème de E-unification avec restriction constante* (Γ, C, η) *si* σ *est solution de* (Γ, C) *et si* $\forall c \in C,\ \forall x \in \eta(c),\ c \notin x\sigma$*. L'ensemble (resp. un ensemble complet) de E-solutions au problème* (Γ, C, η) *est noté* $SU_E(\Gamma, C, \eta)$ *(resp.* $CSU_E(\Gamma, C, \eta)$*).*

La proposition suivante établit le lien entre restriction variable et restriction constante.

Proposition 2.14 *([Bou90a]) Un ensemble complet de E_i-solutions au problème*

$$P_i = P_i' \wedge M_{E_i}(z_1) \wedge \cdots \wedge M_{E_i}(z_q) \wedge Arc_{E_i}(x_1, y_1) \wedge \cdots \wedge Arc_{E_i}(x_n, y_n),$$

est

$$\bigcup_{\xi \in ID_{\mathcal{M}(P_i)}} CSU_{E_i}(P_i\xi, \mathcal{M}(P_i)\xi, \{(x_1\xi, y_1), \ldots, (x_n\xi, y_n)\}) \circ \xi.$$

Les notions de restriction constante et variable étant équivalentes modulo identification des variables, nous n'emploierons plus que l'unification avec restriction constante ou plus simplement *unification avec restriction.*

Elimination de constante

Un problème d'élimination de constante est un cas particulier de problème d'unification avec restriction.

Définition 2.14 *Un* problème de E-élimination de constante *est un problème de E-unification avec restriction constante $(\hat{\sigma}, C, \eta)$ tel que $\hat{\sigma}$ est un problème d'unification en forme résolue et $\eta(c) \subseteq \mathcal{D}om(\sigma)$ pour tout $c \in C$.*

Un problème d'élimination de constante est défini dans [SS89] par un ensemble de couples

$$\{(c_1, t_1), \ldots, (c_n, t_n)\}.$$

Définition 2.15 *Un E-*éliminateur *de $\{(c_1, t_1), \ldots, (c_n, t_n)\}$ est une substitution σ vérifiant $\forall i \in \{1, \ldots, n\}\ \exists u,\ t_i\sigma =_E u$ et $c_i \notin u$.*

Proposition 2.15 *Il existe un éliminateur de*

$$\{(c_1, t_1), \ldots, (c_n, t_n)\}$$

si et seulement si il existe une solution au problème

$$(\bigwedge_{k=1}^{n} x_k =^? t_k, \bigcup_{k=1}^{n} \{c_k\}, \bigcup_{k=1}^{n} \{(c_k, x_k)\}),$$

où les variables $x_1, \ldots, x_n$ sont distinctes entre elles et distinctes de celles de $\bigcup_{k=1}^{n} \mathcal{V}(t_k)$.

Il est impossible d'en dire plus car seule l'existence d'un terme éliminant est requise dans la définition 2.15, qui n'est donc pas suffisante pour obtenir un algorithme de résolution. C'est pourquoi nous allons dorénavant considérer l'ensemble de couples

$$\{(c_1, t_1), \ldots, (c_n, t_n)\}$$

comme une notation abrégée au problème

$$(\bigwedge_{k=1}^{n} x_k =^? t_k, \bigcup_{k=1}^{n} \{c_k\}, \bigcup_{k=1}^{n} \{(c_k, x_k)\}).$$

On pourrait envisager de construire un algorithme d'élimination de constante de façon incrémentale en résolvant successivement chaque équation avec restriction.

Définition 2.16 *Un* problème d'élimination de constante élémentaire *est un problème d'unification avec restriction* $(x=^?t[c], \{c\}, \{(c,x)\})$

Malheureusement, un problème d'unification avec restriction n'est pas stable par instanciation, ce qui interdit *a priori* une approche incrémentale.

Exemple 2.2 *Considérons la théorie* $E = \{f(0,x) = 0\}$. *La substitution* $\sigma = \{x \mapsto f(z,0), y \mapsto 0\}$ *est une solution au problème de E-élimination de constante élémentaire*

$$(x=^?f(z,f(y,c)), \{c\}, \{(c,x)\}).$$

Mais la substitution $\{x \mapsto f(c,0), y \mapsto 0, z \mapsto c\}$ *est une instance de* σ *qui n'est plus solution au problème.*

L'intérêt de la classe des théories définies ci-dessous est de pouvoir exprimer tout problème d'élimination de constante élémentaire par un problème d'unification, ce qui permet ensuite de construire incrémentalement un algorithme d'élimination de constante.

Définition 2.17 *Un théorie E est* stable *si pour tout problème de E-élimination de constante élémentaire* $(x=^?t[c], \{c\}, \{(c,x)\})$, *il existe un problème de E-unification avec constantes* (Γ, C) *tel que*

$$CSU_E(x=^?t[c], \{c\},\ (c,x)\}) = CSU_E(\Gamma, C).$$

Toutes les théories pour lesquelles l'élimination de constante est connue sont stables.

Proposition 2.16 *Si E est une théorie stable alors la E-élimination de constante est finitaire si et seulement si la E-élimination de constante élémentaire est finitaire.*

Un algorithme d'unification avec restriction constante est obtenu grâce à l'utilisation conjointe d'un algorithme d'unification avec constantes et d'un algorithme d'élimination de constante.

Proposition 2.17 *Un ensemble complet de E-solutions au problème* (Γ, C, η) *est*

$$\bigcup_{\sigma \in CSU_E(\Gamma,C)} CSU_E(\hat{\sigma}, C, \eta).$$

Preuve : Correction: Une solution ψ de $(\hat{\sigma}, C, \eta)$ est solution de (Γ, C) puisque $\sigma \in CSU_E(\Gamma, C)$; cette solution vérifie aussi la restriction $\forall c \in C,\ \forall x \in \eta(c),\ c \notin x\psi$, donc ψ est plus précisement solution de (Γ, C, η).

Complétude: Si $\psi \in CSU_E(\Gamma, C, \eta)$ alors ψ est solution de (Γ, C) et il existe $\sigma \in CSU_E(\Gamma, C)$, une substitution μ telle que $\forall x \in \mathcal{V}(\Gamma),\ x\psi =_E x\sigma\mu$. Puisque $\mathcal{D}om(\sigma) \cap \mathcal{VR}an(\sigma) = \emptyset$, on peut supposer sans perte de généralité que μ vérifie $\forall x \in \mathcal{V}(\Gamma), x\mu =_E x\sigma\mu$ et $\psi =^{\mathcal{V}(\Gamma)} \mu$. Les substitutions ψ et μ satisfaisant la restriction $\forall c \in C,\ \forall x \in \eta(c),\ c \notin x\mu$, on peut conclure que μ est solution de $(\hat{\sigma}, C, \eta)$ et qu'il existe $\phi \in CSU_E(\hat{\sigma}, C, \eta)$ telle que $\phi \leq_E^{\mathcal{V}(\Gamma)} \mu =^{\mathcal{V}(\Gamma)} \psi$. □

Cette décomposition n'est intéressante que si l'unification avec constantes est finitaire dans E.

Corollaire 2.4 *Si E est une théorie stable alors la E-unification avec restriction est finitaire si et seulement si la E-unification avec constantes est finitaire.*

Corollaire 2.5 *Si E est une théorie stable alors la $E \cup \emptyset$-unification est finitaire si et seulement si la E-unification avec constantes est finitaire.*

Ce corollaire est une conséquence de l'algorithme d'unification décrit ci-après appliqué au mélange de E et de la théorie vide pour laquelle l'unification avec restriction est unitaire puisque l'élimination de constante élémentaire n'a pas de solution.

2.3.4 Algorithme à base de règles

VA

$$\frac{P \wedge s =^? t}{\exists x : P \wedge s =^? t[x]_\omega \wedge x =^? t_{|\omega}} \quad \textbf{si} \begin{cases} \omega \in AlienPos(t) \\ x \text{ est une nouvelle variable} \end{cases}$$

IE

$$\frac{P \wedge s =^? t}{\exists x : P \wedge x =^? s \wedge x =^? t} \quad \textbf{si} \begin{cases} s \in T(\mathcal{F}_1, \mathcal{X}) \backslash \mathcal{X},\ t \in T(\mathcal{F}_2, \mathcal{X}) \backslash \mathcal{X} \\ x \text{ est une nouvelle variable} \end{cases}$$

E_i-Res$^+$

$$\frac{P_i}{\exists \vec{x_i} : \hat{\sigma}_i} \quad \textbf{si } \sigma_i \in CSU_{E_i}(P_i) \text{ et } \vec{x_i} = \mathcal{V}(\hat{\sigma}_i) \backslash \mathcal{V}(P_i)$$

VarRep

$$\frac{P \wedge x =^? y}{P\{x \mapsto y\} \wedge x =^? y} \quad \textbf{si } x, y \in \mathcal{V}(P)$$

Conflit$^+$

$$\frac{P \wedge x =^? s \wedge x =^? t}{P \wedge x =^? s \wedge x =^? t \wedge M_{E_i}(x)}$$

Si $s(\epsilon) \in \mathcal{F}_1, t(\epsilon) \in \mathcal{F}_2$ et la règle VarRep ne s'applique pas

Cycle$^+$

$$\frac{P \wedge y =^? t[x]}{P \wedge y =^? t[x] \wedge (M_{E_i}(y) \vee Arc_{E_i}(x, y))}$$

Si $y =^? t[x]$ est membre d'un cycle composé de P avec $t \in T(\mathcal{F}_i, \mathcal{X}) \backslash \mathcal{X}$ et la règle VarRep ne s'applique pas

Figure 2.2. Ensemble de règles $\mathcal{RD}$ pour l'unification dans le mélange de théories disjointes

Forme résolue

Les règles $\mathcal{RD}$ de la figure 2.2 sont correctes et complètes d'après les propositions 2.11, 2.12 et 2.4. On vérifie ensuite qu'une forme normale pour $\mathcal{RD}$ est un problème d'unification dont on peut extraire directement un ensemble complet de solutions.

Définition 2.18 *Un problème d'unification P est* séparé *si $P = P_V \wedge P_1 \wedge P_2$. Un problème P séparé est en* forme séquentiellement résolue *si*

- $P_V \wedge P_1$ *est en forme résolue,*
- $P_V \wedge P_2$ *est en forme résolue,*
- $tsf(P_V \wedge P_1) \wedge tsf(P_V \wedge P_2)$ *est en forme séquentiellement résolue.*

Les prédicats M_{E_i} et Arc_{E_i} ont une sémantique dans la théorie E_i mais n'en ont pas dans le mélange $E_1 \cup E_2$, ce qui explique la nécessité d'une forme résolue pour un problème séparé ne contenant plus de sous-problème hétérogène P_H.

Lemme 2.14 *([Bou90a]) Si les règles de $\mathcal{RD}$ appliquées avec un contrôle arbitraire à un problème Γ terminent, alors elles retournent une disjonction équivalente de problèmes d'unification $\exists \vec{x} : P$ où les P sont en forme séquentiellement résolue.*

Terminaison

La preuve de terminaison s'appuie sur les mêmes principes que pour les théories simples. On utilise ici le fait que les classes de variables créant un conflit inter-théories sont incluses nécessairement dans $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$. Les règles Conflit et Cycle ne s'appliquent que si VarRep ne s'applique plus. C'est pourquoi les arguments de M_{E_i} et Arc_{E_i} sont assimilés à des classes de variables modulo $=_V$.

Autres mesures de complexité:

- $NM(P)$ le nombre de classes d'équivalence de $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$ ne figurant pas dans une marque M,
- $NA(P)$ le nombre de classes d'équivalence de $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$ ne figurant pas dans un arc Arc.

Lemme 2.15 Conflit *et* Cycle *font décroître respectivement $NM(P)$ et $NA(P)$.*

Preuve : Les règles Conflit et Cycle ne s'appliquent que si la classe de variables est déjà marquée ou si un couple de classes de variables ne figure pas déjà dans un Arc. On ne revient donc pas sur le travail déjà effectué. □

Lemme 2.16 E_i-Res *ne fait croître ni $NM(P)$ ni $NA(P)$.*

Preuve : E_i-Res ne fait pas croître $NVC(P)$ et ne supprime ni des marques ni des arcs. □

Proposition 2.18 *Le système de règles $\mathcal{RD}$ termine pour tout contrôle.*

Preuve : La preuve est résumée par le tableau suivant:

regles	$TH_{mul}(P)$	$IE(P)$	$NM(P)$	$NA(P)$	$NVC(P)$	$PVR(P)$	$USP(P)$
VA	$\downarrow$						
IE	$=$	$\downarrow$					
Conflit	$=$	$=$	$\downarrow$				
Cycle	$=$	$=$	$=\downarrow$	$\downarrow$			
E_i-Res(1)	$=$	$=$	$=\downarrow$	$=\downarrow$	$\downarrow$		
E_i-Res(2)	$=$	$=$	$=\downarrow$	$=\downarrow$	$=$	$=$	$\downarrow$
VarRep	$=$	$=$	$=$	$=$	$=$	$\downarrow$	

□

Théorème 2.2 *(M. Schmidt-Schauß [SS89]) Si E_1 et E_2 sont deux théories disjointes alors la $E_1 \cup E_1$-unification est finitaire si la E_i-unification avec restriction ($i = 1, 2$) est finitaire.*

Ce théorème a été démontré initialement sans donner d'algorithme déterministe. A. Boudet a ensuite présenté et prouvé l'algorithme sous forme de règles décrit en figure 2.2. La preuve de terminaison semblait la principale difficulté de l'algorithme déterministe. La preuve que nous en avons faite est plus simple, et nous allons voir qu'elle n'a plus raison d'être pour l'algorithme non-déterministe présenté dans ce qui suit.

2.4 Approche par décision

Les règles mises en évidence précédemment sont basées sur la transformation d'un problème i-pur sous une forme résolue équivalente. Elles ne peuvent donc s'appliquer au problème de décision puisque l'on suppose alors ne plus disposer d'un algorithme de E_i-unification mais seulement d'un algorithme de E_i-unifiabilité.

La combinaison d'algorithmes de décision de l'unification a été abordée pour la première fois par M. Schmidt-Schauß [SS89] puis par F. Baader et K. Schulz [BS92] qui ont montré qu'une même méthode pouvait s'appliquer à la fois au problème de décision et au problème de résolution.

2.4.1 Unification avec restriction linéaire

L'idée est de résoudre les problèmes 1-purs et 2-purs obtenus par purification suivant une même restriction interprétée différemment dans les théories E_1 et E_2. Cette restriction est choisie en fonction de la forme séquentiellement résolue attendue.

Définition 2.19 *Soient (Γ, C, η) un problème de E-unification avec restriction, V un ensemble fini de variables incluant $\mathcal{V}(\Gamma)$ et $<$ un ordre linéaire sur $V \cup C$. Le problème (Γ, C, η) est un* problème de E-unification avec restriction linéaire *noté $(\Gamma, C, <)$ si*

$$\forall c \in C,\ \eta(c) = \{v \in \mathcal{V}(\Gamma) \mid v < c\}.$$

Un ensemble (resp. ensemble complet) de E-solutions au problème $(\Gamma, C, <)$ est noté $SU_E^<(\Gamma, C)$ (resp. $CSU_E^<(\Gamma, C)$).

Une même restriction linéaire est utilisée par les deux théories afin de construire des solutions dont la conjonction est une forme séquentiellement résolue.

Solutions combinées

Définition 2.20 *Soient $(\Gamma_1, V_2, <)$ et $(\Gamma_2, V_1, <)$ deux problèmes respectivement 1-pur et 2-pur avec $<$ un ordre linéaire sur $V_1 \oplus V_2 = \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$. La* solution combinée $\sigma_1 \odot \sigma_2$ *de Γ_1 et Γ_2 obtenue à partir de $\sigma_1 \in SU_{E_1}^<(\Gamma_1, V_2)$ et $\sigma_2 \in SU_{E_2}^<(\Gamma_2, V_1)$ est définie comme suit: si x une variable de V_i et $\{y_1, \ldots y_n\}$ l'ensemble des variables de V_j ($i \neq j$) inférieures à x par rapport à $<$, alors $x\sigma = x\sigma_i[y_k \hookleftarrow y_k\sigma]_{k=1,\ldots,n}$.*

Une solution combinée est la substitution associée à une forme séquentiellement résolue.

Proposition 2.19 *Une solution combinée de Γ_1 et Γ_2 est une $E_1 \cup E_2$-solution de $\Gamma_1 \wedge \Gamma_2$.*

Décidabilité

On montre qu'une $E_1 \cup E_2$-solution de $\Gamma_1 \wedge \Gamma_2$ est l'instance d'une solution combinée. Nous allons d'abord établir un lemme permettant de régler le cas des solutions instanciant des variables par des variables ou par des termes non distincts.

Lemme 2.17 *Si ϕ est une solution R-normalisée d'un problème d'unification Γ, alors il existe*

- *une identification $\xi \in ID_{\mathcal{V}(\Gamma)}$,*
- *une solution σ R-normalisée de $\Gamma\xi$,*

telles que $\xi\sigma =^{\mathcal{V}(\Gamma)} \phi$, $\mathcal{R}an(\sigma) \cap \mathcal{X} = \emptyset$ et $\forall x, y \in \mathcal{D}om(\sigma),\ x\sigma \neq y\sigma$ si $x \neq y$.

On suppose dorénavant qu'une substitution σ R-normalisée instancie les variables par des termes distincts qui ne sont pas des variables. Cela va permettre de définir une abstraction par variables à partir d'une solution σ en posant $\pi(x\sigma) = x$ pour $x \in \mathcal{D}om(\sigma)$ et de faciliter ainsi les preuves de complétude, par exemple celle du lemme suivant.

Lemme 2.18 *Si σ est une solution R-normalisée d'un problème d'unification $\Gamma_1 \wedge \Gamma_2$, alors il existe*

- *une identification $\xi \in ID_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$,*
- *un ordre linéaire $<$ sur $V_1 \oplus V_2 = \mathcal{V}((\Gamma_1 \wedge \Gamma_2)\xi)$,*
- *une solution combinée $\sigma_1 \odot \sigma_2$ de $(\Gamma_1\xi, V_2, <)$ et $(\Gamma_2\xi, V_1, <)$,*

tels que $\xi(\sigma_1 \odot \sigma_2) \leq^{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)} \sigma$.

Preuve : Soit σ une solution R-normalisée d'une conjonction $\Gamma_1 \wedge \Gamma_2$ telle que $\mathcal{R}an(\sigma) \cap \mathcal{X} = \emptyset$ et $\forall x, y \in \mathcal{D}om(\sigma), x\sigma \neq y\sigma$ si $x \neq y$. L'abstraction par variables π peut être par conséquent définie comme une bijection telle que $\forall x \in \sigma,\ \pi(x\sigma) = x$. Par cette définition, $x\sigma$ est abstrait par x lorsque $x\sigma$ est sous-terme étranger.

L'ordre linéaire $<$ est choisi tel que $x < y$ si $x\sigma$ est sous-terme strict de $y\sigma$.

On considère les substitutions $\sigma_1 = \sigma^{\pi_1}$ et $\sigma_2 = \sigma^{\pi_2}$. Par le lemme 2.13, σ_1 est E_1-solution de Γ_1 et σ_2 est E_2-solution. Chaque variable de $\mathcal{D}om(\sigma)$ est instanciée dans σ_1 ou σ_2 mais pas dans les deux. On définit $V_1 = \mathcal{D}om(\sigma_1)$ et $V_2 = \mathcal{D}om(\sigma_2)$. Par conséquent si $x \in V_1$ alors $x\sigma_2 = x$ et si $x \in V_2$ alors $x\sigma_1 = x$. Les variables de V_1 sont donc skolémisées dans E_2 et symétriquement les variables de V_2 skolémisées dans E_1. On s'assure ensuite que σ_1 et σ_2 satisfont la restriction linéaire $<$ induit par l'ordre sous-terme strict. Si le terme $x\sigma_i$ $(i = 1, 2)$ contient une variable y instanciée dans σ_j $(j \neq i)$ alors montrons que $x \not< y$. Par construction, si y est une variable de $x\sigma_i$, alors le sous-terme étranger $y\sigma$ est sous-terme de $x\sigma$. Il est sous-terme strict car $x\sigma$ et $y\sigma$ sont différents puisque les variables x et y sont non identifiées. Donc $y < x$. □

Théorème 2.3 *(M. Schmidt-Schauß [SS89], F. Baader et K. Schulz [BS92]) Si E_1 et E_1 sont deux théories disjointes alors la $E_1 \cup E_2$-unification est décidable si la E_i-unification avec restriction linéaire $(i = 1, 2)$ est décidable.*

Ce résultat de décidabilité mène à un résultat de résolution, à condition de savoir construire un ensemble complet de solutions combinées.

Ensemble complet de solutions combinées

Proposition 2.20 *L'ensemble des solutions combinées*

$$\{\sigma = \sigma_1 \odot \sigma_2 \mid \sigma_1 \in CSU^{<}_{E_1}(\Gamma_1, V_2),\ \sigma_2 \in CSU^{<}_{E_2}(\Gamma_2, V_1)\}$$

est un ensemble complet de solutions combinées de $(\Gamma_1, V_2, <)$ *et* $(\Gamma_2, V_1, <)$.

Preuve : Soit $\sigma' = \sigma'_1 \odot \sigma'_2$ avec $\sigma'_i \in SU^{<}_{E_i}(\Gamma_i, V_j)$. Il existe une substitution $\sigma_i \in CSU^{<}_{E_i}(\Gamma_i, V_j)$ tel que $\sigma_i \leq^{V_i}_{E_i} \sigma'_i$. Nous allons montrer que $\sigma = \sigma_1 \odot \sigma_2$ est plus générale que σ', c'est-à-dire

$$\sigma \leq^{V_1 \oplus V_2}_{E_1 \cup E_2} \sigma'.$$

Il existe donc deux substitutions γ_1, γ_2 tel que

$$\sigma'_i =^{V_i}_{E_i} \sigma_i\gamma_i$$

pour $i, j = 1, 2$ et $i \neq j$. Ces substitutions n'instancient pas les variables de $V_1 \oplus V_2$: $\mathcal{D}om(\gamma_1) \cap (V_1 \oplus V_2) = \emptyset$ et $\mathcal{D}om(\gamma_2) \cap (V_1 \oplus V_2) = \emptyset$. On supposera sans perte de généralité que γ_1 (resp. γ_2) est idempotente et instancie des variables introduites par l'unification dans E_1 (resp. E_2), c'est-à-dire $\gamma_1\gamma_1 = \gamma_1$, $\gamma_2\gamma_2 = \gamma_2$ et $\mathcal{D}om(\gamma_1) \cap \mathcal{D}om(\gamma_2) = \emptyset$.

La preuve se fait par induction noethérienne sur l'ordre $<$. Soit x une variable de V_i et $\{y_1, \ldots, y_n\}$ l'ensemble des variables de V_j inférieures à x par rapport à $<$. L'hypothèse d'induction est

$$y\sigma' =_{E_1 \cup E_2} y\sigma\gamma_1\gamma_2$$

pour $y < x$. Cette hypothèse est vérifiée pour la variable minimale z de V_m avec $m = 1, 2$ car par définition

$$z\sigma' = z\sigma'_m =_{E_m} z\sigma_m\gamma_m = z\sigma_m\gamma_1\gamma_2 = z\sigma\gamma_1\gamma_2.$$

Pour la variable x, la définition inductive d'une solution combinée est

$$\begin{array}{rcl} x\sigma' & = & x\sigma'_i[y_k \hookleftarrow y_k\sigma']_{k=1,\ldots,n} \\ & =_{E_i} & x\sigma_i\gamma_i[y_k \hookleftarrow y_k\sigma']_{k=1,\ldots,n} \\ & =_{E_1 \cup E_2} & x\sigma_i\gamma_i[y_k \hookleftarrow y_k\sigma\gamma_1\gamma_2]_{k=1,\ldots,n} \\ & = & (x\sigma_i[y_k \hookleftarrow y_k\sigma]_{k=1,\ldots,n})\gamma_1\gamma_2 \\ & = & x\sigma\gamma_1\gamma_2. \end{array}$$

Nous avons donc montré que $y\sigma' =_{E_1 \cup E_2} y\sigma\gamma_1\gamma_2$ pour $y \leq x$. □

L'algorithme qui en résulte est complètement non-déterministe.

Algorithme

L'algorithme consiste à choisir toutes les identifications et toutes les restrictions linéaires possibles puis résoudre séparement chaque sous-problème pur dans sa théorie en tenant compte de la restriction choisie.

On le présente formellement par la proposition suivante:

Proposition 2.21 *Un ensemble complet de $E_1 \cup E_2$-solutions au problème $\Gamma_1 \wedge \Gamma_2$ est formé de l'ensemble des substitutions $\xi(\sigma_1 \odot \sigma_2)$ telles que*

- $\xi \in ID_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$,
- $<$ *est un ordre linéaire sur* $V_1 \oplus V_2 = \mathcal{V}((\Gamma_1 \wedge \Gamma_2)\xi)$,
- $\sigma_1 \in CSU^{<}_{E_1}(\Gamma_1\xi, V_2)$, $\sigma_2 \in CSU^{<}_{E_2}(\Gamma_2\xi, V_1)$.

L'apport de F. Baader et K. Schulz est d'avoir montré que l'unification avec restriction linéaire est suffisante pour obtenir un algorithme d'unification dans le mélange de théories disjointes. Même si l'on ne connait pas encore de théories dont l'unification avec restriction diffère de l'unification avec restriction linéaire, cela permet d'avoir une idée plus précise sur le non-déterminisme inhérent à l'approche modulaire. Il n'est pas nécessaire en effet de considérer n'importe quelles relations entre variables et variables skolémisées: les relations d'ordres linéaires suffisent.

Théorème 2.4 *(F. Baader et K. Schulz [BS92]) Si E_1 et E_1 sont deux théories disjointes alors la $E_1 \cup E_2$-unification est finitaire si la E_i-unification avec restriction linéaire ($i = 1, 2$) est finitaire.*

2.4.2 Unification avec symboles libres

En appliquant l'algorithme de combinaison au mélange d'une théorie E et de la théorie vide $\emptyset$, on obtient un algorithme de E-unification avec symboles libres, notée $E \cup \emptyset$-unification et encore appelée *E-unification générale*. La théorie équationnelle notée abusivement $E \cup \emptyset$ est celle engendrée par la présentation équationnelle $(\mathcal{F} \cup \mathcal{F}_\emptyset, E)$ où $\mathcal{F}$ et $\mathcal{F}_\emptyset$ sont des ensembles de symboles de fonctions disjoints, ce qui correspond à un mélange de théories disjointes si $\emptyset$ dénote l'ensemble des axiomes

$$\bigcup_{f \in (\mathcal{F}_\emptyset)_p} \{f(x_1, \ldots, x_p) = f(x_1, \ldots, x_p)\}$$

au lieu d'un ensemble vide d'axiomes.

Corollaire 2.6 *Si la E-unification avec restriction linéaire est décidable (resp. finitaire) alors la E-unification générale est décidable (resp. finitaire).*

L'objectif de cette section est de montrer les réciproques.

Problème de décision

Nous allons donc nous intéresser au mélange $E_1 \cup E_2$ avec $E_1 = E$ et $E_2 = \emptyset$.

A chaque restriction linéaire est associée un problème d'unification dans la théorie vide.

Définition 2.21 *Etant donné un problème d'unification avec restriction linéaire* $(\Gamma, C, <)$*,* $\hat{\sigma}_<$ *désigne le problème d'unification défini par*

$$\sigma_< = \{c \mapsto f_c(x_1, \ldots, x_m) \mid \eta(c) = \{x_1, \ldots, x_m\}\},$$

où les symboles de C *sont des variables pour* $\hat{\sigma}_<$ *et* f_c *des symboles libres.*

On peut supposer que pour toute solution R-normalisée ψ' de $\hat{\sigma}_<$, $\pi(c\psi') = c$ pour $c \in C$ puisque les termes de $\mathcal{R}an(\hat{\sigma}_<)$ ne sont pas unifiables. Avec cette hypothèse, $\forall c \in C,\ c\psi'^{\pi_1} = c$.

Lemme 2.19 $SU_E^<(\Gamma, C) \subseteq \{\psi'^{\pi_1} \mid \psi' \in SU_{E\cup\emptyset}(\Gamma \wedge \hat{\sigma}_<)\}$.

Preuve : Si ψ est solution de $(\Gamma, C, <)$ alors on peut construire par induction nœthérienne sur $<$ une substitution ψ' telle que $\psi =^{\mathcal{V}(\Gamma)} \psi'$ et $\forall c \in C,\ c\psi' = f_c(x_1\psi', \ldots, x_m\psi')$. Cette substitution ψ' est donc solution de $\Gamma \wedge \hat{\sigma}_<$, elle vérifie $\psi = \psi'^{\pi_1}$. □

Lemme 2.20 $SU_E^<(\Gamma, C) \supseteq \{\psi'^{\pi_1} \mid \psi' \in SU_{E\cup\emptyset}(\Gamma \wedge \hat{\sigma}_<) \downarrow_R\}$.

Preuve : Si ψ' est solution R-normalisée de $\Gamma \wedge \hat{\sigma}_<$ alors ψ'^{π_1} est solution de Γ, d'après le lemme 2.13. Il reste à montrer que $(\psi')^{\pi_1}$ est aussi solution de $(\Gamma, C, <)$, c'est-à dire qu'elle satisfait la restriction linéaire $<$. Les termes $f_c(x_1\psi', \ldots, x_m\psi')$ sont R-normalisés car les symboles f_c sont libres et ψ' R-normalisée. Donc $\forall c \in C, c\psi' = f_c(x_1\psi', \ldots, x_m\psi')$. Un terme $c\psi'$ ne pouvant être sous-terme de $x_1\psi', \ldots, x_m\psi'$, on a $c \notin (x_k\psi')^{\pi_1}$ pour $k = 1, \ldots, m$. □

Proposition 2.22 *La E-unification avec restriction linéaire est décidable si et seulement si la E-unification générale est décidable.*

Problème de résolution

Pour pouvoir étendre le résultat précédent à la résolution, il suffit d'utiliser la propriété que $\emptyset$ est une théorie non effondrante.

Lemme 2.21 *Si* E_2 *est une théorie non effondrante alors* $s =_{E_1\cup E_2} t \iff s^{\pi_1} =_{E_1} t^{\pi_1}$.

Preuve : La preuve est basée sur la proposition 2.9, ses corollaires et sur le fait que $t^{\pi_1} = (t \downarrow_R)^{\pi_1}$ si E_2 une théorie non effondrante et t un 2-terme. Dans ce qui suit, on note $\breve{t}$ le terme t dont les sous-termes étrangers sont remplacés par leurs R-formes normales

$$\breve{t} = t[\omega \hookleftarrow t_{|\omega} \downarrow_R]_{\omega \in AlienPos(t)}.$$

Supposons $s =_{E_1\cup E_2} t$. Trois cas sont à distinguer:

- Si s et t sont des 1-termes alors $s^{\pi_1} = \breve{s}^{\pi_1} =_{E_1} \breve{t}^{\pi_1} = t^{\pi_1}$.
- Si s est un 1-terme et t un 2-terme alors

$$s^{\pi_1} = \breve{s}^{\pi_1} =_{E_1} (s \downarrow_R)^{\pi_1} = (t \downarrow_R)^{\pi_1} = t^{\pi_1}.$$

- Si s et t sont des 2-termes alors $s^{\pi_1} = t^{\pi_1}$.

□

Corollaire 2.7 *Si E_2 est une théorie non effondrante alors $s \leq_{E_1 \cup E_2} t \Longleftrightarrow s^{\pi_1} \leq_{E_1} t^{\pi_1}$.*

Lemme 2.22 *L'ensemble des substitutions*

$$\{\phi'^{\pi_1} \mid \phi' \in CSU_{E \cup \emptyset}(\Gamma \wedge \hat{\sigma}_<)\}$$

est un $CSU_E^{\leq}(\Gamma, C)$.

Preuve : Nous avons montré en proposition 2.22 que pour toute solution $\psi \in SU_E(\Gamma, C, <)$, il existe ψ' tel que $\psi = \psi'^{\pi_1}$ avec $\psi' \in SU_{E \cup \emptyset}(\Gamma \wedge \hat{\sigma}_<)$. Il existe donc $\phi' \in CSU_{E \cup \emptyset}(\Gamma \wedge \hat{\sigma}_<)$ tel que $\phi' \leq_{E \cup \emptyset}^{\mathcal{V}(\Gamma)} \psi'$, et $\phi'^{\pi_1} \leq_E \psi'^{\pi_1}$ d'après le corollaire 2.7. Donc $\forall \psi \in SU_E(\Gamma, C, <)\ \exists \phi' \in CSU_{E \cup \emptyset}(\Gamma \wedge \hat{\sigma}_<)$ tel que $\phi'^{\pi_1} \leq_E^{\mathcal{V}(\Gamma)} \psi$. □

Théorème 2.5 *(F. Baader et K. Schulz [BS92]). La E-unification avec restriction linéaire est décidable (resp. finitaire) si et seulement si la E-unification générale est décidable (resp. finitaire).*

2.4.3 Disunification

Le problème traité dans ce paragraphe est celui de la résolution de problèmes équationnels formés à la fois d'équation $s =^? t$ et de *diséquations* $s \neq^? t$, qu'on appelle problème de *disunification*. La notion de solution à un problème de disunification $s \neq^? t$ est sujet à deux interprétations possibles

Pour J. Siekmann [Sie90], il s'agit d'une solution symbolique σ telle que $s\sigma \neq t\sigma$ est une diségalité *valide* dans la structure considérée. La notion d'ensemble complet de solutions garde alors son intérêt puisqu'une solution est stable par instanciation. Malheureusement, dans ce contexte la combinaison d'algorithme de disunification retournant un ensemble complet de disunificateurs est vouée à l'échec. La résolution d'une diséquation pure dans sa composante reste correcte mais n'est plus complète...

Exemple 2.3 *Considérons le mélange d'un symbole libre f, $E_1 = \{f(x) = f(x)\}$ et d'un symbole commutatif $+$, $E_2\{x + y = y + x\}$ et la diséquation $z \neq^? x + y$. La substitution $\sigma = \{x \mapsto f(x'), y \mapsto f(y'), z \mapsto f(z')\}$ est un disunificateur de $z \neq x + y$ dans le mélange $E_1 \cup E_2$. Par contre, la substitution $\sigma^{\pi_2} = \{x \mapsto X, y \mapsto Y, z \mapsto Z\}$ n'est bien évidemment pas un disunificateur de $z \neq x + y$ dans E_2. Cet exemple contredit donc une généralisation du lemme 2.13.*

Une alternative est de considérer une solution à un problème de disunification comme étant simplement une assignation satisfaisant le problème [Bür87]. Reste alors à choisir la structure adéquate.

Pour la combinaison, la structure étudiée jusqu'à présent est la théorie équationnelle $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/=_{E_1 \cup E_2}$. On pourra utiliser dans cette structure la négation du lemme 2.13:

$$s\sigma \neq_{E_1 \cup E_2} t\sigma \Leftrightarrow (s\sigma)^{\pi_i} \neq_{E_i} (t\sigma)^{\pi_i},$$

si s, t sont des termes i-purs. Mais dans ce contexte, une instance d'une solution n'est plus une solution. En conséquence, une solution combinée, construite par instanciation, n'est plus nécessairement une solution dans le mélange de théories équationnelles. Ce problème a été résolu par F. Baader et K. Schulz [BS93].

La structure considérée par H. Comon et P. Lescanne [Com88, CL88, Com91] est une algèbre de termes clos qui permet par exemple de résoudre des problèmes liés à la réductibilité inductive. Les algorithmes de disunification d'H. Comon et P. Lescanne suivent une approche par résolution: un problème de disunification est transformé jusqu'à aboutir à une forme résolue équivalente dont la propriété est d'avoir une solution si et seulement elle est différente de $\bot$. Une autre approche, celle de F. Baader et K. Schulz, consiste à montrer comment modifier l'algorithme de combinaison pour décider de l'existence d'une solution close. Nous allons en rappeler les grandes étapes et les principaux résultats.

Purification

On ramène un problème de disunification à une conjonction de problèmes purs $\Gamma_1 \wedge \Gamma_2$ où les seules diséquations sont de la forme $x \neq y$ avec $x, y \in \mathcal{X}$ et $x \neq y$. Pour ce faire, une diséquation $s \neq^? t$ est transformé en $x =^? s \wedge y =^? t \wedge x \neq^? y$.

En corollaire, nous allons avoir à disunifier les variables qui ne s'identifient pas.

Identification

Une identification permet d'identifier les variables dont les solutions ont même forme normale et permet de distinguer les variables qui sont égales à des formes normales distinctes.

Définition 2.22 *Etant donnée ξ une identification, on note $\xi_{\neq}$ le problème de disunification inter-variables suivant:*

$$\xi_{\neq} = \bigwedge_{x,y \in \mathcal{R}an(\xi), x \neq y} x \neq^? y.$$

Il va falloir considérer les problèmes de disunification de la forme $(\Gamma_1 \wedge \Gamma_2)\xi \wedge \xi_{\neq}$.

Existence d'une solution

L'algorithme de décision de l'existence d'une solution pour un problème de disunification dans $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/=_{E_1 \cup E_2}$ est basé sur le théorème suivant qui utilise la disunification avec restriction linéaire.

Théorème 2.6 *([BS93]) Il existe une $E_1 \cup E_2$-solution à $\Gamma_1 \wedge \Gamma_2$ si et seulement si il existe*

- $\xi \in ID_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$,
- *un ordre linéaire* $<$ *sur* $V_1 \oplus V_2 = \mathcal{V}((\Gamma_1 \wedge \Gamma_2)\xi)$,
- *une* E_1*-solution à* $(\Gamma_1\xi \wedge \xi_{\neq}, V_2, <)$,
- *une* E_2*-solution à* $(\Gamma_2\xi \wedge \xi_{\neq}, V_1, <)$.

La principale difficulté consiste à montrer que si $\sigma_1 \in SS^{<}(\Gamma_1, V_2)$ et $\sigma_2 \in SS^{<}(\Gamma_2, V_1)$ alors $(\sigma_1 \odot \sigma_2)\downarrow_R$ est $E_1 \cup E_2$-solution. Il faut absolument que la solution combinée soit R-normalisée pour être une $E_1 \cup E_2$-solution. La réciproque est, elle, inchangée.

On notera que la décision de l'existence d'une solution à un problème de disunification est triviale si l'on dispose d'un algorithme de $E_1 \cup E_2$-unification. On peut alors dans un premier temps résoudre le sous-problème d'unification puis tester à l'aide d'un algorithme de décision de la $E_1 \cup E_2$-égalité si les solutions de variables x, y ne sont pas $E_1 \cup E_2$-égaux si $x \neq^? y$ est une diséquation du problème. La construction modulaire d'un algorithme de décision de la $E_1 \cup E_2$-égalité, ou problème du mot, sera vue au chapitre 4.

L'intérêt de la disunification dans $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/=_{E_1 \cup E_2}$ doit donc relativisé puisque un algorithme de $E_1 \cup E_2$-unification suffit pour résoudre le problème.

Existence d'une solution close

On s'intéresse désormais à la décision de l'existence d'une solution close, c'est-à-dire une substitution σ telle que $\mathcal{VR}an(\sigma) = \emptyset$. Ce problème est identique à celui que se pose H. Comon, mais sera résolu différemment, sans l'aide d'une forme résolue.

A première vue, il serait tentant de vouloir adapter le théorème 2.6 en remplaçant simplement la notion de *solution* par celle de *solution close*.

Pour la correction, si σ_1 est solution close de $(\Gamma_1\xi \wedge \xi_{\neq}, V_2, <)$ (où V_2 est un ensemble de constantes libres) et σ_2 une solution close de $(\Gamma_2\xi \wedge \xi_{\neq}, V_1, <)$ (où V_1 est un ensemble de constantes libres) alors $(\sigma_1 \odot \sigma_2)\downarrow_R$ est une solution de $\Gamma_1 \wedge \Gamma_2$. Il s'agit d'une solution close car $\mathcal{VR}an(\sigma_{1|V_1}) = \mathcal{VR}an(\sigma_{2|V_2}) = \emptyset$ par définition et surtout parce qu'on peut facilement s'assurer de l'existence d'un terme clos plus petit que toute variable. La forme normale d'une substitution close est par conséquent close.

Pour la complétude, si σ est une $E_1 \cup E_2$-solution alors σ^{π_1} et σ^{π_2} sont des solutions satisfaisant une certaine restriction linéaire mais ne sont pas closes car elles peuvent contenir des variables qui ne sont pas instanciées par des termes clos dans l'autre théorie. Par contre, σ^{π_1} et σ^{π_2} sont des solutions restrictives.

Définition 2.23 *([BS93]) Une* E*-solution* σ *est* restrictive *si* $\forall x \in \mathcal{D}om(\sigma), x\sigma \neq_E y \in \mathcal{X}$.

La question est de savoir s'il existe une E-solution à un problème $\Gamma_1 \wedge \Gamma_2$ lorsqu'il existe une E_1-solution restrictive à un problème $(\Gamma_1\xi \wedge \xi_{\neq}, V_2, <)$ et une E_2-solution restrictive à un problème $(\Gamma_2\xi \wedge \xi_{\neq}, V_1, <)$. Il est possible de répondre par l'affirmative mais seulement si les domaines des algèbres $\mathcal{T}(\mathcal{F}_1)/=_{E_1}$ et $\mathcal{T}(\mathcal{F}_2)/=_{E_2}$ sont infinis, auquel cas il est possible de construire une solution close à un problème de disunification inter-variables $\xi_{\neq}$.

Théorème 2.7 *([BS93]) Soient* E_1 *et* E_2 *des théories disjointes telles que les domaines de* $\mathcal{T}(\mathcal{F}_1)/=_{E_1}$ *et* $\mathcal{T}(\mathcal{F}_2)/=_{E_2}$ *sont infinis. Il existe une* $E_1 \cup E_2$*-solution close à* $\Gamma_1 \wedge \Gamma_2$ *si et seulement si il existe*

- $\xi \in ID_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$,
- *un ordre linéaire* $<$ *sur* $V_1 \oplus V_2 = \mathcal{V}((\Gamma_1 \wedge \Gamma_2)\xi)$,
- *une* E_1*-solution restrictive à* $(\Gamma_1\xi \wedge \xi_{\neq}, V_2, <)$,
- *une* E_2*-solution restrictive à* $(\Gamma_2\xi \wedge \xi_{\neq}, V_1, <)$.

L'hypothèse faite sur les théories peut être relâchée sur E_2. Il suffit de supposer $T(\mathcal{F}_1)/=_{E_1}$ infini et que pour tout problème satisfaisable dans E_1, il existe une solution close. En particulier, s'il existe une solution restrictive, alors il existe une solution close. Le fait que seul $T(\mathcal{F}_1)/=_{E_1}$ soit infini suffit pour construire une solution à $\xi_{\neq}$. Cette remarque s'applique au cas où E_1 est la théorie vide et $\mathcal{F}_1$ contient au minimum une constante et un symbole qui ne l'est pas. La théorie E_2 peut être par exemple la théorie associative dont la disunification avec restriction linéaire est décidable mais dont l'unification est infinitaire.

Corollaire 2.8 *([BS93]) Soit* $\mathcal{F}$ *un ensemble de symboles contenant une constante, un symbole non constant et d'autres symboles sur lesquels est formé un ensemble A d'axiomes associatifs. La théorie existentielle de* $\mathcal{T}(\mathcal{F})/=_A$ *est décidable.*

2.5 Comparaison des approches par résolution et décision

La présentation des deux approches s'est faite dans le souci d'homogéneiser les notations et les concepts utilisés par les différents auteurs. Nous avons vu dans un premier temps comment combiner des algorithmes d'unification avec restriction en suivant une démarche par transformation successive de problèmes équationnels. Ensuite, il a été montré que la restriction linéaire suffisait pour l'unification mais aussi pour l'unifiabilité. L'algorithme de combinaison correspondant ne fait plus appel à des règles de transformation mais consiste simplement à rajouter l'information nécessaire à la définition d'une restriction linéaire. Finalement, nous avons rappelé qu'il y a équivalence entre unification avec symboles libres et unification avec restriction linéaire. Il est en effet possible de coder une restriction linéaire par un problème d'unification avec symboles libres alors que le même codage n'est plus valable avec une restriction quelconque, comme le montre l'exemple suivant, tiré de [BS92].

Exemple 2.4 *Le problème d'unification* $(x=^? c \wedge y=^? d, \{c, d\}, \{(c, y), (d, x)\})$ *a pour solution évidente* $\{x \mapsto c, y \mapsto d\}$ *alors que son codage avec symboles libres*

$$x=^? c \wedge y=^? d \wedge c=^? f_c(y) \wedge d=^? f_d(y),$$

où c, d *sont des variables, n'a pas de solution.*

Il n'est donc pas envisageable d'interchanger les notions de restriction et restriction linéaire puisque elles se comportent différemment vis-à-vis de l'unification avec symboles libres. L'unification avec restriction linéaire est donc suffisante pour la combinaison d'algorithmes et permet de par son équivalence avec l'unification générale d'obtenir un résultat

de modularité qui n'a pu être démontré pour l'unification avec restriction. Ces remarques laissent à penser que l'unification avec restriction linéaire ou de façon équivalente l'unification générale est le bon concept à combiner pour le mélange de théories disjointes. Cependant, l'unification avec restriction est utile aussi en pratique puisque:

Proposition 2.23 *Si E est une théorie stable alors la E-unification avec restriction linéaire est finitaire si et seulement si la E-unification avec restriction est finitaire.*

Les règles de transformation de la figure 2.2 peuvent être vues comme une implantation des principes de combinaison qui ont été dégagés, mais plus tard par F. Baader et K. Schulz. Le plus grand intérêt à mon sens de l'approche par décision prise par F. Baader et K. Schulz, dans le prolongement de M. Schmidt-Schauß, est d'avoir pu isoler des concepts de combinaison qui sont centrés sur l'unification avec restriction linéaire. Il en résulte un algorithme de combinaison des solutions totalement non-déterministe faisant appel de façon systématique aux algorithmes d'unification avec restriction linéaire de chaque théorie composante. Cet algorithme de combinaison présente en outre l'avantage de terminer trivialement. On ne pouvait donc rêver plus simple sur le principe. Cette apparente simplicité sur la forme va faciliter l'extension de ces concepts de combinaison, ou *règles de combinaison* à d'autres cas que la seule unification dans le mélange de théories disjointes. Cette approche permet aussi de résoudre un problème de décision, ce qui n'est pas négligeable par exemple pour la déduction ou la programmation avec contraintes, où un algorithme de satisfaisabilité suffit dans beaucoup de situations. Pour le seul problème de décision, l'algorithme de combinaison non-déterministe semble difficile à améliorer dans le cas général puisque aucune transformation ne peut être *a priori* autorisée. Par contre, pour le problème de résolution, il serait tentant de vouloir utiliser la connaissance accrue sur chaque théorie pour essayer de déterminiser, ou rendre opérationnelle la méthode de combinaison décrite par M. Schmidt-Schauß puis par F. Baader et K. Schulz. L'idée d'A. Boudet est justement d'utiliser en premier lieu les algorithmes d'unification, sans restriction, et de ne pratiquer d'étapes non-déterministes que lorsqu'elles s'avèrent nécessaires, c'est-à-dire lors d'un conflit inter-théories. Le choix d'une théorie pour une variable n'est ainsi effectué que lorsque celle-ci apparait résolue dans les deux théories et qu'il y a donc conflit.

La différence fondamentale entre l'algorithme non-déterministe (celui de F. Baader et K. Schulz) et l'algorithme déterministe (celui de A. Boudet) est que dans le premier, la restriction linéaire est imposée dès le départ alors que dans le second, cette restriction (linéaire) se construit au fur et à mesure et seulement si nécessaire. Cette approche déterministe semble idéale à première vue. Cependant, il faut remarquer que la construction de la restriction est relativement coûteuse. Un conflit de théorie peut être interprété comme un constat d'échec à l'approche déterministe: une étape de résolution aurait pu être évitée si le choix d'une théorie avait été fait préalablement. La même remarque est valable pour un cycle composé dont la détection, moins évidente, nécessite cette fois un tri topologique. L'étape de résolution ayant engendré un cycle composé apparait comme superflue puisque elle n'a pas réussi à l'empêcher et qu'il va donc falloir reconsidérer son cas. Le comportement de l'algorithme déterministe est donc nettement moins bon dans le pire cas lorsque toutes les restrictions linéaires doivent être construites pour aboutir à un ensemble complet de solutions. Par contre, il a un intérêt lorsque les algorithmes d'unification des théories composantes ont la particularité d'instancier aussi peu de variables

que possible. C'est d'ailleurs comme cela que A. Boudet motive son algorithme. Il faudrait aussi supposer qu'il y ait aussi peu de variables que possible qui soient introduites dans les solutions. Au vu de toutes ces remarques, un problème qui se résout plus rapidement avec l'algorithme de A. Boudet doit être très particulier.

Une autre critique, plus fondamentale, est liée à la sémantique du prédicat *Arc*. Intuitivement, il correspond à l'ordre linéaire que doit satisfaire la relation d'"Occur-Check" d'une solution. Mais ce prédicat n'est pas interprété par une relation d'ordre sur les variables. Il n'est donc pas tenu compte de l'antisymétrie et de la transitivité de l'ordre linéaire, ce qui conduit à de trop nombreuses éliminations de constante, comme le montre l'exemple suivant:

Exemple 2.5 *Considérons un problème séparé* $P_1 \wedge P_2$, *avec* $P_1 = (x =^? t_1[y][z] \wedge v =^? t'_1[y])$ *et* $P_2 = (y =^? t_2[x] \wedge z =^? t'_2[v])$. *Les variables apparaissant dans les termes ont été mises entre crochets. Ce problème contient deux cycles composés et pour les résoudre, on est amené à éliminer successivement* x *dans* t_2 *et* y *dans* t_1, *c'est-à-dire à considérer* $Arc_{E_1}(y, x)$ *et* $Arc_{E_2}(x, y)$, *ce qui n'est pas nécessaire d'après l'approche par décision puisque un ordre linéaire suffit.*

Par conséquent, le prédicat *Arc* n'est qu'une grossière approximation de l'ordre linéaire. Beaucoup de solutions superflues sont de ce fait calculées et n'auraient pas besoin de l'être.

Dans l'algorithme déterministe, l'élimination de constante n'est pas le seul outil pour la résolution des cycles. Il faut aussi utiliser l'unification avec constantes. Ce dernier mécanisme peut paraître surprenant à ce stade mais s'explique par l'absence d'un choix de théorie pour une variable du cycle. L'utilisation des algorithmes d'unification avec constantes et élimination de constante n'est pas clairement séparée comme elle l'est dans la construction d'un algorithme d'unification avec restriction. L'unification avec constantes n'est pas à proprement parler un mécanisme pour la résolution d'un cycle composé mais plutôt de quoi réparer un oubli car rien ne préjuge de la forme définitive du problème séparé obtenu après avoir choisi une théorie pour chaque variable d'un cycle composé.

L'algorithme d'unification à base de règles présenté par A. Boudet est intéressant à bien des égards. La preuve qu'il en fait suit des normes bien établies. La preuve de *Terminaison* est comme toujours dans ce contexte la partie la plus délicate à réaliser alors qu'elle devient triviale dans l'approche de F. Baader et K. Schulz. La preuve de *Correction* a nécessité l'introduction de nouveaux prédicats pour la résolution de cycle inter-théories par des règles correctes et complètes. Les prédicats M_{E_i} et Arc_{E_i} ont pourtant un statut particulier puisqu'ils n'ont pas de sémantique dans le mélange $E_1 \cup E_2$ mais seulement dans la théorie E_i. Grâce au lemme 2.13, il est bien sûr possible de faire correspondre une $E_1 \cup E_2$-solution R-normalisée à une E_i-solution par i-abstraction mais cela ne permet pas stricto sensu de parler de règles de Conflit et Cycle préservant l'ensemble des solutions.

Au vu de ces remarques, le problème de combinaison a permis de mettre en évidence les limites du formalisme à base de règles de transformation. Une version à base de règles de l'algorithme de combinaison s'accorde mal en effet au caractère intrinsèquement non-déterministe de la combinaison. Les problèmes rencontrés pour la preuve de *Correction* et *Terminaison* de l'algorithme "déterministe" sont artificiels car non pas liés à la combinaison en elle même mais plutôt à l'approche par règles de transformation. L'algorithme d'unification à base de règles représente finalement un contrôle sur la construction d'une

restriction linéaire, ce qui est en contradiction avec la séparation souhaitée entre contrôle et transformation. Il est pourtant des cas de combinaison où le non-déterminisme est relativement restreint voire inexistant. Dans ces cas, une approche à base de règles de transformation retrouve tout son intérêt, comme nous l'avons vu pour le mélange de théories régulières et non effondrantes. Mais en général, les algorithmes de combinaison présentés dans la suite le seront en suivant une approche par décision. Nous donnerons des *principes de combinaison* pour préciser comment

- purifier un problème,
- identifier les variables,
- choisir une théorie pour chaque variable,
- choisir un ordre linéaire.

Mais il ne s'agira pas de règles de transformation comme celles, en principe déterministes, définissant un algorithme d'unification à base de règles.

2.6 Procédures d'élimination de constante

Le problème de l'élimination de constante joue un rôle majeur dans la combinaison d'algorithmes d'unification. Nous allons rappeler les algorithmes connus pour certaines théories spécifiques. Ensuite nous verrons des procédures plus générales basées, elles, sur la surréduction. Mais il vain d'espérer un algorithme d'élimination de constante universel puisque ce problème, tout comme l'unification, est indécidable en général.

Proposition 2.24 *Il existe une théorie dans laquelle l'élimination de constante est indécidable.*

Preuve : Considérons la théorie $DA(*,+) \cup \{h(x,y,y) = g(y)\}$ dont l'unification est indécidable car l'unification dans DA est indécidable [Sza79] et les symboles h, g disjoints de $+$ et $*$. Les éliminateurs de c dans $h(c,s,t)$ sont les unificateurs de s et t. L'élimination de c dans $h(c,s,t)$ est donc elle aussi indécidable. □

Il existe aussi des théories dans lesquelles l'élimination de constante est moins complexe que l'unification. Considérons par exemple la théorie $A0 = A(+) \cup \{x + 0 = 0\}$. Eliminer une constante c dans un terme t est équivalent dans $A0$ à unifier t et 0, un problème de $A0$-unification finitaire. Par contre, la $A0$-unification est en général infinitaire.

L'élimination de constante est fortement liée à la résolution de cycles composés mais ne suffit pas pour résoudre des cycles dans une seule théorie.

Proposition 2.25 *Il existe une théorie équationnelle, un terme t et un cycle $x =^? t[x]_\omega$, $\omega \neq \epsilon$, dont l'unification est décidable mais pour lequel la résolution du problème d'élimination de constante élémentaire $(x =^? t[c]_\omega, \{c\}, \{(c,x)\})$ est indécidable.*

Preuve : Il suffit de considérer la théorie $DA(*,+) \cup \{h(x,y,y) = g(y,y), h(g(y,z),y,z) = g(y,z)\}$ et le problème $x =^? h(x,s,u)$. Un unificateur principal est $\{x \mapsto g(s,u)\}$. Mais l'élimination de c dans $h(c,s,u)$ est, lui, équivalent à unifier s et u, un problème indécidable dans cette théorie. □

Le problème encore ouvert concerne le lien entre unification avec constantes et élimination de constante. La question est:
"Existe-t-il une théorie dont l'unification avec constantes est finitaire mais dont l'élimination de constante est infinitaire?"

Nous ne connaissons pas à l'heure actuelle de théories qui ne soient pas stables et qui pourraient donc permettre de répondre positivement à la question.

2.6.1 Exemples de théories stables et finitaires

On rappelle ici les résultats connus concernant l'élimination de constante.

Nous avons déjà mentionné que ce problème est trivial dans les théories régulières. Dans ce cas, soit la forme résolue satisfait la restriction linéaire, soit elle ne la satisfait pas et le problème n'a pas de solution. On a donc simplement un test syntaxique à effectuer.

Il existe pourtant des théories intéressantes en pratique et qui ne sont pas régulières. On peut citer par exemple les anneaux booléens (BR) et les groupes abéliens (AG) qui sont toutes les deux des théories dont l'unification est unitaire.

Théorème 2.8 *(M. Schmidt-Schauß [SS89]) L'élimination de constante dans BR est unitaire.*

La solution satisfaisant une restriction est obtenue à partir d'un unificateur principal par annulation des coefficients des constantes à éliminer. Le même principe est utilisé pour l'élimination de constante dans les groupes abéliens.

Théorème 2.9 *(M. Schmidt-Schauß [SS89]) L'élimination de constante dans AG est unitaire.*

Il existe finalement relativement peu d'axiomes utiles en pratique et non réguliers. Citons par exemple l'axiome inverse, $Inv(*, e) = \{x * i(x) = e\}$, pour AG et l'axiome absorbant, $Abs(*, 0) = \{x * 0 = 0\}$ pour BR. A. Boudet s'est intéressé à l'unification dans $AC0 = AC(*) \cup Abs(*, 0)$ en montrant comment réutiliser l'unification dans AC. L'élimination de constante dans $AC0$ est équivalent à unifier les termes avec 0 lorsqu'ils contiennent des constantes à éliminer.

Théorème 2.10 *(A. Boudet [Bou90a]) L'élimination de constante dans $AC0$ est finitaire.*

Après ce bref récapitulatif, mentionnons un résultat prouvé au chapitre 6 concernant l'élimination de constante dans les algèbres primales qui sont approximativement des algèbres pour lesquelles toute fonction peut être représentée par un terme.

Théorème 2.11 *L'élimination de constante dans les algèbres primales est unitaire.*

L'algorithme d'élimination de constante que nous présentons au chapitre 6 est basé sur l'unification qui a la propriété d'être unitaire dans les algèbres primales.

2.6.2 Procédures par surréduction

De même que pour l'unification, la surréduction permet d'obtenir une procédure complète d'énumération des solutions pour le problème de l'élimination de constante.

Considérons une théorie équationnelle présentée par un système de réécriture R et un ensemble d'axiomes E.

Définition 2.24 *Un terme s se* surréduit *modulo E en un terme t s'il existe une position $\omega \in \mathcal{FP}os(s)$, un renommage d'une règle $l \to r$ de R et une substitution σ tels que*

- $\sigma \in CSU_E(t_{|\omega} =^? l)$,
- $t = (s[r]_\omega)\sigma$,

ce qu'on note $s \leadsto_E t$ ou simplement $s \leadsto t$ lorsque E est vide.

La relation $\leadsto_E$ est appelée surréduction modulo E ou simplement surréduction lorsque E est vide. On note $s \leadsto^*_{E,\sigma} t$ s'il existe une dérivation de surréduction

$$s = t_1 \leadsto_{E,\sigma_1} t_2 \leadsto_{E,\sigma_2} \cdots \leadsto_{E,\sigma_{n-1}} t_{n-1} = t$$

telle que $\sigma = \sigma_1\sigma_2 \cdots \sigma_{n-1}$.

Considérons d'abord le cas où E est vide.

Proposition 2.26 *Soit $(x =^? t[c], \{c\}, \{(c, x)\})$ un problème d'élimination de constante élémentaire. Si R est un système de réécriture convergent alors*

$$\{\{x \mapsto u\}\sigma \mid t \leadsto^*_\sigma u, c \notin u\}$$

est un $CSU_R(x =^? t[c], \{c\}, \{(c, x)\})$.

Preuve : (Correction). Si $t \leadsto^*_\tau u$ tel que $c \notin u$ alors $t\tau \to^*_R u$ et σ est un R-éliminateur de c dans t.

(Complétude). Soit τ une substitution R-normalisée. Si τ est un R-éliminateur de c dans t alors $t\tau \to^*_R u \downarrow_R$ avec $c \notin u \downarrow_R$. Si l'on suppose en effet $c \in u \downarrow_R$ alors il existe u' tel que $c \notin u'$ et $u' \to^*_R u \downarrow_R$, ce qui est absurde puisque les règles $l \to r$ de R vérifient $\mathcal{V}(l) \supseteq \mathcal{V}(r)$.

D'après le lemme de relèvement [Hul80] établissant le lien entre réécriture et surréduction, nous avons ensuite $t \leadsto^*_\sigma v$ avec $v \leq u \downarrow_R$ et $\sigma \leq^{\mathcal{V}(t)}_R \tau$. Puisque $c \notin u \downarrow_R$ et $v \leq u \downarrow_R$, on a $c \notin v$ et $\{x \mapsto v\}\sigma \leq^{\mathcal{V}(t)}_R \{x \mapsto u \downarrow_R\}\tau$. □

Corollaire 2.9 *(M. Schmidt-Schauß [SS89]) Soit R un système de réécriture convergent. S'il existe une stratégie de surréduction complète et terminante alors*

- *l'élimination de constante élémentaire est finitaire,*
- *R est stable.*

Ce résultat peut être étendu à un système de réécriture R convergent modulo E puisque E est alors nécessairement une théorie régulière.

Proposition 2.27 *Soit* $(x=^?t[c], \{c\}, \{(c,x)\})$ *un problème d'élimination de constante élémentaire. Si* R *est un système de réécriture convergent modulo* E*, alors*

$$\{\{x \mapsto u\}\sigma \mid t \leadsto^*_{E,\sigma} u, c \notin u\}$$

est un $CSU_{R\cup E}(x=^?t[c], \{c\}, \{(c,x)\})$.

Preuve : (Correction). Si $t \leadsto^*_{E,\tau} u$ tel que $c \notin u$ alors $t\tau \to^*_{R\cup E} u$ et σ est un $R \cup E$-éliminateur de c dans t.

(Complétude). Soit τ une substitution normalisée pour $\to_{R,E}$. Si τ est un $R \cup E$-éliminateur de c dans t alors $t\tau \to^*_{R/E} u \downarrow_{R,E}$ avec $c \notin u \downarrow_{R,E}$. On ne peut pas en effet avoir $c \in u \downarrow_{R,E}$ car $s \to_{R,E} t$ implique $\mathcal{V}(s) \supseteq \mathcal{V}(t)$ puisque E est un ensemble d'axiomes réguliers.

D'après le lemme de relèvement [Kir85a] établissant le lien entre réécriture et surréduction modulo E, nous avons ensuite $t \leadsto^*_{E,\sigma} v$ avec $v \leq_E u \downarrow_{R,E}$ et $\sigma \leq^{\mathcal{V}(t)}_{R\cup E} \tau$.

L'hypothèse de régularité des axiomes de E permet encore une fois d'affirmer que $c \notin v$ lorsque $c \notin u \downarrow_{R,E}$ et $v \leq_E u \downarrow_{R,E}$. La substitution τ est donc une instance d'une solution obtenue par surréduction: $\{x \mapsto v\}\sigma \leq^{\mathcal{V}(t)}_{R\cup E} \{x \mapsto u \downarrow_R\}\tau$ avec $c \notin v$. □

Corollaire 2.10 *Soit* R *un système de réécriture convergent modulo* E*. S'il existe une stratégie de surréduction modulo* E *complète et terminante alors*

- *l'élimination de constante élémentaire est finitaire,*
- $R \cup E$ *est stable.*

La surréduction modulo AC fournit par exemple un algorithme d'élimination de constante dans la théorie $AC0 = \{x * 0 \to 0\} \cup AC(*)$.

2.7 Modularité de l'unification

Cette section se veut un bref récapitulatif des résultats concernant la modularité de l'unification; ceux-ci sont des conséquences directes des algorithmes introduits précédemment.

Le théorème central, énoncé ci-dessous, concerne toutes les théories mais l'unification avec symboles libres, ou unification générale.

Théorème 2.12 *(F. Baader et K. Schulz [BS92]). Si* E_1 *et* E_1 *sont deux théories disjointes alors la* $E_1 \cup E_2$*-unification générale est décidable (resp. finitaire) si et seulement si la* E_i*-unification générale (*$i = 1, 2$*) est décidable (resp. finitaire).*

Preuve : La $(E_1 \cup \emptyset) \cup E_2$-unification est décidable (resp. finitaire) si la $(E_1 \cup \emptyset)$-unification et la E_2-unification avec restriction linéaire sont décidables (resp. finitaires), c'est-à-dire si la $E_1 \cup \emptyset$-unification générale et la E_2-unification générale sont décidables (resp. finitaires), où la $E_1 \cup \emptyset$-unification générale est la E_1-unification générale. □

Si les théories E_1 et E_2 sont stables alors il n' y a pas de différence entre unification générale et unification avec constantes.

Théorème 2.13 *Si E_1 et E_1 sont deux théories stables et disjointes alors la $E_1 \cup E_2$-unification avec constantes est finitaire si et seulement si la E_i-unification avec constantes ($i = 1, 2$) est finitaire.*

Les théories régulières sont stables puisque l'élimination de constante élémentaire n'a pas de solution. Si elles sont en plus non effondrantes alors le filtrage sur une constante n'a pas non plus de solution et l'on obtient le même résultat que celui vu pour les théories simples en guise d'introduction.

Théorème 2.14 *(K. Yelick [Yel87]). Si E_1 et E_1 sont deux théories régulières, non effondrantes et disjointes alors la $E_1 \cup E_2$-unification est finitaire si et seulement si la E_i-unification ($i = 1, 2$) est finitaire.*

Bürckert [Bür90] a montré qu'il existe une théorie dans laquelle l'unification avec constantes est indécidable alors que l'unification est décidable. La même question se pose à présent entre unification générale et unification avec constantes. Un élément de réponse a déjà été apporté par Narendran et Otto [NO90] qui ont exhibé une théorie dont la résolution d'une équation quelconque est décidable alors que la résolution d'un système d'équations ne l'est plus. Puisque un système $\bigwedge_{k=1}^{m} s_k =^? t_k$ est équivalent à une seule équation $f(s_1, \ldots, s_m) =^? f(t_1, \ldots, t_m)$ avec f un symbole libre, on peut en conclure que:

Proposition 2.28 *Il existe une théorie dont l'unification générale d'une équation est indécidable alors que l'unification d'une équation est décidable.*

Dans le même ordre d'idée, on peut se demander s'il existe une théorie dont l'unification générale est infinitaire alors que l'unification avec constantes est finitaire. Une telle théorie ne peut pas être stable.

3

Unification dans les mélanges de théories non disjointes

La construction modulaire d'un algorithme d'unification est cette fois fondée sur la décomposition d'une théorie E en deux sous-théories E_1 et E_2 partageant éventuellement des symboles et dont la combinaison des algorithmes d'unification respectifs permet d'en obtenir un pour le mélange de théories $E = E_1 \cup E_2$.

Les conditions que nous donnons vont donc dans le sens de la décomposabilité et s'apparentent ainsi beaucoup aux conditions suffisantes pour la modularité de propriétés liées aux systèmes de réécriture [KO92]. L'idée est d'étendre la méthode initiée pour le cas disjoint qui consiste à introduire un système de réécriture convergent formé de deux sous-systèmes (un par théorie) sans paire critique.

L'existence d'un tel système de réécriture combiné permet de pouvoir résoudre sans perte de complétude un sous-problème pur dans sa théorie composante. Bien qu'il s'agisse là d'une propriété essentielle pour la réutilisation "intelligente" d'un algorithme d'unification, elle ne permet pas pour autant de régler les conflits inter-théories qui surviennent lorsque les solutions respectives des algorithmes des deux théories composantes doivent être composées. Le problème de la combinaison des solutions est fondamentalement différent dans le contexte du mélange non disjoint, suivant que l'on dispose de quoi énumerer un ensemble complet fini de solutions ou que l'on sache seulement décider de l'existence d'une solution.

La principale difficulté provient du fait qu'une variable peut se trouver désormais instanciée dans les deux théories par un terme formé de symboles partagés. Les méthodes que nous allons présenter consistent à introduire explicitement, lorsque c'est possible, un nombre fini de contextes partagés soit pour résoudre les conflits inter-théories, soit pour éviter leurs apparitions. Deux algorithmes sont ainsi proposés: l'un s'appuie sur la possibilité de transformer un problème en une forme résolue alors que l'autre pratique des transformations beaucoup plus élémentaires, ce qui lui permet de s'appliquer à la fois au problème de décision et à celui de résolution. Mais la différence sur les formes des mélanges de théories auxquelles s'appliquent ces deux algorithmes ne permet pas de conclure que l'un est plus général que l'autre. Ils fonctionnent d'ailleurs différemment puisque chacun d'eux a sa propre vision d'un sous-problème formé exclusivement de symboles partagés. Pour l'un, il s'agit d'un sous-problème pur simultanément dans les deux théories alors que pour l'autre, ce sous-problème doit être résolu à part, dans la théorie vide.

Comme précédemment dans le cas du mélange de théories disjointes, nous allons intro-

duire le problème de la combinaison des solutions par une forme de mélange pour laquelle un conflit inter-théories n'a pas de solution.

Ce chapitre est le fruit d'une étude menée en collaboration avec E. Domenjoud et F. Klay.

3.1 Théories décomposables

On considère deux théories équationnelles E_1 et E_2 dont les signatures respectives sont $\mathcal{F}_1$ et $\mathcal{F}_2$ avec $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ et $\mathcal{F}_1 \cap \mathcal{F}_2 = \mathcal{SF}$ (abbréviation de "$\mathcal{S}$hared $\mathcal{F}$unctions").

Un moyen pour éviter les paires critiques entre théories est de limiter les égalités ayant un symbole partagé en tête. Nous allons supposer que les égalités dont les deux membres ont un symbole partagé en tête sont décomposables et que les autres égalités s'orientent en règles de réécriture ayant des symboles non partagées en tête des membres gauches.

L'ordre de réduction $>$ introduit dans le cas disjoint sert maintenant aussi à orienter correctement les égalités de manière à ce que les symboles partagés puissent être vus comme des constructeurs.

Définition 3.1 *Une théorie $(\mathcal{F}, E) = (\mathcal{F}_1, E_1) \cup (\mathcal{F}_2, E_2)$ est* décomposable *s'il existe un ordre de réduction $>$ sur $\mathcal{T}(\mathcal{F} \cup \mathcal{X})$ tel que*

- $(E_i, >) = \{s \to t \mid s =_{E_i} t,\ s > t,\ s,t \in \mathcal{T}(\mathcal{F} \cup \mathcal{X})\}$ *est un système de réécriture convergent sur $\mathcal{T}(\mathcal{F} \cup \mathcal{X})$ vérifiant $=_{E_i} = \overset{*}{\longleftrightarrow}_{(E_i,>)}$.*
- *Les symboles* partagés *de $\mathcal{SF} = \mathcal{F}_1 \cap \mathcal{F}_2$ sont des constructeurs pour $(E_i, >)$ i.e.*

$$\forall f \in \mathcal{SF},\ f(t_1, \ldots, t_p) \downarrow_{(E_i,>)} = f(t_1 \downarrow_{(E_i,>)}, \ldots, t_p \downarrow_{(E_i,>)}).$$

Le mélange de théories à signatures disjointes est l'exemple trivial de théorie décomposable car $\mathcal{SF}$ est vide. Il suffit de prendre un ordre de simplification total sur $\mathcal{T}(\mathcal{F} \cup \mathcal{X})$.

Le mélange de théories représenté par l'union de systèmes de réécriture canoniques respectant la discipline de constructeurs [MT91] est lui aussi décomposable lorsque les symboles partagés sont des constructeurs. Il suffit de prendre la relation de réécriture pour ordre de réduction.

Corollaire 3.1 *Si $E = E_1 \cup E_2$ est une théorie décomposable alors*

- $\forall s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}),\ \forall f \in \mathcal{SF},$

$$s = f(s_1, \ldots, s_p) =_{E_i} f(t_1, \ldots, t_p) = t \Leftrightarrow \forall k \in \{1, \ldots, p\},\ s_k =_{E_i} t_k.$$

  Les symboles de $\mathcal{SF}$ sont dits ω-libres *[Kir85a].*

- $\forall s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}),\ \forall f, g \in \mathcal{SF},$

$$s = f(s_1, \ldots, s_p) \neq_{E_i} g(t_1, \ldots, t_p) = t \quad si \quad f \neq g$$

3.1.1 Système de réécriture combiné

On suppose dorénavant que $E = E_1 \cup E_2$ est une théorie décomposable.

Les systèmes de réécriture associés à chaque théorie composante sont équivalents à des systèmes n'ayant pas de symboles partagés en tête des membres gauches. Soit

$$E_i^{>} = \{l\sigma \to r\sigma \mid l =_{E_i} r,\ l, r \in \mathcal{T}(\mathcal{F}_i, \mathcal{X}), \mathcal{R}an(\sigma) \subseteq \mathcal{T}(\mathcal{F} \cup \mathcal{X}), l\sigma > r\sigma, l(\epsilon) \in \mathcal{F}_i \backslash \mathcal{SF}\}.$$

Le système de réécriture $E_1^{>} \cup E_2^{>}$ a le même pouvoir de preuve que E si cette théorie est décomposable.

Lemme 3.1 $t \xrightarrow{*}_{E_i^{>}} t \downarrow_{(E_i,>)}$

Preuve : Par induction sur $|t|$.

Si $t = x \in \mathcal{X}$ alors $x = x \downarrow_{(E_i,>)}$, sinon $>$ n'est pas un ordre de réduction.

Si $t(\epsilon) = f \in \mathcal{SF}$ alors, par hypothèse d'induction,

$$t = f(t_1, \ldots, t_p) \xrightarrow{*}_{E_i^{>}} f(t_1 \downarrow_{(E_i,>)}, \ldots, t_p \downarrow_{(E_i,>)}) = t \downarrow_{(E_i,>)} .$$

Si $t(\epsilon) \in \mathcal{F}_j \backslash \mathcal{SF}$ alors $t \downarrow_{(E_i,>)} = f(t_1, \ldots, t_p) \downarrow_{(E_i,>)} = f(t_1 \downarrow_{(E_i,>)}, \ldots, t_p \downarrow_{(E_i,>)})$, sinon $>$ n'est pas un ordre de réduction. C'est donc le même cas que le précédent.

Considérons le cas où $t(\epsilon) \in \mathcal{F}_i \backslash \mathcal{SF}$. Si t est i-pur alors il existe une substitution σ et une égalité $l =_{E_i} r$ telles que $t = l = l\sigma =_{E_i} r\sigma = t \downarrow_{(E_i,>)}$. Sinon, par hypothèse d'induction, $t \xrightarrow{*}_{E_i^{>}} t[\omega \hookleftarrow t_{|\omega} \downarrow_{(E_i,>)}]_{\omega \in AlienPos(t)} = u$. Il existe alors une substitution σ et une égalité $l =_{E_i} r$ telles que $u = l\sigma =_{E_i} r\sigma = u \downarrow_{(E_i,>)} = t \downarrow_{(E_i,>)}$ où $\mathcal{R}an(\sigma) = \{t_{|\omega} \downarrow_{(E_i,>)} \mid \omega \in AlienPos(t)\}$. □

Proposition 3.1 *Les systèmes de réécriture $E_1^{>}$, $E_2^{>}$ et $E_1^{>} \cup E_2^{>}$ sont convergents.*

Preuve : Le système de réécriture $E_i^{>}$ est convergent car $\xleftrightarrow{*}_{E_i^{>}} = \xleftrightarrow{*}_{(E_i,>)}$ d'après le lemme précédent.

Les superpositions entre règles de $E_1^{>}$ et $E_2^{>}$ sont équivalentes à des superpositions à position variable car les têtes des règles de $E_1^{>}$ et $E_2^{>}$ sont distinctes. □

On note R le système de réécriture inclus dans $E_1^{>} \cup E_2^{>}$ et défini par:

$$R = \bigcup_{i=1}^{2} \{l\sigma \to r\sigma \mid l =_{E_i} r,\ \mathcal{R}an(\sigma) \subseteq \mathcal{T}(\mathcal{F} \cup \mathcal{X}),\ l\sigma > r\sigma,\ l(\epsilon) \in \mathcal{F}_i \backslash \mathcal{SF},$$

$$x\sigma \text{ est } E_1^{>} \cup E_2^{>}\text{-normalisée pour } x \in \mathcal{V}(r) \backslash \mathcal{V}(l)\}.$$

Corollaire 3.2 $s =_{E_1 \cup E_2} t \iff s \downarrow_R = s \downarrow_{E_1^{>} \cup E_2^{>}} = t \downarrow_{E_1^{>} \cup E_2^{>}} = t \downarrow_R$.

Les symboles partagés sont ω-libres dans E.

Proposition 3.2 *Les symboles partagés sont des constructeurs pour R:*

$$\forall f \in \mathcal{SF},\ (f(s_1, \ldots, s_p)) \downarrow_R = f(s_1 \downarrow_R, \ldots, s_p \downarrow_R).$$

3.1.2 Abstraction

L'abstraction par variables n'a pas à être définie pour les termes en forme normale dont la tête est partagée, qui ne seront jamais abstraits. Il en résulte une définition de la i-abstraction un peu moins opérationnelle puisqu'elle consiste soit à normaliser les sous-termes apparemment étrangers, soit à abstraire par variables des sous-termes étrangers qui le restent par normalisation.

Définition 3.2 *Une abstraction par variables π est une bijection de*

$$\{u\downarrow_R \mid u \in T(\mathcal{F}\cup\mathcal{X}) \text{ et } u\downarrow_R(\epsilon) \in \mathcal{F}\backslash\mathcal{SF}\}$$

vers un ensemble de variables de $\mathcal{X}$.

La i-abstraction d'un terme t est le terme t^{π_i} défini inductivement comme suit:
- *Si $t = x \in \mathcal{X}$ alors $t^{\pi_i} = x$,*
- *Si $t = f(s_1,\ldots,s_p)$ et $f \in \mathcal{F}_i$ alors $t^{\pi_i} = f(s_1^{\pi_i},\ldots,s_p^{\pi_i})$*

sinon si $t\downarrow_R(\epsilon) \notin \mathcal{F}_i \cup \mathcal{X}$ alors $t^{\pi_i} = \pi(t\downarrow_R)$ sinon $t^{\pi_i} = t\downarrow_R$.

La i-abstraction d'un terme t est un terme i-pur si les sous-termes étrangers de t sont R-normalisés.

Remarque 3.1 *Si s est un terme i-pur et σ une substitution R-normalisée alors $(s\sigma)^{\pi_i} = s\sigma^{\pi_i}$ est i-pur.*

Cette définition de l'abstraction est nettement plus adaptée à l'unification, où l'on abstrait une substitution R-normalisée, qu'au problème du mot, où l'abstraction ne signifie plus seulement remplacer un sous-terme étranger par une variable.

Un terme ayant un symbole partagé en tête a intuitivement un ensemble de sous-termes étrangers différent suivant que le symbole de tête est vu dans $\mathcal{F}_1$ ou $\mathcal{F}_2$. Cette notion doit donc être redéfinie par rapport à une théorie E_i.

Définition 3.3 *L'ensemble des* positions i-étrangères *d'un terme t est*

$$AlienPos_i(t) = \{\omega \neq \epsilon \mid t(\omega) \notin \mathcal{F}_i \cup \mathcal{X} \text{ et } \forall\omega' \in \mathcal{P}os(t),\ \omega' < \omega \Rightarrow t(\omega') \in \mathcal{F}_i\}.$$

L'ensemble des sous-termes i-étrangers *de t est $AST_i(t) = \{t_{|\omega} \mid \omega \in AlienPos_i(t)\}$. L'ensemble des positions étrangères de t est $AlienPos(t) = AlienPos_1(t) \cup AlienPos_2(t)$. L'ensemble des sous-termes étrangers de t est $AST(t) = AST_1(t) \cup AST_2(t)$.*

On remarquera que $AST(t) = AST_i(t)$ si $t(\epsilon) \in \mathcal{F}_i\backslash\mathcal{SF}$.

Exemple 3.1 *Soit $t = f(a, a+a)$ un terme tel que $\mathcal{F}_1 = \{f,+\}, \mathcal{F}_2 = \{f,a\}$ et $E_1 = E_2 = \emptyset$. Les abstractions de t sont $t^{\pi_1} = f(\pi(a),\pi(a)+\pi(a))$ et $t^{\pi_2} = f(a,\pi(a+a))$ alors que l'ensemble des sous-termes étrangers est $AST(t) = \{a\} \cup \{a+a\}$*

3.1.3 Résolution dans une composante

Nous allons voir que la preuve de complétude de la résolution dans une composante est similaire à celle vue pour le cas du mélange de théories disjointes. On suit la même progression que pour le cas disjoint, en montrant d'abord sous quelle condition les i-abstractions d'un terme et de sa forme normale sont E_i-égales.

Proposition 3.3 *Si s est un i-terme dont les sous-termes i-étrangers sont R-normalisés alors s^{π_i} est un terme i-pur tel que $s^{\pi_i} =_{E_i} (s \downarrow_R)^{\pi_i}$.*

Preuve : Si $s \rightarrow_{R,\omega} t$ alors $s(\omega) \in \mathcal{F}_i$ puisque les sous-termes i-étrangers sont R-normalisés et plus précisément $s(\omega) \in \mathcal{F}_i \backslash \mathcal{SF}$ car un symbole partagé ne peut apparaitre en tête des membres gauches des règles de R. Le terme s se réécrit donc en t par une règle de $E_i^>$ pour laquelle les sous-termes étrangers figurent dans la partie instance, ce qui implique $s^{\pi_i} =_{E_i} t^{\pi_i}$. Par définition de R, cette règle n'introduit pas de nouveaux sous-termes étrangers réductibles. De plus, lorsque t n'est pas un i-terme, il s'agit soit d'une variable, soit d'un sous-terme étranger de s, c'est-à-dire que t est R-normalisé.

Il suffit ensuite d'une preuve par induction sur $\rightarrow_R$ pour prouver la proposition. □

Corollaire 3.3 *Si s est un terme i-pur et σ une substitution R-normalisée alors*

$$(s\sigma)^{\pi_i} =_{E_i} (s\sigma \downarrow_R)^{\pi_i}.$$

Preuve : Si s n'est pas un i-terme mais une variable, alors $s\sigma = s\sigma \downarrow_R$ et

$$(s\sigma)^{\pi_i} = (s\sigma \downarrow_R)^{\pi_i}.$$

□

Lemme 3.2 *Si s, t sont des termes i-purs et σ une substitution R-normalisée alors*

$$s\sigma =_E t\sigma \Leftrightarrow s\sigma^{\pi_i} =_{E_i} t\sigma^{\pi_i}.$$

Preuve : $s\sigma =_{E_1 \cup E_2} t\sigma \Leftrightarrow s\sigma^{\pi_i} = (s\sigma)^{\pi_i} =_{E_i} (s\sigma \downarrow_R)^{\pi_i} = (t\sigma \downarrow_R)^{\pi_i} =_{E_i} (t\sigma)^{\pi_i} = t\sigma^{\pi_i}$. □

3.1.4 Combinaison des solutions

Les hypothèse faites jusqu'à présent ont servi à montrer comment résoudre un problème pur dans sa composante. Mais nous n'avons résolu que la moitié du problème puisque il s'agit de résoudre une conjonction de problèmes respectivement 1-pur et 2-pur. Une première solution serait de s'intéresser aux cas de conflits inter-théories qui font qu'une conjonction de formes résolues pures n'est pas une forme résolue. Une seconde solution serait de ne pas introduire *a priori* de conflits inter-théories en s'imposant une restriction sur les solutions recherchées. Ces deux approches seront étudiées successivement. Elles nécessitent en général toutes deux d'autres hypothèses que la décomposabilité d'une théorie. Dans le cas de théories disjointes, nous avons eu recours à l'unification avec restriction linéaire. Mais cette notion de restriction linéaire ne peut être directement utilisée dans le cas non disjoint puisqu'un terme (R-normalisé) à tête partagée n'est pas forcément abstrait par une variable dans au moins l'une des théories composantes. Les hypothèses que nous serons amenés à faire permettent de gérer au mieux cette situation délicate qui empêche une combinaison des solutions comme dans le cas disjoint.

3.2 Approche par résolution

Nous commençons par l'étude d'un cas simple où l'unification élémentaire est suffisante parce que les conflits inter-théories n'ont pas de solution. Un cas plus général sera ensuite considéré.

3.2.1 Le cas régulier et non effondrant

L'algorithme de combinaison déterministe que nous présentons maintenant généralise celui vu pour le mélange des théories régulières non effondrantes mais disjointes.

Théorie faiblement partagée

Définition 3.4 *Une théorie $(\mathcal{F}, E)$ est* faiblement partagée *par rapport à un ensemble de symboles de fonctions $\mathcal{SF} \subseteq \mathcal{F}$ si*

- *les symboles de $\mathcal{SF}$ sont ω-libres,*
- $\forall s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X}),\ \forall f \in \mathcal{SF}$,

$$s = f(s_1, \ldots, s_p) \neq_E t \quad si \quad t(\epsilon) \neq f$$

Proposition 3.4 *L'union de deux théories $(\mathcal{F}_1, E_1)$ et $(\mathcal{F}_2, E_2)$ faiblement partagées par rapport à un même ensemble de symboles de fonctions $\mathcal{SF} = \mathcal{F}_1 \cap \mathcal{F}_2$ est une théorie décomposable faiblement partagée par rapport à $\mathcal{SF}$.*

Preuve : L'ordre de simplification vu dans le cas disjoint peut être réutilisé car il n'y a pas à orienter des égalités dont l'un des membres a un symbole de tête partagé et pas l'autre. L'hypothèse de ω-liberté et l'absence d'autre égalité ayant un symbole partagé en tête suffit pour que les symboles de $\mathcal{SF}$ soient des constructeurs pour R. □

Une théorie faiblement partagée ne peut pas être effondrante. L'hypothèse supplémentaire de régularité permet de résoudre le plus simplement possible les conflits inter-théories comme dans le cas disjoint.

Conflit de théories

On utilise le lemme 3.2 comme dans le cas disjoint, ainsi que le fait qu'il n'y a pas d'égalité mixte $l =_E r$ avec $l(\epsilon) \in \mathcal{SF}$ et $r(\epsilon) \notin \mathcal{SF}$.

Proposition 3.5 *Si E_1 et E_2 sont des théories faiblement partagées, alors il n'existe pas de solution au problème $P = (x =^? t_1 \wedge x =^? t_2)$ tel que*

- $t_1 \in T(\mathcal{F}_1, \mathcal{X}) \backslash \mathcal{X}$,
- $t_2 \in T(\mathcal{F}_2, \mathcal{X}) \backslash \mathcal{X}$,
- $t_1(\epsilon) \notin \mathcal{SF}$ *ou* $t_2(\epsilon) \notin \mathcal{SF}$ *ou* $t_1(\epsilon), t_2(\epsilon) \in \mathcal{SF}, t_1(\epsilon) \neq t_2(\epsilon)$.

Preuve : Par l'absurde. S'il existe une solution au problème P, alors E_1 ou E_2 n'est pas faiblement partagée, parce qu'il existe forcément une égalité ayant comme membre une variable ou une égalité dont l'un des membres est à tête partagée et l'autre pas, ou encore une égalité dont les membres n'ont pas les mêmes symboles partagés au sommet. □

Résolution de cycle

Il suffit, comme nous l'avions annoncé dans le cas disjoint, de s'intéresser aux cycles composés. Mais la notion de cycle composé doit être légèrement adaptée lorsque des symboles sont partagés pour éviter de le confondre avec un cycle pouvant être résolu uniquement dans une théorie.

Définition 3.5 *Un* cycle composé *est un cycle*

$$y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1]$$

tel que

- $t_1, \ldots, t_n \in T(\mathcal{F}_1, \mathcal{X}) \backslash \mathcal{X}$,
- $s_1, \ldots, s_n \in T(\mathcal{F}_2, \mathcal{X}) \backslash \mathcal{X}$,
- $x_1, \ldots, x_n, y_1, \ldots, y_n$ *sont des variables distinctes.*
- $\exists k \in \{1, \ldots, n\},\ t_k \notin T(\mathcal{SF}, \mathcal{X}) \backslash \mathcal{X}$,
- $\exists k \in \{1, \ldots, n\},\ s_k \notin T(\mathcal{SF}, \mathcal{X}) \backslash \mathcal{X}$,

Un cycle qui n'est pas composé est soit 1-pur, soit 2-pur.

Proposition 3.6 *Si E_1 et E_2 sont des théories faiblement partagées, régulières et non effondrantes, alors il n'existe pas de solution à un cycle composé.*

Preuve : On peut toujours réarranger un cycle composé par généralisation d'un contexte partagé et sa propagation dans l'autre théorie pour que

$$P = (y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1])$$

vérifie $t_1(\epsilon), \ldots, t_n(\epsilon) \in \mathcal{F}_1 \backslash \mathcal{SF}$ et $s_1(\epsilon), \ldots, s_n(\epsilon) \in \mathcal{F}_2 \backslash \mathcal{SF}$. Si σ est une solution R-normalisée de P alors

$$\forall m \in \{1, \ldots, n\},\ x_m\sigma(\epsilon) \in \mathcal{F}_2 \backslash \mathcal{SF},$$

$$\forall m \in \{1, \ldots, n\},\ y_m\sigma(\epsilon) \in \mathcal{F}_1 \backslash \mathcal{SF},$$

sinon il existe une égalité ayant comme membre une variable ou une égalité dont l'un des membres a une tête partagée et pas l'autre; et

$$\forall m \in \{1, \ldots, n\},\ x_m\sigma^{\pi_1} \in \mathcal{V}(y_m\sigma),$$

$$\forall m \in \{2, \ldots, n\},\ y_m\sigma^{\pi_2} \in \mathcal{V}(x_{m-1}\sigma),$$

$$y_1\sigma^{\pi_2} \in \mathcal{V}(x_n\sigma),$$

sinon il existe une égalité non régulière. Dans ce cas, $y_1\sigma$ est sous-terme strict de $x_n\sigma$ et $x_n\sigma$ est sous-terme strict de $y_1\sigma$, ce qui est absurde. □

VA

$$\frac{P \wedge s =^? t}{\exists x : P \wedge s =^? t[x]_\omega \wedge x =^? t_{|\omega}} \quad \textbf{si} \begin{cases} \omega \in AlienPos(t) \\ x \text{ est une nouvelle variable} \end{cases}$$

IE

$$\frac{P \wedge s =^? t}{\exists x : P \wedge x =^? s \wedge x =^? t}$$

$$\textbf{si} \begin{cases} s \in T(\mathcal{F}_1, \mathcal{X})\backslash\mathcal{X},\ t \in T(\mathcal{F}_2, \mathcal{X})\backslash\mathcal{X},\ s \notin T(\mathcal{SF}, \mathcal{X}) \text{ ou } t \notin T(\mathcal{SF}, \mathcal{X}) \\ x \text{ est une nouvelle variable} \end{cases}$$

$E_i - \mathsf{Res}^+$

$$\frac{P_i}{\exists \vec{x_i} : \hat{\sigma}_i} \quad \textbf{si } \sigma_i \in CSU_{E_i}(P_i) \text{ et } \vec{x_i} = \mathcal{V}(\hat{\sigma}_i)\backslash\mathcal{V}(P_i)$$

VarRep

$$\frac{P \wedge x =^? y}{P\{x \mapsto y\} \wedge x =^? y} \quad \textbf{si } x, y \in \mathcal{V}(P)$$

Conflit

$$\frac{P \wedge x =^? s \wedge x =^? t}{\bot} \quad \textbf{si} \begin{cases} s \in T(\mathcal{F}_1, \mathcal{X})\backslash\mathcal{X}, \\ t \in T(\mathcal{F}_2, \mathcal{X})\backslash\mathcal{X}, \\ s(\epsilon) \notin \mathcal{SF} \text{ ou } t(\epsilon) \notin \mathcal{SF} \text{ ou } s(\epsilon), t(\epsilon) \in \mathcal{SF}, s(\epsilon) \neq t(\epsilon) \end{cases}$$

Dec

$$\frac{P \wedge x =^? f(s_1, \ldots, s_p) \wedge x =^? f(t_1, \ldots, t_p)}{P \wedge x =^? f(s_1, \ldots, s_p) \wedge \bigwedge_{k=1}^{p} s_k =^? t_k} \quad \textbf{si} \text{ aucune autre règle ne s'applique}$$

Cycle

$$\frac{P \wedge y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1]}{\bot}$$

si $y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1]$ est un cycle composé.

Figure 3.1. Ensemble de règles $\mathcal{RAS}$ pour l'unification dans le mélange de théories faiblement partagées, régulières et non effondrantes

Algorithme

La structure de données est rigoureusement identique à celle prise pour les théories disjointes, un problème P se décomposant en $P_H \wedge P_V \wedge P_1 \wedge P_2$. La différence majeure est que les problèmes P_1 et P_2 ont en commun les équations sur $T(\mathcal{SF}, \mathcal{X})$.

La preuve de terminaison de l'ensemble $\mathcal{RAS}$ des règles données en figure 3.1 est similaire à celle du cas disjoint. Il faut toutefois tenir compte des variables qui apparaissent résolues par un terme dont la tête est partagée.

Définition 3.6 *Une variable est $\mathcal{SF}$-résolue dans P si $P_H \wedge P_1 \wedge P_2 = (Q \wedge x =^? t)$ avec $x \notin \mathcal{V}(Q) \cup \mathcal{V}(t), t(\epsilon) \in \mathcal{SF}$. Une classe de variables est $\mathcal{SF}$-résolue s'il en existe un représentant $\mathcal{SF}$-résolu dans le problème obtenu en appliquant jusqu'à saturation la règle* VarRep.

On suppose dorénavant que si P est de la forme $P = (Q \wedge x =^? t)$ avec $t \in \mathcal{T}(\mathcal{SF}, \mathcal{X})\backslash\mathcal{X}$,

alors x est une variable $\mathcal{SF}$-résolue c'est-à-dire que le remplacement de x par t dans $P_1 \wedge P_2 \wedge P_H$ est implicitement effectué au retour de chaque règle de transformation.

Autre mesure de complexité: $NVR_{\mathcal{SF}}(P)$ est le nombre de classes de variables de $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$ non $\mathcal{SF}$-résolues.

La règle Dec s'applique à une variable à la fois $\mathcal{SF}$-résolue dans P_1 et $\mathcal{SF}$-résolue dans P_2 tandis que P_H est vide: elle fait donc décroître $NVR_{\mathcal{SF}}(P)$.

Proposition 3.7 *L'application des règles $\mathcal{RAS}$ à un problème Γ termine pour tout contrôle et retourne une disjonction équivalente de problèmes d'unification $\exists\vec{x} : P$ où les P sont en forme séquentiellement résolue.*

Preuve : La preuve est résumée par le tableau suivant:

regles	$TH_{mul}(P)$	$IE(P)$	$NVR_{\mathcal{SF}}(P)$	$NVC(P)$	$PVR(P)$	$USP(P)$
VA	$\downarrow$					
IE	$=$	$\downarrow$				
Dec	$=$	$=$	$\downarrow$			
E_i-Res(0)	$=$	$=$	$\downarrow$			
E_i-Res(1)	$=$	$=$	$=\downarrow$	$\downarrow$		
E_i-Res(2)	$=$	$=$	$=\downarrow$	$=$	$=$	$\downarrow$
VarRep	$=$	$=$	$=$	$=$	$\downarrow$	

La règle E_i-Res a plusieurs comportements. Elle engendre une équation $x=^?t$, $x \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, $t \in \mathcal{T}(\mathcal{SF}, \mathcal{X})\backslash\mathcal{X}$ et $NVR_{\mathcal{SF}}(P)$ décroît, ou sinon elle engendre une équation $x=^?y$ entre variables et $NVC(P)$ décroît, ou sinon elle rend P_i résolu tout en préservant l'état de P_j, $j \neq i$, c'est-à-dire que $USP(P)$ décroît.

Les règles Conflit et Cycle ne sont pas reproduites ici car elles mènent directement à des formes normales. □

Théorème 3.1 *Soit $E_1 \cup E_2$ une théorie décomposable en deux théories E_1 et E_2 faiblement partagées, régulières et non effondrantes. La $E_1 \cup E_2$-unification est finitaire si la E_i-unification ($i = 1, 2$) est finitaire.*

3.2.2 Généralisation

La principale difficulté pour la combinaison des solutions est qu'une solution dans l'union $E_1 \cup E_2$ peut instancier une variable dans les deux théories, E_1 et E_2. Il n'est plus possible de choisir une théorie pour une variable et de s'interdire d'instancier cette variable dans l'autre théorie.

L'idée que nous allons développer consiste à deviner les contextes partagés pouvant apparaitre en tête de certaines formes normales, celles représentant des classes de termes "mixtes".

Définition 3.7 *Un contexte* partagé *est un terme de $T(\mathcal{SF} \cup \{[]\})$.*

Définition 3.8 *Un théorie E décomposable est* bornée *s'il existe un ensemble* **fini** *de contextes partagés $\mathcal{SC}$ tel que*

$$\{C[t_1, \ldots, t_q] \mid C \in \mathcal{SC}, t_1(\epsilon), \ldots, t_q(\epsilon) \in \mathcal{F}\backslash\mathcal{SF}\} = \{t\downarrow_R \mid t\downarrow_R(\epsilon) \in \mathcal{SF} \text{ et } t(\epsilon) \in \mathcal{F}\backslash\mathcal{SF}\}$$

On suppose disposer d'un oracle permettant de déterminer $\mathcal{SC}$ lorsque une théorie décomposable est bornée.

Exemple 3.2 *Considérons la théorie $E = E_1 \cup E_2$ avec $E_i = \{f(h_i(x)) = g_i(f(x))\}$ pour $i = 1, 2$ avec le symbole f partagé par E_1 et E_2. Le symbole f est ω-libre et la précédence $g_2 > g_1 > f > h_2 > h_1$ permet d'orienter les égalités afin de ne pas faire apparaître le symbole f en tête des membres gauches de règles. La théorie E est donc décomposable et admet comme borne $\mathcal{SC} = \{f([])\}$.*

La théorie $E = E_1 \cup E_2$ avec $E_i = \{f(x) = g_i(f(x))\}$ est encore décomposable mais elle n'est pas bornée puisqu'on peut instancier la variable x par un terme à symboles partagés arbitrairement grand. Nous verrons ultérieurement comment traiter cette théorie en suivant une approche par décision.

La théorie $E = E_1 \cup E_2$ avec $E_i = \{c = g_i(f(x))\}$ est cette fois bornée par une constante. La notion de théorie décomposable bornée permet donc de capturer une forme de mélange dont les constantes ne sont pas les seuls symboles partagés (f l'est aussi dans cet exemple).

Cette définition autorise donc certaines égalités entre deux termes dont l'un a une tête partagée et l'autre pas. Elle empêche cependant les théories effondrantes, sauf à une condition.

Proposition 3.8 *Si une théorie décomposable bornée est effondrante alors les symboles partagés sont des constantes.*

Preuve : Soit E une théorie effondrante ayant un symbole partagé f non constant, par exemple unaire. Il existe une égalité $l[x] =_E x$ et par conséquent $l[f^n(x)] = f^n(x)$ où n est un entier arbitrairement grand. Le terme l a forcément en tête un symbole non partagé, sinon E n'est pas décomposable. □

Le partage des constantes n'a donc pas de conséquence sur la forme des théories qu'il est possible de mélanger. Mais nous n'irons pas jusqu'à dire que c'est le seul cas intéressant lorsque les signatures sont non disjointes.

L'ensemble des substitutions d'un ensemble de variables V par des contextes partagés est construit à l'aide d'un ensemble de variables W de taille $|W| = \sum_{C \in \mathcal{SC}} |\mathcal{CP}os(C)|$ et que l'on supposera disjoint de l'ensemble des variables $\mathcal{V}(P)$ du problème P à résoudre. Soient $\mathcal{SC}[W] = \{C[w_1, \ldots, w_q] \in \mathcal{T}(\mathcal{SF}, W) \mid C \in \mathcal{SC}\}$ et

$$SUBS_V^{\mathcal{SC}} = \{\rho \mid \mathcal{D}om(\rho) \subseteq V, \mathcal{R}an(\rho) \subseteq \mathcal{SC}[W]\}.$$

Ces substitutions sont appliquées lorsqu'apparaissent des conflits de théories.

3.2.3 Algorithme

L'algorithme a la même caractéristique que celui proposé par A. Boudet dans le cas disjoint: résoudre les conflits inter-théories au fur et à mesure de leurs apparitions par la construction progressive d'une restriction linéaire. On reprend exactement la même terminologie et la même structure de données que celles utilisées jusqu'à présent. Nous allons

cependant introduire de nouvelles notations pour simplifier l'écriture et se restreindre à un problème avec restriction linéaire.

Etant donné un ordre linéaire $<$ sur $V_1 \oplus V_2$, le problème

$$\mathsf{Linear}(V_1, V_2, <) = \left\{ \begin{array}{ll} & \bigwedge_{v_2 \in V_2} M_{E_1}(v_2) \wedge \bigwedge_{v_1 \in V_1} M_{E_2}(v_1) \\ \wedge & \bigwedge_{v_1 \in V_1 < v_2 \in V_2} Arc_{E_1}(v_2, v_1) \wedge \bigwedge_{v_2 \in V_2 < v_1 \in V_1} Arc_{E_2}(v_1, v_2) \end{array} \right.$$

exprime la restriction linéaire à l'aide de marques et d'arcs.

Etant donnés une variable x, un terme t et $\rho \in SUBS_V^{\mathcal{SC}}$, le problème d'unification

$$\mathsf{Inst}(x, t, \rho) = \exists \mathcal{V}(x\rho) : (x\rho =^? t \wedge x =^? x\rho \wedge \bigwedge_{w \in \mathcal{V}(x\rho)} (M_{E_1}(w) \vee M_{E_2}(w)))$$

va remplacer le problème $x =^? t$ lorsque le terme t a une tête non partagée.

Finalement, on note en abrégé $M_{E_i}(W)$ le problème $\bigwedge_{w \in W} M_{E_i}(w)$ composé de marques dans E_i.

On s'intéresse uniquement à résoudre un problème séparé, c'est-à-dire une conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes purs, en le transformant par un système de règles $\mathcal{RND}$ jusqu'à obtenir une forme normale. Les mesures de complexité utilisées sont fonctions des variables de ce problème initial.

Le système $\mathcal{RND} = \{\mathsf{VarRep}, \mathsf{Conflits}, \mathsf{DecRep}, \mathsf{Res}\}$ est composé de seulement quatre règles:

- VarRep et DecRep gèrent l'interaction entre les deux théories. La règle DecRep a pour but de propager dans l'autre théorie un contexte partagé généralisé par l'introduction de nouvelles variables. Cette règle permet en particulier de réarranger un cycle composé de sorte que les termes qui le forment aient tous des têtes non partagées.
- Conflits permet de choisir une théorie ou un contexte lorsque une variable apparait résolue par un terme ayant une tête non partagée.
- Res consiste à appliquer, sous les hypothèses 2.1, les algorithmes de E_1-unification puis E_2-unification suivant une même restriction linéaire.

Correction

La règle essentielle du système $\mathcal{RND}$ est Res, qui fait appel aux algorithmes de E_i-unification ($i = 1, 2$) suivant la même restriction étendue aux variables déjà marquées mais pas encore intégrées à l'ordre linéaire. La preuve que cette règle préserve l'ensemble des solutions est basée sur la proposition suivante.

Proposition 3.9 *Soit σ une substitution R-normalisée dont les termes de $\mathcal{R}an(\sigma)$ sont des termes non-variables et distincts deux à deux. Si σ est une solution de*

$$P_1 \wedge P_2 \wedge M_{E_1}(W_2) \wedge M_{E_2}(W_1) \wedge \mathsf{Linear}(V_1, V_2, <)$$

avec $W_1 \cap V_1 = W_2 \cap V_2 = W_1 \cap W_2 = \emptyset$, alors σ est solution d'un problème

$$\hat{\sigma}_1 \wedge \hat{\sigma}_2 \wedge \mathsf{Linear}(V_1 \cup W_1, V_2 \cup W_2, \ll)$$

où $\ll$ est un ordre linéaire sur $(V_1 \cup W_1) \oplus (V_2 \cup W_2)$ tel que $\ll \supseteq <$ et $\sigma_1 \in CSU_{E_1}^{\ll}(P_1, V_2 \cup W_2), \sigma_2 \in CSU_{E_2}^{\ll}(P_2, V_1 \cup W_1)$.

Preuve : Si σ est une solution R-normalisée alors nécessairement

$$\forall x \in V_i \cup W_i, x\sigma(\epsilon) \in \mathcal{F}_i \backslash \mathcal{SF},$$

sinon $\sigma^{\pi_1} \notin SU_{E_1}(P_1, V_2 \cup W_2)$ ou $\sigma^{\pi_2} \notin SU_{E_2}(P_2, V_1 \cup W_1)$. L'ordre linéaire $\ll$ tel que $x \ll y$ si $x\sigma$ est sous-terme de $y\sigma$ contient $<$, sinon $\sigma^{\pi_1} \notin SU^{<}_{E_1}(P_1, V_2)$ ou $\sigma^{\pi_2} \notin SU^{<}_{E_2}(P_2, V_1)$. On vérifie ensuite comme d'habitude par contraposée que les solutions σ^{π_1} et σ^{π_2} satisfont la même restriction linéaire $\ll$. □

Les problèmes sont à identifier au préalable afin de tenir compte de toutes les solutions R-normalisées.

Les variables marquées le sont par la règle Conflits qui consiste à fixer une théorie ou un contexte partagé de $\mathcal{SC}$.

Proposition 3.10 *La règle* Conflits *préserve l'ensemble des solutions.*

Preuve : Soit σ une solution R-normalisée de $P \wedge x =^? t$ avec $t(\epsilon) \notin \mathcal{SF}$.

Si $x\sigma(\epsilon) \in \mathcal{F}_i \backslash \mathcal{SF}$ alors $x\sigma^{\pi_j} \in \mathcal{X}$ pour $j \neq i$.

Si $x\sigma \in \mathcal{X}$ alors $x\sigma^{\pi_1} = x\sigma^{\pi_2} \in \mathcal{X}$.

Si $x\sigma(\epsilon) \in \mathcal{SF}$ alors il existe $C \in \mathcal{SC}$ tel que $x\sigma = C[t_1, \ldots, t_q]$ avec $t_1(\epsilon), \ldots, t_q(\epsilon) \in \mathcal{F} \backslash \mathcal{SF}$ puisque $E_1 \cup E_2$ est bornée. Par conséquent, il existe $\rho \in SUBS^{\mathcal{SC}}_{\{x\}}$ et une substitution ϕ identique à σ sauf pour $\mathcal{VR}an(\rho)$ telles que $x\sigma = x\rho\phi =_{E_1 \cup E_2} t\sigma = t\phi = x\phi$. Plus précisément, $\forall w_k \in \mathcal{VR}an(\rho),\ w_k\phi = t_k$ avec $w_k\phi^{\pi_1} \in \mathcal{X}$ ou $w_k\phi^{\pi_2} \in \mathcal{X}$. □

Une équation $x =^? t$ avec $t(\epsilon) \in \mathcal{SF}$ peut elle aussi créer un conflit de théories ou un cycle composé. La règle de remplacement DecRep permet de propager le contexte partagé en tête de t en généralisant les sous-termes à têtes non partagées par des nouvelles variables. Le problème obtenu reste un problème séparé. La preuve de correction de cette règle ne pose pas de difficulté.

Complétude

Les variables qui vont figurer dans un problème Inst sont uniquement celles qui engendrent des conflits inter-théories.

Définition 3.9 *Une variable est* conflictuelle *pour un problème séparé si elle apparait résolue dans les deux théories ou si elle est membre d'un cycle composé. Une classe de variables est* conflictuelle *s'il existe une variable de la classe qui est conflictuelle pour le problème obtenu en appliquant jusqu'à saturation la règle* VarRep*.*

Une conjonction de formes résolues pures est en forme séquentiellement résolue si et seulement si elle ne contient pas de classe de variables conflictuelle.

Le système de règles $\mathcal{RND}$ a été conçu pour qu'un problème séparé soit en forme normale si et seulement si c'est une forme séquentiellement résolue.

Proposition 3.11 *Un problème séparé est en forme séquentiellement résolue si et seulement si il est irréductible par le système de règles $\mathcal{RND}$.*

Preuve : ($\Leftarrow$) Si un problème séparé contient une forme non résolue alors Res s'applique.

Si c'est une conjonction de formes résolues sans être une forme séquentiellement résolue, alors il existe une classe de variables conflictuelle et VarRep s'applique s'il n'y a pas de variable conflictuelle ou sinon Conflits ou DecRep s'applique, mais uniquement lorsque cette variable n'est pas déjà marquée. Dans le cas contraire, il existe nécessairement une variable marquée n'apparaissant pas dans la restriction linéaire et Res s'applique.

($\Rightarrow$) Si un problème séparé est en forme séquentiellement résolue alors VarRep ne s'applique pas et il n'existe pas de variable conflictuelle: ni Conflits ni DecRep ne peut s'appliquer. Il en est de même pour Res car les problèmes purs sont résolus et toutes les variables marquées figurent dans la restriction linéaire. □

Terminaison

La preuve de terminaison repose sur le principe qu'un conflit de théories ou un cycle composé est relatif à au moins une classe de variables du problème initial. Celle-ci est utilisée dans la construction d'une restriction linéaire et ce choix ne sera jamais remis en cause. Il faut cependant se méfier des nouvelles variables introduites par DecRep qui pourraient devenir conflictuelles si elles n'étaient toutes directement ajoutées à la restriction linéaire.

Proposition 3.12 *Si une conjonction de formes résolues n'est pas séquentiellement résolue, alors il existe une classe de variables conflictuelle incluse dans* $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$ *où* $=_V$ *est la relation d'équivalence engendrée par* P_V*.*

Preuve : Les variables introduites par DecRep peuvent être conflictuelles si elles sont marquées au lieu d'être $\mathcal{SF}$-résolues. Dans ce cas, elles seront intégrées à la restriction linéaire à la prochaine application de Res. Il ne peut plus y avoir conflit de théories pour ces variables après application de Res. D'autres variables sont également introduites par E_1-unification d'une part et E_2-unification d'autre part mais sont supposées disjointes et ne sont donc pas conflictuelles. Par conséquent, les seules classes de variables conflictuelles non incluses dans $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$ figurent dans la restriction linéaire après Res. Si le nouveau problème contient encore un cycle composé, celui-ci concerne nécessairement une classe de variables ne figurant pas dans la restriction linéaire, c'est-à-dire une classe dans $\mathcal{V}(\Gamma_1 \wedge \Gamma_2)/=_V$. □

La règle Conflits fait décroître $NM(P)$ ou $NVR_{\mathcal{SF}}(P)$ alors que DecRep fait décroître $NVR_{\mathcal{SF}}(P)$.

La règle Res ne fait pas croître $NM(P)$ et $NVR_{\mathcal{SF}}(P)$ mais fait décroître des mesures de complexité déjà utilisées pour le mélange de théories disjointes.

Lemme 3.3 *La règle* Res *fait décroître* $NA(P)$, $NVR_{\mathcal{SF}}(P)$, $NVC(P)$, $NM(P)$ *ou* $USP(P)$.

Preuve : Si le problème est une conjonction de formes résolues alors les marques contiennent au moins une classe de variables conflictuelle non encore incluse dans la restriction linéaire et $NA(P)$ décroît. Sinon, le problème n'est pas une conjonction de

VarRep

$$\frac{P \wedge x =^? y}{P\{x \mapsto y\} \wedge x =^? y} \quad \textbf{si } x, y \in \mathcal{V}(P)$$

Conflits

$$\frac{P \wedge x =^? t}{\bigvee_{\rho \in SUBS^{SC}_{\{x\}}} P\rho \wedge \mathsf{Inst}(x, t, \rho)}$$

si

- VarRep ne s'applique pas,
- $t(\epsilon) \in \mathcal{F} \backslash \mathcal{SF}$,
- x est une variable conflictuelle non marquée.

DecRep

$$\frac{P \wedge x =^? C[t_1, \ldots, t_q]}{\bigvee_{\rho \in SUBS^{SC}_{\vec{v}}} \exists \vec{v} : P\{x \mapsto C[v_1, \ldots, v_q]\rho\} \wedge x =^? C[v_1, \ldots, v_q]\rho \wedge \bigwedge_{k=1}^{q} \mathsf{Inst}(v_k, t_k, \rho)}$$

si

- VarRep ne s'applique pas,
- $C \in \mathcal{T}(\mathcal{SF}, \mathcal{X})$,
- $t_1(\epsilon), \ldots, t_q(\epsilon) \in \mathcal{F} \backslash \mathcal{SF}$,
- $\vec{v} = \{v_1, \ldots, v_q\}$ est un ensemble de nouvelles variables,
- x est une variable conflictuelle.

Res$^+$

$$\frac{(P_1 \wedge M_{E_1}(W_2)) \wedge (P_2 \wedge M_{E_2}(W_1)) \wedge \mathsf{Linear}(V_1, V_2, <)}{\hat{\xi}_1 \wedge \hat{\xi}_2 \wedge (\exists \vec{x_1} : \hat{\sigma}_1) \wedge (\exists \vec{x_2} : \hat{\sigma}_2) \wedge \mathsf{Linear}(V_1 \cup W_1\xi_1, V_2 \cup W_2\xi_2, \ll)}$$

si

- VarRep ne s'applique pas,
- $(P_1 \wedge M_{E_1}(W_2))$ ou $(P_2 \wedge M_{E_2}(W_1))$ n'est pas résolu, ou $W_1 \neq \emptyset$ ou $W_2 \neq \emptyset$.
- $W_i \cap V_i = \emptyset$ pour $i = 1, 2$,
- $\xi_i \in ID_{W_i} \circ ID^{V_i}_{W_i}$ pour $i = 1, 2$,
- $\ll$ est un ordre linéaire sur $(V_1 \cup W_1\xi_1) \oplus (V_2 \cup W_2\xi_2)$ contenant $<$,
- $\sigma_1 \in CSU^{\ll}_{E_1}(P_1\xi_2 \wedge \hat{\xi}_1, V_2 \cup W_2\xi_2), \sigma_2 \in CSU^{\ll}_{E_2}(P_2\xi_1 \wedge \hat{\xi}_2, V_1 \cup W_1\xi_1)$,
- $\vec{x_1} = \mathcal{V}(\hat{\sigma}_1) \backslash \mathcal{V}(P_1)$, $\vec{x_2} = \mathcal{V}(\hat{\sigma}_2) \backslash \mathcal{V}(P_2)$ et $\vec{x_1} \cap \vec{x_2} = \emptyset$.

Figure 3.2. Ensemble de règles $\mathcal{RND}$ pour l'unification dans une théorie décomposable bornée

formes résolues. Supposons que Res engendre une équation $x=^?t$, $x \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ et $t \in \mathcal{T}(\mathcal{SF}, \mathcal{X})\backslash\mathcal{X}$. Dans ce cas, $USP(P)$ ne décroît pas nécessairement, mais $NVR_{\mathcal{SF}}(P)$ décroît. Considérons maintenant le cas contraire, où $USP(P)$ décroît. Supposons que soit engendrée par Res une équation $x=^?y$ telle que $PVR(P)$ augmente. Deux cas de figure se présentent: soit $x, y \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ et $NVC(P)$ décroît, soit $x \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$, $y \notin \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ où y est forcément marquée alors que x ne l'est pas. Dans ce dernier cas, $NM(P)$ décroît. Tous les autres cas contredisent l'hypothèse que $PVR(P)$ augmente. D'autre part, si $PVR(P)$ n'augmente pas, alors $USP(P)$ décroît toujours. □

Proposition 3.13 *Le système de règles $\mathcal{RND}$ termine pour tout contrôle et retourne une disjonction de formes séquentiellement résolues équivalente au problème initial $\Gamma_1 \wedge \Gamma_2$.*

Preuve : La preuve de terminaison est résumée par le tableau suivant:

regles	$NVR_{\mathcal{SF}}(P)$	$NM(P)$	$NA(P)$	$NVC(P)$	$PVR(P)$	$USP(P)$
Conflits(1)	$=$	$\downarrow$				
Conflits(2)	$\downarrow$					
DecRep	$\downarrow$					
Res(0)	$\downarrow$					
Res(1)	$=\downarrow$	$=\downarrow$	$\downarrow$			
Res(2)	$=\downarrow$	$=\downarrow$	$=\downarrow$	$\downarrow$		
Res(3)	$=\downarrow$	$\downarrow$				
Res(4)	$=\downarrow$	$=\downarrow$	$=\downarrow$	$=\downarrow$	$=$	$\downarrow$
VarRep	$=$	$=$	$=$	$=$	$\downarrow$	

□

Théorème 3.2 *Si $E_1 \cup E_2$ est une théorie décomposable bornée alors la $E_1 \cup E_2$-unification est finitaire si la E_i-unification avec restriction linéaire ($i = 1, 2$) est finitaire.*

Les théories faiblement partagées sont des théories bornées dont l'ensemble des contextes $\mathcal{SC}$ est vide.

Corollaire 3.4 *Si $E_1 \cup E_2$ est une théorie décomposable faiblement partagée alors la $E_1 \cup E_2$-unification est finitaire si la E_i-unification avec restriction linéaire ($i = 1, 2$) est finitaire.*

L'algorithme simplifié correspondant ressemble beaucoup à celui connu dans le mélange de théories disjointes. La seule différence se situe dans la règle DecRep qui, comme VarRep, gère l'interaction entre les théories.

Une autre cas exemplaire est donné par une théorie décomposable ne partageant que des constantes, qui est forcément bornée par l'ensemble $\mathcal{SC}$ de ces constantes.

Corollaire 3.5 *Si $E_1 \cup E_2$ est une théorie décomposable partageant des constantes alors la $E_1 \cup E_2$-unification est finitaire si la E_i-unification avec restriction linéaire ($i = 1, 2$) est finitaire.*

On remarquera qu'il suffit de ne pas avoir de E_i-égalité entre constantes pour que $E_1 \cup E_2$ soit décomposable.

Nous allons voir que ce résultat concernant le mélange avec constantes partagées peut être étendu au problème de décision.

3.3 Approche par décision

La résolution effective des problèmes purs n'est plus requise ici. On suppose ne disposer que d'algorithmes permettant de décider si un problème est unifiable. L'idée consiste à borner non plus les contextes partagés apparaissant en tête des formes normales permettant de passer d'une théorie à l'autre mais plutôt les instances partagées suffisantes pour valider toute égalité.

3.3.1 Hypothèses

Nous allons maintenant considérer des théories dont on peut supposer "sans perte de généralité" que les égalités ont un nombre borné de symboles partagés consécutifs, qu'ils apparaissent en tête ou ailleurs. Cela signifie qu'il existe pour cette théorie un entier n tel qu'on puisse toujours se restreindre à une égalité ne contenant dans chacun de ses membres qu'un nombre de symboles partagés consécutifs inférieur à n. Pour formaliser cette propriété, on abstrait certains sous-termes à têtes partagées par des variables. Soient ϱ une bijection de

$$\{u\downarrow_R \mid u \in \mathcal{T}(\mathcal{F},\mathcal{X}) \text{ et } u\downarrow_R(\epsilon) \in \mathcal{SF}\}$$

vers un ensemble de variables de $\mathcal{X}$ et Ω une application qui a tout terme t de $\mathcal{T}(\mathcal{F},\mathcal{X})$ associe un ensemble de positions $\Omega(t)$ tel que

$$\Omega(t) \subseteq \{\omega \in \mathcal{P}os(t) \mid t(\omega) \in \mathcal{SF}\}.$$

On appelle *abstraction partagée* par rapport à (ϱ,Ω) une application $\varrho_\Omega : \mathcal{T}(\mathcal{F},\mathcal{X}) \to \mathcal{T}(\mathcal{F},\mathcal{X})$ qui à tout terme t associe $t^\varrho = t[\omega \hookleftarrow \varrho(t_{|\omega}\downarrow_R)]_{\omega\in\Omega(t)}$.

Nous supposerons toujours que le codomaine de ϱ est disjoint de celui de π, c'est-à-dire que ces deux abstractions définissent conjointement une bijection de $\mathcal{T}(\mathcal{F}\cup\mathcal{X})\downarrow_R$ vers un ensemble de variables.

Les termes sont abstraits lorsqu'ils contiennent plus d'un nombre fixé de symboles partagés consécutifs. La hauteur de partage d'un terme t est intuitivement le nombre maximal de symboles partagés consécutifs sur une branche de t:

Définition 3.10 *La* hauteur de partage *d'un terme t, notée $hp(t)$, est définie inductivement par*

- $hp(x) = 0$ *si* $x \in \mathcal{X}$,
- $hp(c) = 1$ *si* $c \in \mathcal{SF}_0$,
- $hp(c) = 0$ *si* $c \in \mathcal{F}_0 \backslash \mathcal{SF}$,
- $hp(f(s_1,\ldots,s_p)) = \max(1+\max_{s_i(\epsilon)\in\mathcal{SF}}(hp(s_i)), \max_{s_i(\epsilon)\notin\mathcal{SF}}(hp(s_i)))$ *si* $f \in \mathcal{SF}$,
- $hp(f(s_1,\ldots,s_p)) = \max_{i\in\{1,\ldots,p\}}(hp(s_i))$ *si* $f \notin \mathcal{SF}$.

Définition 3.11 *Un* facteur de partage *d'une théorie équationnelle $(\mathcal{F},E)$ par rapport à un ensemble de symboles de fonctions $\mathcal{SF} \subseteq \mathcal{F}$ est un entier n tel qu'il existe une abstraction partagée ϱ_Ω préservant l'égalité dans E i.e.*

$$\forall s,t \in \mathcal{T}(\mathcal{F},\mathcal{X}),\ s =_E t \Leftrightarrow s^\varrho =_E t^\varrho$$

et pour laquelle les termes du codomaine de ϱ_Ω ont des hauteurs de partage inférieures à n.

On suppose disposer d'un oracle permettant de déterminer un facteur de partage pour une théorie donnée, s'il existe.

Dans le cas extrême où l'on peut se ramener à des E-égalités ne contenant plus de symboles partagés alors ces symboles partagés ne sont pas seulement ω-libres mais libres.

Remarque 3.2 *Si une théorie décomposable E admet un facteur de partage nul alors soit $\mathcal{SF} = \emptyset$, soit $\mathcal{SF}$ est un ensemble de symboles libres pour E.*

Dans le cas élémentaire où $\mathcal{SF}$ est un ensemble de constantes alors ϱ_Ω est l'application identité.

Remarque 3.3 *Si $E_1 \cup E_2$ est une théorie décomposable ne partageant que des constantes alors E_1, E_2 et $E_1 \cup E_2$ admettent 1 comme facteur de partage.*

L'existence d'un facteur de partage est une propriété modulaire pour les théories faiblement partagées qui sont des théories non effondrantes.

Proposition 3.14 *Si $E_1 \cup E_2$ est une théorie décomposable en deux théories faiblement partagées E_1 et E_2 admettant respectivement n_1 et n_2 comme facteurs de partage alors $E_1 \cup E_2$ admet $\max(n_1, n_2)$ comme facteur de partage.*

Preuve : On suppose l'existence d'abstractions partagées $\varrho_{1\Omega_1}$ et $\varrho_{2\Omega_2}$ préservant respectivement l'égalité de E_1 et E_2, dont les facteurs de partage sont respectivement n_1 et n_2.

Nous allons dans un premier temps définir une abstraction partagée pour $E = E_1 \cup E_2$ à partir de celles supposées pour E_1 et E_2. Etant donnée une bijection ϱ quelconque, l'application Ω est construite inductivement comme suit:

- $\Omega(x) = \{\}$,
- $\Omega(t) = \{\epsilon\}$ si $t(\epsilon) \in \mathcal{SF}$,
- $\Omega(t) = \Omega_i(t^{\pi_i}) \cup \{\omega.\omega' \mid \omega \in AlienPos(t), (t_{|\omega})^{\pi_i} \in \mathcal{V}(t^{\pi_i}), \omega' \in \Omega(t_{|\omega})\}$ si $t(\epsilon) \in \mathcal{F}_i \backslash \mathcal{SF}$ $(i = 1, 2)$.

Il est facile ensuite de vérifier par induction sur la hauteur de théories que pour tout terme t, on a $hp(t^\varrho) \leq n$ où $n = \max(n_1, n_2)$. Il s'agit de montrer ensuite que l'abstraction ϱ_Ω préserve la E-égalité. Pour cela, on prouve par induction sur la hauteur de théories que $l^\varrho =_E (l \downarrow_R)^\varrho$.

Si l est un terme i-pur alors $l =_{E_i} (l \downarrow_R)^{\pi_i}$, $l^{\varrho_i} =_{E_i} ((l \downarrow_R)^{\pi_i})^{\varrho_i}$ et $l^\varrho = l^{\varrho_i} =_{E_i} ((l \downarrow_R)^{\pi_i})^{\varrho_i} = ((l \downarrow_R)^{\pi_i})^\varrho$, par définition de ϱ_Ω. En instanciant par $(\pi^{-1})^\varrho$, on obtient $l^\varrho = l^\varrho(\pi^{-1})^\varrho =_E ((l \downarrow_R)^{\pi_i})^\varrho(\pi^{-1})^\varrho = (l \downarrow_R)^\varrho$.

Considérons un i-terme l hétérogène. Puisque E_i est faiblement partagée et donc non effondrante, l^{π_i} est un terme i-pur et $l^{\pi_i} =_{E_i} (l \downarrow_R)^{\pi_i}$. Il existe par hypothèse une abstraction partagée $\varrho_{i\Omega_i}$ telle que $(l^{\pi_i})^{\varrho_i} =_{E_i} ((l \downarrow_R)^{\pi_i})^{\varrho_i}$ puis $(l^{\pi_i})^{\varrho_i}(\pi^{-1})^\varrho =_E ((l \downarrow_R)^{\pi_i})^{\varrho_i}(\pi^{-1})^\varrho = (l \downarrow_R)^\varrho$ en instanciant la précédente E_i-égalité par $(\pi^{-1})^\varrho$. D'après l'hypothèse d'induction, on a $(l^{\pi_i})^{\varrho_i}(\pi^{-1})^\varrho =_E l^\varrho$ et donc $l^\varrho =_E (l \downarrow_R)^\varrho$.

En conclusion, si $l =_E r$ alors $l \downarrow_R = r \downarrow_R$ et $l^\varrho =_E (l \downarrow_R)^\varrho = (r \downarrow_R)^\varrho =_E r^\varrho$. □

On suppose dorénavant que $E = E_1 \cup E_2$ est une théorie décomposable admettant un facteur de partage n grâce à une abstraction partagée ϱ_Ω.

3.3.2 Algorithme

Un terme ou un problème d'unification est dit *strictement pur* s'il est pur et ne contient pas de symboles partagés. Un terme ou un problème d'unification est *partagé* s'il ne contient que des symboles partagés.

Lemme 3.4 *Soit E une théorie décomposable. Si s, t sont des termes strictement purs et σ une substitution R-normalisée alors*

$$s\sigma =_E t\sigma \Leftrightarrow s\sigma^{\varrho} =_E t\sigma^{\varrho}.$$

Preuve : Les symboles partagés ne pouvant apparaitre que dans la substitution σ, on a $(s\sigma)^{\varrho} =_E s\sigma^{\varrho}$ et $(t\sigma)^{\varrho} =_E t\sigma^{\varrho}$. La substitution σ est une instance de σ^{ϱ} qui contient au plus n symboles partagés consécutifs. □

La principale caractéristique de l'algorithme est de résoudre des problèmes strictement purs en appliquant explicitement toutes les substitutions de $SUBS_V^{\mathcal{SC}}$ où $\mathcal{SC} = \{C \mid C \in \mathcal{T}(\mathcal{SF}, []), hp(C) \leq n\}$.

Un problème d'unification hétérogène Γ est équivalent après purification (où les symboles partagés sont vus comme étrangers) à une conjonction de problèmes $\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2$ respectivement partagé, strictement 1-pur et 2-pur. La résolution du problème Γ_0 s'effectue dans la théorie vide $\emptyset$, où les symboles partagés sont considérés comme libres.

La 0-abstraction d'un terme t est $t^{\pi_0} = (t^{\pi_1})^{\pi_2} \in \mathcal{T}(\mathcal{SF}, \mathcal{X})$.

Lemme 3.5 *Si s, t sont des termes partagés et σ une substitution R-normalisée alors*

$$s\sigma =_E t\sigma \Leftrightarrow s\sigma^{\pi_0} =_{\emptyset} t\sigma^{\pi_0}.$$

Preuve :

$$\begin{array}{rcl} s\sigma =_E t\sigma & \Leftrightarrow & s\sigma^{\pi_1} =_{E_1} t\sigma^{\pi_1} \text{ car } s,t \text{ sont 1-purs.} \\ & \Leftrightarrow & s(\sigma^{\pi_1})^{\pi_2} =_{E_2} t(\sigma^{\pi_1})^{\pi_2} \text{ car } s,t \text{ sont 2-purs.} \\ & \Leftrightarrow & s\sigma^{\pi_0} =_{\emptyset} t\sigma^{\pi_0} \text{ car les symboles de } \mathcal{SF} \text{ sont } \omega\text{-libres dans } E_2. \end{array}$$

□

L'algorithme que nous proposons consiste à réutiliser celui dans le mélange de théories disjointes pour résoudre la conjonction $\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2$ grâce aux algorithmes d'unification dans $\emptyset, E_1, E_2$.

Théorème 3.3 *Soit $E_1 \cup E_2$ une théorie décomposable admettant un facteur de partage. La $E_1 \cup E_2$-unification est décidable (resp. finitaire) si la E_i-unification ($i = 1, 2$) avec restriction linéaire est décidable (resp. finitaire).*

Preuve : Soit ϕ une solution R-normalisée d'une conjonction $\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2$ où Γ_0 est un problème partagé et Γ_1, Γ_2 des problèmes respectivement strictement 1-pur et strictement 2-pur. On peut supposer sans perte de généralité que toutes les variables s'instancient par des termes non variables et distincts entre eux, sinon les problèmes n'ont pas été identifiés et doivent l'être au préalable. D'autre part et d'après l'existence d'un facteur de partage, on peut supposer qu'il existe une substitution $\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$ et une substitution σ telles que

- $\forall x \in \mathcal{V}(\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2), x\phi = x\rho\sigma$,
- σ est une substitution R-normalisée telle que $\pi(x\sigma) = x$ si $x\sigma(\epsilon) \in \mathcal{F}\backslash\mathcal{SF}$,
- σ est une substitution R-normalisée telle que $\varrho(x\sigma) = x$ si $x\sigma(\epsilon) \in \mathcal{SF}$,
- $\sigma \in SS_E(\Gamma_0\rho \wedge \Gamma_1\rho \wedge \Gamma_2\rho)$,

et σ est obtenue à partir d'une solution où il est fait abstraction des termes ayant un symbole partagé en tête. Plus précisement, $(\sigma_{|V_1\cup V_2})^\varrho \in SS_E(\Gamma_1\rho \wedge \Gamma_2\rho, V_0)$ où $V_i = \{x \mid x\sigma(\epsilon) \in \mathcal{F}_i\backslash\mathcal{SF}\}$ pour $i = 1, 2$ et $V_0 = \{x \mid x\sigma(\epsilon) \in \mathcal{SF}\}$. On a donc $\sigma_1 = ((\sigma_{|V_1\cup V_2})^\varrho)^{\pi_1} \in SS_{E_1}(\Gamma_1\rho, V_0 \cup V_2)$ et $\sigma_2 = ((\sigma_{|V_1\cup V_2})^\varrho)^{\pi_2} \in SS_{E_2}(\Gamma_2\rho, V_0 \cup V_1)$. La substitution $\sigma_0 = \sigma^{\pi_0}$ est solution de $\Gamma_0\rho$ dans $\emptyset$, et plus précisement $\sigma_0 \in SS_\emptyset(\Gamma_0\rho, V_1 \cup V_2)$. Une variable de V_i $(i = 0, 1, 2)$ n'est donc instanciée que par σ_i. L'ordre linéaire $<$ est choisi tel que $x < y$ si $x\sigma$ est sous-terme strict de $y\sigma$. On s'assure ensuite que σ_0, σ_1 et σ_2 satisfont la restriction linéaire $<$. Si le terme $x\sigma_i$ $(i = 0, 1, 2)$ contient une variable $y \in V_j$ $(j \neq i)$ alors montrons que $x \not< y$. Par construction:

- Si le terme $x\sigma_i$ $(i = 1, 2)$ contient une variable $y \in V_0$, alors $y\sigma$ est sous-terme de $x\sigma$.
- Si le terme $x\sigma_i$ $(i = 1, 2)$ contient une variable $y \in V_j$ avec $j = 1, 2$ et $i \neq j$, alors $y\sigma$ est sous-terme étranger de $x\sigma$.
- Si le terme $x\sigma_0$ contient une variable $y \in V_1 \cup V_2$ alors $y\sigma$ est sous-terme de $x\sigma$.

Dans tous les cas, $y\sigma$ est sous-terme strict de $x\sigma$ car $x\sigma$ et $y\sigma$ sont différents puisque x et y n'ont pas été identifiées. Donc $y < x$ par définition de $<$.

La solution ϕ R-normalisée de $\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2$ est donc égale à $\rho\sigma$ où σ est une instance d'une solution combinée $\sigma_0 \odot (\sigma_1 \odot \sigma_2)$ de $\Gamma_0\rho \wedge (\Gamma_1\rho \wedge \Gamma_2\rho)$ dans les théories $\emptyset, E_1$ et E_2. □

Un ensemble complet d'unificateurs est obtenu en utilisant l'algorithme de combinaison dans le mélange de théories disjointes sur la disjonction

$$\bigvee_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_0\wedge\Gamma_1\wedge\Gamma_2)}} (\Gamma_0\rho \wedge \Gamma_1\rho \wedge \Gamma_2\rho)$$

avec les algorithmes d'unification dans les théories $\emptyset$, E_1 et E_2 pour résoudres les sous-problèmes respectivement partagé et strictement purs.

3.3.3 Applications

La notion de facteur de partage est complètement abstraite et inutilisable directement. Il est donc primordial de montrer que la classe de théories admettant un facteur de partage est non vide et qu'elle contient même le cas important en pratique des partages de constantes.

Lorsque $\mathcal{SF}$ est vide, les théories E_1, E_2 et $E_1 \cup E_2$ admettent des facteurs de partage nuls. L'ensemble des substitutions $SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1\wedge\Gamma_2)}$ est vide et l'on retrouve bien l'algorithme de combinaison dans les théories disjointes. Lorsque $\mathcal{SF}$ est maintenant un ensemble de

constantes alors $E_1, E_2, E_1 \cup E_2$ admettent 1 comme facteur de partage. Dans ce cas particulier, $SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$ représente l'ensemble des substitutions dont le codomaine est un ensemble de constantes.

Corollaire 3.6 *Si $E_1 \cup E_2$ est un théorie décomposable dont les symboles partagés sont des constantes alors la $E_1 \cup E_2$-unification est décidable (resp. finitaire) si la E_i-unification ($i = 1, 2$) avec restriction linéaire est décidable (resp. finitaire).*

Rappelons encore une fois qu'une théorie partageant des constantes est décomposable s'il n'y a pas de E_i-égalités entre constantes.

Les constantes partagées n'est pas le seul cas admettant 1 comme facteur de partage.

Exemple 3.3 *Soient $E_1 = \{f(x) = g(x)\}$ et $E_2 = \{f(x) = h(x)\}$ avec la précédence $h > g > f$ où f est partagé. Il est facile de vérifier que $E_1 \cup E_2$ admet 1 comme facteur de partage.*

Cette approche permet donc de traiter par combinaison des cas "d'école" qui ne pouvaient pas l'être par le passé malgré leur déconcertante simplicité.

Exemple 3.4 *(Suite de l'Exemple 3.2). Considérons la théorie $E = E_1 \cup E_2$ avec $E_i = \{f(x) = g_i(f(x))\}$ et qui admet 1 comme facteur de partage. L'équation*

$$g_1(f(x)) =^? g_2(f(f(y)))$$

se résout de la façon suivante:

Après purification, on obtient la conjonction

$$z =^? g_1(v_1) \quad \wedge \quad z =^? g_2(v_2) \quad \wedge \quad (v_1 =^? f(x) \wedge v_2 =^? f(f(y))).$$

Après instanciation de ce problème par $\rho = \{z \mapsto f(w)\}$, il s'agit de résoudre

$$f(w) =^? g_1(v_1) \quad \wedge \quad f(w) =^? g_2(v_2) \quad \wedge \quad (v_1 =^? f(x) \wedge v_2 =^? f(f(y))).$$

Après résolution des équations strictement pures dans les théories respectives, on obtient

$$v_1 =^? f(w) \wedge v_2 =^? f(w).$$

Ces équations sont ensuite reportées dans le problème partagé pour aboutir au problème

$$f(w) =^? f(x) \wedge f(w) =^? f(f(y))$$

qui est équivalent dans $\emptyset$ à $x =^? w \wedge w =^? f(y)$. Le dernier problème étant en forme séquentiellement résolue, il n'est plus besoin d'imposer une restriction linéaire et la solution principale est celle attendue: $\{x \mapsto f(y)\}$. Les variables z, v_1, v_2, w sont supposées existentiellement quantifiées et n'apparaissent donc plus dans cette solution.

L'existence d'un facteur de partage permet donc de traiter des théories non bornées.

Exemple 3.5 *La théorie $E = E_1 \cup E_2$ avec $E_i = \{g_i(f(x), f(y)) = g_i(f(y), f(x))\}$ est décomposable car les théories E_i sont faiblement partagées. Nous avons vu précédemment un algorithme de combinaison de l'unification pour les théories faiblement partagées. Mais la connaissance d'un facteur de partage valant 1 pour les théories E_i permet cette fois de combiner les algorithmes de décision de l'unification. On note au passage que la surréduction ne termine pas et ne permet pas d'obtenir un algorithme de E-unification.*

Il existe par conséquent des théories décomposables (ne partageant pas uniquement les constantes) pour lesquelles on dispose à la fois d'un algorithme de combinaison de l'unification et d'un algorithme de combinaison de décision de l'unification.

3.4 Codage dans le mélange de théories disjointes

Nous avons présenté un algorithme d'unification s'inspirant beaucoup de celui existant dans les théories disjointes. Il est possible en fait de repositionner, dans certains cas, un problème d'unification dans une théorie décomposable avec facteur de partage en un problème d'unification dans le mélange de deux théories disjointes. Une fois ce lien établi, nous pourrons ainsi nous contenter dans une certaine mesure de l'étude du cas disjoint lorsque l'on s'intéressera à des problèmes d'unification spécifiques.

Le passage à un mélange de théories disjointes s'effectue en dupliquant l'ensemble des symboles partagés $\mathcal{SF}$ pour obtenir $\mathcal{SF}_1$ et $\mathcal{SF}_2$:

$$\mathcal{SF}_i = \{f_i \mid f \in \mathcal{SF}\}.$$

Soit $\mathcal{F}'_i = (\mathcal{F}_i \backslash \mathcal{SF}) \cup \mathcal{SF}_i$. On considère la présentation $(\mathcal{F}'_i, E'_i)$ obtenue en rajoutant l'indice i aux symboles partagés de $(\mathcal{F}_i, E_i)$. Pour passer d'une signature à l'autre nous utiliserons le système de réécriture $I = \{f_i(x_1, \ldots, x_n) \to f(x_1, \ldots, x_n) \mid f \in \mathcal{SF}, |f| = n\}$ qui est évidemment convergent.

Supposons la purification déjà effectuée et qu'il ne reste plus qu'à résoudre dans $E_1 \cup E_2$ une conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes d'unification respectivement strictement 1-pur et 2-pur. Si Γ_1 et Γ_2 sont indicées par 1 et 2 pour obtenir Γ'_1 et Γ'_2, alors une $E'_1 \cup E'_2$-solution de $\Gamma'_1 \wedge \Gamma'_2$ est une $E_1 \cup E_2$-solution de $\Gamma_1 \wedge \Gamma_2$ par effacement des indices (i.e. normalisation suivant $\to_I$).

Proposition 3.15 *Soient s, t des termes de $\mathcal{T}(\mathcal{F}'_1 \cup \mathcal{F}'_2, \mathcal{X})$.*
Si $s =_{E'_1 \cup E'_2} t$ alors $s \downarrow_I =_{E_1 \cup E_2} t \downarrow_I$.

Preuve : Les théorèmes valides dans E' le sont dans n'importe quel modèle de E', en particulier dans E où l'interprétation des symboles f_1, f_2 est identique pour $f \in \mathcal{SF}$. □

Réciproquement, pour la complétude, il s'agit de montrer qu'une $E_1 \cup E_2$-solution de $\Gamma_1 \wedge \Gamma_2$ est obtenue en oubliant les indices d'une $E'_1 \cup E'_2$-solution de $\Gamma'_1 \wedge \Gamma'_2$. Pour ce faire, nous allons établir un codage qui permet de passer d'un terme de $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})$ à un terme de $\mathcal{T}(\mathcal{F}'_1 \cup \mathcal{F}'_2, \mathcal{X})$. Etant donné $t \in \mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})$, on note t' le terme de $\mathcal{T}(\mathcal{F}'_1 \cup \mathcal{F}'_2, \mathcal{X})$ identique à t sauf aux positions où figurent un symbole partagé qui est alors indicé par la théorie du symbole non partagé qui le précède.

Ce codage ne s'applique qu'à un ensemble de termes T dont les symboles de tête ne sont pas partagées. On note T' l'ensemble issu du codage de T. De même, on note σ' la substitution $\{x \mapsto (x\sigma)' \mid x \in \mathcal{D}om(\sigma)\}$ si σ vérifie $x\sigma(\epsilon) \notin \mathcal{SF}$ pour $x \in \mathcal{D}om(\sigma)$.

Une abstraction par variables $\pi : T \downarrow_R \to \mathcal{X}$ définit elle aussi une abstraction par variables $\pi'_i : T \downarrow'_R \to \mathcal{X}$ par $\pi'(t') = \pi(t)$. La i-abstraction $t'^{\pi'_i}$ d'un terme t' est ensuite définie comme précédemment dans le cas disjoint.

Lemme 3.6 *Soit $\Gamma_1 \wedge \Gamma_2$ une conjonction de problèmes strictement purs. Si σ est une solution R-normalisée de $\Gamma_1 \wedge \Gamma_2$ n'ayant pas de symbole partagé en tête alors σ' est une $E'_1 \cup E'_2$-solution de $\Gamma'_1 \wedge \Gamma'_2$*

Preuve : Pour tout $i = 1, 2$ et n'importe quelle équation $s =^? t$ de Γ_i, on a:

$$\begin{array}{lll} & s\sigma =_{E_1 \cup E_2} t\sigma & \\ \Leftrightarrow & (s\sigma)^{\pi_i} =_{E_i} (t\sigma)^{\pi_i} \text{ par lemme 3.2} & (1) \\ \Leftrightarrow & (s\sigma')^{\pi'_i} =_{E'_i} (t\sigma')^{\pi'_i} & (2) \end{array}$$

(2) est simplement un renommage de (1) car

- les symboles indicés par j apparaissent nécessairement dans les sous-termes étrangers qui sont remplacés par de nouvelles variables.
- les variables qui sont des i-abstractions de sous-termes étrangers sont inchangées par définition de π'.

Ensuite (2) implique $((s\sigma')^{\pi'_i})\pi'^{-1} =_{E'_i} ((t\sigma')^{\pi'_i})\pi'^{-1}$. Cette égalité est identique à $s\sigma' =_{E'_i} t\sigma'$. Donc $s\sigma' =_{E'_1 \cup E'_2} t'\sigma$ et $\sigma = \sigma' \downarrow_I$ par définition de σ'. □

Théorème 3.4 *Soit $E_1 \cup E_2$ une théorie décomposable admettant un facteur de partage. Un algorithme d'unification dans le mélange de deux théories équationnelles disjointes avec en entrée*

$$\bigvee_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}} (\Gamma_1\rho) \wedge (\Gamma_2\rho)$$

fournit un ensemble complet de $E_1 \cup E_2$-solutions à une conjonction de problèmes d'unification $\Gamma_1 \wedge \Gamma_2$ strictement purs.

Preuve : Pour toute solution R-normalisée σ de $\Gamma_1 \wedge \Gamma_2$, il existe d'après le lemme 3.6 une substitution ϕ R-normalisée et un $\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}$ tels que ϕ' est une $E'_1 \cup E'_2$-solution de $((\Gamma_1\rho)' \wedge (\Gamma_2\rho)')$ et $\rho\phi \leq^{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)} \sigma$. Par conséquent,

$$\bigcup_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}} SU_{E'_1 \cup E'_2}((\Gamma_1\rho)' \wedge (\Gamma_2\rho)') \downarrow_I \circ\rho$$

est un $CSU_{E_1 \cup E_2}(\Gamma_1 \wedge \Gamma_2)$. L'ensemble de solutions peut être ensuite remplacé par un ensemble complet de solutions. Soit $\psi \in CSU_{E'_1 \cup E'_2}((\Gamma_1\rho)' \wedge (\Gamma_2\rho)')$ tel que $\psi \leq^{\mathcal{V}((\Gamma_1\rho)' \wedge (\Gamma_2\rho)')}_{E'_1 \cup E'_2} \phi'$, on a

$$\begin{array}{lll} \forall x \in \mathcal{V}((\Gamma_1\rho)' \wedge (\Gamma_2\rho)'),\ x\sigma & =_{E_1 \cup E_2} & x\phi' \downarrow_I \\ & =_{E_1 \cup E_2} & x(\psi\mu) \downarrow_I \\ & =_{E_1 \cup E_2} & x\psi \downarrow_I \mu \downarrow_I . \end{array}$$

Donc

$$\bigcup_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)}} CSU_{E'_1 \cup E'_2}((\Gamma_1\rho)' \wedge (\Gamma_2\rho)') \downarrow_I \circ\rho$$

est un $CSU_{E_1 \cup E_2}(\Gamma_1 \wedge \Gamma_2)$. □

Ce résultat fournit encore une fois un algorithme d'unification pour une théorie décomposable partageant les constantes. La transformation d'un problème hétérogène en une conjonction de problèmes $\Gamma_1 \wedge \Gamma_2$ strictement purs nécessite de poser en parallèle

une substitution $\rho \in SUBS_V^{\mathcal{SC}}$ pour un $V \in \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$ qui sera ensuite prolongée par l'ensemble des substitutions $SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_1 \wedge \Gamma_2)\setminus V}$ lors de la réutilisation de l'algorithme de combinaison dans le mélange de théories disjointes.

Le mélange avec constantes partagées est le cas où l'unification dans la théorie vide du problème partagé se limite donc à la plus simple expression.

Cette forme de mélange n'est pas pour autant à sous-estimer car elle apparait naturellement lorsqu'il est possible de transformer, par abstraction de termes clos, une théorie équationnelle en une union de deux théories. Les constantes introduites par abstraction sont alors partagées par ces deux théories et n'ont pas à être égales entre elles. Les théories closes avec symboles AC [NR93], les théories minces [CHJ92] avec symboles AC ou théories SHAC [DK93], sont des exemples pour lesquels le problème de l'unification peut en conséquence se traiter par combinaison.

4
Combinaison d'algorithmes de décision

L'utilisation d'un algorithme d'unification élémentaire est aussi une façon constructive de décider d'une contrainte équationnelle dont les variables, les inconnues, sont existentiellement quantifiées. L'unification élémentaire n'étant pas toujours adaptée à la combinaison, on lui préfère l'unification avec constantes libres pour les théories régulières effondrantes et dans le cas général l'unification avec symboles libres. La notion de constante libre correspond en logique du premier ordre à une variable universellement quantifiée [Bür90]. Un algorithme de combinaison de l'unification a ainsi pour objet la décision dans l'algèbre libre $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/=_{E_1 \cup E_2}$ de contraintes équationnelles

$$\forall \vec{y} : (\exists \vec{x} : \Gamma)$$

où $\Gamma = (\bigwedge_{k \in K} s_k =^? t_k)$ et $\mathcal{V}(\Gamma) = \vec{x} \oplus \vec{y}$, par utilisation des algorithmes de décision du même type de contraintes dans $\mathcal{T}(\mathcal{F}_1, \mathcal{X})/=_{E_1}$ et $\mathcal{T}(\mathcal{F}_2, \mathcal{X})/=_{E_2}$.

Le problème abordé indirectement dans ce chapitre est celui de la décision de contraintes équationnelles spécifiques que sont:

- le problème du mot, $\forall \vec{y} : s =^? t$ avec $\vec{y} = \mathcal{V}(s) \cup \mathcal{V}(t)$.
- le filtrage, $\forall \vec{y} : (\exists \vec{x} : s =^? t)$ avec $\mathcal{V}(t) \subseteq \vec{y}$.

Les valeurs associées aux inconnues $\vec{x}$ peuvent être évidemment obtenues par l'algorithme de combinaison de l'unification avec constantes, s'il existe un tel algorithme dans chaque théorie composante. Cette hypothèse est trop forte pour bon nombre de théories pour lesquelles l'unification est indécidable alors que le problème du mot ou le filtrage est décidable. De plus, l'utilisation d'un algorithme d'unification pour résoudre un problème du mot ou de filtrage s'avère souvent très inefficace.

C'est pourquoi nous allons chercher à combiner des algorithmes de décision spécialisés. Il ne suffit pas pour autant de remplacer chaque algorithme d'unification par un plus spécifique dans l'algorithme de combinaison. En cherchant à résoudre des problèmes hétérogènes spécifiques, on risque en effet d'avoir à considérer des problèmes purs ayant perdu en spécificité. Nous allons voir sous quelles conditions cette spécificité peut être maintenue au cours de la résolution. Cela va nous conduire à adapter les règles de combinaison de l'unification d'abord pour le problème du mot puis le filtrage et finalement pour le problème de la combinaison des solutions.

Pour clôre ce chapitre, nous allons reprendre les travaux précurseurs de Nelson et Oppen [NO79, Nel81] qui se sont intéressés à la combinaison d'algorithmes de décision dans des théories non équationnelles. Les règles que nous présentons dans ce cadre sont celles qui s'avèrent en définitive essentielles à la combinaison. Elles doivent être cependant adaptées pour s'appliquer au modèle de combinaison particulier que représente $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/=_{E_1 \cup E_2}$.

4.1 Problème du mot

L'objectif de cette section est d'adapter les techniques de combinaison à un problème d'unification très particulier où les membres gauche et droit d'une équation ne contiennent que des constantes libres. Ce problème correspond à la décision de l'égalité dans une théorie équationnelle, ou problème du mot.

Nous allons nous intéresser à la construction incrémentale d'un algorithme de décision du problème du mot dans l'union de théories équationnelles $E = E_1 \cup E_2$ à partir de ceux disponibles dans les théories équationnelles E_1 et E_2.

La construction incrémentale d'algorithmes de décision du problème du mot est fortement reliée aux différents aspects de la modularité de système de réécriture. Si les théories équationnelles E_1 et E_2 sont engendrées par des systèmes de réécriture canoniques, combiner des algorithmes de décision du problème du mot se limite à l'étude de la modularité de la convergence. Notre approche est plus générale car elle ne présuppose pas l'existence d'un système de réécriture canonique pour chaque théorie.

La combinaison du problème du mot a déjà été abordée dans le cadre de la combinaison de l'unification par E. Tiden [Tid86] et M. Schmidt-Schauß [SS89]. Ces deux auteurs prouvent la décidabilité du problème du mot dans le mélange de théories disjointes. La méthode est basée sur une transformation préalable des termes. Cette méthode a été reprise plus tard par T. Nipkow [Nip91] dans le cadre de la combinaison du filtrage.

La méthode que nous proposons pour le mélange de théories disjointes est analogue à celles existantes mais sa compréhension est grandement facilitée par l'utilisation du système de réécriture combiné R convergent sur les termes clos introduit au chapitre 2. La réécriture par ce système de réécriture R infini n'étant pas opérationnelle, l'idée est de trouver un moyen pour simuler la mise en forme normale suivant R. Nous verrons comment substituer la notion de *forme réduite par couches* à celle de forme normale. Cette nouvelle forme canonique a l'avantage de pouvoir se calculer.

Hypothèse 4.1 *Les théories équationnelles E_1 et E_2 ont des signatures disjointes.*

On précise d'abord comment ramener le problème du mot dans le mélange de théories à un problème dans une composante. Nous verrons ensuite que la combinaison des solutions obtenues par chaque algorithme est triviale puisque il s'agit alors de prendre simplement la conjonction des valeurs de vérité retournées par chaque algorithme de décision du problème du mot.

4.1.1 Forme réduite par couches

La principale étape consiste à résoudre le problème dans une seule composante. Comme la notion de substitution n'a pas d'intérêt pour le problème du mot, on aimerait avoir un

lemme de projection de la forme

$$s =_E t \Leftrightarrow s^{\pi_i} =_{E_i} t^{\pi_i}.$$

Malheureusement cette équivalence n'est pas toujours valide. Pour qu'elle le soit, on pourrait supposer s et t R-normalisés. L'objectif est de déterminer une condition plus faible sur les termes en introduisant une autre forme canonique.

Définition 4.1 *Un terme t est en* forme réduite par couches (f.r.c)

- *si t est une variable,*
- *ou si $t(\epsilon)$ et $t \downarrow_R (\epsilon)$ sont des symboles dans la même théorie et les sous-termes étrangers de t sont en forme réduite par couches.*

Lemme 4.1 *Si s est en forme réduite par couches et $s \rightarrow_R t$ alors t est en forme réduite par couches et $s^{\pi_i} =_{E_i} t^{\pi_i}$.*

Preuve : On peut supposer que s est un i-terme puisque une variable est irréductible par R.

La preuve s'effectue par induction sur la hauteur de théories.

Si s est i-pur alors t est forcément un i-terme dont les sous-termes étrangers sont irréductibles lorsqu'ils existent. Donc t est en forme réduite par couches.

Si $s \rightarrow_{R,\omega} t$ avec ω inférieure à toute position de $AlienPos(t)$, alors $s \rightarrow_{E_i,\omega} t$.

Deux cas se présentent:

1. Si $t(\omega) \in \mathcal{F}_j, j \neq i$ alors t est un sous-terme étranger de s. En supposant $\omega = \epsilon$, $t \downarrow_R (\epsilon) = t(\epsilon) \in \mathcal{F}_j$ puisque les sous-termes étrangers de s sont en forme réduite par couches, ce qui contredit que s est en forme réduite par couches puisque $s \downarrow_R (\epsilon) = t \downarrow_R (\epsilon) \in \mathcal{F}_j$. Donc $\omega \neq \epsilon$ et $t_{|\omega}$ est sous-terme étranger en forme réduite par couches d'un i-terme t qui est donc lui aussi en forme réduite par couches.
2. Si $t(\omega) \in \mathcal{F}_i$ alors les sous-termes étrangers de t sont ceux de s plus éventuellement d'autres qui sont forcément irréductibles par définition de R.

Lorsque ω est supérieure ou égale à une position de $AlienPos(s)$, t est forme réduite par couches d'après l'hypothèse d'induction.

En définitive, t est toujours en forme réduite par couches. La seconde assertion est ensuite une conséquence directe de la proposition 2.9. □

Lorsque s n'est pas un i-terme alors l'égalité $s^{\pi_i} =_{E_i} t^{\pi_i}$ est simplement $x = x$ avec $\pi(s \downarrow_R) = x$ et n'a donc peu d'intérêt.

Corollaire 4.1 *Si s et t sont deux termes en forme réduite par couches alors*

$$s =_E t \Longleftrightarrow s^{\pi_i} =_{E_i} t^{\pi_i}.$$

Soient $t_i[s_1,\ldots,s_m] et t'_i[s_{m+1},\ldots,s_k]$ des i-termes.

$$\texttt{Egal}_E(t_i[s_1,\ldots,s_m], t'_i[s_{m+1},\ldots,s_k]) = \texttt{Egal}_{E_i}(t_i[x_1,\ldots,x_m], t'_i[x_{m+1},\ldots,x_k])$$

où $x_1,\ldots,x_k$ sont des variables telles que

- $\forall p \in \{1,\ldots,k\},\ x_p = s_p$ si $s_p \in \mathcal{X}$.
- $\forall p,q \in \{1,\ldots,k\},\ x_p = x_q$ ssi $\texttt{Egal}_E(s_p, s_q)$.

Figure 4.1. Algorithme de décision de l'égalité de termes en f.r.c

Preuve : D'après le lemme précédent nous avons $s^{\pi_i} =_{E_i} (s\downarrow_R)^{\pi_i}$ et $t^{\pi_i} =_{E_i} (t\downarrow_R)^{\pi_i}$. Si $s =_E t$ alors $s\downarrow_R = t\downarrow_R$ et $s^{\pi_i} =_{E_i} (s\downarrow_R)^{\pi_i} = (t\downarrow_R)^{\pi_i} =_{E_i} t^{\pi_i}$. Réciproquement, si $s^{\pi_i} =_{E_i} t^{\pi_i}$ alors $s =_E s^{\pi_i}\pi^{-1} =_{E_i} t^{\pi_i}\pi^{-1} =_E t$. □

Pour faciliter la lecture de l'algorithme, on introduit au préalable une nouvelle notation pour représenter l'opération de i-abstraction. C'est celle utilisée dans [Nip91].

Définition 4.2 *Etant donné t un i-terme, $\mathcal{VP}os_i(t)$ désigne l'ensemble des positions variables de t pour lesquelles les symboles des positions préfixes sont dans $\mathcal{F}_i$:*

$$\mathcal{VP}os_i(t) = \{p \in \mathcal{VP}os(t) \mid \forall q \leq p,\ t(q) \in \mathcal{F}_i\}$$

et $\mathcal{V}_i(t)$ l'ensemble des variables de t aux positions de $\mathcal{VP}os_i(t)$. On note $t_i[s_1,\ldots,s_m]_{p_1,\ldots,p_m}$, abrégé en $t_i[s_1,\ldots,s_m]$, le i-terme t dont l'ensemble des positions $\{p_1,\ldots,p_m\} = \mathcal{VP}os_i(t) \cup AlienPos(t)$ vérifie $t_{|p_k} = s_k$ pour $k = 1,\ldots,m$. Le terme $t_i[x_1,\ldots,x_m]_{p_1,\ldots,p_m}$, abrégé en $t_i[x_1,\ldots,x_m]$, est celui obtenu en remplaçant les termes $s_1,\ldots,s_m$ par les variables $x_1,\ldots,x_m \in \mathcal{X}$:

$$t_i[x_1,\ldots,x_m]_{p_1,\ldots,p_m} = t[p_1 \hookleftarrow x_1]\ldots[p_m \hookleftarrow x_m].$$

Proposition 4.1 *Soient E_1 et E_2 des théories disjointes. La $E_1 \cup E_2$-égalité de termes en forme réduite par couches est décidable si la E_i-égalité est décidable pour $i = 1, 2$.*

Preuve : Considérons la figure 4.1. Il s'agit de montrer par récurrence sur la hauteur de théories que $\texttt{Egal}_E(t,t')$ retourne vrai si et seulement si $(t =_E t')$ lorsque t, t' sont en forme réduite par couches.

- Si t, t' sont des termes i-purs alors $\texttt{Egal}_E(t,t')$ retourne vrai si et seulement si $(t =_{E_i} t')$, qui est équivalent à $(t =_E t')$ puisque les règles de R servant à normaliser t et t' sont forcément dans $E_i^{>}$.
- En supposant vraie l'hypothèse de récurrence sur les sous-termes étrangers de t et t', il existe une abstraction par variables π (une application) telle que

$$\forall q \in \{1,\ldots,k\}, \pi(s_q\downarrow_R) = x_q.$$

Par conséquent $\texttt{Egal}_E(t_i[s_1,\ldots,s_m], t'_i[s_{m+1},\ldots,s_k])$ retourne vrai si et seulement si

$$\begin{array}{ll} & (t_i[x_1,\ldots,x_m] =_{E_i} t'_i[x_{m+1},\ldots,x_k]) \\ \Leftrightarrow & (t^{\pi_i} =_{E_i} t'^{\pi_i}) \\ \Leftrightarrow & (t =_E t'), \end{array}$$

par le corollaire 4.1.

□

Pour les théories non effondrantes, la normalisation d'un terme ne peut pas changer la théorie du symbole de tête et n'importe quel terme est donc en forme réduite par couches. L'algorithme ci-dessus permet déjà d'affirmer:

Corollaire 4.2 *Soient E_1 et E_2 des théories disjointes non effondrantes. Le problème du mot dans $E_1 \cup E_2$ est décidable si le problème du mot dans E_i est décidable pour $i = 1, 2$.*

Exemple 4.1 *Considérons $E_1 = A(+)$ et $E_2 = C(\star)$. L'équation $(a + (a \star b)) + (b \star a) =^?_{E_1 \cup E_2} a + ((a \star b) + (a \star b))$ est vraie si et seulement si $(a + V) + X =^?_{E_1} a + (Y + Z)$ avec $V = X = Y = Z$ car $a \star b =_{E_2} b \star a$. L'équation hétérogène est donc vraie puisque $(a + V) + V =_{E_1} a + (V + V)$.*

La précédente proposition peut être généralisée à l'ensemble des théories dans la mesure où il est possible de calculer une forme réduite par couches.

4.1.2 Calcul d'une forme réduite par couches

Il reste à étudier la possibilité de calculer la forme réduite par couches d'un terme quelconque. Nous allons voir comment utiliser les algorithmes de décision du problème du mot pour effondrer un terme sur un sous-terme étranger. Le risque de considérer comme étranger un terme qui ne l'est pas toujours est alors évité. Des effondrements successifs par un processus ascendant vont permettre de construire une forme réduite par couches.

Définition 4.3 *Un i-terme t s'*effondre *sur une variable ou un sous-terme étranger u de t si $t^{\pi_i} =_{E_i} u^{\pi_i}$. Le terme u est* effondrant *pour t.*

Le E_i-théorème correspondant est effondrant car dans les deux cas u^{π_i} est une variable.

Exemple 4.2 *Supposons $E_1 = \{x + x = x\}$ et E_2 la théorie vide avec $F_2 = \{f, a\}$, le 1-terme $f(a) + f(a)$ s'effondre sur $f(a)$.*

Si les sous-termes étrangers de t sont en forme réduite par couches alors on sait décider de l'égalité entre sous-termes étrangers et il est donc possible de décider si un sous-terme étranger est effondrant pour t.

Proposition 4.2 *L'effondrement d'un i-terme t sur un élément de $\mathcal{V}_i(t) \cup AST(t)$ est décidable si les sous-termes étrangers de t sont en forme réduite par couches et si la E_1-égalité et E_2-égalité sont décidables.*

Soient $t_i[s_1, \ldots, s_m]$ un i-terme et k un indice entre 1 et m.

$$\texttt{Effondrer}(t_i[s_1, \ldots, s_m], s_k) = \texttt{Egal}_{E_i}(t_i[x_1, \ldots, x_m], x_k)$$

où $x_1, \ldots, x_m$ sont des variables telles que

- $\forall p \in \{1, \ldots, m\},\ x_p = s_p$ si $s_p \in \mathcal{X}$.
- $\forall p, q \in \{1, \ldots, m\},\ x_p = x_q$ ssi $\texttt{Egal}_E(s_p, s_q)$.

Figure 4.2. Effondrement d'un terme

Preuve : Considérons la figure 4.2. Par hypothèse, il existe une abstraction par variables π (une application) telle que

$$\forall p \in \{1, \ldots, m\}, \pi(s_p \downarrow_R) = x_p.$$

Ainsi

$$\texttt{Effondrer}(t_i[s_1, \ldots, s_m], s_k) = (t_i[x_1, \ldots, x_m] =_{E_i} x_k) = (t^{\pi_i} =_{E_i} s_k^{\pi_i}).$$

□

Définition 4.4 *Soit $t_i[s_1, \ldots, s_m]$ un i-terme dont les variables et les sous-termes étrangers sont $s_1, \ldots, s_m$. $t \Downarrow$ désigne le terme défini comme suit:*

- *$c \Downarrow = c$ si c est une variable ou une constante.*
- *$(t_i[s_1, \ldots, s_m]) \Downarrow = t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ s'il n'existe pas de terme effondrant pour $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$.*
- *$(t_i[s_1, \ldots, s_m]) \Downarrow = u$ si u est effondrant pour $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$.*

Proposition 4.3 *$t \Downarrow$ est un terme en forme réduite par couches, E-égal à t, qui est calculable si la E_i-égalité est décidable pour $i = 1, 2$.*

Preuve : Par récurrence sur la hauteur de théories.

- Si t est un terme i-pur alors $t \downarrow_R$ est une variable si et seulement celle-ci est effondrante pour t. On dispose alors d'un algorithme pour décider de $t[x] =_{E_i} x$.
- S'il existe un algorithme pour calculer les formes réduites par couches de $s_1, \ldots, s_m$, alors l'effondrement sur une variable ou un sous-terme étranger est décidable. Si un effondrement est possible alors $t \Downarrow$ est en forme réduite par couches. Si aucun effondrement n'est possible alors la normalisation suivant R du terme $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ ne peut pas changer la théorie du symbole de tête. Le terme $t \Downarrow = t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ est donc en forme réduite par couches.

  Dans les deux cas $t \Downarrow$ est E-égal à t car $s_1 =_E s_1 \Downarrow, \ldots, s_m =_E s_m \Downarrow$.

$t \Downarrow$ est calculé récursivement comme suit:

```
Si t ∈ X ∪ (F)_0 alors t ⇓:= t
Si t = (t_i[s_1, ..., s_m]) alors
   t' := t_i[s_1 ⇓, ..., s_n ⇓]
   Pour u ∈ V_i(t') ∪ AST(t') faire
        Si Effondrer(t', u) alors t ⇓:= u
        Sinon t ⇓:= t'
        FinPour
   FinSi
```

Figure 4.3. Calcul de la forme réduite par couches

□

Théorème 4.1 *(M. Schmidt-Schauß [SS89]) Soient E_1 et E_2 des théories disjointes. Le problème du mot dans $E_1 \cup E_2$ est décidable si le problème du mot dans E_i est décidable pour $i = 1, 2$.*

Preuve : Il existe une abstraction par variables π telle que $\texttt{Egal}_E(t \Downarrow, t' \Downarrow)$ retourne vrai si et seulement si

$$\begin{aligned} & ((t \Downarrow)^{\pi_i} =_{E_i} (t' \Downarrow)^{\pi_i}) \\ \Leftrightarrow \quad & (t \Downarrow =_E t' \Downarrow) \\ \Leftrightarrow \quad & (t =_E t') \end{aligned}$$

□

Exemple 4.3 *Considérons $E_1 = A(+)$ et $E_2 = CI(*)$. L'équation*

$$((a + b) * (a + b)) + (a * b) =^?_{E_1 \cup E_2} a + (b + (b * a))$$

est vraie si et seulement si $(a + b) + (a * b) =^?_{E_1 \cup E_2} a + (b + (b * a))$ est vraie. Les membres de cette équation sont maintenant en forme réduite par couches. Le problème est ensuite équivalent à $(a + b) + V =^?_{E_1} a + (b + X)$ avec $V = X$ puisque $a * b =_{E_2} b * a$. L'équation hétérogène est finalement vraie puisque $(a + b) + V =_{E_1} a + (b + V)$.

4.1.3 Règles de combinaison

Les règles de combinaison du problème du mot peuvent être présentées en suivant la démarche suivie pour l'unification. Cette forme de description peut être vue comme une dérécursivation possible de l'algorithme vu précédement qui consistait à calculer les formes normales puis à pratiquer l'abstraction. La différence majeure se situe dans l'introduction explicite de nouvelles variables et équations résolues.

Présentons informellement les grandes lignes de l'algorithme:

La première étape consiste à calculer les formes réduites par couches des membres des équations. Les variables apparaissant dans les membres des équations initiales sont toutes skolémisées. Des variables non skolémisées sont introduites au cours de la purification mais sont existentiellement quantifiées. Une nouvelle variable et une équation résolue sont introduites pour chaque sous-terme étranger s abstrait. Il s'agit ensuite d'identifier les nouvelles variables faisant abstraction de sous-termes étrangers potentiellement équivalents. On obtient des conjonctions d'équations résolues ensuite transformées en équations i-pures dont les variables sont toutes skolémisées dans E_i. Ces équations peuvent être finalement remplacées par leurs formes résolues grâce aux seuls algorithmes de décision du problème du mot.

La correction des règles de combinaison du problème du mot peut se faire au travers des règles de combinaison de l'unification. Mais la spécificité du problème du mot et la mise en forme réduite par couches des termes permet d'éliminer beaucoup de non-déterminisme dans l'utilisation des règles de combinaison de l'unification:

- Une variable introduite par purification est nécessairement instanciée dans la théorie du terme abstrait car celui-ci est clos et en forme réduite par couches. Le choix de la théorie est donc complètement déterministe.
- Le choix d'un ordre linéaire est lui aussi déterministe car il n'y a pas de cycle composé.
- L'identification des variables ne mène à une solution que si elle identifie des termes ayant leurs symboles de tête dans la même théorie.

Il faut ensuite pouvoir résoudre le problème décomposé à l'aide des algorithmes de décision du problème du mot. Deux formes d'équations coexistent encore dans le problème d'unification $e_1 \wedge e_2$ après purification et identification.

Soit $s=^? t$ est un équation i-pure dont les membres sont clos. L'algorithme de décision du problème du mot est dans ce cas susceptible de la résoudre.

Soit $x=^? s$ et x figure dans un autre équation i-pure résolue: $x=^? t$. L'étape de fusion engendre alors l'équation i-pure $s=^? t$. Lorsque x n'apparait plus nulle part ailleurs dans e_i alors l'équation $x=^? s$ est retirée de e_i car x est existentiellement quantifiée.

Finalement toutes les équations sont mises en formes résolues $\bot$ ou $\top$ grâce aux algorithmes de décision du problème du mot dans E_1 et E_2. La combinaison des solutions de chaque composante est triviale dans ce contexte car elle consiste simplement en la conjonction des formes résolues.

Pour conclure, remarquons que les adaptations des règles de combinaison de l'unification au problème du mot se situent en pré et post-traitement. Une phase préliminaire est nécessaire pour la mise en forme réduite par couches des membres des équations. Ensuite et avant la résolution proprement dite, il faut encore rajouter une phase de fusion permettant de reconstruire un problème du mot afin que l'on puisse utiliser les algorithmes disponibles dans chaque théorie. Ces deux phases seront encore présentes pour le filtrage qui est le prochain problème d'unification particulier traité dans ce chapitre.

Proposition 4.4 *Les règles de combinaison du problème du mot appliquées à une équation $s=^? t$ retournent $\top$ si $s =_{E_1 \cup E_2} t$, $\bot$ sinon.*

1. Mise en forme réduite par couches

   Appliquer tant que possible la règle suivante:

$$\frac{e \wedge s =^? t}{e \wedge s \Downarrow =^? t \Downarrow}$$

2. Purification

   Appliquer tant que possible la règle suivante:

   Abstraction Variable

$$\frac{e \wedge s =^? t}{\exists x : e \wedge s[x]_\omega =^? t \wedge x =^? s_{|\omega}}$$

   si

   - $\omega \in AlienPos(s)$,
   - x est une nouvelle variable.

3. Identification de variables

$$\frac{e_1 \wedge e_2}{\bigvee_{\xi \in ID_{\mathcal{V}(e_1 \wedge e_2)}} (e_1 \wedge e_2)\xi}$$

4. Fusion

   Appliquer tant que possible les règles suivantes:

$$\frac{e \wedge x =^? s \wedge x =^? t}{e \wedge s =^? t \wedge x =^? s}$$

$$\frac{\exists x : e \wedge x =^? s}{e} \quad \textbf{si } x \notin \mathcal{V}(e)$$

5. Decide

   Appliquer tant que possible les règles suivantes:

$$\frac{s =^? t}{\top} \quad \textbf{si } s =_{E_i} t$$

$$\frac{s =^? t}{\bot} \quad \textbf{si } s \neq_{E_i} t$$

Figure 4.4. Règles de combinaison du problème du mot

Exemple 4.4 *Considérons à nouveau l'exemple 4.3 et l'équation*

$$((a+b)*(a+b))+(a*b)=^?a+(b+(b*a)).$$

Cette équation est équivalente après mise en forme réduite par couches et purification à $(a+b)+V=^?a+(b+X)\wedge V=^?a*b\wedge X=^?b*a$. *L'équation* $(a+b)+V=^?a+(b+X)$, *où* V *et* X *sont skolémisées, est équivalente à* $\bot$ *et mène donc à un échec. En appliquant l'identification* $\{X\mapsto V\}$ *on doit considérer* $((a+b)+V=^?a+(b+V))\wedge(V=^?a*b\wedge V=^?b*a)$. *Après fusion, il s'agit finalement de résoudre* $(a+b)+V=^?a+(b+V)$ *dans* E_1 *et* $a*b=^?b*a$ *dans* E_2 *qui sont toutes deux équivalentes à* $\top$.

4.1.4 Extension au mélange de théories non disjointes

La notion d'abstraction dans le mélange de théories non disjointes est différente de celle utilisée dans le cas disjoint. Un sous-terme étranger n'est remplacé par une variable que si sa forme normale reste un sous-terme étranger. Lorsque par exemple cette forme normale a un symbole de tête qui est partagé alors elle se substitue au sous-terme étranger. L'abstraction d'un terme quelconque est donc une opération délicate et là encore il est intéressant d'avoir une notion de forme réduite par couches, dans certains cas calculable, et qui ait les mêmes propriétés que la forme normale suivant le système de réécriture combiné R défini à partir d'une théorie décomposable $E=E_1\cup E_2$ (cf. chapitre 3).

Définition 4.5 *Un terme* t *est en* forme réduite par couches (f.r.c) *si*

- t *est une variable,*
- *ou* $t(\epsilon), t\downarrow_R(\epsilon)\in\mathcal{SF}$,
- *ou* $t(\epsilon), t\downarrow_R(\epsilon)\in\mathcal{F}_i\backslash\mathcal{SF}$,

et si les sous-termes étrangers de t *sont en forme réduite par couches.*

On déduit de cette définition que:

Proposition 4.5 *Si* t *est un terme en forme réduite par couches alors* t^{π_i} *est un terme* i*-pur tel que* $t^{\pi_i}=_{E_i}(t\downarrow_R)^{\pi_i}$.

Preuve : Si t est un i-terme alors $t^{\pi_i}=(t[\omega\hookleftarrow t_{|\omega}\downarrow_R]_{\omega\in AlienPos(t)})^{\pi_i}=_{E_i}(t\downarrow_R)^{\pi_i}$ d'après la proposition 3.3. Si t n'est pas un i-terme alors $t^{\pi_i}=(t\downarrow_R)^{\pi_i}$. □

Corollaire 4.3 *Si* s,t *sont deux termes en forme réduite par couches alors*

$$s=_{E_1\cup E_2}t\Leftrightarrow s^{\pi_i}=_{E_i}t^{\pi_i}.$$

Preuve : $s=_{E_1\cup E_2}t\Leftrightarrow s\downarrow_R=t\downarrow_R\Leftrightarrow s^{\pi_i}=_{E_i}(s\downarrow_R)^{\pi_i}=(t\downarrow_R)^{\pi_i}=_{E_i}t^{\pi_i}$. □

La $E_1\cup E_2$-égalité de deux termes en forme réduite par couches est donc décidable si l'on sait décider de la E_i-égalité pour $i=1,2$. Mais pour donner une méthode opérationnelle de calcul d'une forme réduite par couches, la décision de la E_i-égalité ne suffit plus.

Définition 4.6 *Un i-terme t tel que $t(\epsilon) \notin \mathcal{SF}$* change de tête *dans la théorie E_i s'il existe un terme u tel que*

- $t^{\pi_i} =_{E_i} u^{\pi_i}$,
- *u est effondrant pour t ou $u(\epsilon) \in \mathcal{SF}$.*

On dit que u est un terme changeant la tête de t.

On notera que u n'est plus forcément un sous-terme de t et peut avoir son symbole de tête dans $\mathcal{SF}$.

Proposition 4.6 *Soit t un i-terme tel que $t(\epsilon) \notin \mathcal{SF}$. Si*

- *tout problème de décision de E_i-filtrage*

$$\forall \mathcal{V}(r) : (\exists x_1, \ldots, x_m : f(x_1, \ldots, x_m) =^?_{E_i} r) \qquad f \in \mathcal{SF}$$

  est décidable.
- *les sous-termes étrangers de t sont en forme réduite par couches,*
- *la E_1-égalité et E_2-égalité sont décidables,*

alors il est possible de décider de l'existence d'un terme u changeant la tête de t dans E_i et d'en exhiber un lorsqu'il existe.

Preuve : Par hypothèse, l'égalité entre sous-termes étrangers de t est décidable et l'on sait donc décider de l'existence d'un $u \in \mathcal{V}_i(t) \cup AST(t)$ tel que $t^{\pi_i} =_{E_i} u^{\pi_i}$.

D'autre part, il existe un terme u changeant la tête de t si et seulement si il existe $f \in \mathcal{SF}$ et une substitution σ tels que

$$t^{\pi_i} =_{E_i} f(x_1\sigma, \ldots, x_m\sigma),$$

où $x_1, \ldots, x_m$ sont des variables n'apparaissant pas dans t^{π_i}. Par hypothèse, on sait d'abord décider

$$\forall \mathcal{V}(r) : (\exists x_1, \ldots, x_m : f(x_1, \ldots, x_m) =^?_{E_i} t^{\pi_i}) \qquad f \in \mathcal{SF}$$

puis obtenir une solution σ pour les variables $x_1, \ldots, x_m$, s'il en existe une, en utilisant un algorithme de semi-décision (à la Gallier et Snyder [GS89]). Le terme $f(x_1\sigma, \ldots, x_m\sigma)$ change la tête de t. La construction de t^{π_i} est là encore décidable car les sous-termes étrangers de t sont supposés être en forme réduite par couches. □

Proposition 4.7 *Soit t un i-terme tel que $t(\epsilon) \notin \mathcal{SF}$. Si t n'est pas en forme réduite par couches et les sous-termes étrangers de t sont en forme réduite par couches alors il existe un terme u changeant la tête de t dans E_i.*

Preuve : On a $t^{\pi_i} =_{E_i} (t \downarrow_R)^{\pi_i}$ d'après la preuve de la proposition 4.5. Posons $u = t \downarrow_R$. Par hypothèse, $u(\epsilon) \in \mathcal{SF}$ ou $u(\epsilon) \notin \mathcal{F}_i$.

Si $u(\epsilon) \in \mathcal{SF}$ alors u change la tête de t.

Si $u(\epsilon) \notin \mathcal{F}_i$ alors il existe nécessairement $s \in \mathcal{V}_i(t) \cup AST(t)$ tel que $u =_{E_1 \cup E_2} s$ et donc $t^{\pi_i} =_{E_i} s^{\pi_i}$ car E_i est consistante. □

Proposition 4.8 *Si t est un terme tel que $t(\epsilon) \in \mathcal{SF}$ et dont les sous-termes étrangers sont en forme réduite par couches, alors t est en forme réduite par couches.*

Preuve : Puisque tous les symboles partagés sont des constructeurs pour R, $t(\epsilon) = t \downarrow_R (\epsilon)$. □

La notation $t_i[s_1, \ldots, s_m]$ définie initialement dans le cas disjoint est reprise ici en prenant toutefois en compte le changement de définition d'un sous-terme étranger dans le cas non-disjoint. On note par exemple $f[a, a + a]$ le terme $f(a, a + a)$ si $f \in \mathcal{SF}$ et $\{a, +\} \notin \mathcal{SF}$. Une couche partagée est donc une couche à part entière.

Définition 4.7 *On note $t \Downarrow$ le terme défini comme suit:*

- *$c \Downarrow = c$ si c est une variable ou une constante.*
- *$(t_i[s_1, \ldots, s_m]) \Downarrow = t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ si $t_i(\epsilon) \in \mathcal{SF}$,*
- *$(t_i[s_1, \ldots, s_m]) \Downarrow = t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ s'il n'existe pas de terme changeant la tête de $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ dans E_i.*
- *$(t_i[s_1, \ldots, s_m]) \Downarrow = u$ si u est un terme changeant la tête de $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ dans E_i.*

Proposition 4.9 *$t \Downarrow$ est un terme en forme réduite par couches, E-égal à t, qui est calculable si la E_i-égalité et les problèmes de décision de E_i-filtrage*

$$\forall \mathcal{V}(r) : (\exists x_1, \ldots, x_m : f(x_1, \ldots, x_m) =^?_{E_i} r) \qquad f \in \mathcal{SF}$$

sont décidables pour $i = 1, 2$.

Preuve : On montre que $t \Downarrow$ est en forme réduite par couches par récurrence sur la hauteur de théories et d'après les propositions 4.7 et 4.8. Pour calculer un terme changeant la tête d'un terme, il suffit d'utiliser la proposition 4.6. □

Théorème 4.2 *Soit $E_1 \cup E_2$ une théorie décomposable. Le problème du mot dans $E_1 \cup E_2$ est décidable si le problème du mot dans E_i et les problèmes de décision de E_i-filtrage*

$$\forall \mathcal{V}(r) : (\exists x_1, \ldots, x_m : f(x_1, \ldots, x_m) =^?_{E_i} r) \qquad f \in \mathcal{SF}$$

sont décidables pour $i = 1, 2$.

Lorsque les symboles partagés $\mathcal{SF}$ sont uniquement des constantes, il suffit de savoir décider des problèmes $\forall \mathcal{V}(r) : c =^?_{E_i} r$ pour le calcul des formes réduites par couches. Ces problèmes sont trivialement des problèmes du mot dans E_i.

Corollaire 4.4 *Soit $E_1 \cup E_2$ une théorie décomposable partageant uniquement des constantes. Le problème du mot dans $E_1 \cup E_2$ est décidable si le problème du mot dans E_i est décidable pour $i = 1, 2$.*

Lorsque les symboles partagés ne peuvent pas apparaitre au "sommet", les problèmes de décision de E_i-filtrage, ayant justement pour objet de remonter un symbole partagé au sommet, n'ont pas de solution.

Corollaire 4.5 *Soit $E_1 \cup E_2$ une théorie décomposable en deux théories E_1 et E_2 faiblement partagées. Le problème du mot dans $E_1 \cup E_2$ est décidable si le problème du mot dans E_i est décidable pour $i = 1, 2$.*

4.2 Filtrage

Le problème de combinaison du filtrage consiste à combiner des algorithmes de filtrage dans deux théories équationnelles E_1 et E_2 afin d'obtenir un algorithme de filtrage dans l'union $E_1 \cup E_2$ des théories équationnelles.

Ce problème a déjà été abordé par T. Nipkow [Nip91] en imposant des restrictions syntaxiques sur la forme des égalités puisque seul le cas des théories régulières est traité avec succès. Pour le résoudre, il ne suffit pas de connecter un algorithme de filtrage à la place d'un algorithme d'unification dans l'algorithme de combinaison de l'unification. En cherchant à résoudre un problème de filtrage hétérogène à l'aide de l'algorithme de combinaison de l'unification, on risque en effet de considérer d'autres problèmes équationnels qui ne sont plus simplement des problèmes de filtrage, mais à nouveau des problèmes d'unification.

Nous allons présenter un algorithme de combinaison du filtrage, ou de décision du filtrage, qui s'inspire des techniques de combinaison introduites par F. Baader et K. Schulz [BS92] pour l'unification. Cet algorithme est complet pour une large classe de problèmes:

- Les conjonctions d'équations de filtrage $s \leq^? t$, ou filtre-équations dont les membres gauches s sont purs dans une théorie. On suppose alors l'existence dans chaque théorie d'un algorithme de filtrage avec restriction linéaire.

- Les problèmes de filtrage combinables qui sont grossièrement les problèmes équationnels pouvant être résolus sans l'aide d'un algorithme d'unification. Cette propriété est décidable pour un problème donné en supposant l'existence d'un algorithme pour résoudre dans chaque théorie, filtre-équations et équations résolues suivant une restriction linéaire.

- Tout problème de filtrage si les théories sont *partiellement linéaires*, une généralisation des théories linéaires incluant les théories régulières et non effondrantes. On pourra alors se contenter de la connaissance d'un algorithme de filtrage pour chaque théorie, la restriction linéaire étant superflue même pour la combinaison d'algorithmes de décision du filtrage.

Nous verrons aussi que le filtrage dans des théories non partiellement linéaires n'est pas un outil suffisamment puissant pour le problème de combinaison. Il est impossible de se passer dans ce contexte de l'unification.

Nous allons d'abord considérer le cas du mélange de théories équationnelles disjointes.

Un problème de filtrage est considéré ici comme un problème d'unification particulier [Bür90].

Définition 4.8 *Un* problème de E-filtrage *est un problème de E-unification avec constantes* (Γ, C) *tel que*

- $\Gamma = (\bigwedge_{k \in K} s_k =^? t_k)$,
- $\bigcup_{k \in K} \mathcal{V}(t_k) \subseteq C \subseteq \mathcal{V}(\Gamma)$.

On note un problème de E-filtrage sous la forme $\bigwedge_{k \in K} s_k \leq^? t_k$. L'ensemble C des variables skolémisées *est noté $\mathcal{GV}(\Gamma)$. Voici d'autres sous-ensembles de $\mathcal{V}(\Gamma)$ que nous distinguerons:*

- $\mathcal{RV}(\Gamma) = \bigcup_{k\in K} \mathcal{V}(t_k)$, *l'ensemble des variables skolémisées dans les membres droits.*
- $\overline{\mathcal{RV}}(\Gamma) = \mathcal{GV}(\Gamma)\backslash\mathcal{RV}(\Gamma)$, *l'ensemble des variables skolémisées qui ne sont pas dans les membres droits.*
- $\mathcal{CV}(\Gamma) = \{t_k \mid t_k \in \mathcal{X}, k \in K\}$, *l'ensemble des variables membres droits.*

Remarquons qu'un problème de E-unification avec constantes (Γ, C) ou de E-unification avec restriction constante (Γ, C, η) est par définition encore un problème de E-unification. C'est ce qui permet d'avoir directement une sémantique pour les solutions des problèmes $((\Gamma, C), C', \eta)$ de E-filtrage avec restriction constante.

4.2.1 Problème de filtrage étendu

La première étape de l'algorithme de combinaison consiste à purifier un problème de filtrage hétérogène par l'introduction de nouvelles variables et d'équations résolues. Les problèmes devant être résolus après purification sont donc des conjonctions de problèmes de filtrage et de formes résolues qu'on appelera problèmes de filtrage étendus.

Définition 4.9 *Un problème de* E-filtrage étendu *est un problème d'unification avec constantes*

$$(\Gamma \wedge \hat{\sigma}, C)$$

où (Γ, C) *est un problème de* E*-filtrage et* σ *une substitution telle que* $\mathcal{D}om(\sigma)\cap\mathcal{VR}an(\sigma) = \emptyset$ *et* $\mathcal{V}(\Gamma) \cap \mathcal{D}om(\sigma) = \emptyset$.

L'ensemble des variables résolues *d'un problème de filtrage étendu* $\Gamma\wedge\hat{\sigma}$ *est* $\mathcal{SV}(\Gamma\wedge\hat{\sigma}) = \mathcal{D}om(\sigma)$.

L'introduction de nouvelles variables par purification ne pose pas de difficulté dans les membres gauches des filtre-équations contenant déjà des variables. Par contre, celles introduites dans les membres droits est plus problématique car elles changent le statut des ces termes qui ne sont plus clos. L'étude spécifique de la purification des membres droits sera détaillée dans la prochaine section. On se limitera à une première purification "gauche".

Définition 4.10 *Soient* $i, j \in \{1, 2\}$ *et* $i \neq j$. *Une filtre-équation* $(s \leq^? t)$ *est* i-pure à gauche *si* s *est* i*-pur. Un problème de filtrage est* i-pur à gauche *s'il est formé de filtre-équations* i*-pures à gauche. Un problème de filtrage étendu* $\Gamma \wedge \hat{\sigma}$ *est* i-pur à gauche *si* Γ *est* i*-pur à gauche et* $\hat{\sigma}$ *est* i*-pur. Un problème de* (E_1, E_2)*-filtrage étendu est une conjonction de problèmes de filtrage étendu purs à gauche* $(\Gamma_1 \wedge \hat{\sigma}_1) \wedge (\Gamma_2 \wedge \hat{\sigma}_2)$ *telle que* $\mathcal{D}om(\sigma_1) \cap \mathcal{D}om(\sigma_2) = \emptyset$ *et* $\forall x \in \mathcal{D}om(\sigma_i)\ \forall y \in \mathcal{V}(x\sigma_i),\ x \notin \mathcal{V}(y\sigma_j)$ *pour* $i, j = 1, 2$ *et* $i \neq j$.

Nous venons de définir les problèmes obtenus après une première étape de purification.

Proposition 4.10 *Un problème de* $E_1 \cup E_2$*-filtrage est équivalent à un problème de* (E_1, E_2)*-filtrage étendu.*

Preuve : Le problème de (E_1, E_2)-filtrage étendu est obtenu par application répétée de la règle en figure 4.5. □

Les variables n'apparaissant résolues que dans une seule des deux théories, la conjonction des formes résolues $\hat{\sigma}_1 \wedge \hat{\sigma}_2$ est très spécifique. Nous verrons comment profiter de cette situation.

Abstraction Variable

$$\frac{\Gamma \wedge p(s,t)}{\exists x : \Gamma \wedge p^?(s[\omega \hookleftarrow x], t) \wedge x =^? s_{|\omega}}$$

où

- $\omega \in AlienPos(s)$,
- x est une nouvelle variable,
- $p \in \{=, \leq\}$.

Figure 4.5. Purification gauche pour le filtrage

4.2.2 Résolution dans une composante

Chaque filtre-équation est maintenant i-pure à gauche. Nous nous sommes contentés pour l'instant de purifier le membre gauche qui contenait déjà des variables. Par contre, l'introduction de variables dans le membre droit doit être évitée puisqu'elle change la nature du problème qui devient alors une équation. Une solution serait d'abstraire le membre droit puis de résoudre la filtre-équation i-pure ainsi obtenue. Comme pour le problème du mot, une abstraction aveugle peut s'avérer dangereuse.

Exemple 4.5 *Considérons $E_1 = \{x+x = x\}$ et E_2 la théorie vide avec un symbole unaire f. La filtre-équation $f(f(x)) \leq^?_E f(f(a) + f(a))$ est équivalente à $f(f(x)) \leq^?_E f(f(a))$. Mais la filtre-équation $f(f(x)) \leq^?_{E_2} f(C)$ obtenue par 2-abstraction du membre droit de $f(f(x)) \leq^?_E f(f(a) + f(a))$ n'a pas de solution alors que $f(f(x)) \leq^?_{E_2} f(f(a))$, filtre-équation équivalente à $f(f(x)) \leq^?_{E_2} f(f(a))$, a pour unique solution $\{x \mapsto a\}$.*

Le membre droit devrait être préalablement R-normalisé pour éviter de considérer comme étrangers des sous-termes qui ne le sont plus par remplacement d'égal par égal. A nouveau, nous utiliserons la normalisation opérationnelle fournie par la mise en forme réduite par couches introduite initialement pour le problème du mot.

Lemme 4.2 *Soient $(s \leq^? t)$ une filtre-équation i-pure à gauche, t un terme en forme réduite par couches et σ une substitution R-normalisée. Alors*

$$s\sigma =_E t \iff s\sigma^{\pi_i} =_{E_i} t^{\pi_i}.$$

Preuve : Puisque σ est R-normalisée, nous avons $(s\sigma)^{\pi_i} =_{E_i} ((s\sigma) \downarrow_R)^{\pi_i}$. De même, $t^{\pi_i} =_{E_i} (t \downarrow_R)^{\pi_i}$ car t et $t \downarrow_R$ sont en forme réduite par couches.

Si $s\sigma =_E t$ alors $(s\sigma) \downarrow_R = t \downarrow_R$ et $(s\sigma)^{\pi_i} =_{E_i} ((s\sigma) \downarrow_R)^{\pi_i} = (t \downarrow_R)^{\pi_i} =_{E_i} t^{\pi_i}$. Réciproquement, si $s\sigma^{\pi_i} =_{E_i} t^{\pi_i}$ alors $s\sigma = s\sigma^{\pi_i}\pi^{-1} =_{E_i} t^{\pi_i}\pi^{-1} =_E t$. □

Ce lemme signifie qu'une E-solution à une filtre-équation i-pure à gauche $s \leq^?_E t$ est l'instance d'une E_i-solution à la filtre-équation i-pure obtenue grâce à la réduction par couches du membre droit t puis i-abstraction de $t \Downarrow$. Le calcul de la forme réduite par

couches $t \Downarrow$ est rendu possible puisque le filtrage et donc le problème du mot est décidable dans chaque théorie. La i-abstraction de $t \Downarrow$ ne pose pas non plus de difficulté puisque l'égalité entre ses sous-termes étrangers (en f.r.c) est décidable.

En pratique, nous n'allons pas effectuer de i-abstraction mais plutôt purifier le membre droit en forme réduite par couches. Cela correspond à introduire une nouvelle variable et aussi l'équation entre cette variable et le terme qu'elle abstrait. Le fait que le terme abstrait soit clos et en forme réduite par couches implique que le choix de la théorie pour cette variable est complètement déterministe. Il est fixé par la tête du terme abstrait. Ainsi, une variable introduite par purification dans un membre droit en forme réduite par couches sera considérée comme une variable skolémisée dans la théorie où sera résolue la filtre-équation.

Définition 4.11 *Soit Γ_i le problème d'unification i-pur obtenu à partir d'un problème de filtrage étendu Γ par application de la règle de purification droite en figure 4.9.*

On note $\mathcal{GSV}(\Gamma_i)$ la réunion de $\mathcal{GV}(\Gamma)$ et de l'ensemble des variables introduites par purification droite et figurant comme membres d'une équation de Γ_i.

Il s'agit des variables skolémisées de Γ et des variables de Γ_i introduites par purification des membres droits dont la particularité est de s'instancier nécessairement dans E_i. Compte tenu de l'introduction de nouvelles variables, il faudra ensuite pouvoir les identifier comme cela a été fait dans la version non-déterministe (i.e. avec identifications) de l'algorithme de combinaison du problème du mot.

En vertu des lemmes 2.13 et 4.2, on peut conclure qu'un problème de filtrage étendu i-pur à gauche est résolue dans sa théorie de façon correcte et complète.

4.2.3 Combinaison des solutions

Nous avons vu comment ramener un problème de filtrage étendu pur à gauche à un problème de filtrage étendu pur. Il reste à déterminer comment résoudre une conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes de filtrage étendu, chacun pur dans sa théorie respective. Nous allons réutiliser la technique utilisée pour l'unification qui consiste à résoudre chaque problème suivant une même restriction rendant plus aisée la combinaison des solutions.

Cette technique de combinaison est valable pour toute conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes d'unification purs et en particulier une conjonction de problèmes de filtrage étendu. Pour ce faire, nous avons à résoudre des problèmes d'unification avec restriction linéaire $(\Gamma'_1, V_2, <)$ et $(\Gamma'_2, V_1, <)$, où $<$ est un ordre linéaire sur $V_1 \oplus V_2 = \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$.

La résolution d'un problème de filtrage étendu suivant une restriction linéaire peut s'effectuer par l'utilisation successive d'un algorithme de filtrage puis d'un algorithme d'élimination de constante.

Théorème 4.3 *Le filtrage étendu avec restriction est finitaire si et seulement si le filtrage et l'élimination de constante sont finitaires.*

Preuve : S'il existe un algorithme de filtrage étendu avec restriction alors il existe à fortiori un algorithme de filtrage et un autre algorithme pour tenir compte de la restriction dans une forme résolue, c'est-à-dire un algorithme d'élimination de constante.

Réciproquement, un algorithme de filtrage étendu avec restriction peut être construit en deux phases, une première pour la résolution du problème de filtrage et une seconde pour tenir compte de la restriction à l'aide d'un algorithme d'élimination de constante. On suit donc scrupuleusement la même construction que celle donnée dans [BS92] pour l'unification avec restriction. □

Le fait d'avoir à résoudre un problème de filtrage étendu suivant une certaine restriction linéaire ne change pas sa nature.

Proposition 4.11 *Si Γ est un problème de E-filtrage étendu alors $(\Gamma, C, <)$ est un problème de E-filtrage étendu avec restriction linéaire.*

Preuve : La skolémisation d'une variable x transforme une équation $x =^?_E t$ en $t \leq^?_E x$ si $x \in C \cap \mathcal{SV}(\Gamma)$. Dans les autres cas, une filtre-équation reste une filtre-équation et une équation résolue reste une équation résolue. □

Malheureusement, il faut encore tenir compte des problèmes de filtrage étendu identifiés. Nous avons délibérement repoussé l'application des identifications en fin de processus car c'est elle qui pose problème. Une identification doit être alors compatible avec la restriction linéaire déjà imposée. Les variables identifiées sont instanciées dans la même théorie et se comportent pareillement vis-à-vis de la restriction linéaire.

Définition 4.12 *Soit $<$ un ordre linéaire sur $V_1 \oplus V_2$. Une identification ξ dans $ID_{V_1 \oplus V_2}$ est* compatible *avec $<$ si*

- $\forall x, y \in \mathcal{D}om(\xi),\ x\xi = y\xi \Rightarrow x, y \in V_i$,
- $x < y,\ x\xi = x'\xi,\ y\xi = y'\xi \Rightarrow x' < y'$.

L'identification $\xi_{|V_i}$ est notée ξ_i.

Une identification peut facilement changer la nature d'un problème de filtrage étendu Γ puisqu'il suffit que deux variables de $\mathcal{SV}(\Gamma)$ soient identifiées pour engendrer un problème d'unification qui ne puisse pas toujours être résolu par filtrage.

Nous allons distinguer certains cas pour lesquels l'identification de variables n'engendre pas de problème d'unification.

Filtrage pur à gauche

L'identification de variables n'est pas gênante s'il n'y a pas d'équations introduites par purification de membres gauches, i.e. $\mathcal{SV}(\Gamma_1) = \mathcal{SV}(\Gamma_2) = \emptyset$.

Définition 4.13 *Une conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes de filtrage respectivement 1-pur et 2-pur à gauche est un problème de filtrage pur à gauche.*

Théorème 4.4 *Le $E_1 \cup E_2$-filtrage pur à gauche est décidable (resp. finitaire) si le E_i-filtrage pur à gauche avec restriction linéaire est décidable (resp. finitaire).*

Preuve : Le lemme 4.2 stipule que la résolution d'un problème de filtrage pur à gauche est équivalente à la résolution d'une conjonction $\Gamma_1 \wedge \Gamma_2$ de problèmes de filtrage purs par mise en forme réduite par couches et abstraction des membres droits. Un problème de filtrage pur identifié reste un problème de filtrage pur. La combinaison des solutions se fait ensuite par filtrage avec restriction linéaire. Les solutions combinées obtenues ne sont pas encore tout à fait celles du problème de filtrage pur à gauche initial. Les variables skolémisées introduites par i-abstraction doivent en effet être substituées par les sous-termes étrangers (en f.r.c) correspondants, en appliquant la substitution π^{-1} qui est l'inverse de l'abstraction par variables. □

Exemple 4.6 *Considérons $E_1 = A(+), E_2 = A(*)$ deux théories dont le filtrage est finitaire mais l'unification infinitaire. Pour résoudre le problème de $E_1 \cup E_2$-filtrage pur à gauche*

$$x + (y + z) \leq^? (a + b) + (a * b) \ \wedge \ z * a \leq^? (a * b) * a,$$

nous avons à considérer la conjonction de problèmes purs

$$(x + (y + z) \leq^? (a + b) + c) \ \wedge \ (z * a \leq^? (a * b) * a).$$

Les variables x, y seront instanciées dans la théorie E_1 et z dans la théorie E_2. La variable z est skolémisée dans E_1 et identifiée à c. La filtre-équation

$$x + (y + c) \leq^? (a + b) + c$$

est résolue dans E_1 et a pour solution $\{x \mapsto a, y \mapsto b\}$. La filtre-équation

$$z * a \leq^? (a * b) * a$$

est résolue dans E_2 et a pour solution $\{z \mapsto a * b\}$. En combinant ces deux solutions, on obtient finalement l'unique solution $\{x \mapsto a, y \mapsto b, z \mapsto a * b\}$. On remarquera que c n'apparait plus dans la solution et qu'il n'est donc même pas nécessaire de remplacer c par le terme abstrait $a * b$ dans la solution.

Il est important de noter que ce résultat est valable sans condition sur les théories (signatures disjointes mis à part). Nous allons désormais nous intéresser à des classes de théories qui se comportent bien par rapport à l'identification de variables.

Théories régulières non effondrantes

La phase de purification (cf figure 4.6) peut être adaptée dans ce cas pour ne pas introduire seulement une équation résolue mais des filtre-équations. La raison en est qu'un sous-terme étranger apparaissant dans le membre gauche de $s\sigma =_{E_1 \cup E_2} t$ est $E_1 \cup E_2$-égal à un sous-terme étranger de t qui est donc clos. L'adaptation a donc pour objet d'"identifier" directement un sous-terme étranger du membre gauche à un autre du membre droit. On obtient ainsi des problèmes de filtrage pur à gauche à la place de problèmes de filtrage étendu. Le théorème suivant est donc une conséquence du précédent.

Théorème 4.5 *Soient E_1 et E_2 deux théories régulières non effondrantes. Le $E_1 \cup E_2$-filtrage est décidable (resp. finitaire) si le E_i-filtrage est décidable (resp. finitaire).*

Abstraction Variable(RNE)

$$\frac{\Gamma \wedge s \leq^? t}{\bigvee_{(\omega,\omega')\in AlienPos(s)\times AlienPos(t)} \exists x : \Gamma \wedge s[\omega \hookleftarrow x] \leq^? t \wedge s_{|\omega} \leq^? t_{|\omega'} \wedge x \leq^? t_{|\omega'}}$$

si $s \notin T(\mathcal{F}_i, \mathcal{X})$, $s(\epsilon), t(\epsilon) \in \mathcal{F}_i$ et x une nouvelle variable.

Conflit

$$\frac{\Gamma \wedge s \leq^? t \Downarrow}{\bot}$$

si $s(\epsilon) \in \mathcal{F}_i, t(\epsilon) \notin \mathcal{F}_i$.

Figure 4.6. Purification dans les théories régulières non effondrantes

Exemple 4.7 *Considérons $E_1 = A(+), E_2 = C(\star)$ et la filtre-équation*

$$(y + z) + (x \star a) \leq^? a + (b + (a \star b)).$$

Par application de la règle Abstraction Variable(RNE)*, nous obtenons le problème de filtrage*

$$(y + z) + v \leq^? a + (b + (a \star b)) \ \wedge \ v \leq^? a \star b \ \wedge \ x \star a \leq^? a \star b.$$

Pour le problème de filtrage 1*-pur à gauche , on résoud d'abord*

$$(y + z) + v \leq^? a + (b + c) \ \wedge \ v \leq^? c$$

puis on remplace c par le terme abstrait $a \star b$ dans la solution. On obtient alors

$$\{y \mapsto a, z \mapsto b, v \mapsto a \star b\}.$$

La résolution du problème de filtrage 2*-pur donne $\{x \mapsto b\}$. En combinant les solutions, on a finalement la solution attendue*

$$\{x \mapsto b, y \mapsto a, z \mapsto b\}.$$

La variable v est supposée existentiellement quantifiée, elle n'apparait donc pas dans la solution.

On peut remarquer qu'il n'est même pas nécessaire de connaitre des algorithmes de E_i-filtrage avec restriction linéaire parce que le filtrage dans les théories régulières a la particularité de retourner des solutions closes et ne peut donc pas introduire de cycle composé nécessitant l'introduction d'une restriction linéaire.

Lemme 4.3 *Si E est une théorie régulière alors la restriction aux variables du problème d'une solution à un problème de E-filtrage est close.*

Preuve : Soit $s \leq^?_E t$ une filtre-équation et σ une solution telles que $s\sigma =_E t$. Puisque E est régulière, on a $\mathcal{V}(s\sigma) = \mathcal{V}(t)$ et donc $\mathcal{VR}an(\sigma_{|\mathcal{V}(s)}) \subseteq \mathcal{V}(t)$, où $\mathcal{V}(t)$ est un ensemble de variables skolémisées. □

Filtre-Equation(Non Eff)

$$\frac{\Gamma \wedge s \leq^? t}{\bigvee_{\omega' \in \Omega'} \exists x : \Gamma \wedge s[\omega \hookleftarrow x] \leq^? t \Downarrow \wedge s_{|\omega} \leq^? (t \Downarrow)_{|\omega'} \wedge x \leq^? (t \Downarrow)_{|\omega'}}$$

si

- $s \notin T(\mathcal{F}_i, \mathcal{X})$, $s(\epsilon) \in \mathcal{F}_i$, $t \Downarrow (\epsilon) \notin \mathcal{X}$,
- $\omega \in AlienPos(s)$,
- x est une nouvelle variable,
- $\Omega' = \{\epsilon\}$ si $t \Downarrow (\epsilon) \notin \mathcal{F}_i$, sinon $\Omega' = AlienPos(t \Downarrow)$.

Filtre-Equation(Eff)

$$\frac{\Gamma \wedge s \leq^? t}{\exists x : \Gamma \wedge s[\omega \hookleftarrow x] \leq^? t \wedge t_{|\omega} =^? x} \quad \textbf{si} \begin{cases} \omega \in AlienPos(s) \\ x \text{ est une nouvelle variable} \end{cases}$$

Equation(Non Eff)

$$\frac{\Gamma \wedge p^?(u[x]_{\omega'}, t) \wedge s =^? x}{\Gamma \wedge (\bigwedge_{\omega \in \Omega} p^?(u[\omega' \hookleftarrow s_{|\omega}], t)) \wedge s[\omega \hookleftarrow x]_{\omega \in \Omega} =^? x} \quad \textbf{si } \Omega \subseteq AlienPos(s) \text{ et } p \in \{=, \leq\}$$

Equation(Eff)

$$\frac{\Gamma \wedge s =^? x}{\exists y : \Gamma \wedge s[\omega \hookleftarrow y] =^? x \wedge s_{|\omega} =^? y} \quad \textbf{si} \begin{cases} \omega \in AlienPos(s) \\ y \text{ est une nouvelle variable} \end{cases}$$

Figure 4.7. Purification dans les théories régulières

Théories régulières

Une variable introduite par purification d'une filtre-équation peut maintenant s'instancier dans deux théories:

- celle où la variable apparait dans un membre gauche d'une filtre-équation. Par filtrage, cette variable sera égale à un terme clos. Dans l'autre théorie, elle figure comme membre d'une équation résolue et sera skolémisée. La purification de l'autre membre de cette équation permet ensuite soit d'"effacer" une couche, soit d'engendrer une nouvelle équation résolue (cf. figure 4.7).
- celle où la variable apparait résolue dans une équation. Comme précédemment dans les théories non effondrantes, cette équation est en fait une filtre-équation résolue.

L'identification de variables non skolémisées n'a pas à être imposée puisqu'il suffit de tester l'égalité entre les termes clos associés qui sont obtenus soit par filtrage, soit par la purification décrite en figure 4.7.

Le résultat suivant avait déjà été montré par T. Nipkow [Nip91].

Théorème 4.6 *Si E_1 et E_2 sont deux théories régulières, alors le $E_1 \cup E_2$-filtrage est finitaire si et seulement si le E_i-filtrage est finitaire.*

Les algorithmes que nous proposons sont basés sur les propositions suivantes:

Proposition 4.12 *Soit Γ un problème de filtrage. L'ensemble des substitutions*

$$\xi_1\xi_2(\sigma_1 \odot \sigma_2)$$

tel que

- $\Gamma_1 \wedge \Gamma_2$ *est un problème de filtrage étendu pur à gauche obtenu en appliquant tant que possible les règles de la figure 4.7 avec en entrée* Γ,
- $V_2 \supseteq \mathcal{SV}(\Gamma_1)$, $V_1 \supseteq \mathcal{SV}(\Gamma_2)$,
- $V_1 \oplus V_2 = \mathcal{V}(\Gamma_1 \wedge \Gamma_2)$,
- $\xi_1 \in ID_{V_1}, \xi_2 \in ID_{V_2}$ *avec* $(\mathcal{D}om(\xi_1) \cup \mathcal{D}om(\xi_2)) \cap \mathcal{GV}(\Gamma) = \emptyset$,
- $\sigma_1 \in CSU_{E_1}(\Gamma_1\xi_2, V_2)$, $\sigma_2 \in CSU_{E_2}(\Gamma_2\xi_1, V_1)$,
- $\forall i \in \{1,2\},\ \forall x, y \in V_i,\ x\sigma_i =_{E_i} y\sigma_i \Leftrightarrow x\xi_i = y\xi_i$,

forme un ensemble complet de solutions de Γ.

Preuve : Remarquons tout d'abord que les problèmes $(\Gamma_1\xi_2, V_2)$ et $(\Gamma_2\xi_1, V_1)$ sont des problèmes de filtrage d'après les conditions imposées sur V_1 et V_2. Le dernier point est nécessaire parce que l'identification de variables n'est effectuée que dans la théorie où elles sont skolémisées alors qu'elle devrait l'être dans les deux théories. Les solutions σ_1 et σ_2 ne peuvent donc être combinées directement. Il faut au préalable déterminer puis combiner les instances ϕ_i de σ_i telles que

$$\forall x, y \in V_i, x \neq y,\ x\phi_i =_{E_i} y\phi_i \text{ si } x\xi_i = y\xi_i.$$

Mais comme E_1 et E_2 sont des théories régulières, les substitutions σ_1 et σ_2 sont closes. Les seules instances modulo respectivement E_1 et E_2 sont donc les substitutions σ_1 et σ_2. Ainsi, pour que σ_1 et σ_2 puissent être combinées, il faut et il suffit que

$$\forall x, y \in V_i, x \neq y,\ x\sigma_i =_{E_i} y\sigma_i \text{ si } x\xi_i = y\xi_i,$$

où le test de E_i-égalité est décidable puisque l'on dispose d'un algorithme de E_i-filtrage. On notera que les variables de $\mathcal{GV}(\Gamma)$ ne sont pas à identifier (leurs identifications mènent à un échec dans la théorie vide). □

Exemple 4.8 *Considérons $E_1 = A(+), E_2 = CI(*)$ et la filtre-équation*

$$(x * y) + (a + b) \leq^? (a + a) + b.$$

La solution $\{x \mapsto a, y \mapsto a\}$ s'obtient en considérant

- $v + (a + b) \leq^? (a + a) + b \ \wedge\ v =^? x * y$,
- $V_1 = \{v, a, b\}$, $V_2 = \{x, y\}$,

PurifGauche
$$\frac{\Gamma \wedge s \leq^? t}{\Gamma \wedge s[\omega \hookleftarrow x] \leq^? t \wedge x =^? s_{|\omega}} \quad \textbf{si } \omega \in AlienPos(s) \text{ et } x \text{ est une nouvelle variable}$$

Fusion
$$\frac{\Gamma \wedge x \leq^? t \wedge x =^? s}{\Gamma \wedge x \leq^? t \wedge s \leq^? t}$$

FiltreRes
$$\frac{\Gamma \wedge s \leq^? t}{\Gamma \wedge \bigwedge_{k \in K} x_k \leq^? t_k} \quad \textbf{si } s \in \mathcal{T}(\mathcal{F}_i, \mathcal{X}) \backslash \mathcal{X}, \{x_k \mapsto t_k\}_{k \in K} \in CSU_{E_i}(s \leq^? t \Downarrow)$$

Suppression
$$\frac{\Gamma \wedge x \leq^? t \wedge x \leq^? t'}{\Gamma \wedge x \leq^? t} \quad \textbf{si } t =_E t'$$

Echec
$$\frac{\Gamma \wedge x \leq^? t \wedge x \leq^? t'}{\bot} \quad \textbf{si } t \neq_E t'$$

Figure 4.8. Ensemble de règles $\mathcal{RMR}$ pour le filtrage dans le mélange de théories régulières

- $\xi_1 = \{v \mapsto a\}$, $\xi_2 = \{\}$,
- $\sigma_1 = \{\} = CSU_{E_1}(a+(a+b) \leq^? (a+a)+b)$, $\sigma_2 = \{x \mapsto a, y \mapsto a\} = CSU_{E_2}(x*y \leq^? a)$.

Les solutions σ_1 et σ_2 peuvent être dans ce cas particulier trivialement combinées.

Algorithme de filtrage déterministe

Après avoir vu un algorithme non déterministe même dans sa phase de purification, nous allons présenter un algorithme déterministe sous la forme de règles de transformation données en figure 4.8. La différence avec l'approche précédente est de ne pas se préocupper de l'introduction d'une équation résolue $x =^? s$. Dans les théories régulières, cette variable x apparaissant par ailleurs dans une filtre-équation sera forcément associée à un terme clos. La résolution d'une filtre-équation engendre en effet une conjonction de filtre-équations résolues, par exemple $x \leq^? t$. C'est ce qui permet ensuite de transformer $x =^? s$ et $x \leq^? t$ en $s \leq^? t$ et $x \leq^? t$.

(**Forme Résolue**) On peut facilement vérifier que les formes normales par rapport à $\mathcal{RMR}$ sont les conjonctions de filtre-équations résolues

$$\bigwedge_{k \in K} x_k \leq^? t_k$$

ou $\bot$ ou $\top$. Supposons que Γ n'est pas une conjonction de filtre-équations résolues. S'il existe une équation $x =^? s$, soit $x \leq^? t$ est une filtre-équation de Γ et Fusion s'applique, soit il existe dans Γ une filtre-équation $s[x] \leq^? t$ ayant une occurrence de x dans le membre gauche non variable et PurifGauche ou FiltreRes s'applique. Sinon, s'il existe une filtre-équation non encore résolue, alors PurifGauche ou FiltreRes s'applique. Dans le cas où toutes les filtre-équations sont résolues, il existe nécessairement deux filtre-équations $x \leq^? t$ et $x \leq^? t'$ dans Γ et Suppression ou Echec s'applique.

Par conséquent, une règle de transformation de $\mathcal{RMR}$ peut toujours être appliquée à Γ.

Proposition 4.13 *(***Terminaison***) Le système de règles $\mathcal{RMR}$ termine pour tout contrôle et retourne une forme résolue équivalente au problème de filtrage initial.*

Preuve : Pour tout Γ, on considère les mesures de complexité suivantes:

- TH_{mul} est le multi-ensemble des hauteurs de théories des sous-termes étrangers des termes de Γ.
- NEQ est le nombre d'équations dans Γ.
- NNV est le nombre de filtre-équations de Γ qui n'ont pas une variable en membre gauche.
- $NMEQ$ est le nombre de filtre-équations de Γ.

Ces mesures de complexité sont comparées de façon lexicographique. La situation est décrite ci-dessous.

regles	TH_{mul}	NEQ	NNV	$NMEQ$
PurifGauche	$\downarrow$			
Fusion	$=$	$\downarrow$		
FiltreRes	$=$	$=$	$\downarrow$	
Suppression	$=$	$=$	$=$	$\downarrow$
Echec	$=$	$=$	$=$	$\downarrow$

□

On ne peut pas avoir de résultat de décision du filtrage suivant la même approche.

Le principe développé dans la prochaine section généralise le cas des théories régulières et non effondrantes dans le sens où ne seront identifiées deux variables que si l'une est égale à un terme clos. L'identification de variables n'implique pas alors l'unification de sous-termes étrangers mais le filtrage sur un terme clos.

Les théories linéaires semblent par exemple vérifier ce principe car une variable d'un membre gauche d'une égalité ne doit être identifiée qu'à une variable du membre droit.

La combinaison de théories régulières avec des théories linéaires pose pourtant problème comme le montre l'exemple suivant inspiré de [Nip91].

Exemple 4.9 *Considérons trois théories $E_1 = \{f(x) = a\}$, $E_2 = \{g(x, x) = x\}$ et*

$$E_3 = DA = \begin{cases} x + (y + z) &= (x + y) + z \\ x * (y + z) &= x * y + x * z \\ (x + y) * z &= x * z + y * z \end{cases}$$

La théorie E_1 est linéaire et les théories E_2, E_3 sont régulières. Le filtrage est décidable dans chacune des théories:

- *la E_1-unification avec constantes est en effet trivialement décidable,*
- *la E_2-unification est considérée par exemple dans [Her87],*
- *la théorie E_3 a la particularité d'être décidable [Sza82] pour le filtrage et indécidable pour l'unification.*

Nous allons montrer que le $E_1 \cup E_2 \cup E_3$-filtrage est indécidable en utilisant le fait que la E_3-unification est indécidable.
Soient $s, t \in \mathcal{T}(\{+, *\}, \mathcal{X})$. La filtre-équation

$$f(g(f(s), f(t)) \leq^? a$$

est unifiable dans $E_1 \cup E_2 \cup E_3$ si et seulement si

$$\begin{array}{lcl} & g(f(s), f(t)) =^? f(x) & \textit{est unifiable} \\ \Leftrightarrow & f(s) =^? f(t) & \textit{est unifiable} \\ \Leftrightarrow & s =^? t & \textit{est unifiable} \end{array}$$

L'équation $s =^? t$ est unifiable dans $E_1 \cup E_2 \cup E_3$ si et seulement si $s =^? t$ est unifiable dans $E_3 = DA$. Or la DA-unification est indécidable. Par conséquent, le $E_1 \cup E_2 \cup E_3$-filtrage est indécidable alors que le filtrage est décidable dans les théories E_1, E_2 et E_3.

Nous verrons comment combiner le filtrage de théories linéaires et de théories régulières lorsque ces dernières sont non effondrantes, ce qui n'est pas le cas de l'exemple précédent à cause de E_2.

4.2.4 Problème de filtrage combinable

Les variables skolémisées apparaissant dans les membres droits et celles n'apparaissant que dans les membres gauches sont sensiblement différentes. Les premières sont liées à des termes clos, les secondes sont éventuellement égales dans l'autre théorie à des termes non clos. L'identification des secondes entre elles est très fâcheuse.

L'idée que nous allons développer dans cette section consiste à déterminer précisément les théories pour lesquelles les identifications posant problème sont superflues. Le problème provient de l'identification de variables skolémisées dans une théorie mais instanciées dans l'autre. Les variables skolémisées dans un membre gauche d'une filtre-équation d'une théorie peuvent apparaître résolues ailleurs. L'identification de ces variables entre elles engendre donc un problème d'unification dans l'autre théorie. De même, l'identification de ces variables avec un membre droit est elle aussi délicate puisque ce membre droit peut apparaitre résolu dans une autre théorie. Ces remarques conduisent à la définition suivante:

Définition 4.14 *Un problème de E-filtrage étendu Γ suivant une restriction linéaire $<$ est* combinable *si pour toute identification ξ telle que*

1. *soit $\xi \in ID_{\overline{\mathcal{RV}}(\Gamma)}$,*

2. *soit* $\xi \in ID^{\mathcal{CV}(\Gamma)}_{\overline{\mathcal{RV}}(\Gamma)}$ *et aucun membre gauche d'une filtre-équation* $l\xi \leq^? x$ *de* $\Gamma\xi$ *n'a une unique occurrence de* x *si* $x \notin \mathcal{V}(l)$,

on a $SU^{\leq}_E(\Gamma\xi) = SU^{\leq}_E(\Gamma)\xi$. *La théorie équationnelle* E *est* à filtrage combinable *si tout problème de* E*-filtrage étendu est combinable.*

L'intérêt de rejeter les identifications à la fin du calcul des solutions est finalement de complètement pouvoir s'en débarrasser durant la combinaison des solutions.

L'objectif du prochain paragraphe est de caractériser les théories à filtrage combinable.

Théories partiellement linéaires

La définition que nous donnons est volontairement abstraite afin de recouvrir l'ensemble des théories à filtrage combinable. Il s'agit intuitivement des théories pour lesquelles il est possible d'éliminer une variable d'un terme ou d'effondrer un terme sur une variable sans avoir à pratiquer d'identification. Nous verrons que de nombreuses théories connues vérifient les propriétés suivantes sur l'égalité.

Définition 4.15 *Un terme* r *élimine un ensemble* V *de* l *si* $\forall x \in V,\ x \in \mathcal{V}(l)$ *et* $x \notin \mathcal{V}(r)$.

Une théorie E *est* partiellement linéaire *si pour tout terme linéaire* l *et toute identification* ξ *sur* $\mathcal{V}(l)$, *on a*

- *pour tout terme* r *tel que* $l\xi =_E r$ *et* r *élimine* $V\xi$ *de* $l\xi$ *il existe* d *tel que* $l =_E d$ *et* d *élimine* V *de* l,
- $l\xi =_E x$ *implique* $l =_E x$ *si* $x\xi = x$.

Les théories linéaires sont partiellement linéaires.

Les théories régulières sont partiellement linéaires si elles sont non effondrantes, auquel cas le second point de la définition est toujours satisfait. Il existe en pratique peu d'axiomes $l = x$ effondrants satisfaisant ce second point. On peut citer par exemple les axiomes de projection $f(x_1, \ldots, x_m) = x_i$ avec $i \in \{1, \ldots, m\}$ et $x + 0 = x$.

La théorie engendrée par le seul axiome effondrant $x + x = x$ est régulière mais n'est pas partiellement linéaire puisque $x + x = x$ ne peut être linéarisée, c'est-à-dire qu'il n'existe pas d'axiome $x + y = x$ ou $y + x = x$. Pour la même raison, la théorie booléenne n'est pas non plus partiellement linéaire.

Il existe cependant des théories partiellement linéaires qui ne sont ni régulières ni linéaires.

Exemple 4.10 *Nous allons exhiber une théorie partiellement linéaire pour laquelle le filtrage étendu est finitaire.*

Soit DAZ *la théorie équationnelle présentée par les axiomes* DA *et* Z*:*

$$DA = \left\{ \begin{array}{lcl} x + (y + z) & = & (x + y) + z \\ x * (y + z) & = & x * y + x * z \\ (x + y) * z & = & x * z + y * z \end{array} \right.$$

$$Z = \left\{ \begin{array}{lcl} x + 0 & = & 0 \\ x * 0 & = & 0 \end{array} \right.$$

Si $l\xi$ est un terme contenant le symbole 0 alors l le contient aussi et $l =_Z 0$. Sinon $l\xi$ est toujours égal modulo DA à un terme ayant les mêmes variables, auncune variable ne pouvant être éliminée ou effondrée. La théorie DAZ qui n'est ni régulière à cause de Z ni linéaire à cause de l'axiome de distributivité est donc partiellement linéaire.

On s'intéresse à présent à l'existence d'un algorithme de DAZ-filtrage étendu avec restriction. Le DAZ-filtrage est finitaire car le DA-filtrage est finitaire [Sza82]. Le lecteur pourra en effet facilement vérifier que

- $CSU_{DAZ}(s \leq^? t)$ *est vide si* s *contient* 0 *et* t *ne contient pas* 0.
- $CSU_{DAZ}(s \leq^? t) = CSU_{DAZ}(s =^? 0) = \bigcup_{x \in \mathcal{V}(s)} \{x \mapsto 0\}$ *si* t *contient* 0.
- $CSU_{DAZ}(s \leq^? t) = CSU_{DA}(s \leq^? t)$ *si* s *et* t *ne contiennent pas* 0.

La DAZ-élimination de constante est elle aussi finitaire car équivalente au DAZ-filtrage sur 0. *Par conséquent le DAZ-filtrage étendu avec restriction est encore finitaire alors que la DAZ-unification est infinitaire puisque la DA-unification est infinitaire.*

Le lemme suivant confirme la possibilité de tester la propriété de *filtrage combinable* sur des ensembles complets de solutions.

Lemme 4.4 *Si ξ est une identification sur des variables skolémisées compatible avec une restriction linéaire $<$ alors un $CSU_E^<(\Gamma)\xi$ est un $CSU_E^<(\Gamma\xi)$ si et seulement si $SU_E^<(\Gamma\xi) = SU_E^<(\Gamma)\xi$.*

Preuve : ($\Leftarrow$) Pour la correction, on a $CSU_E^<(\Gamma)\xi \subseteq SU_E^<(\Gamma)\xi = SU_E^<(\Gamma\xi)$ et pour la complétude,

$$\forall \psi \in SU_E^<(\Gamma\xi) = SU_E^<(\Gamma)\xi \exists \sigma \in CSU_E^<(\Gamma)\xi,\ \sigma \leq_E^{\mathcal{V}(\Gamma)} \psi.$$

Donc $CSU_E^<(\Gamma)\xi$ est un $CSU_E^<(\Gamma\xi)$.

($\Rightarrow$) Par hypothèse,

$$\forall \psi \in SU_E^<(\Gamma\xi) \exists \sigma \in CSU_E^<(\Gamma)\xi,\ \sigma \leq_E^{\mathcal{V}(\Gamma)} \psi.$$

Pour toute substitution ϕ satisfaisant la restriction linéaire $<$ telle que $\sigma \leq_E^{\mathcal{V}(\Gamma)} \phi$ et $\sigma \in CSU_E^<(\Gamma)\xi$, on a $\phi \in SU_E^<(\Gamma)\xi$. Par conséquent, $SU_E^<(\Gamma\xi) \subseteq SU_E^<(\Gamma)\xi$. L'inclusion $SU_E^<(\Gamma)\xi \subseteq SU_E^<(\Gamma\xi)$ est toujours vérifiée ce qui permet d'en conclure que $SU_E^<(\Gamma)\xi = SU_E^<(\Gamma\xi)$. □

Proposition 4.14 *Une théorie équationnelle est partiellement linéaire si et seulement si elle est à filtrage combinable.*

Preuve : ($\Leftarrow$) Pour tout terme linéaire l dont les variables $\mathcal{V}(l)$ sont skolémisées, il suffit de considérer soit $x =^? l$ suivant la restriction linéaire $<$ sur $V \cup C$ avec $V = \{x\}$ et $C = \mathcal{V}(l)$, soit l'équation close $x =^? l$ avec $x \in \mathcal{V}(l)$.

($\Rightarrow$) Il suffit de prouver que

$$\forall \psi \in SU_E^<(\Gamma\xi) \exists \phi \in CSU_E^<(\Gamma), \phi\xi \leq_E^{V(\Gamma)} \psi$$

puisque $CSU_E^<(\Gamma)\xi \subseteq SU_E^<(\Gamma\xi)$. Soit ψ une substitution telle que $(\psi\xi) \in SU_E^<(\Gamma\xi)$. Un problème de filtrage étendu Γ est composé de filtre-équations et d'équations.

1. Soit $s \leq^? t$ une filtre-équation de Γ. Comme $\mathcal{D}om(\xi) \subseteq \overline{\mathcal{RV}}(\Gamma)$, on a $s(\psi\xi) =_E t\xi = t$.

   Si t est de hauteur non nulle alors toutes les variables skolémisées de $s\psi$ identifiées par ξ sont éliminées par t. La définition 4.15 implique qu'il existe u tel que $s\psi =_E u$ avec $u =_E u\xi =_E (s\psi)\xi =_E t$.

   Si t est une variable skolémisée x alors $s(\psi\xi) = (s\psi)\xi =_E x$. D'après la définition 4.15, on a $(s\psi) =_E x$.

2. Soit $x =^? t$ une équation résolue de Γ et $x(\psi\xi) =_E t(\psi\xi)$. Si pour tenir compte de la restriction linéaire $<$, $x(\psi\xi)$ élimine une variable skolémisée de $t(\psi\xi)$ alors la définition 4.15 implique l'existence d'une substitution ϕ telle que $x\phi =_E t\phi$ avec $\phi\xi = (\psi\xi)$ sur $\mathcal{V}(t)$ et $(x\phi)\xi =_E t\phi\xi =_E t(\psi\xi) =_E x(\psi\xi)$. Sinon $\{x \mapsto t\}$ est une solution de $x =^? t$ suivant la restriction linéaire $<$ avec $\{x \mapsto t\}\xi \leq_E^{\{x\}} (\psi\xi)$. Le terme solution associé à x ne peut être remis en cause par la considération d'autres équations car x n'apparait résolue qu'une fois dans Γ.

□

Modularité de la partielle linéarité

Pour qu'une approche modulaire de la construction d'algorithmes de filtrage (étendu) ait un intérêt, il faut d'abord s'assurer de la modularité de la partielle linéarité.

La modularité de cette propriété se démontre en projetant toute $E_1 \cup E_2$-égalité sur une égalité dans E_i, théorie supposée partiellement linéaire. Avant de projeter une égalité hétérogène dans une seule composante, il faut que les termes soient mis en forme réduite par couches. On montre dans un premier temps que la forme réduite par couches d'un terme identifié est égal à l'identification de sa forme réduite par couches et ce grâce à la linéarité des axiomes effondrants de chaque théorie.

Lemme 4.5 *Si E_1 et E_2 sont deux théories partiellement linéaires, t un terme linéaire et ξ une identification de $ID_{\mathcal{V}(t)}$ alors*

$$(t\xi) \Downarrow = (t \Downarrow)\xi.$$

Preuve : Par récurrence sur la hauteur de théories.

- Si t est i-pur alors, par hypothèse $t\xi =_{E_i} x$ si et seulement si $t =_{E_i} x$ (avec $x \notin \mathcal{D}om(\xi)$) et donc $(t\xi) \Downarrow = (t \Downarrow)\xi$.
- Supposons le lemme vrai pour les sous-termes étrangers. Alors

  $$t_i[(s_1\xi) \Downarrow, \ldots, (s_m\xi) \Downarrow] = t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi.$$

  Il s'agit de montrer que $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi$ s'effondre sur $u\xi$ si et seulement si $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ s'effondre sur u.
  Si $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi$ s'effondre sur un sous terme $u\xi$ alors par définition

  $$(t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi)^{\pi_i} =_{E_i} (u\xi)^{\pi_i}.$$

Il existe donc une identification ξ_i telle que

$$t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]^{\pi_i}\xi_i =_{E_i} u^{\pi_i}$$

puis

$$t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]^{\pi_i} =_{E_i} u^{\pi_i}$$

car E_i est partiellement linéaire. Par conséquent $(t\xi) \Downarrow = (t \Downarrow)\xi$ s'il existe un sous-terme effondrant.

Réciproquement, s'il existe un sous-terme u effondrant pour $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ alors par définition

$$t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]^{\pi_i} =_{E_i} u^{\pi_i}$$

et $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow] =_E u$ avec u sous-terme de $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$. En identifiant cette égalité par ξ, on obtient

$$t_i[(s_1\xi) \Downarrow, \ldots, (s_m\xi) \Downarrow] = t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi =_E u\xi.$$

Puis finalement

$$(t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi)^{\pi_i} =_{E_i} (u\xi)^{\pi_i}$$

car les sous-termes étrangers sont en forme réduite par couches. Par contraposée, si $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]\xi$ ne s'effondre pas alors $t_i[s_1 \Downarrow, \ldots, s_m \Downarrow]$ ne s'effondre pas non plus et donc $(t\xi) \Downarrow = (t_i[s_1 \Downarrow, \ldots, s_m \Downarrow])\xi = (t \Downarrow)\xi$.

□

Proposition 4.15 *Si E_1 et E_2 sont deux théories partiellement linéaires alors $E_1 \cup E_2$ est une théorie partiellement linéaire.*

Preuve : Considérons une égalité $l\xi =_E r$ telle que $(l\xi) \Downarrow$ et $r \Downarrow$ ont des symboles de tête dans $\mathcal{F}_i$. D'après le corollaire 4.1, on a

$$((l\xi) \Downarrow)^{\pi_i} =_{E_i} (r \Downarrow)^{\pi_i},$$

puis

$$((l \Downarrow)\xi)^{\pi_i} =_{E_i} (r \Downarrow)^{\pi_i}$$

d'après le lemme 4.5. Il existe donc une identification ξ_i sur $\mathcal{V}((l \Downarrow)^{\pi_i})$ telle que

$$((l \Downarrow)^{\pi_i})\xi_i =_{E_i} (r \Downarrow)^{\pi_i}$$

et $(r \Downarrow)^{\pi_i}$ élimine un ensemble de variables V_i faisant abstraction de termes avec des occurrences de variables de V:

$$\forall x \in V \exists x_i \in V_i,\ x \in x_i\pi^{-1}.$$

Puisque E_i est partiellement linéaire, il existe d tel que $((l \Downarrow)^{\pi_i}) =_{E_i} d$ et d élimine V_i. Ensuite $l =_E (l \Downarrow) =_E ((l \Downarrow)^{\pi_i})\pi^{-1} =_E d\pi^{-1}$ avec $V \cap \mathcal{V}(d\pi^{-1}) = \emptyset$ puisque $\mathcal{V}(d) \cap V_i = \emptyset$.

De la même façon, $l\xi =_E x$ implique successivement $((l\xi) \Downarrow)^{\pi_i} =_{E_i} x$, $((l \Downarrow)\xi)^{\pi_i} =_{E_i} x$ puis $(l \Downarrow)^{\pi_i}\xi_i =_{E_i} x$ et $(l \Downarrow)^{\pi_i} =_{E_i} x$ par hypothèse sur E_i. On reconstruit ensuite une E-égalité par application de π^{-1}, $l =_E ((l \Downarrow)^{\pi_i})\pi^{-1} =_E x\pi^{-1} = x$. □

Réciproquement, si l'une des théories E_1 ou E_2 n'est pas partiellement linéaire alors $E_1 \cup E_2$ ne l'est pas non plus puisque les égalités entre termes i-purs sont conservés par mélange: s'il existe une égalité i-pure rendant une théorie E_i non partiellement linéaire alors cette égalité fera de même pour l'union $E_1 \cup E_2$.

Filtrage étendu vs Unification

On s'intéresse davantage au filtrage étendu suivant une restriction linéaire qui est fortement lié à la résolution conjointe avec une forme résolue dans la théorie vide.

Définition 4.16 *Un problème de E-filtrage* librement étendu *est un problème équationnel*

$$\Gamma \wedge \hat{\sigma}_{\emptyset}$$

où Γ *est un problème de E-filtrage étendu et* $\sigma_{\emptyset}$ *une substitution formée de symboles libres telle que*

- $\mathcal{D}om(\sigma_{\emptyset}) \cap \mathcal{R}an(\sigma_{\emptyset}) = \emptyset$,
- $\forall x, y \in \mathcal{D}om(\sigma_{\emptyset}), x \neq y,\ CSU_{\emptyset}(x\sigma_{\emptyset} =^{?} y\sigma_{\emptyset}) = \emptyset$.

Une variable peut apparaitre résolue dans la théorie E et dans la théorie vide mais sera forcément instanciée dans la théorie vide qui est non effondrante. De plus, deux variables résolues et instanciées dans la théorie vide ne pourront être identifiées à cause de l'hypothèse faite sur la substitution $\sigma_{\emptyset}$.

Proposition 4.16 *Si le E-filtrage librement étendu est décidable (resp. finitaire) alors le E-filtrage étendu avec restriction linéaire est décidable (resp. finitaire).*

Preuve : Il suffit de prendre la substitution $\sigma_<$ obtenue à partir d'un restriction linéaire $<$ et qui permet de résoudre un problème d'unification avec restriction linéaire $(\Gamma, C, <)$ grâce au problème d'unification avec symboles libres $\Gamma \wedge \hat{\sigma}_<$. Dans le cas particulier où $\Gamma \wedge \hat{\sigma}_<$ est un problème de filtrage librement étendu, on est alors capable de résoudre un problème de filtrage étendu avec restriction linéaire $(\Gamma, C, <)$. □

Dans le cas où E n'est pas partiellement linéaire, la notion de filtrage librement étendu est trop forte car elle coincide avec l'unification en présence de symboles libres.

Proposition 4.17 *Si E n'est pas partiellement linéaire alors le* $E \cup \emptyset$*-filtrage librement étendu est équivalent à la* $E \cup \emptyset$*-unification.*

Preuve : On montre comment coder tout problème d'unification en un problème de filtrage librement étendu. L'apport des symboles libres permet de créer des sous-termes étrangers aux positions adéquates dans des termes ne pouvant être linéarisés. Ces sous-termes étrangers sont ensuite forcément abstraits par des variables skolémisées dans la théorie E. Lorsque E n'est pas partiellement linéaire, trois situations rendant l'identification nécessaire se produisent. La première est dûe à une identification de variables éliminées, la seconde à l'identification de variables aidant à éliminer une autre variable, et la troisième est à effectuer pour identifier une variable effondrante.

1. Il existe un terme linéaire l, une identification $\xi \in ID_{\mathcal{V}(l)}$ tels que $l\xi$ contient une variable x aux positions $\Omega = \{\omega_1, \ldots, \omega_n\}$ pour $n > 1$ et une égalité $l\xi =_E r$ avec $x \notin \mathcal{V}(r)$ pour laquelle le problème de filtrage

$$l[\omega_1 \hookleftarrow h(t_1)][\ldots][\omega_n \hookleftarrow h(t_n)] \leq^{?} r$$

est équivalent au problème d'unification

$$t_1 =^? \ldots =^? t_n$$

si h est un symbole libre et les positions de l hormis Ω sont closes.

2. Il existe un terme l linéaire et une identification $\xi \in ID_{\mathcal{V}(l)}$ tels que
   - une variable non éliminable x apparait aux positions $\Omega = \{\omega_1, \ldots, \omega_n\}$ de $l\xi$ pour $n > 1$,
   - une variable éliminable y est à une position ω' de $l\xi$,

   pour lesquels le problème de filtrage librement étendu

$$\exists y, z : z =^? l[\omega_1 \hookleftarrow h(t_1)][\ldots][\omega_n \hookleftarrow h(t_n)][y]_{\omega'} \wedge y =^? f(z)$$

   est équivalent au problème d'unification

$$t_1 =^? \ldots =^? t_n$$

   si h, f sont des symboles libres et les positions de l hormis Ω sont closes.

3. Il existe un terme l linéaire et une identification $\xi \in ID_{\mathcal{V}(l)}$ tels que $l\xi$ contient une variable x aux positions $\Omega = \{\omega_1, \ldots, \omega_n\}$ $(n > 1)$ pour lesquels le problème de filtrage librement étendu

$$\exists x : l[\omega_1 \hookleftarrow h(t_1)][\ldots][\omega_n \hookleftarrow h(t_n)] =^? x \wedge x =^? h(t_n)$$

   est équivalent au problème d'unification

$$t_1 =^? \ldots =^? t_n$$

   si h est un symbole libre et les positions de l hormis Ω sont closes.

□

Le filtrage librement étendu n'a donc pas d'intérêt pour les théories non partiellement linéaires car l'on dispose alors de l'unification avec restriction permettant d'appliquer l'algorithme de combinaison de l'unification.

Après ce premier résultat négatif, nous allons voir que la décidabilité du filtrage librement étendu est modulaire dans les théories partiellement linéaires. L'algorithme correspondant est décrit ci-après sous forme de règles.

4.2.5 Règles de combinaison

Les règles de combinaison que nous proposons n'engendrent que des problèmes qui peuvent être résolus par utilisation conjointe du filtrage et de l'élimination de constante.

Ces règles de combinaison ont pour objet de transformer un problème de (E_1, E_2)-filtrage étendu Γ en des conjonctions de problèmes de filtrage étendu avec restriction linéaire $(\Gamma_1, V_2, <)$ et $(\Gamma_2, V_1, <)$ tels que

- Γ_1 est un problème de filtrage étendu 1-pur,

Abstraction Close

$$\frac{\Gamma \wedge s \leq^? t}{\exists v_1, \ldots, v_m : \Gamma \wedge s =^? (t \Downarrow)[\omega_1 \hookleftarrow v_1][\ldots][\omega_m \hookleftarrow v_m] \wedge \bigwedge_{k=1}^{m} v_k \leq^? (t \Downarrow)_{|\omega_k}}$$

où

- $s \in T(\mathcal{F}_i, \mathcal{X})$,
- $\{\omega_1, \ldots, \omega_m\} = AlienPos(t \Downarrow)$ si $t \Downarrow (\epsilon) \in \mathcal{F}_i$, sinon $\{\omega_1, \ldots, \omega_m\} = \{\epsilon\}$,
- $\{v_1, \ldots, v_m\}$ est un ensemble de nouvelles variables.

Figure 4.9. Purification droite pour le filtrage

- Γ_2 est un problème de filtrage étendu 2-pur,
- V_1 est un ensemble de variables instanciées dans E_1 et skolémisées dans E_2,
- V_2 est un ensemble de variables instanciées dans E_2 et skolémisées dans E_1,
- $<$ est un ordre linéaire sur $V_1 \oplus V_2$.

L'étape de Purification introduit aussi des variables dans les membres droits en forme réduite par couches. Le choix de théories est déterministe pour ces variables et sera pris en compte à l'étape suivante.

L'étape de Choix de la restriction linéaire consiste à fixer les ensembles de variables V_1 et V_2 en tenant compte qu'une variable figurant dans un membre droit d'une filtre-équation est skolémisée dans la théorie où sera résolue cette filtre-équation. Les variables sont ensuite identifiées de telle manière que ces identifications engendrent soit de nouvelles filtre-équations, soit de nouvelles équations résolues mais en aucun cas d'équations qui ne puissent être résolues par un algorithme de filtrage.

Définition 4.17 *Une identification* ξ préserve la spécificité *d'un problème de filtrage étendu* (Γ, C) *si* $(\Gamma \wedge \hat{\xi}, C)$ *est égal après remplacement à un problème de filtrage étendu.*

L'étape de Fusion transforme le problème équationnel obtenu après identification en une conjonction de problèmes de filtrage étendu purs équivalente.

L'étape de Combinaison réduit le problème équationnel en conjonctions de formes résolues pures qui sont obtenues par filtrage étendu avec la même restriction linéaire.

Un problème équationnel obtenu finalement par Resolution est en forme séquentiellement résolue. Il est transformé ensuite en une forme résolue d'arbre par application répétée de la règle de Remplacement.

Proposition 4.18 *Les règles de combinaison du filtrage en figure 4.10 retournent un ensemble complet de solutions si les problèmes de filtrage étendu à résoudre sont combinables.*

Preuve : Nous avons à montrer que

1. Purification

$$\frac{\Gamma}{\Gamma_1 \wedge \Gamma_2}$$

où Γ_1 est 1-pur et Γ_2 est 2-pur. Le problème $\Gamma_1 \wedge \Gamma_2$ est obtenu en appliquant tant que possible la règle de la figure 4.9.

2. Application non-déterministe.
Choix de la restriction linéaire et identification des variables

$$\frac{\Gamma_1 \wedge \Gamma_2}{(\Gamma_1\xi_2 \wedge \hat{\xi}_1, V_2, <) \wedge (\Gamma_2\xi_1 \wedge \hat{\xi}_2, V_1, <)}$$

où

- $V_1 \supseteq \mathcal{GSV}(\Gamma_1)$, $V_2 \supseteq \mathcal{GSV}(\Gamma_2)$,
- $<$ est un ordre linéaire sur $V_1 \oplus V_2 = \mathcal{V}(\Gamma_1 \wedge \Gamma_2) \backslash \mathcal{GV}(\Gamma)$,
- ξ est une identification compatible avec $<$ telle que ξ_1 préserve la spécificité de $(\Gamma_1\xi_2, V_2)$ et ξ_2 préserve la spécificité de $(\Gamma_2\xi_1, V_1)$.

3. Fusion

Appliquer tant que possible les règles suivantes:

Fusion(Proj)

$$\frac{(\Gamma_i \wedge s =^? t, V_j, <)}{(\Gamma_i \wedge s \leq^? t, V_j, <)} \quad \textbf{si } \mathcal{V}(t) \in V_j, j \neq i$$

Fusion(Filtre)

$$\frac{(\Gamma_i \wedge x =^? s \wedge x \leq^? t \Downarrow, V_j, <)}{(\Gamma_i \wedge s \leq^? t \Downarrow \wedge x \leq^? t \Downarrow, V_j, <)}$$

4. Combinaison

Appliquer tant que possible la règle suivante:

Resolution

$$\frac{(\Gamma_1, V_2, <) \wedge (\Gamma_2, V_1, <)}{\hat{\sigma}_1 \wedge \hat{\sigma}_2} \quad \textbf{si } \sigma_i \in CSU^{\leq}_{E_i}(\Gamma_i, V_j), j \neq i$$

5. Solution combinée

Appliquer tant que possible la règle suivante:

Remplacement

$$\frac{\Gamma \wedge x =^? t}{\Gamma\{x \mapsto t\} \wedge x =^? t} \quad \textbf{si } x \in \mathcal{V}(\Gamma) \text{ et } t \notin \mathcal{X}$$

Figure 4.10. Règles de combinaison du filtrage

- l'ensemble des solutions combinées des problèmes $(\Gamma_1, V_2, <)$ et $(\Gamma_2, V_1, <)$ considérés à l'étape de Combinaison forme un ensemble complet de solutions au problème de filtrage étendu Γ.
- Ces problèmes sont effectivement des problèmes de filtrage étendu.

Pour le premier point, il s'agit de vérifier que le choix de la restriction linéaire et de l'identification de variables est suffisant. Les variables de $\mathcal{GSV}(\Gamma_i)$ sont forcément skolémisées dans E_j, la mise en forme réduite par couches permet de s'en assurer. Les variables de $\mathcal{GV}(\Gamma) = \mathcal{GSV}(\Gamma_1) \cap \mathcal{GSV}(\Gamma_2) = V_1 \cap V_2$ sont même skolémisées dans E_1 et E_2. Pour ce qui est des identifications, leur nombre est relativement limité et nous allons avoir à utiliser l'hypothèse que $(\Gamma_1, V_2, <)$ et $(\Gamma_2, V_1, <)$ sont combinables. Montrons d'abord l'intérêt de pouvoir identifier après la résolution. Soit ξ une identification compatible avec $<$. Si l'on suppose $SU^{<}_{E_1}(\Gamma_1\xi_2) = SU^{<}_{E_1}(\Gamma_1)\xi_2$ et $SU^{<}_{E_2}(\Gamma_2\xi_1) = SU^{<}_{E_2}(\Gamma_2)\xi_1$, alors il est facile de montrer par induction sur $<$ que l'ensemble des solutions combinées

$$SU^{<}_{E_1}(\Gamma_1\xi_2 \wedge \hat{\xi}_1) \odot SU^{<}_{E_2}(\Gamma_2\xi_1 \wedge \hat{\xi}_2)$$

est inclus dans

$$SU^{<}_{E_1}(\Gamma_1) \odot SU^{<}_{E_2}(\Gamma_2).$$

En toute généralité nous aurions à effectuer une identification $\xi' = \xi'_1\xi'_2$ telle que $\xi'_i \in ID_{V_i}$ compatible avec $<$ se décompose en deux identifications $\xi'_i = \mu_i\xi_i$ où

$$\mu_i \in ID^{\mathcal{GSV}(\Gamma_i)}_{V_i \setminus \mathcal{SV}(\Gamma_j)} \circ ID^{V_i}_{\mathcal{SV}(\Gamma_j)}$$

et $\xi_i \in ID^{V_i \setminus \mathcal{GSV}(\Gamma_i)}_{V_i \setminus \mathcal{SV}(\Gamma_j)}$ ou plus précisément

$$\xi_i \in ID^{V_i \setminus \mathcal{GSV}(\Gamma_i)}_{V_i \setminus (\mathcal{GSV}(\Gamma_i) \cup \mathcal{SV}(\Gamma_j))}$$

puisque l'on peut supposer sans perte de généralité que les variables de $\mathcal{GSV}(\Gamma_i)$ sont les représentants des classes d'équivalence définis par une identification. L'ensemble des variables de $V_i \setminus \mathcal{GSV}(\Gamma_i)$ pouvant apparaître dans $\mathcal{V}(\Gamma_j)$ est inclus dans $\overline{\mathcal{RV}}(\Gamma_j) \cup \mathcal{SV}(\Gamma_j)$. Comme μ_i se charge d'identifier une variable de $\mathcal{SV}(\Gamma_j) = \mathcal{CV}((\Gamma_i, V_j))$ à une autre du membre gauche, on peut en conclure que ξ_i satisfait la condition donnée en définition 4.14. En supposant les problèmes $\Gamma_1\mu_2 \wedge \hat{\mu}_1$ et $\Gamma_2\mu_1 \wedge \hat{\mu}_2$ filtre-combinables, nous avons par définition

$$SU^{<}_{E_1}((\Gamma_1\mu_2 \wedge \hat{\mu}_1)\xi_2) = SU^{<}_{E_1}(\Gamma_1\mu_2 \wedge \hat{\mu}_1)\xi_2,$$

$$SU^{<}_{E_2}((\Gamma_2\mu_1 \wedge \hat{\mu}_2)\xi_1) = SU^{<}_{E_2}(\Gamma_2\mu_1 \wedge \hat{\mu}_2)\xi_1,$$

et donc

$$SU^{<}_{E_1}(\Gamma_1\xi'_2 \wedge \hat{\xi}'_1) \odot SU^{<}_{E_2}(\Gamma_2\xi'_1 \wedge \hat{\xi}'_2)$$

est inclus dans

$$SU^{<}_{E_1}(\Gamma_1\mu_2 \wedge \hat{\mu}_1) \odot SU^{<}_{E_2}(\Gamma_2\mu_1 \wedge \hat{\mu}_2).$$

Il reste à montrer que les problèmes équationnels $\Gamma_i\mu_j \wedge \hat{\mu}_i$ sont équivalents à des problèmes de E_i-filtrage étendu. On utilise ici le fait que $\mathcal{SV}(\Gamma_1) \cap \mathcal{SV}(\Gamma_2) = \emptyset$ et

donc $\mathcal{D}om(\mu_{i|\mathcal{SV}(\Gamma_j)}) \cap \mathcal{SV}(\Gamma_i\mu_j) = \emptyset$. Cette identification ne peut pas créer un cycle dans E_i car μ_i sera choisie de manière à n'identifier une variable x_i de $\mathcal{SV}(\Gamma_j)$ qu'à une variable y_i de t_j si $x_i =^? t_j$ est l'équation de Γ_j ayant l'occurrence de x_i, pourvu évidemment que x_i ne soit pas déjà identifiée à une variable de $\mathcal{SV}(\Gamma_j) \cup \mathcal{GSV}(\Gamma_i)$. Par hypothèse, il n'existe pas d'équation $y_i =^? t_i[x_i]$ dans Γ_i et donc pas de cycle.

D'un autre côté, $\mu_{i|V_i \setminus \mathcal{SV}(\Gamma_j)}$ identifiant exclusivement sur des variables égales à des termes clos dans E_i, on transforme

$$x =^? s \wedge y \leq^? t \Downarrow \wedge x =^? y$$

en

$$s \leq^? t \Downarrow \wedge y \leq^? t \Downarrow \wedge x =^? y.$$

□

Etant donné un problème de filtrage étendu, savoir s'il est ou non combinable est décidable car le filtrage est supposé décidable dans chaque théorie.

Exemple 4.11 *Considérons E_1 la théorie booléenne non partiellement linéaire de signature $\{\neg, +, \cdot, 0, 1\}$.*

- *Soient E_2 la théorie vide $\emptyset$ et f un symbole libre. Pour résoudre la filtre-équation $x + f(x) \leq^? 1$, nous devons d'abord la purifier. On obtient $x + c \leq^? 1 \wedge c =^? f(x)$ avec $c \in V_2$ car E_2 est une théorie non effondrante. La résolution dans les booléens de la filtre-équation engendre $x =^? \neg c + c \cdot y$ créant ainsi un cycle composé avec $c =^? f(x)$. Pour le casser, on fixe un ordre linéaire $x < c$. La résolution suivant cette restriction linéaire donne la solution $\{x \mapsto 1\}$. Cette substitution forme un ensemble complet de solutions car l'unique problème de filtrage étendu solvable est combinable puisque ne contenant qu'une variable skolémisée c.*
- *Soient $E_2 = \{g(x, y) = g(y, x)\}$ et la filtre-équation $x + g(y, z) \leq^? g(a, b) + g(b, a)$. Après purification, nous obtenons $x + c \leq^? c' \wedge c =^? g(y, z) \wedge c' =^? g(a, b)$ avec $c, c' \in V_2$ car E_2 est une théorie non effondrante. L'unique façon d'obtenir une solution à la filtre-équation $x + c \leq^? c'$ est d'identifier c et c'. La résolution de $x + c \leq^? c$ dans les booléens produit $\{x \mapsto c \cdot v\}$ où v est une nouvelle variable. Dans la théorie E_2, l'identification de c et c' a pour conséquence d'avoir à considérer la filtre-équation $g(y, z) \leq^? g(a, b)$ dont les solutions sont $\{y \mapsto a, z \mapsto b\}$ et $\{y \mapsto b, z \mapsto a\}$. En recombinant les solutions de chaque théorie, on obtient finalement*

$$\{x \mapsto g(a, b) \cdot v, y \mapsto a, z \mapsto b\},$$

$$\{x \mapsto g(b, a) \cdot v, y \mapsto b, z \mapsto a\}$$

formant un ensemble complet de solutions puisque l'unique problème de filtrage étendu est évidemment combinable.

L'algorithme de combinaison du filtrage ne renvoie pas toujours un ensemble complet de solutions, cela dépend si le problème de filtrage étendu donné à résoudre est combinable ou non. Nous allons dorénavant nous intéresser aux théories pour lesquelles l'algorithme est complet.

4.2.6 Discussion

L'algorithme de combinaison vu précédemment s'applique aux théories dont tous les problèmes de filtrage étendu sont combinables, c'est-à-dire les théories partiellement linéaires.

Théorème 4.7 *Si E_1 et E_2 sont deux théories partiellement linéaires alors le (E_1, E_2)-filtrage étendu est décidable (resp. finitaire) si le E_i-filtrage étendu ($i = 1, 2$) avec restriction linéaire est décidable (resp. finitaire).*

Le théorème peut être généralisé car l'algorithme permet en fait la résolution de problèmes de $E_1 \cup E_2$-filtrage librement étendu.

Théorème 4.8 *Si E_1 et E_2 sont deux théories partiellement linéaires alors le $E_1 \cup E_2$-filtrage librement étendu est décidable (resp. finitaire) si et seulement si le E_i-filtrage librement étendu ($i = 1, 2$) est décidable (resp. finitaire).*

Preuve : Un problème de $E_1 \cup E_2$-filtrage librement étendu est équivalent à une conjonction $\Gamma_1 \wedge \Gamma_2 \wedge \Gamma_0$ de problèmes de filtrage étendu purs dans E_1, E_2 et $\emptyset$ qui sont toutes des théories partiellement linéaires. L'algorithme vu jusqu'à présent pour deux théories s'applique tout aussi bien à trois théories partiellement linéaires. Une identification μ se décompose maintenant en trois identifications μ_1, μ_2, μ_0 sur les ensembles de variables V_1, V_2, V_0 instanciées respectivement dans les théories E_1, E_2 et $\emptyset$. Nous avons à combiner trois ensembles de solutions

$$(SU^{<}_{\emptyset}(\Gamma_0\mu_1\mu_2 \wedge \hat{\mu}_0) \odot SU^{<}_{E_1}(\Gamma_1\mu_0\mu_2 \wedge \hat{\mu}_1)) \odot SU^{<}_{E_2}(\Gamma_2\mu_0\mu_1 \wedge \hat{\mu}_2).$$

Les problèmes à résoudre ne sont plus toujours équivalents à des problèmes de filtrage étendu car $\mathcal{SV}(\Gamma_0) \cap \mathcal{SV}(\Gamma_1 \wedge \Gamma_2)$ est éventuellement non vide. Les identifications μ_0 à considérer dans $\emptyset$ vérifient

$$\mu_0 \in ID^{\mathcal{GSV}(\Gamma_0)}_{V_0 \setminus \mathcal{SV}(\Gamma_1 \wedge \Gamma_2)} \circ ID^{V_0}_{\mathcal{SV}(\Gamma_1 \wedge \Gamma_2)}.$$

Comme $V_0 \supseteq \mathcal{SV}(\Gamma_0)$, il est maintenant possible d'identifier deux variables instanciées dans la théorie vide mais cette identification mène à un échec dans la théorie $\emptyset$ par définition de Γ_0 qui est une forme résolue dont les termes ne sont pas unifiables. Les problèmes à résoudre à l'étape de Combinaison restent donc combinables.

Le $E_1 \cup E_2$-filtrage librement étendu est par conséquent décidable (resp. finitaire) si le E_i-filtrage étendu avec restriction linéaire est décidable (resp. finitaire). Pour clôre la preuve, il suffit de remarquer qu'un algorithme de E_i-filtrage étendu avec restriction linéaire peut être construit grâce à un algorithme de E_i-filtrage librement étendu d'après la proposition 4.16. □

Ce résultat appliqué à la combinaison d'une théorie partiellement linéaire et de la théorie vide, elle aussi partiellement linéaire, complète la proposition 4.17.

Corollaire 4.6

- *Si E est partiellement linéaire alors le $E \cup \emptyset$-filtrage librement étendu est décidable (resp. finitaire) si et seulement si le E-filtrage librement étendu est décidable (resp. finitaire).*
- *Si E n'est pas partiellement linéaire alors le $E \cup \emptyset$-filtrage librement étendu est décidable (resp. finitaire) si et seulement si la $E \cup \emptyset$-unification est décidable (resp. finitaire).*

Nous savons désormais comment résoudre des problèmes hétérogènes un peu plus généraux que les problèmes de filtrage. Si l'on se restreint aux problèmes de filtrage, la restriction linéaire n'est même plus nécessaire car le filtrage dans une théorie partiellement linéaire ne peut introduire de cycle composé. Les théories partiellement linéaires généralisent en ce sens les théories régulières.

Lemme 4.6 *Soit E une théorie partiellement linéaire et Γ un problème de filtrage. On peut supposer sans perte de généralité que*

$$\forall \sigma \in CSU_E(\Gamma), \forall x \in \mathcal{V}(\Gamma), \forall c \in \overline{\mathcal{RV}}(\Gamma),\ c \notin x\sigma.$$

Preuve : Soit σ une solution de Γ et σ' la substitution obtenue à partir de σ par renommage des constantes $c \in \overline{\mathcal{RV}}(\Gamma)$ en $c' \notin \mathcal{GV}(\Gamma)$. La substitution σ' ne contient plus de variable skolémisée de $\overline{\mathcal{RV}}(\Gamma)$. Pour chaque filtre-équation $s \leq^?_E t$ de Γ on a $s\sigma =_E t$ et trois cas sont envisageables:

- S'il existe plusieurs occurrences de $c \in \overline{\mathcal{RV}}(\Gamma)$ dans $s\sigma$ alors $s\sigma' =_E t$ par hypothèse de partielle linéarité de E.
- S'il existe une occurrence de $c \in \overline{\mathcal{RV}}(\Gamma)$ dans $s\sigma$ alors $s\sigma' =_E t$ est un renommage de $s\sigma =_E t$.
- S'il n'existe pas d'occurrence de $c \in \overline{\mathcal{RV}}(\Gamma)$ alors $s\sigma' = s\sigma =_E t$.

La substitution σ' est donc une solution de Γ. Lorsque $\sigma \leq_E^{\mathcal{V}(\Gamma)} \phi$ alors $\sigma = \sigma'\{c' \mapsto c\} \leq_E^{\mathcal{V}(\Gamma)} \phi$ et donc $\sigma' \in CSU_E(\Gamma)$ si $\sigma \in CSU_E(\Gamma)$. □

Les variables skolémisées pouvant créer un cycle composé sont celles figurant uniquement dans les membres gauches des filtre-équations. Si elles n'apparaissent plus dans les solutions, il ne peut y avoir introduction de cycle composé.

Théorème 4.9 *Si E_1 et E_2 sont deux théories partiellement linéaires alors le $E_1 \cup E_2$-filtrage est décidable (resp. finitaire) si et seulement si le E_i-filtrage ($i = 1, 2$) est décidable (resp. finitaire).*

Preuve : Considérons une conjonction de problèmes de filtrage étendu

$$(\Gamma_1 \wedge \hat{\sigma}_1, V_2, <) \wedge (\Gamma_2 \wedge \hat{\sigma}_2, V_1, <)$$

obtenu à l'étape de Combinaison à partir d'un problème de filtrage Γ. Le problème équationnel $\hat{\sigma}_1 \wedge \hat{\sigma}_2$ est forcément en forme séquentiellement résolue d'après les identifications effectuées. D'après le lemme 4.6, la résolution de (Γ_1, V_2) et (Γ_2, V_1) n'introduit pas de cycle composé. Par conséquent, il existe un ordre linéaire $<$ pour lequel la conjonction $(\Gamma_1 \wedge \hat{\sigma}_1, V_2, <) \wedge (\Gamma_2 \wedge \hat{\sigma}_2, V_1, <)$ a une solution suivant la restriction linéaire $<$ si (Γ_1, V_2) et (Γ_2, V_1) ont chacun une solution. □

Exemple 4.12 *Considérons $E_1 = \{f(x,y) = y\}$ une théorie linéaire, $E_2 = C(\star)$ une théorie régulière non effondrante et la filtre-équation*

$$f(y \star y, y \star z) \leq^? a \star b.$$

La Purification *mène au système suivant:*

$$(f(v,w) \leq^? c) \wedge (v =^? y \star y \wedge w =^? y \star z \wedge c =^? a \star b)$$

où les variables v, w, c sont E_2-instanciées car E_2 est non effondrante. L'identification $\{v \mapsto c\}$ à appliquer sur $f(v,w) \leq^? c$ mène à un échec. Par contre, la filtre-équation est équivalente à $\top$ pour l'identification $\{w \mapsto c\}$. Nous avons alors à identifier w et c dans l'autre théorie et à résoudre $y \star z \leq^? a \star b$, ce qui conduit à l'ensemble complet de solutions $\{y \mapsto a, z \mapsto b\}$ et $\{y \mapsto b, z \mapsto a\}$.

Exemple 4.13 *(Suite de l'Exemple 4.9). La théorie régulière E_3 n'est pas partiellement linéaire car l'axiome idempotent $x + x = x$ n'est pas linéaire. L'exemple 4.9 n'est donc pas un contre-exemple au théorème 4.9 qui ne s'applique pas au $E_1 \cup E_2 \cup E_3$-filtrage.*

Le théorème 4.9 s'applique évidemment à la plus simple des théories partiellement linéaires, à savoir la théorie vide.

Corollaire 4.7 *Si E est une théorie partiellement linéaire alors le E-filtrage avec symboles libres est décidable (resp. finitaire) si et seulement si le E-filtrage est décidable (resp. finitaire).*

On rappelle que les théories linéaires et les théories régulières non effondrantes sont partiellement linéaires.

Le seul résultat connu jusqu'à présent ne concernait que le mélange de théories régulières qui ne sont pas partiellement linéaires. L'algorithme de filtrage dans le mélange de théories régulières nécessite la construction effective des solutions de chaque théorie et ne peut donc être utilisé pour combiner des algorithmes de décision. Le filtrage dans les théories régulières n'est donc pas à proprement parler un cas particulier de celui dans les théories partiellement linéaires. Ainsi, l'on ne dispose pas de résultat de décidabilité pour la combinaison du filtrage étendu dans les théories régulières alors que dans les théories partiellement linéaires nous avons le théorème 4.8 et son corollaire:

"Le $E_1 \cup \ldots \cup E_n \cup \emptyset$-filtrage librement étendu est décidable (resp. finitaire) si le $E_i \cup \emptyset$-filtrage librement étendu est décidable (resp. finitaire)."

Si maintenant l'une des théories E_i n'est pas partiellement linéaire alors le $E_1 \cup \ldots \cup E_n \cup \emptyset$-filtrage librement étendu est équivalent à la $E_1 \cup \ldots E_n \cup \emptyset$-unification. Le problème est dans ce cas celui résolu par F. Baader et K. Schulz dans [BS92]. C'est pourquoi la combinaison d'algorithmes de filtrage est traitée ici de façon exhaustive même si le filtrage étendu pourrait paraitre à certains égards comme une trop forte extension du filtrage. Le cas très particulier des théories régulières en est la parfaite illustration. Toutefois, avoir à considérer des équations résolues et donc des problèmes de filtrage étendu semble nécessaire à une approche modulaire pour laquelle la purification joue un rôle majeur.

4.2.7 Extension au mélange de théories non disjointes

La notion de forme réduite par couches adaptée aux théories décomposables permet de transformer un problème de filtrage Γ en une conjonction $\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2$ de problèmes tout aussi spécifiques. Nous avons vu au chapitre 2 sous quelle condition il est possible de résoudre cette conjonction en considérant les problèmes

$$\bigvee_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{V}(\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2)}} (\Gamma_0\rho) \wedge (\Gamma_1\rho) \wedge (\Gamma_2\rho)$$

et l'algorithme de combinaison dans les théories disjointes mais en utilisant les algorithmes d'unification dans $\emptyset, E_1, E_2$.

Les spécificités du problème $\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2$ et des théories supposées partiellement linéaires permettent d'utiliser des algorithmes de filtrage au lieu des algorithmes d'unification. Cette optimisation est justifiée par la mise en forme réduite par couches des membres droits des filtre-équations qui force les variables introduites dans les membres droits à être skolémisées et rend, par la même occasion, inutile l'instanciation d'une variable de $\mathcal{RV}(\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2)$ par un contexte partagé. Il suffit donc de considérer les problèmes

$$\bigvee_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{RV}(\Gamma_0 \wedge \Gamma_1 \wedge \Gamma_2)}} (\Gamma_0\rho) \wedge (\Gamma_1\rho) \wedge (\Gamma_2\rho)$$

qui mènent à des problèmes de filtrage par application des règles de combinaison du filtrage dans le mélange des théories disjointes. La restriction sur les identifications à appliquer ne dépend pas en effet de la forme du mélange, disjoint ou non, mais uniquement de la propriété de partielle linéarité satisfaite par $\emptyset, E_1, E_2$ et de la façon de combiner les solutions qui est identique au cas disjoint. Le résultat concernant le filtrage dans les théories partiellement linéaires peut donc être étendu aux théories décomposables.

Théorème 4.10 *Soit $E_1 \cup E_2$ une théorie décomposable, partiellement linéaire et admettant un facteur de partage. Le $E_1 \cup E_2$-filtrage est décidable (resp. finitaire) si le E_i-filtrage $(i = 1, 2)$ est décidable (resp. finitaire).*

On notera que dans le cas particulier du mélange avec constantes partagées, il est possible de ramener simplement le problème à une disjonction

$$\bigvee_{\rho \in SUBS^{\mathcal{SC}}_{\mathcal{RV}(\Gamma_1 \wedge \Gamma_2)}} (\Gamma_1\rho)' \wedge (\Gamma_2\rho)'$$

dans le mélange de théories disjointes et se servir ensuite du fait qu'une forme réduite par couches pour $E_1 \cup E_2$ est aussi une forme réduite par couches pour $E_1' \cup E_2'$.

N'oublions pas non plus que le calcul d'une forme réduite par couches utilise des algorithmes de décision du filtrage dans E_1 et E_2. L'existence d'algorithmes retournant un ensemble complet de solutions à un problème de filtrage arbitraire se révèle en effet une hypothèse trop forte puisqu'il suffit d'utiliser un algorithme de semi-décision pour déterminer une solution s'il en existe une.

Il n'est même pas besoin de supposer l'existence d'un facteur de partage pour $E_1 \cup E_2$ lorsque les théories E_1 et E_2 sont régulières. Dans ce cas, les problèmes à résoudre après purification sont deux problèmes de filtrage purs à gauche dont les solutions sont forcément closes. La combinaison des solutions se ramène alors simplement au problème du mot. L'algorithme est identique à celui présenté en figure 4.8.

Théorème 4.11 *Soit $E_1 \cup E_2$ une théorie décomposable régulière. Le $E_1 \cup E_2$-filtrage est finitaire si le E_i-filtrage $(i = 1, 2)$ est finitaire.*

Il est remarquable que l'hypothèse de décomposabilité avec régularité ne suffit pas pour l'unification.

4.3 Elimination de constante

Nous allons nous intéresser au plus simple des problèmes équationnels, la *forme résolue*, dont l'unification élémentaire est unitaire car triviale et ce dans n'importe quelle théorie.

Par contre, dans le cadre de la combinaison, l'unification élémentaire est supplantée par l'unification avec restriction pour laquelle même une forme résolue pose des difficultés si elle ne satisfait pas la restriction.

4.3.1 Forme résolue librement étendue

Nous avons introduit le problème de l'élimination de constante comme étant celui de l'unification d'une forme résolue avec *restriction*.

La résolution suivant une restriction *linéaire* peut, elle, se traduire de façon équationnelle dans le mélange avec des symboles libres. La définition d'une forme résolue librement étendue est analogue à celle introduite pour le filtrage.

Définition 4.18 *Une E-forme résolue* librement étendue *est un $E \cup \emptyset$-problème équationnel*

$$\hat{\sigma} \wedge \hat{\sigma}_\emptyset$$

où $\hat{\sigma}$ est une forme résolue et $\sigma_\emptyset$ une substitution formée de symboles libres telle que

- $\mathcal{D}om(\sigma_\emptyset) \cap \mathcal{R}an(\sigma_\emptyset) = \emptyset$,
- $\forall x, y \in \mathcal{D}om(\sigma_\emptyset), x \neq y,\ CSU_\emptyset(x\sigma_\emptyset =^? y\sigma_\emptyset) = \emptyset$.

Lorsqu'il y a conflit entre une variable apparaissant résolue dans E et dans $\emptyset$ alors cette variable sera forcément instanciée dans $\emptyset$. L'hypothèse de non-unifiabilité permet de s'assurer que les variables résolues dans $\emptyset$ et skolémisées dans E ne pourront être identifiées.

Proposition 4.19 *Si la E-unification de forme résolue librement étendue est décidable (resp. finitaire) alors la E-unification de forme résolue avec restriction linéaire est décidable (resp. finitaire).*

Preuve : A nouveau, d'après Baader & Schulz [BS92], on peut coder la restriction linéaire en posant $\Gamma \wedge \sigma_<$, où $\sigma_<$, une substitution construit à partir de $<$, vérifie les hypothèses de la définition 4.18. □

Il est remarquable que cette proposition n'est valable que pour une restriction linéaire. Si ϕ est une solution de $\hat{\sigma}$ suivant une restriction quelconque, alors

$$\bigwedge_{c \in C} c =^? f_c(x_1\phi, \ldots, x_m\phi)$$

n'est pas nécessairement en forme résolue et il n'est donc pas possible de construire immédiatement par composition une solution à $\hat{\sigma} \wedge \hat{\sigma}_\emptyset$.

Théorème 4.12 *Si E_1 et E_2 sont deux théories partiellement linéaires et non effondrantes alors la $E_1 \cup E_2$-unification de forme résolue librement étendue est décidable (resp. finitaire) si et seulement si la E_i-unification de forme résolue librement étendue est décidable (resp. finitaire).*

Preuve : C'est une conséquence directe de la proposition 4.18. Une forme résolue est un cas particulier de problème de filtrage étendu. Les problèmes équationnels devant être résolus par application de l'algorithme de filtrage ne restent pas toujours en forme résolue. On peut en effet introduire des filtre-equations avec des variables skolémisées en membres droits. Celles-ci n'ont pas de solution car les théories sont supposées non effondrantes. □

Nous avons établi une correspondance entre élimination de constante et unification de forme résolue suivant une restriction *quelconque* et non pas seulement linéaire. C'est pourquoi nous ne pouvons remplacer la notion d'unification de forme résolue librement étendue par celle d'élimination de constante qui est trop forte pour les besoins de la combinaison. On doit alors se restreindre à l'élimination de constante élémentaire.

Théorème 4.13 *Si E_1 et E_2 sont deux théories partiellement linéaires et non effondrantes alors la $E_1 \cup E_2$-élimination de constante élémentaire est décidable (resp. finitaire) si et seulement si la E_i-élimination de constante élémentaire est décidable (resp. finitaire).*

Preuve : Le problème d'élimination de constante élémentaire consistant à éliminer une seule constante c dans un seul terme t est équivalent à l'unification de la forme résolue librement étendue

$$x =^? t[c] \wedge c =^? f_c(x).$$

□

Un algorithme d'élimination de constante élémentaire ne permet pas de résoudre une conjonction de problèmes élémentaires car la notion d'élimination n'est pas stable par instanciation. Mais en pratique, il a toujours été possible jusqu'à présent d'exprimer un problème d'élimination de constante élémentaire par un problème unification. Si l'unification dans la théorie en question est finitaire alors une conjonction de problèmes élémentaires d'élimination peut se résoudre incrémentalement en considérant successivement chacun des problèmes. Il n'y a dans ce cas pas de différence entre élimination et élimination élémentaire et par conséquent entre unification avec restriction quelconque et linéaire qui s'obtiennent toutes les deux à l'aide d'un algorithme d'élimination élémentaire.

4.3.2 Combinaison des solutions

Nous nous intéressons maintenant à combiner des algorithmes d'élimination de constante en vue de résoudre une conjonction de formes résolues, chacune pure dans une théorie.

Lorsque une conjonction de formes résolues pures est donnée en entrée, l'algorithme d'unification dans le mélange de théories équationnelles nécessite encore la connaissance d'un algorithme d'unification pour résoudre les problèmes obtenus après identification et choix de théories pour les variables. Se restreindre aux théories partiellement linéaires et non effondrantes permet de ne pas avoir à pratiquer d'identification et de ne pas avoir à choisir de théories.

Théorème 4.14 *Soient E_1 et E_2 deux théories partiellement linéaires et non effondrantes, $\hat{\sigma}_1$ une E_1-forme résolue et $\hat{\sigma}_2$ une E_2-forme résolue. La $E_1 \cup E_2$-unification de $\hat{\sigma}_1 \wedge \hat{\sigma}_2$ est décidable (resp. finitaire) si la E_i-élimination de constante est décidable (resp. finitaire).*

Preuve : Les seuls problèmes équationnels unifiables à l'étape de combinaison de l'algorithme de filtrage étendu sont des formes résolues suivant une restriction linéaire. □

L'élimination de constante est donc le seul outil nécessaire pour la combinaison des solutions dans les théories partiellement linéaires non effondrantes. Cette notion a été introduite initialement afin de résoudre les cycles composés mais elle s'accompagne en général d'une phase d'unification pour les équations et filtre-équations obtenues respectivement par identification et choix de théories.

Exemple 4.14 *Considérons les théories linéaires non effondrantes $E_1 = AC0(+)$ et $E_2 = \emptyset$ et la conjonction de formes résolues*

$$x =^? y + z \wedge z =^? f(y, x).$$

L'élimination de constante dans $AC0(+)$ est équivalente au filtrage sur 0. La forme résolue $x =^? y + z$ suivant la restriction $x < z$ a donc pour solution $\{x \mapsto 0, y \mapsto 0\}$ dans $AC0(+)$. La forme résolue $z =^? f(y, z)$ suivant la même restriction $x < z$ a pour solution $\{z \mapsto f(y, z)\}$. En combinant ces deux solutions, on obtient

$$\{x \mapsto 0, y \mapsto 0, z \mapsto f(0, 0)\}$$

qui est l'unique solution. En effet, l'élimination de constante dans la théorie vide n'ayant pas de solution, la restriction linéaire $z < x$ conduit à un échec.

1. Appliquer tant que possible

   Conflit
   $$\frac{(\hat{\sigma}_1 \wedge x =^? s_1) \wedge (x =^? s_2 \wedge \hat{\sigma}_2)}{\bot} \quad \textbf{si } s_1(\epsilon) \in \mathcal{F}_1, s_2(\epsilon) \in \mathcal{F}_2$$

2. Appliquer la règle suivante pour toute restriction linéaire $<$ sur $V_1 \oplus V_2 = \mathcal{V}(\hat{\sigma}_1 \wedge \hat{\sigma}_2)$ telle que $V_i \subseteq \mathcal{D}om(\sigma_i)$ pour $i = 1, 2$

   Resolution
   $$\frac{(\hat{\sigma}_1 \wedge \hat{\sigma}_2)}{(\hat{\gamma}_1 \wedge \hat{\gamma}_2)} \quad \textbf{si } \gamma_i \in CSU^{<}_{E_i}(c_i, V_j), j \neq i$$

3. Solution combinée

   Appliquer tant que possible la règle suivante:

   Remplacement
   $$\frac{\Gamma \wedge x =^? t}{\Gamma\{x \mapsto t\} \wedge x =^? t} \quad \textbf{si } x \in \mathcal{V}(\Gamma) \text{ et } t \notin \mathcal{X}$$

Figure 4.11. Règles de combinaison des solutions dans les théories partiellement linéaires et non effondrantes

4.4 Satisfaisabilité

Le problème étudié ici est différent de ceux évoqués jusqu'à présent même s'il s'énonce de la même façon:
"Comment construire un algorithme de satisfaisabilité dans l'union de deux théories $Th_1 \cup Th_2$ *à partir de ceux existants dans* Th_1 *et* Th_2 *?"*

On s'intéresse désormais à savoir s'il existe un modèle de $Th_1 \cup Th_2$ satisfaisant une formule arbitraire, alors qu'auparavant le modèle était fixé. En particulier, lorsque $Th_1 \cup Th_2 = E_1 \cup E_2$ est un ensemble d'égalités, il se peut que le modèle de $\mathcal{M}od(E_1 \cup E_2)$ satisfaisant une formule ne soit pas $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/ =_{E_1 \cup E_2}$. Le théorème de Birkhoff, bien qu'établissant le lien entre égalité modulo $=_{E_1 \cup E_2}$ et égalité dans $\mathcal{M}od(E_1 \cup E_2)$, ne permet pas de ramener le problème à la satisfaisabilité modulo $=_{E_1 \cup E_2}$.

Dans le cadre de la vérification de programmes, on est amené à tester la satisfaisabilité de formules complexes dans des structures de données comme les tableaux, les listes et l'arithmétique entière ou réelle. Ces structures de données interagissent entre elles puisque on peut concevoir des listes de réels ou des tableaux dont les indices sont des entiers, comme l'illustre la formule

$$v[i + j] \neq v[i + i]$$

formée d'un opérateur [] de sélection d'un élément de tableau et de l'addition + sur les entiers. Cette formule est satisfaisable si $i \neq j$. Supposons maintenant que i et j sont liées par une formule

$$i + k = j \wedge 0 \leq k \leq 1 \wedge k * k \leq 0.$$

Cette formule est satisfaisable dans les entiers et implique $i = j$ puisque $k = 0$. Mais la conjonction avec la formule précédente est alors insatisfaisable.

Le problème de la satisfaisabilité de formules hétérogènes a été résolu pour l'arithmétique de Presburger avec symboles non interprétés par Shostak [Sho79] puis en présence de tableaux (d'entiers) par Suzuki et Jefferson [SJ77]. Une approche modulaire est suivie par Nelson et Oppen [NO79, Nel81] avec pour objectif de combiner des algorithmes de satisfaisabilité dans les théories du premier ordre de certaines structures de données. L'objectif de cette section est de d'abord présenter ces travaux relativement anciens et l'algorithme de combinaison déterministe qui en résulte. Ensuite, nous verrons des règles de combinaison qui s'apparentent beaucoup à celles vues jusqu'à présent pour l'unification, le problème du mot ou le filtrage.

Ces règles de transformations sont liées à la construction d'un modèle pour l'union de deux théories disjointes satisfaisant une conjonction de deux formules si chacune d'elles est pure et satisfaisable dans une théorie. On pourra se ramener à une conjonction de deux formules pures par une étape préliminaire de purification. Mais il ne suffit pourtant pas de tester la satisfaisabilité des formules obtenues par purification. Il va encore falloir pratiquer l'identification de variables de façon indéterministe. Une vision moins déterministe de l'algorithme présenté dans [NO79] avait déjà été esquissée par Oppen [Opp80]. L'étape indéterministe d'identification de variables que nous proposons est identique à celle utilisée par F. Baader et K. Schulz pour la disunification [BS93]. Des variables x, y non identifiées doivent être contraintes à satisfaire la diségalité $x \neq y$.

4.4.1 Exemples de théories

Nous donnons quelques exemples d'axiomatisation en logique du premier ordre de "structures de données" dont les algorithmes de décision de la satisfaisabilité des formules sans quantification sont connus et pourront être ensuite combinés par la méthode introduite par la suite. Nous supposerons dorénavant que toutes les formules considérées sont universellement quantifiées, ou sans quantification. Les exemples suivants sont tirés de [NO79, Opp80].

- La satisfaisabilité dans la théorie $\mathcal{R}$ des réels avec l'opérateur d'addition présentée par l'ensemble d'axiomes R

$$R = \begin{cases} x + 0 = x \\ x + (-x) = 0 \\ (x + y) + z = x + (y + z) \\ x + y = y + x \\ x \leq x \\ x \leq y \vee y \leq x \\ x \leq y \wedge y \leq x \Rightarrow x = y \\ x \leq y \wedge y \leq z \Rightarrow x \leq z \\ x \leq y \Rightarrow x + z \leq y + z \\ 0 \neq 1 \\ 0 \leq 1 \end{cases}$$

  est décidable par un algorithme basé sur le simplexe [Nel81].

- La satisfaisabilité dans la théorie $\mathcal{T}$ des tableaux présentée par T l'ensemble des axiomes

$$T = \begin{cases} \mathsf{store}(v,i,e)[i] = e \\ i \neq j \Rightarrow \mathsf{store}(v,i,e)[j] = v[j] \\ \mathsf{store}(v,i,v[i]) = v \\ \mathsf{store}(\mathsf{store}(v,i,e),i,f) = \mathsf{store}(v,i,f) \\ i \neq j \Rightarrow \mathsf{store}(\mathsf{store}(v,i,e),j,f) = \mathsf{store}(\mathsf{store}(v,j,f),i,e) \end{cases}$$

  est décidable [DS78]. La sélection du i-ème élément de v, $\mathsf{select}(v,i)$, a été abrégée en $v[i]$.

- La satisfaisabilité dans la théorie vide $\emptyset$ ou théorie de l'égalité.

Nous n'avons introduit que le strict nécessaire pour illustrer le fonctionnement de l'algorithme de combinaison, qui va permettre par exemple de résoudre le problème de satisfaisabilité dans l'union des théories $R \cup T$, $T \cup \emptyset$ ou $R \cup \emptyset$.

4.4.2 Algorithme déterministe

Le déroulement de l'algorithme est d'abord présenté de façon informelle sur un exemple pour que le lecteur puisse apprécier son comportement sans surprise.

Considérons la formule

$$(\varphi_{R\cup\emptyset}) \qquad x + f(x) - f(y) \neq z + y \wedge x \leq y \wedge y + z \leq x \wedge z \geq 0$$

dans la théorie $R \cup \emptyset$. L'introduction de variables aux positions étrangères et des équations résolues correspondantes permet de transformer la formule $\varphi_{R\cup\emptyset}$ en une conjonction $\varphi_R \wedge \varphi_\emptyset$,

$$(\varphi_R) \qquad x + X - Y \neq z + y \wedge x \leq y \wedge y + z \leq x \wedge z \geq 0,$$

$$(\varphi_\emptyset) \qquad X = f(x) \wedge Y = f(y)$$

satisfaisable dans $R \cup \emptyset$ si et seulement si φ l'est.

L'interaction entre les théories R et $\emptyset$ se fait au travers des formules communes aux deux théories, déduites par l'une et propagées par l'autre. Il s'agit typiquement d'égalités impropres $x = y$ ou de diségalités $x \neq y$. On pourrait s'inquiéter du nombre potentiellement infini de formules impropres encore appelées résidus dans [NO79]. Heureusement, seules les égalités entre deux variables sont à propager. Ainsi, de la formule φ_R on peut déduire que $x = y$. Cette déduction est automatique car nous disposons dans R d'un algorithme de décision de la satisfaisabilité et donc indirectement de la validité par exemple de $\varphi_R \Rightarrow x = y$ dans R. L'égalité $x = y$ est ensuite propagée dans $\varphi_\emptyset$, pour obtenir

$$(\varphi_R) \qquad x + X - Y \neq z + y \wedge x \leq y \wedge y + z \leq x \wedge z \geq 0 \wedge x = y,$$

$$(\varphi_\emptyset) \qquad X = f(x) \wedge Y = f(y) \wedge x = y.$$

L'algorithme de décision dans $\emptyset$ permet de déduire $X = Y$ propagée dans φ_R. Le nouvel état est

$$(\varphi_R) \qquad x + X - Y \neq z + y \wedge x \leq y \wedge y + z \leq x \wedge z \geq 0 \wedge x = y \wedge X = Y,$$

$$(\varphi_\emptyset) \qquad X = f(x) \wedge Y = f(y) \wedge x = y \wedge X = Y$$

avec φ_R devenu insatisfaisable car équivalent à

$$X - Y \neq 0 \wedge X = Y$$

vu que $x = y$ et $z = 0$. L'algorithme retourne finalement insatisfaisable et ce résultat est correct puisque toutes les transformations effectuées sont valides dans R ou $\emptyset$, donc dans $R \cup \emptyset$.

Si nous étions arrivé à un stade où chaque formule est satisfaisable dans sa théorie et aucune équation impropre déductible, alors l'algorithme aurait retourné satisfaisable. Toute la difficulté de la preuve d'adéquation faite dans [NO79], que nous ne développerons pas ici, est de démontrer que la formule est bien dans ce cas satisfaisable dans $R \cup \emptyset$.

La propagation des équations n'est cependant pas aussi simple qu'il y parait. Déduire n égalités $x_1 = y_1, \ldots, x_n = y_n$ d'une formule φ est équivalent à déduire la conjonction $\wedge_{i=1}^n x_i = y_i$ de φ. Par contre, il existe des formules pour lesquelles une disjonction $\vee_{i=1}^n x_i = y_i$ est déduite sans que l'une des égalités $x_i = y_i$ le soit individuellement.

Définition 4.19 *Soient $2n$ variables $x_1, \ldots, x_n, y_1, \ldots y_n$. Une formule φ est convexe si φ implique une disjonction $\vee_{i=1}^n x_i = y_i$ alors φ implique $x_i = y_i$ pour un $i \in \{1, \ldots, n\}$.*

Une théorie est convexe si toutes ses formules sont convexes.

On peut montrer que R est convexe par des considérations géométriques dans $\mathcal{R}$. Mais T n'est pas convexe car

$$\mathsf{store}(v, i, e)[j] = x \wedge v[j] = y \Rightarrow x = y \vee x = e$$

sans que $\mathsf{store}(v, i, e)[j] = x \wedge v[j] = y$ n'implique individuellement $x = y$ ou $x = e$. La théorie des entiers avec l'addition n'est pas non plus convexe car

$$0 \leq x \leq 1 \wedge y = 0 \wedge z = 1 \Rightarrow x = y \vee x = z.$$

Il en est de même pour la théorie des réels avec multiplication car

$$x * y = 0 \wedge z = 0 \Rightarrow x = y \vee y = z.$$

La non convexité d'une théorie n'empêche pas l'algorithme de combinaison de s'appliquer, elle ne fait que le compliquer. Chaque équation d'une disjonction doit être en effet prise en compte dans un branchement différent.

Algorithme par propagation d'égalités

Soient Th_1 et Th_2 deux théories formées sur des signatures disjointes et φ une formule hétérogène.

1. Transformer φ en une conjonction de formules $\varphi_1 \wedge \varphi_2$ respectivement 1-pure et 2-pure.

2. (Insatisfaisable ?) Si φ_1 est insatisfaisable dans Th_1 ou φ_2 insatisfaisable dans Th_2 alors retourner insatisfaisable.

3. (Propagation) Si $Th_i \models \varphi_i \Rightarrow x = y$ et $Th_i \not\models \varphi_j \Rightarrow x = y$ alors

$$\varphi_1 := (\varphi_1 \wedge x = y) \text{ et } \varphi_2 := (\varphi_2 \wedge x = y)$$

   puis aller en 2.

4. (Branchement ?) Si φ_i implique une disjonction

$$\bigvee_{i=1}^{n} x_i = y_i$$

   sans impliquer individuellement chaque équation alors relancer n fois l'algorithme avec $\varphi_1 \wedge \varphi_2 \wedge x_i = y_i$ en entrée. Retourner satisfaisable si l'une au moins des n exécutions retourne satisfaisable. Sinon retourner insatisfaisable.

5. Retourner satisfaisable.

Cet algorithme termine puisqu'on ne peut créer indéfiniment des égalités entre un nombre fini et constant de variables. Par comparaison, la terminaison de l'algorithme de combinaison de l'unification donné par A. Boudet est beaucoup plus délicate même s'il est aussi basé sur la propagation des équations entre variables. La différence se situe au niveau de l'ensemble des variables qui est dans ce cas modifié au cours du déroulement de l'algorithme.

Le principal inconvénient de l'algorithme que nous venons de présenter est d'avoir à utiliser les algorithmes de décision à deux niveaux, l'un pour la satisfaisabilité des formules et l'autre pour la déduction des égalités à propager.

Nous allons adopter une autre démarche, fondamentalement différente, consistant à ne pas déduire des égalités mais à imposer une même identification de variables dans les deux théories.

L'algorithme que nous présentons formellement dans le prochain paragraphe est à l'algorithme de Nelson et Oppen, ce qu'est l'algorithme de F. Baader et K. Schulz à celui de A. Boudet. La preuve s'appuie sur la construction effective d'un modèle de $Th_1 \cup Th_2$ satisfaisant une conjonction de formules pures, chacune déjà satisfaisable dans sa théorie.

Le prochain paragraphe va permettre de formaliser certains concepts et de préciser les théories pour lesquelles l'algorithme de combinaison est adéquat. En effet, même si nous n'en avions pas fait mention jusqu'à présent, l'algorithme de Nelson et Oppen ne s'applique pas toujours.

4.4.3 Modèle pour la combinaison

Dans ce qui suit, une *formule* est une formule de logique du premier ordre sans quantification.

Définition 4.20 *Une* théorie Th *est un ensemble de formules. Une formule* φ *est* valide *dans une théorie* Th *si tout modèle de* Th *est modèle de* φ*, ce qu'on note* $Th \models \varphi$*. Une formule* φ *est* satisfaisable *dans une théorie* Th *s'il existe un modèle de* Th *qui satisfait* φ*. Une formule* φ *est* insatisfaisable *dans une théorie* Th *si elle n'est pas satisfaisable dans* Th*.*

Corollaire 4.8 *Une formule* φ *est valide dans* Th *si* $\neg\varphi$ *est insatisfaisable dans* Th*.*

Preuve : Par définition φ est valide dans Th signifie que φ est valide dans tous les modèles de Th, ce qui est équivalent à $\neg\varphi$ insatisfaisable dans tous les modèles de Th. □

Un algorithme de décision de la satisfaisabilité fournit aussi un algorithme de décision de la validité et réciproquement. Cela explique pourquoi on peut décider des formules

$$Th \models \varphi \Rightarrow x = y$$

et donc "déduire" les équations entre deux variables x et y de φ à propager par l'algorithme de Nelson et Oppen.

L'objet de ce paragraphe est de montrer comment construire un modèle de $Th_1 \cup Th_2$ satisfaisant une conjonction de formules $\varphi_1 \wedge \varphi_2$ si φ_1 est satisfaisable dans Th_1 et f_2 satisfaisable dans Th_2. Les modèles $\mathcal{M}_1$ de Th_1 et $\mathcal{M}_2$ de Th_2 satisfaisant respectivement φ_1 et φ_2 doivent vérifier une même propriété.

Définition 4.21 *(G. Nelson [Nel81]) Une théorie* Th *est* stable-infinie *si pour toute formule* φ *satisfaisable dans* Th *il existe un modèle de* Th *satisfaisant* φ *ayant un domaine infini.*

Les théories utilisées en pratique axiomatisent des modèles dont les domaines sont infinis: Réels, Entiers, Termes, Tableaux, Listes, etc ...

Définition 4.22 *Une théorie* Th axiomatise *un modèle* $\mathcal{M}$ *si toute formule valide dans* $\mathcal{M}$ *l'est aussi dans* Th *et réciproquement.*

Corollaire 4.9 *Une théorie axiomatisant un modèle de domaine infini est stable-infinie.*

Les théories $R, T, \emptyset$ sont donc stable-infinies.

Lorsque une théorie est stable-infinie et φ satisfaisable dans cette théorie alors il existe un modèle infini dénombrable satisfaisant φ puisque il possible de prendre un domaine dénombrable de termes clos à la place du domaine infini quelconque. Les constantes correspondent aux valeurs d'une assignation des variables de φ évaluant φ à vrai.

Deux théories stable-infinies ont quasiment un même domaine d'interprétation car deux ensembles infinis dénombrables peuvent être mis en bijection. L'existence d'une telle bijection permet d'envisager maintenant la construction d'un modèle de $Th_1 \cup Th_2$, pourvu que les signatures des théories soient disjointes.

Il faut cependant que deux variables prennent une même valeur simultanément dans les deux théories. C'est pourquoi nous allons effectuer une même identification de variables dans les deux formules pures. Mais comme précédemment pour la disunification (cf. chapitre 2), les variables non identifiées doivent être en plus contraintes à satisfaire les diségalités $\xi_{\neq}$ (cf. définition 2.22).

Le résultat suivant établit comment réutiliser les algorithmes de satisfaisabilité de chaque théorie composante.

Théorème 4.15 *Si Th_1 et Th_2 sont deux théories stable-infinies à signatures disjointes alors une formule $\varphi_1 \wedge \varphi_2$ est satisfaisable dans la théorie $Th_1 \cup Th_2$ si et seulement si il existe une identification ξ telle que $\varphi_1\xi \wedge \xi_{\neq}$ est satisfaisable dans Th_1 et $\varphi_2\xi \wedge \xi_{\neq}$ est satisfaisable dans Th_2.*

Preuve : On suppose les théories Th_1 et Th_2 formées respectivement sur une $(\mathcal{F}_1, \mathcal{P}_1)$-signature et $(\mathcal{F}_2, \mathcal{P}_2)$-signature.

($\Rightarrow$) Il existe un modèle $\mathcal{M}$ de $Th_1 \cup Th_2$ et une assignation $\alpha : \mathcal{X} \to M$ tels que $\underline{\alpha}(\varphi_1 \wedge \varphi_2)$ est vrai. Pour l'identification ξ définie sur $\mathcal{V}(\varphi_1 \wedge \varphi_2)$ comme suit: $x\xi = y\xi$ si $\alpha(x) = \alpha(y)$, on a encore $\underline{\alpha}((\varphi_1 \wedge \varphi_2)\xi \wedge \xi_{\neq})$ vrai, qui est équivalent à $\underline{\alpha}(\varphi_1\xi \wedge \xi_{\neq})$ vrai et $\underline{\alpha}(\varphi_2\xi \wedge \xi_{\neq})$ vrai. Ainsi $\mathcal{M}$ est un modèle de Th_i satisfaisant $\varphi_i\xi \wedge \xi_{\neq}$.

($\Leftarrow$) Il existe un modèle $\mathcal{M}_i$ de Th_i et une assignation $\alpha_i : \mathcal{X} \to M_i$ tels que $\underline{\alpha_i}(\varphi_i\xi \wedge \xi_{\neq})$ est vrai. On peut supposer sans perte de généralité que M_1 et M_2 sont des domaines infinis dénombrables. Ils peuvent être donc mis en bijection. On note h une bijection de M_1 vers M_2 telle que

$$\forall x \in \mathcal{V}((\varphi_1 \wedge \varphi_2)\xi \wedge \xi_{\neq}),\ h(\alpha_1(x)) = \alpha_2(x)$$

La relation d'équivalence sur $M_1 \cup M_2$ engendrée par h est notée $\equiv$. Les assignations α_1 et α_2 définissent une assignation $\alpha : \mathcal{V}(\varphi_1 \wedge \varphi_2) \to (M_1 \cup M_2)/_{\equiv}$ de la façon suivante:

$$\forall x \in \mathcal{V}(\varphi_1 \wedge \varphi_2),\ \alpha(x) = [\alpha_1(x\xi)]/_{\equiv} = [\alpha_2(x\xi)]/_{\equiv}$$

La structure $\mathcal{M}$ de domaine $(M_1 \cup M_2)/_{\equiv}$ est définie comme suit sur les symboles de fonctions et prédicats:

$$\begin{array}{lrcl}
\forall f \in \mathcal{F}_1, & f_{\mathcal{M}}([a_1]/_{\equiv}, \ldots, [a_n]/_{\equiv}) & = & [f_{\mathcal{M}_1}(a_1, \ldots, a_n)]/_{\equiv} \\
\forall f \in \mathcal{F}_2, & f_{\mathcal{M}}([b_1]/_{\equiv}, \ldots, [b_n]/_{\equiv}) & = & [f_{\mathcal{M}_2}(b_1, \ldots, b_n)]/_{\equiv} \\
\forall p \in \mathcal{P}_1, & p_{\mathcal{M}}([a_1]/_{\equiv}, \ldots, [a_n]/_{\equiv}) & = & p_{\mathcal{M}_1}(a_1, \ldots, a_n) \\
\forall p \in \mathcal{P}_2, & p_{\mathcal{M}}([b_1]/_{\equiv}, \ldots, [b_n]/_{\equiv}) & = & p_{\mathcal{M}_2}(b_1, \ldots, b_n)
\end{array}$$

Cette structure est modèle de $Th_1 \cup Th_2$ et satisfait $\varphi_1 \wedge \varphi_2$ car $\underline{\alpha}(\varphi_1 \wedge \varphi_2)$ est vrai. □

Corollaire 4.10 *La propriété* stable-infinie *est modulaire par union disjointe: $Th_1 \cup Th_2$ est stable-infinie si Th_1 et Th_2 sont deux théories stable-infinies à signatures disjointes.*

Preuve : Le domaine $(M_1 \cup M_2)/_{\equiv}$ du modèle $\mathcal{M}$ construit pour la preuve du théorème 4.15 est infini. □

1. Purification
$$\frac{(\varphi)_{Th_1 \cup Th_2}}{(\varphi_1 \wedge \varphi_2)_{Th_1 \cup Th_2}}$$
où $\varphi_1 \wedge \varphi_2$ est obtenue à partir de φ en appliquant tant que possible la règle

   Abstraction Variable
$$\frac{(\varphi \wedge p(t_1, \ldots, t_n))_{Th_1 \cup Th_2}}{(\varphi \wedge p(\ldots, t_l[\omega \hookleftarrow x], \ldots) \wedge x = t_{l|\omega})_{Th_1 \cup Th_2}}$$
   où

   - $p \in \mathcal{P}_i \cup \{=\}$,
   - $l \in \{1, \ldots, n\}$,
   - $\omega \in AlienPos(t)$,
   - x est une nouvelle variable.

2. Identification des variables
$$\frac{(\varphi_1 \wedge \varphi_2)_{Th_1 \cup Th_2}}{\bigvee_{\xi \in ID_{\mathcal{V}(\varphi_1 \wedge \varphi_2)}} ((\varphi_1 \xi \wedge \xi_{\neq})_{Th_1} \wedge (\varphi_2 \xi \wedge \xi_{\neq})_{Th_2})}$$

3. Satisfaisabilité

   Appliquer tant que possible les règles suivantes:

$$\frac{(\varphi_i)_{Th_i}}{\top} \quad \textbf{si } \varphi_i \text{ satisfaisable dans } Th_i$$

$$\frac{(\varphi_i)_{Th_i}}{\bot} \quad \textbf{si } \varphi_i \text{ est insatisfaisable dans } Th_i$$

Figure 4.12. Règles de combinaison de la satisfaisabilité

4.4.4 Règles de combinaison

Par souci de commodité, nous supposerons que la formule φ à décider est mise en forme normale disjonctive, chaque conjonction de littéraux étant décidée séparément. Le résultat définitif est obtenu en prenant la disjonction des formules $\bot$ ou $\top$ qui se réduit à $\bot$ (insatisfaisable) ou $\top$ (satisfaisable).

La proposition suivante est une conséquence directe du théorème 4.15.

Proposition 4.20 *Si Th_1 et Th_2 sont deux théories stable-infinies à signatures disjointes et φ une conjonction de littéraux, alors les règles de combinaison de la figure 4.12 retourne $\top$ si φ est satisfaisable dans $Th_1 \cup Th_2$, $\bot$ sinon.*

L'algorithme de combinaison ainsi décrit se comporte de la même manière dans les théories convexes et non convexes.

Exemple 4.15 *Considérons l'exemple tiré de [NO79],*

$$(\varphi_T) \qquad \mathsf{store}(v,i,e)[j] = x \wedge v[j] = y,$$

$$(\varphi_R) \qquad x > e \wedge x > y.$$

Pour les identifications ξ identifiant x et y, on a $\varphi_R\xi$ insatisfaisable dans R. Pour les identifications ξ identifiant x et e, on a $\varphi_R\xi$ insatisfaisable dans R. Pour les autres identifications, on doit considérer la conjonction des diségalités $x \neq y \wedge x \neq e \wedge \ldots$ et alors $\varphi_T\xi \wedge \xi_{\neq}$ est insatisfaisable. En définitive, la formule $\varphi_R \wedge \varphi_T$ est insatisfaisable dans $R \cup T$ car $\varphi_R\xi \wedge \xi_{\neq}$ est insatisfaisable dans R ou $\varphi_T\xi \wedge \xi_{\neq}$ est insatisfaisable dans T pour toute identification ξ.

L'algorithme proposé par Nelson et Oppen aurait déduit

$$T \models \varphi_T \Rightarrow x = y \vee x = e$$

puis propagé séparement $x = y$ et $x = e$.

4.4.5 Application aux théories équationnelles

On s'intéresse naturellement à appliquer l'algorithme de combinaison de la satisfaisabilité à deux théories E_1 et E_2 composées exclusivement d'axiomes égalitaires.

Un modèle particulier de E_i est l'algèbre libre

$$\mathcal{T}(\mathcal{F}_i, \mathcal{X})/ =_{E_i}$$

engendrée par l'ensemble d'axiomes E_i. Le résultat fondamental dû à G .Birkhoff [Bir35, BM70] permet d'étudier le problème de validité dans E_i grâce à $\mathcal{T}(\mathcal{F}_i, \mathcal{X})/ =_{E_i}$ puisque

$$E_i \models s_i = t_i \Leftrightarrow \mathcal{T}(\mathcal{F}_i, \mathcal{X})/ =_{E_i} \models s_i = t_i.$$

Mais cette équivalence n'est valable que pour une égalité $s_i = t_i$ ou plus exactement une conjonction d'égalités. Un algorithme de décision du problème du mot dans E_i ne fournit ainsi qu'un algorithme de satisfaisabilité des formules

$$\neg(\bigwedge_{k \in K} s_{i,k} = t_{i,k}) = \bigvee_{k \in K} s_{i,k} \neq t_{i,k}.$$

Une formule pure obtenue par purification n'a pas cette forme et ne pourra donc pas être décidée. Les règles de combinaison de la satisfaisabilité ne permettent pas la décision du problème du mot dans $E_1 \cup E_2$ qui a été traité en début de chapitre.

Même la conjonction de formules pures

$$\varphi_1 \wedge \varphi_2 = (\bigvee_{k_1 \in K_1} s_{1,k_1} \neq t_{1,k_1}) \wedge (\bigvee_{k_2 \in K_2} s_{2,k_2} \neq t_{2,k_2})$$

ne peut être décidée car les formules obtenues après identification des variables sont encore trop complexes pour l'algorithme disponible dans chaque théorie équationnelle.

En conclusion, l'algorithme de combinaison de la satisfaisabilité n'est pas adapté à l'union disjointe des théories équationnelles. Nous avons vu en début de chapitre quelles

sont les étapes supplémentaires nécessaires à la décision du problème du mot dans l'union des théories équationnelles. Il n'est pas étonnant en effet qu'un algorithme de satisfaisabilité pour un modèle particulier, en l'occurrence

$$\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/=_{E_1 \cup E_2},$$

soit une spécialisation d'un algorithme de satisfaisabilité pour l'ensemble des modèles de $E_1 \cup E_2$.

L'extension de l'algorithme de combinaison de la satisfaisabilité à des mélanges de théories non disjointes n'a pas encore été menée, mais il semble que la construction du modèle pour la preuve du théorème 4.15 se généralise très bien lorsque seules les constantes sont partagées et qu'il n'y a pas deux constantes représentant le même élément du domaine. Cette hypothèse sera reprise pour la définition au chapitre 6 d'un langage de contraintes dans lequel un algorithme de résolution symbolique s'obtient par combinaison.

5

Résolution de contraintes dans les algèbres finies

Ce chapitre est consacré à l'étude d'algorithmes de résolution dans des structures algébriques basées sur des domaines finis. Nous nous intéresserons en premier lieu aux anneaux et algèbres booléennes où l'unification a la propriété d'être unitaire. Deux méthodes anciennes ont été récemment redécouvertes. Nous les rappellerons en les appliquant uniquement à l'unification dans l'anneau booléen à deux éléments ou à l'algèbre de Boole associée.

Il y a souvent confusion entre l'algèbre de Boole et les booléens, entre l'algèbre initiale et la variété équationnelle engendrée par les axiomes booléens. Nous verrons que cette confusion est pleinement justifiée. En conséquence, l'algorithme d'unification dans l'algèbre initiale fournit un algorithme d'unification dans la variété équationnelle. Par ce biais, le problème est repositionné dans un contexte plus standard en théorie de l'unification.

Les propriétés de l'algèbre de Boole se généralisent à d'autres algèbres finies dites primales qui par définition ont une structure algébrique suffisamment riche pour permettre d'exprimer n'importe quelle fonction par un terme. Nous allons poursuivre l'approche initialisée dans [Büt89, Büt88, BES+90] qui consiste à plonger toute algèbre finie dans une algèbre primale par adjonction d'opérateurs.

L'objectif est de mettre en évidence des classes d'algèbres ayant les mêmes bonnes propriétés que celles de l'algèbre de Boole. Le concept d'algèbre primale sera ainsi généralisé à d'autres structures potentiellement infinies, comme par exemple la structure des pseudo-booléens qui contient les entiers et les booléens. L'algorithme de résolution de contrainte introduit pour les algèbres finies sera étendu à ces structures mélangeant l'arithmétique sur les entiers et un ou plusieurs domaines finis.

5.1 Anneaux booléens

Les anneaux booléens ont suscité beaucoup d'intérêt en déduction automatique. C'est une théorie équationnelle dont la complétion modulo des égalités AC termine et donne un système de réécriture pour décider du problème du mot. L'unification dans les anneaux booléens a elle aussi connu un vif regain d'interêt ces dernières années car elle offre la particularité d'être unitaire.

5.1.1 Problème du mot

La classe des anneaux booléens est définie par l'ensemble des axiomes BR construit sur l'ensemble des symboles de fonctions $\mathcal{F}_{BR} = \{0, 1, \wedge, \oplus\}$.

$$BR = \begin{cases} x \oplus 0 = x \\ x \oplus x = 0 \\ x \wedge 0 = 0 \\ x \wedge 1 = x \\ x \wedge x = x \\ x \wedge (y \oplus z) = (x \wedge y) \oplus (x \wedge z) \\ x \oplus x = 0 \\ x \wedge y = y \wedge x \\ (x \wedge y) \wedge z = x \wedge (y \wedge z) \\ x \oplus y = y \oplus x \\ (x \oplus y) \oplus z = x \oplus (y \oplus z) \\ x \wedge y = y \wedge x \\ (x \wedge y) \wedge z = x \wedge (y \wedge z) \end{cases}$$

Un *anneau booléen* est un modèle de BR.

Le système de règles de réécriture obtenu après complétion modulo les axiomes d'associativité et commutativité de $\wedge$ et $\oplus$ a été donné pour la première fois par Hsiang et Dershowitz [HD83]. Il peut s'obtenir de façon automatique par un logiciel de complétion supportant l'unification associative-commutative [KK86].

$$\begin{array}{rcl} x \oplus 0 & \rightarrow & x \\ x \oplus x & \rightarrow & 0 \\ x \wedge 0 & \rightarrow & 0 \\ x \wedge 1 & \rightarrow & x \\ x \wedge x & \rightarrow & x \\ x \wedge (y \oplus z) & \rightarrow & (x \wedge y) \oplus (x \wedge z) \end{array}$$

La réécriture par ces règles et modulo l'associativité-commutativité de $\wedge$ et $\oplus$ fournit un algorithme de décision de l'égalité dans les anneaux booléens. L'étroite liaison existant entre anneaux et algèbres booléennes permet de dériver un système de réécriture de termes contenant éventuellement le $^-$ (non) et le $\vee$ (ou). Il suffit de rajouter les règles suivantes:

$$\begin{array}{rcl} x \vee y & \rightarrow & (x \wedge y) \oplus x \oplus y \\ \bar{x} & \rightarrow & x \oplus 1 \end{array}$$

pour transposer le problème dans les anneaux booléens.

L'égalité dans les algèbres booléennes est donc elle aussi décidable par réécriture. Le système canonique utilisé n'est cependant pas construit sur les seuls symboles de fonctions $\{0, 1, \wedge, ^-, +\}$ définissant une algèbre booléenne. La définition d'une algèbre booléenne sera précisée ultérieurement.

L'existence d'un système canonique pourrait motiver l'utilisation de techniques de surréduction afin de construire une procédure d'unification, voire un algorithme. Cette idée est à proscrire de par les propriétés exemplaires de cette théorie concernant l'unification.

5.1.2 Unification

La résolution d'équations dans les anneaux booléens a été traitée depuis fort longtemps, l'une des deux méthodes présentées dans ce qui suit étant due à Boole, l'autre plus récente a été présentée par Löwenheim au début du siècle. Ces deux algorithmes de résolution ont été repris récemment par U. Martin & T. Nipkow [MN86, MN89] et W. Büttner & Simonis [BS87] dans un cadre plus conforme à la théorie de l'unification. L'objectif de ce paragraphe est d'exposer ces différents algorithmes et d'en montrer les avantages et surtout les inconvénients.

Par souci de commodité, ces algorithmes sont présentés uniquement dans l'algèbre initiale de BR de domaine $\{0,1\}$, c'est-à-dire l'anneau booléen à deux éléments $\{0,1\}$ noté $\mathbf{2}$, même s'ils sont valables pour tout modèle de BR.

Définition 5.1 *Les termes* booléens t *formés sur les symboles $\mathcal{F}_{BR}$ contenant n variables $x_1, \ldots, x_n$ seront notés si nécessaire $t(x_1, \ldots, x_n)$ et sont interprétés comme des fonctions booléennes de $\mathbf{2}^n \to \mathbf{2}$. Si $\underline{b} \in \mathbf{2}^n$ alors $t(\underline{b})$ correspond à la valeur de cette fonction en $\underline{b}$.*

Lemme 5.1 *Le problème d'unification entre termes booléens $t_1 =^? t_2$ est équivalent au problème de filtrage $t_1 \oplus t_2 =^? 0$.*

Preuve : Les équations $t_1 =^? t_2$, $t_1 \oplus t_2 =^? t_2 \oplus t_2$ puis finalement $t_1 \oplus t_2 =^? 0$ par l'axiome $x \oplus x = 0$ ont le même ensemble de solutions. □

Seules les équations $t =^? 0$ seront donc considérées par la suite.

L'unification dans les anneau booléens est unitaire. Elle s'apparente donc à l'unification dans la théorie vide, elle aussi unitaire. Pourtant, à la différence de la théorie vide, il existe un nombre infini d'unificateurs principaux équivalents entre eux. En effet, il existe un unique terme dans chaque classe d'équivalence modulo $\emptyset$ mais une infinité dans chaque classe d'équivalence modulo BR. Pour choisir un bon unificateur principal, il va donc falloir des critères comme celui introduit ci-dessous.

Définition 5.2 *Un unificateur σ de $t =^? 0$ tel que $\mathcal{VR}an(\sigma) = \mathcal{V}(t)$ est* reproductif *si pour tout $\underline{b}$* solution *de $t =^? 0$, i.e. $t(\underline{b}) = 0$, on a*

$$(x_1\sigma(\underline{b}), \ldots, x_n\sigma(\underline{b})) = \underline{b}.$$

Un unificateur reproductif possède donc la particularité de représenter l'ensemble des solutions.

Proposition 5.1 *Un unificateur reproductif est un unificateur principal.*

Preuve : Etant donnés ϕ un unificateur et σ un unificateur reproductif, on montre que $\sigma \leq_{BR}^{\mathcal{V}(t)} \phi$.

Soit $m = \mathcal{V}(t\phi)$ le nombre de variables apparaissant dans l'équation substituée. Pour tout m-uplet $\underline{b'}$ on a

$$\forall i \in \{1, \ldots, n\},\ x_i\sigma(x_1\phi(\underline{b'}), \ldots, x_n\phi(\underline{b'})) = x_i\phi(\underline{b'})$$

puisque ϕ est un unificateur de $t =^? 0$, c'est-à-dire

$$t(x_1\phi(\underline{b'}), \ldots, x_n\phi(\underline{b'})) = 0.$$

Nous avons donc $\forall x_i \in \mathcal{V}(t),\ x_i\sigma\phi = x_i\phi$ et par conséquent $\sigma \leq_{BR}^{\mathcal{V}(t)} \phi$. □

L'avantage des méthodes de Boole et Löwenheim est de retourner directement un unificateur reproductif et donc principal.

Théorème 5.1 *(Méthode de Boole [Boo47]). Soit un problème d'unification*

$$t(x_1, \ldots, x_n) =^? 0.$$

Si la substitution σ est définie par

$$\sigma = \{x_1 \mapsto (t(0, x_2, \ldots, x_n) \oplus t(1, x_2, \ldots, x_n) \oplus 1) \wedge x_1 \oplus t(0, x_2, \ldots, x_n)\}\sigma',$$

et σ' un unificateur principal de

$$t(0, x_2, \ldots, x_n) \wedge t(1, x_2, \ldots, x_n) =^? 0$$

alors σ est un unificateur principal de $t =^? 0$.

Pour le début de la récurrence, il s'agira de remarquer que l'identité est un unificateur reproductif si $t = 0$.

Preuve : (Idée). On se contente de vérifier par récurrence sur le nombre de variables n que si σ est un unificateur de $t =^? 0$ alors il est reproductif.

Lorsque $n = 0$, on peut remarquer que l'identité est un unificateur reproductif si $t = 0$.

Considérons $n > 0$ et une solution $\underline{b}$ de $t =^? 0$. Soit $\underline{b} = (0, b_2, \ldots, b_n)$ et $x_1\sigma(\underline{b}) = t(0, b_2, \ldots, b_n) = 0$. Soit $\underline{b} = (1, b_2, \ldots, b_n)$ et $x_1\sigma(\underline{b}) = t(0, b_2, \ldots, b_n) \oplus t(0, b_2, \ldots, b_n) \oplus 1 = 1$. Dans tous les cas, la première composante est reproduite. En supposant σ' reproductif, on obtient

$$(x_1\sigma(\underline{b}), x_2\sigma'(\underline{b}), \ldots, x_n\sigma'(\underline{b})) = (x_1\sigma(\underline{b}), b_2, \ldots, b_n) = (b_1, \ldots, b_n).$$

□

Exemple 5.1 *Considérons le terme $t = (x_1 \oplus x_2) \wedge x_1$. L'unificateur principal est*

$$\sigma = \{x_1 \mapsto (0 \oplus (1 \oplus x_2) \oplus 1) \wedge x_1\}\sigma',$$

avec σ' unificateur principal de $0 \wedge x_2 =^? 0$. Comme $0 \wedge x_2 =_{BR} 0$, on peut prendre la substitution identité pour σ'. Après simplification, on obtient $\sigma = \{x_1 \mapsto x_1 \wedge x_2\}$ et l'on vérifie que

$$t\sigma = (x_1 \wedge x_2 \oplus x_2) \wedge x_1 = x_1 \wedge x_2 \oplus x_1 \wedge x_2 = 0.$$

L'algorithme implanté dans CHIP et présenté dans [BS87] se base sur la méthode de Boole. Les variables introduites par unification sont renommées, et toutes les variables du problème à unifier ne sont pas nécessairement instanciées.

Le renommage des variables dans l'unificateur principal permet de considérer l'unificateur principal comme un problème équationnel en forme résolue, et d'adopter ainsi une approche par règles de transformation de problèmes équationnels.

Les variables n'ont pas à être toutes instanciées si le test d'arrêt est $t =_{BR} 0$ avec t non clos, au lieu de $t = 0$ avec t clos. Cette optimisation a déjà été pratiquée dans l'exemple 5.1.

Une autre implantation de la méthode figure dans [RT90].

L'autre méthode dûe à Löwenheim nécessite la connaissance d'une solution particulière, un problème annexe traité dans [MN88] pour le cas général et qui n'est pas considéré ici car évident dans **2**.

Théorème 5.2 *(Méthode de Löwenheim [Löw08]). Soit $\underline{b}'$ une solution particulière de*

$$t(x_1, \ldots, x_n) =^? 0.$$

La substitution σ définie par

$$\sigma = \bigcup_{i=1}^{n} \{x_i \mapsto x_i \oplus t(x_1, \ldots, x_n) \wedge (x_i \oplus b'_i)\}$$

est un unificateur principal de $t =^? 0$.

Preuve : Cette méthode retourne un unificateur reproductif: si $t(\underline{b}) = 0$ alors

$$\begin{aligned}(x_1\sigma(\underline{b}), \ldots, x_n\sigma(\underline{b})) &= (b_1 \oplus t(\underline{b}) \wedge (b_1 \oplus b'_1), \ldots, b_n \oplus t(\underline{b}) \wedge (b_n \oplus b'_n)) \\ &= (b_1 \oplus 0 \wedge (b_1 \oplus b'_1), \ldots, b_n \oplus 0 \wedge (b_n \oplus b'_n)) \\ &= (b_1, \ldots, b_n).\end{aligned}$$

Pour les non solutions $\underline{b}$ vérifiant $t(\underline{b}) = 1$, on obtient

$$\begin{aligned}(x_1\sigma(\underline{b}), \ldots, x_n\sigma(\underline{b})) &= (b_1 \oplus (b_1 \oplus b'_1), \ldots, b_n \oplus (b_n \oplus b'_n)) \\ &= (b'_1, \ldots, b'_n).\end{aligned}$$

□

L'ensemble des solutions est reproduit et on associe aux non solutions une solution particulière $\underline{b}'$. Ce principe sera généralisé dans la section suivante.

Exemple 5.2 *Considérons le terme $t = x_1 \wedge x_2$ et la solution particulière $(0, 0)$ telle que $t(0, 0) = 0$. Un unificateur principal est*

$$\sigma = \{x_1 \mapsto x_1 \oplus x_1 \wedge x_2, x_2 \mapsto x_2 \oplus x_1 \wedge x_2\}$$

et l'on vérifie que $t\sigma = (x_1 \oplus x_1 \wedge x_2) \wedge (x_2 \oplus x_1 \wedge x_2) = x_1 \wedge x_2 \oplus x_1 \wedge x_2 = 0$.

Le gros inconvénient des deux méthodes, surtout de la seconde, est d'introduire autant de variables que de variables initialement dans le problème. L'unification ne simplifie pas le problème en nombre de variables. L'algorithme [BES+90] exposé plus en détail par la suite n'a pas cet inconvénient, il introduit un nombre minimal de variables. Le problème après unification s'en trouve simplifié, même si l'ensemble des solutions est conservé. Le prix à payer est le calcul cette fois de l'ensemble des solutions et non plus, comme pour la méthode de Löwenheim, d'une seule solution.

5.2 Algèbres booléennes et primales

Nous avons déjà introduit de façon implicite une algèbre booléenne en définissant le $\vee$ (ou) et le $^-$ (non) par rapport à un anneau booléen. Il aurait été tout aussi naturel [Rud74] d'introduire d'abord la notion d'algèbre booléenne puis celle d'anneau en posant $x \oplus y = (\bar{x} \wedge y) \vee (x \wedge \bar{y})$. La présentation équationnelle BA sur l'ensemble des symboles $\mathcal{F}_{BA} = \{\mathbf{0}, \mathbf{1}, \wedge, \vee, ^-\}$ ci-dessous est celle tirée de [SA91].

$$BA = \begin{cases} x \vee \mathbf{0} = x \\ x \wedge \mathbf{1} = x \\ x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \\ x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \\ (x \vee y) \wedge y = y \\ (x \wedge y) \vee y = y \\ x \vee \bar{x} = \mathbf{1} \\ x \wedge \bar{x} = \mathbf{0} \\ x \wedge y = y \wedge x \\ (x \wedge y) \wedge z = x \wedge (y \wedge z) \\ x \vee y = y \vee x \\ (x \vee y) \vee z = x \vee (y \vee z) \end{cases}$$

Une *algèbre booléenne* est un modèle de BA.

La structure d'anneau booléen a souvent été préférée en déduction automatique à celle d'algèbre booléenne car elle offre, nous l'avons déjà évoqué, un parfait exemple de réécriture modulo un ensemble d'égalités AC alors que la présentation équationnelle BA n'admet pas de système réécriture canonique [SA91].

Pourtant, la réécriture modulo AC, bien qu'elle fournisse une méthode élégante de décision de l'égalité, n'est pas un outil particulièrement efficace. Par ailleurs, le fait que l'égalité dans BR soit aussi celle dans $\mathbf{2}$ incite à ne considérer que l'anneau booléen $\mathbf{2}$ dans lequel le problème du mot est simplement un problème de combinatoire.

Nous allons nous intéresser plus particulièrement aux algèbres *finies* dont le domaine et l'ensemble des opérations sont finis. Un exemple de base est l'algèbre de Boole $\mathcal{B}$ à deux éléments $\{0, 1\}$. Il est bien connu dans cette algèbre que toute fonction de $\{0,1\}^m$ vers $\{0,1\}$ peut être représentée par un terme qui est la somme de ses monômes. Cette décomposition symbolique est du plus grand interêt pour l'unification. Les algèbres vérifiant cette propriété bien singulière sont dites *primales*.

Dans un premier temps, on précise la relation entre terme et fonction.

Soit $\mathcal{A}$ une $\mathcal{F}$-algèbre de domaine A avec $\mathcal{F}$ un ensemble fini de symboles. Un terme t formé sur l'ensemble des symboles de fonctions $\mathcal{F}$ et un ensemble de variables $x_1, \ldots, x_m$ définit une fonction $t^{\mathcal{A}}$ de A^m vers A comme suit:

$$\forall (a_1, \ldots, a_m) \in A^m,\ t^{\mathcal{A}}(a_1, \ldots, a_m) = \underline{\alpha}(t),$$

où α est une assignation qui associe la valeur a_i à la variable x_i pour tout $i \in [1 \ldots m]$.

Réciproquement, si toute fonction est l'interprétation fonctionnelle d'un terme alors $\mathcal{A}$ est dite *primale* [Fos53, Nip90].

Définition 5.3 *Une $\mathcal{F}$-algèbre $\mathcal{A}$ est* primale *si toute fonction $f : A^m \to A$ d'arité m non nulle est égale à $t^{\mathcal{A}}$ pour un terme t de $T(\mathcal{F}, \mathcal{X})$.*

Une algèbre primale est par définition une algèbre engendrée par les termes. Mais la réciproque est fausse puisque l'algèbre de termes avec un ensemble fini de symboles n'est pas primale.

Proposition 5.2 *Si $\mathcal{A}$ est une $\mathcal{F}$-algèbre primale ayant un ensemble de symboles de fonctions $\mathcal{F}$ fini, alors $\mathcal{A}$ est une algèbre finie.*

Preuve : Si le domaine A de $\mathcal{A}$ est infini, alors

$$\{f_A \circ g \mid g : A^m \to A^p, f_A \in (\mathcal{F}_p)_A\}$$

est strictement inclus dans $\{A^m \to A\}$. □

Nous supposerons dorénavant un ensemble fini de symboles de fonctions $\mathcal{F}$ et sous cette condition, une algèbre primale est une algèbre finie. Mais la réciproque est évidemment fausse. L'idée de W. Büttner pour manipuler de façon homogène l'ensemble des algèbres finies est d'enrichir leurs structures algébriques pour que toutes deviennent primales. Nous allons nous placer dans le même contexte.

Définition 5.4 *Soit $\mathcal{A}$ une $\mathcal{F}$-algèbre de domaine $A = \{0, \ldots, n-1\}$. L'*algèbre finie enrichie $\overline{\mathcal{A}}$ *a pour domaine A et pour symboles de fonctions*

$$\overline{\mathcal{F}} = \mathcal{F} \cup \{\bot, [1], \ldots, [n-2], \top, C_0, \ldots, C_{n-1}, +, \cdot\}$$

interprétés comme suit:

- $\bot_A = 0$
- $\top_A = n-1$
- $\forall i \in A \setminus \{0, n-1\},\ [i]_A = i$
- $\forall i \in A, \forall x \in A,\ C_{iA}(x) = n-1$ si $x = i, 0$ sinon
- $\forall (x,y) \in A^2,\ x +_A y = \max(x,y)$
- $\forall (x,y) \in A^2,\ x \cdot_A y = \min(x,y)$.

Exemple 5.3 *L'algèbre de domaine $A = \{0,1\}$ muni des seuls opérateurs d'enrichissement est l'algèbre de Boole $\mathcal{B}$ à un renommage près des symboles: le symbole $+$ correspondant à $\vee$ (ou), $\cdot$ à $\wedge$ (et), C_0 à $^-$ (non). L'opérateur C_1 est l'identité: $C_1(x) = x$. L'algèbre de Boole sera souvent pris pour les exemples; c'est la plus intéressante "sur le papier" car la combinatoire inhérente au domaine y est limitée.*

L'algèbre $\overline{\mathcal{A}}$ est primale puisque toute fonction $f : A^m \to A$ est égale à l'interprétation fonctionnelle du terme

$$\sum_{(a_1,\ldots,a_m) \in A^m} C_{a_1}(x_1) \cdots C_{a_m}(x_m) \cdot [f(a_1, \ldots, a_m)] \qquad (POST)$$

où $[f(a_1, \ldots, a_m)]$ est la constante dont l'interprétation vaut la valeur de f en $(a_1, \ldots, a_m)$. Ce terme représente la table de vérité de la fonction f. L'adjonction de nouveaux opérateurs permet ainsi une représentation symbolique d'une fonction sur un domaine fini.

5.3 Une présentation équationnelle pour les algèbres primales

Nous allons maintenant construire une théorie équationnelle qui, par remplacement d'égal par égal, permet d'obtenir une forme canonique pour tout terme t qui est la décomposition de $(POST)$ de la fonction $t^{\mathcal{A}}$. La table de vérité de l' interprétation fonctionnelle d'un terme t peut donc être obtenue par raisonnement équationnel à partir de la connaissance des tables de vérité des fonctions $\overline{\mathcal{F}}_A$.

Définition 5.5 *Soit AF l'ensemble fini d'axiomes sur $\mathcal{T}(\overline{\mathcal{F}}, \mathcal{X})$ donné en figure 5.1.*

$$
\begin{array}{rcll}
x + (y + z) & = & (x + y) + z & (1) \\
x + y & = & y + x & (2) \\
x \cdot (y \cdot z) & = & (x \cdot y) \cdot z & (3) \\
x \cdot y & = & y \cdot x & (4) \\
x \cdot (y + z) & = & x \cdot y + x \cdot z & (5) \\
x + (y \cdot z) & = & (x + y) \cdot (x + z) & (6) \\
x + \bot & = & x & (7) \\
x + \top & = & \top & (8) \\
x \cdot \bot & = & \bot & (9) \\
x \cdot \top & = & x & (10) \\
x + x & = & x & (11) \\
x \cdot x & = & x & (12) \\
\forall f \in \overline{\mathcal{F}}_p, \forall i \in A,\ C_i(f(x_1, \ldots, x_p)) & = & \sum_{f_A(i_1,\ldots,i_p)=i} C_{i_1}(x_1) \cdots C_{i_p}(x_p) & (13) \\
\forall i \in A,\ C_i([i]) & = & \top & (14) \\
\forall (i,j) \in A^2, i \neq j,\ C_i(x) \cdot C_j(x) & = & \bot & (15) \\
\sum_{i=0}^{n-1} C_i(x) & = & \top & (16) \\
\sum_{i=0}^{n-1} C_i(x) \cdot [i] & = & x & (17)
\end{array}
$$

Figure 5.1. La présentation équationnelle AF

Lorsque $n = 2$ et $\mathcal{F} = \emptyset$ alors cet ensemble d'axiomes est une autre présentation de la théorie équationnelle BA. Nous verrons que la présentation AF joue le même rôle pour $\overline{\mathcal{A}}$ que BA pour l'algèbre de Boole $\mathcal{B}$.

Exemple 5.4 *L' égalité $C_i([j]) = \bot$ avec $i \neq j$ est un théorème dans AF:*
$\forall (i,j) \in A^2, i \neq j,$

$$
\begin{array}{rcl}
\bot & =_{AF} & C_i([j]) \cdot C_j([j]) \text{ par } (15) \\
& =_{AF} & C_i([j]) \cdot \top \text{ par } (14) \\
& =_{AF} & C_i([j]) \text{ par } (10)
\end{array}
$$

Définition 5.6 *La* forme canonique *d'un terme t de $T(\overline{\mathcal{F}}, \mathcal{X})$ est*

$$\sum_{\{\alpha : \mathcal{V}(t) \to A\}} \prod \alpha(\mathcal{V}(t)) \cdot [\underline{\alpha}(t)] \qquad (CAN)$$

Cette forme canonique est à comparer à la décomposition précédente ($POST$) où f vaut $t^{\mathcal{A}}$ et $f(a_1,\ldots,a_m)$ vaut $\underline{\alpha}(t)$. Afin de simplifier la notation, le produit $\prod_{x\in V} C_{\alpha(x)}(x)$ sera désormais désigné par $\prod \alpha(V)$ et appelé atome. Cette terminologie provient de [BES+90] où l'ensemble des interprétations fonctionnelles $\prod \alpha(V)$ sont les atomes d'une algèbre booléenne de fonctions.

Remarque : La somme vide est définie égale à $\bot$ et le produit vide égal à $\top$. Avec ces définitions, pour un terme clos on a

$$t =_{AF} \sum_{\{\alpha:\emptyset\to A\}} \top \cdot [\underline{\alpha}(t)] =_{AF} [\underline{\alpha}(t)],$$

où $\underline{\alpha}$ est l'unique homomorphisme de $\mathcal{T}(\overline{\mathcal{F}})$ vers $\overline{\mathcal{A}}$ puisque $\mathcal{T}(\overline{\mathcal{F}})$ est l'algèbre initiale de la classe des $\overline{\mathcal{F}}$-algèbres.

Lemme 5.2 *Un terme t de $T(\overline{\mathcal{F}},\mathcal{X})$ est égal modulo AF à sa forme canonique.*

Preuve : On prouve tout d'abord par induction structurelle sur les termes que

$$\forall i\in A, \forall t\in T(\overline{\mathcal{F}},\mathcal{X}),\ C_i(t) =_{AF} \sum_{\{\alpha:\mathcal{V}(t)\to A|\underline{\alpha}(t)=i\}} \prod \alpha(\mathcal{V}(t)). \qquad (CAN_i)$$

Cette égalité est vrai pour les variables:

$$\forall x\in\mathcal{X},\ C_i(x) =_{AF} \sum_{\{\alpha:\{x\}\to A|\alpha(x)=i\}} C_{\alpha(x)}(x)$$

car il s'agit ici d'une égalité syntaxique. Pour les constantes, on a:

$$\forall j\in A,\ C_i([j]) =_{AF} \sum_{\{\alpha:\emptyset\to A|\alpha([j])=i\}} \top$$

Si $j=i$ alors la somme est non vide et donc $C_i([j]) =_{AF} \top$, sinon la somme est vide et $C_i([j]) =_{AF} \bot$.

Considérons maintenant un terme $t=f(t_1,\ldots,t_p)$. On suppose (CAN_i) vrai sur les sous termes $t_1,\ldots,t_p$. Il faut s'intéresser plus particulièrement au produit de $C_{i_1}(t_1)$ et $C_{i_2}(t_2)$ qui est égal modulo AF à la somme des produits

$$\prod \alpha_1(\mathcal{V}(t_1)) \cdot \prod \alpha_2(\mathcal{V}(t_2)),$$

pour tout couple

$$(\alpha_1,\alpha_2)\in\{\alpha_1:\mathcal{V}(t_1)\to A \mid \underline{\alpha_1}(t_1)=i_1\}\times\{\alpha_2:\mathcal{V}(t_2)\to A \mid \underline{\alpha_2}(t_2)=i_2\}$$

tel que

$$\forall x\in\mathcal{V}(t_1)\cap\mathcal{V}(t_2),\ \alpha_1(x)=\alpha_2(x). \qquad (\star)$$

Les assignations $\alpha_1\cup\alpha_2$, uniquement définies lorsque (α_1,α_2) satisfont la condition $(\star)$, sont les assignations $\alpha:\mathcal{V}(t_1)\cup\mathcal{V}(t_2)\to A$ telles que $\underline{\alpha}(t_1)=i_1$ et $\underline{\alpha}(t_2)=i_2$. Le produit

$$\prod(\alpha_1\cup\alpha_2)(\mathcal{V}(t_1)\cup\mathcal{V}(t_2))$$

est égal modulo AF à

$$\prod \alpha_1(\mathcal{V}(t_1)) \cdot \prod \alpha_2(\mathcal{V}(t_2)),$$

puisque $(\cdot)$ est idempotent. Les produits associés aux couples ne satisfaisant pas la condition $(\star)$ sont AF-égaux à $\bot$ par l'axiome (15). Par conséquent

$$C_{i_1}(t_1) \cdot C_{i_2}(t_2) =_{AF} \sum_{\{\alpha:\mathcal{V}(t_1)\cup\mathcal{V}(t_2)\to A \mid \wedge_{j=1}^{2} \underline{\alpha}(t_j)=i_j\}} \prod \alpha(\mathcal{V}(t_1)\cup\mathcal{V}(t_2)),$$

et plus généralement, lorsque le produit est itéré, on obtient

$$\prod_{j=1}^{p} C_{i_j}(t_j) =_{AF} \sum_{\{\alpha:\cup_{j=1}^{p}\mathcal{V}(t_j)\to A \mid \wedge_{j=1}^{p} \underline{\alpha}(t_j)=i_j\}} \prod \alpha(\cup_{j=1}^{p}\mathcal{V}(t_j)).$$

En utilisant ensuite l'axiome (13), on peut conclure que $C_i(t)$ est AF-égal à la somme des produits $\prod \alpha(\mathcal{V}(t))$ tels que il existe $(i_1, \ldots, i_p) \in A^p$ vérifiant $f_A(i_1, \ldots, i_p) = i$ et la conjonction

$$\bigwedge_{j=1}^{p} \underline{\alpha}(t_j) = i_j.$$

Ou encore

$$C_i(t) =_{AF} \sum_{\{\alpha:\mathcal{V}(t)\to A \mid \underline{\alpha}(t)=i\}} \prod \alpha(\mathcal{V}(t)).$$

Finalement par l'axiome (17), on trouve la forme canonique annoncée pour tout terme t:

$$t =_{AF} \sum_{\{\alpha:\mathcal{V}(t)\to A\}} \prod \alpha(\mathcal{V}(t)) \cdot [\underline{\alpha}(t)].$$

□

Il est possible de rajouter aux variables de t d'autres variables V car

$$\begin{aligned} \sum_{\{\alpha:V\to A\}} \prod \alpha(V) &=_{AF} \prod_{x\in V} \Big(\sum_{\{\alpha:\{x\}\to A\}} C_{\alpha(x)}(x) \Big) \\ &=_{AF} \prod_{x\in V} \top \text{ by (16)} \\ &=_{AF} \top. \end{aligned}$$

Ensuite

$$\begin{aligned} t &=_{AF} \sum_{\{\alpha:\mathcal{V}(t)\to A\}} \top \cdot \prod \alpha(\mathcal{V}(t)) \cdot [\underline{\alpha}(t)] \\ &=_{AF} \sum_{\{\alpha:\mathcal{V}(t)\cup V\to A\}} \prod \alpha(\mathcal{V}(t)\cup V) \cdot [\underline{\alpha}(t)]. \end{aligned}$$

Cette adjonction de variables dans la forme canonique va servir pour la preuve de la prochaine proposition. Celle-ci permet d'expliquer pourquoi l'unification dans l'algèbre de Boole et l'unification dans la classe des modèles de BA est identique. Une propriété analogue existe entre $\overline{\mathcal{A}}$ et AF qui justifie l'utilisation de termes à la place de fonctions.

Proposition 5.3 *Les $\overline{\mathcal{F}}$-algèbres $\overline{\mathcal{A}}$ et $\mathcal{T}(\overline{\mathcal{F}},\mathcal{X})/=_{AF}$ ont le même ensemble de théorèmes équationnels: pour toute égalité $(s=t)$, $\mathcal{A}\models s=t$ si et seulement si $s=_{AF}t$.*

Preuve : ($\Rightarrow$: une égalité valide dans $\overline{\mathcal{A}}$ est une AF-égalité.). Le lemme 5.2 établit que

$$s=_{AF}\sum_{\{\alpha:\mathcal{V}(s)\cup\mathcal{V}(t)\to A\}}\prod\alpha(\mathcal{V}(s)\cup\mathcal{V}(t))\cdot[\underline{\alpha}(s)]$$

et

$$t=_{AF}\sum_{\{\alpha:\mathcal{V}(t)\cup\mathcal{V}(s)\to A\}}\prod\alpha(\mathcal{V}(t)\cup\mathcal{V}(s))\cdot[\underline{\alpha}(t)].$$

Avec par hypothèse

$$\forall\alpha\in ASS_A^{\mathcal{X}},\ \underline{\alpha}(s)=\underline{\alpha}(t).$$

Par conséquent $s=_{AF}t$.

($\Leftarrow$) Il suffit de vérifier que tous les axiomes de la présentation AF sont valides dans $\overline{\mathcal{A}}$. □

Proposition 5.4 *Les $\overline{\mathcal{F}}$-algèbres $\overline{\mathcal{A}}$ et $\mathcal{T}(\overline{\mathcal{F}})/=_{AF}$ sont isomorphes.*

Preuve : Par application du lemme 5.2 aux termes clos, on obtient

$$\forall t\in T(\overline{\mathcal{F}}),\ t=_{AF}[h(t)],\qquad(\star\star)$$

où h est l'unique homomorphisme de $\mathcal{T}(\overline{\mathcal{F}})$ vers $\overline{\mathcal{A}}$. C'est une surjection. En considérant $\mathcal{T}(\overline{\mathcal{F}})$ quotientée par $=_{AF}$ on obtient une bijection puisque

- si $h(s)=h(t)$ alors $s=_{AF}t$ par $(\star\star)$,
- si $h(s)\neq h(t)$ alors $\overline{\mathcal{A}}\not\models h(s)]=[h(t)]$ et $[h(s)]\neq_{AF}[h(t)]$, grâce à la proposition 5.3. Donc $s\neq_{AF}t$.

□

Corollaire 5.1 *La présentation $(\overline{\mathcal{F}},AF)$ est ω-complète, i.e. les algèbres $\mathcal{T}(\overline{\mathcal{F}},\mathcal{X})/=_{AF}$ et $\mathcal{T}(\overline{\mathcal{F}})/=_{AF}$ ont les mêmes théorèmes équationnels.*

Ce résultat peut s'appliquer à toute algèbre primale $\mathcal{A}$ pours lesquelles les opérateurs $+,\cdot,C_0,\ldots,C_{n-1}$ sont alors inutiles puisque il existe des termes

$$t_+,t_\cdot,t_{C_0},\ldots,t_{C_{n-1}},t_\bot,\ldots,t_\top$$

dont les interprétations fonctionnelles sont respectivement les fonctions

$$+_A,\cdot_A,C_{0A},\ldots,C_{n-1A},\bot_A,\ldots,\top_A.$$

Il est alors possible de les réécrire dans les axiomes de AF pour obtenir une présentation PA. Cette méthode "par réécriture" n'est absolument pas opérationnelle mais donne un schéma de construction d'une présentation équationnelle pour n'importe quelle algèbre

primale. Pour formaliser cette traduction, il suffit de considérer le système de réécriture suivant:

$$TRANS = \begin{cases} x + y & \rightarrow & t_+ \\ x \cdot y & \rightarrow & t_. \\ C_0(x) & \rightarrow & t_{C_0} \\ \vdots & & \\ C_{n-1}(x) & \rightarrow & t_{C_{n-1}} \\ \bot & \rightarrow & t_\bot \\ \vdots & & \\ \top & \rightarrow & t_\top \end{cases}$$

où les axiomes non orientés correspondants sont par définition valides dans $\overline{\mathcal{A}}$. Ce système de réécriture est trivialement convergent:

- La terminaison peut être prouvée en prenant un ordre de simplification comme un RPO où les symboles de fonctions $\overline{\mathcal{F}} \backslash \mathcal{F}$ sont supérieurs que tous les symboles de $\mathcal{F}$ suivant l'ordre de précédence.
- La confluence locale est dûe à l'absence de paires critiques.

Pour tout terme $t \in \mathcal{T}(\overline{\mathcal{F}}, \mathcal{X})$ sa forme normale par rapport à $TRANS$, désignée par $t \downarrow_{TRANS}$, est un terme de $\mathcal{T}(\mathcal{F}, \mathcal{X})$. Soit PA la présentation obtenue par normalisation par rapport à $TRANS$ des axiomes AF:

$$PA = \{(l \downarrow_{TRANS} = r \downarrow_{TRANS}) | (l = r) \in AF\}.$$

On pourra remarquer que tout axiome de PA est valide dans A. Il est ensuite facile de prouver par induction sur la longueur des preuves $=_{AF}$ que

$$\forall s, t \in T(\overline{\mathcal{F}}, \mathcal{X}),\ s =_{AF} t \Longrightarrow s \downarrow_{TRANS} =_{PA} t \downarrow_{TRANS} .$$

La réciproque est aussi vraie: $s \downarrow_{TRANS} =_{PA} t \downarrow_{TRANS}$ implique que $s \downarrow_{TRANS} =_{\mathcal{A}} t \downarrow_{TRANS}$ avec $s =_{\overline{\mathcal{A}}} s \downarrow_{TRANS}$ et $t =_{\overline{\mathcal{A}}} t \downarrow_{TRANS}$. Ainsi $s =_{\overline{\mathcal{A}}} t$ et $s =_{AF} t$.

Comme les termes de $T(\mathcal{F}, \mathcal{X})$ sont en forme normale par rapport à $TRANS$, nous avons

$$\forall s, t \in T(\mathcal{F}, \mathcal{X}),\ s =_{PA} t \Longleftrightarrow s =_{AF} t \Longleftrightarrow s =_{\overline{\mathcal{A}}} t \Longleftrightarrow s =_{\mathcal{A}} t.$$

De façon analogue, on peut observer que $\mathcal{T}(\mathcal{F})/=_{PA}$ et $\mathcal{A}$ sont isomorphes, ce qui conduit au théorème suivant:

Théorème 5.3 *Une algèbre primale est l'algèbre initiale d'une présentation équationnelle ω-complète.*

Ce théorème va justifier les techniques de combinaison mises en œuvre dans le chapitre suivant.

5.4 Unification dans les algèbres finies

Les méthodes classiques d'unification booléenne ont été étendues aux algèbres primales par T. Nipkow [Nip88, Nip90]. Nous allons à présent les appliquer à notre contexte en reprenant rigoureusement les mêmes notations que celles utilisées pour l'unification dans **2**, ainsi que la notion d'unificateur reproductif.

Lemme 5.3 *Le problème d'unification* $t_1 =^? t_2$ *entre termes* $t_1, t_2 \in \mathcal{T}(\overline{\mathcal{F}}, \mathcal{X})$ *est équivalent à un problème de filtrage* $t =^? 0$.

Preuve : Le terme t vaut

$$C_0(\sum_{i \in A} C_i(t_1) \cdot C_i(t_2)).$$

□

Dorénavant, seules les équations $t =^? 0$ seront considérées.

La méthode de Boole ou élimination successive de variables est basée sur le lemme suivant.

Lemme 5.4 *L' équation* $t(x_1, \ldots, x_m) =^? 0$ *a une solution si et seulement si*

$$\prod_{\underline{i} \in A^m} t(\underline{i}) = 0.$$

La construction d'un unificateur reproductif par élimination successive nécessite un terme *discriminant* pour sélectionner un m-uplet lorsqu'il est ou non solution.

Définition 5.7 *Le terme* discriminant $d(x, y, z)$ *est défini comme suit:*

$$d(x, y, z) = C_0(x) \cdot y + (\sum_{i \neq 0} C_i(x)) \cdot z$$

La fonction correspondante vaut y si $x = 0$ sinon elle vaut z. Ce terme est utilisé pour la construction d'un unificateur reproductif.

Théorème 5.4 *(Méthode de Boole). Soit un problème d'unification*

$$t(x_1, \ldots, x_m) =^? 0.$$

Si la substitution σ *est définie par*

$$\sigma = \{x_1 \mapsto (d(t(x_1, \ldots, x_m), x_1, \sum_{i \in A} C_0(t(i, x_2, \ldots, x_m)) \cdot [i])\}\sigma',$$

et σ' *un unificateur principal de*

$$\prod_{i \in A} t([i], x_2, \ldots, x_m) =^? 0$$

alors σ *est un unificateur principal de* $t =^? 0$.

Preuve : On montre par récurrence sur le nombre de variables que

$$(x_1\sigma(\underline{i}), \ldots, x_m\sigma(\underline{i})) = \underline{i}$$

si $t(\underline{i}) = 0$, ou sinon il existe un m-uplet $\underline{j}$ tel que $t(\underline{j}) = 0$ et

$$(x_1\sigma(\underline{i}), \ldots, x_m\sigma(\underline{i})) = \underline{j}.$$

Si $t(i_1, \underline{i}) = 0$ alors

$$\begin{aligned} x_1\sigma(i_1, \underline{i}) &= d(t(i_1, x_2\sigma'(\underline{i}), \ldots, x_m\sigma'(\underline{i})), i_1, \ldots) \\ &= d(t(\underline{i}), i_1, \ldots) \end{aligned}$$

car σ' satisfait l'hypothèse de récurrence. Ensuite, comme $t(i_1, \underline{i}) = 0$, on a $x_1\sigma(\underline{i}) = i_1$. Pour les autres variables d'indice $k > 1$, $x_k\sigma(\underline{i}) = x_k\sigma'(\underline{i}) = i_k$ par hypothèse.

Si $t(i_1, \underline{i}) \neq 0$ alors

$$\begin{aligned} x_1\sigma(i_1, \underline{i}) &= d(t(i_1, x_2\sigma'(\underline{i}), \ldots, x_m\sigma'(\underline{i})), i_1, \ldots) \\ &= d(t(i_1, j_2, \ldots, j_m), i_1, \ldots) \end{aligned}$$

car σ' satisfait l'hypothèse de récurrence. Si $t(i_1, j_2, \ldots, j_m) = 0$ alors $j_1 = i_1$, sinon il existe un élément j_1 tel que $t(j_1, \ldots, j_m) = 0$ et

$$j_1 = \sum_{i \in A} C_0(t(j_1, \ldots, j_m)) \cdot i.$$

Ainsi il existe $(j_1, \ldots, j_m)$ tel que $t(j_1, \ldots, j_m) = 0$ et

$$(x_1\sigma(i_1, \underline{i}), \ldots, x_m\sigma'(i_1, \underline{i})) = (j_1, \ldots, j_m)$$

avec $t(j_1, \ldots, j_m) = 0$, c'est-à-dire

$$(x_1\sigma(i_1, \underline{i}), \ldots, x_m\sigma(i_1, \underline{i})) = (j_1, \ldots, j_m)$$

puisque $x_k\sigma = x_k\sigma'$ pour $k > 1$.

Par conséquent, pour tout $\underline{i} \in A^m$, on a

$$t(x_1\sigma(\underline{i}), \ldots, x_m\sigma(\underline{i})) = 0,$$

et la substitution σ est donc un unificateur de $t =^? 0$. Cet unificateur est principal puisque reproductif. □

Exemple 5.5 *Soit la $\overline{\mathcal{F}}$-algèbre finie de domaine $\{0, 1, 2\}$ avec $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, C_0, C_1, C_2, +, \cdot, *_3\}$ et $*_3$ interprété par la multiplication modulo 3.*

Considérons le problème $x_1 *_3 x_2 =^? 0$. Un unificateur principal est

$$\sigma = \{x_1 \mapsto C_0(x_1 *_3 x_2) \cdot x_1 + (C_0(x_2) \cdot \mathbf{1} + C_0(\mathbf{2} *_3 x_2) \cdot \mathbf{2}) \cdot (C_1(x_1 *_3 x_2) + C_2(x_1 *_3 x_2))\}\sigma'$$

où σ' est la substitution identité puisque $(0 *_3 x_2) \cdot x_2 =^? 0 \Leftrightarrow 0 =^? 0$.

On remarquera que l'interprétation de $*_3$ ne sert que pour le test d'arrêt.

Le terme discriminant permet aussi de construire un unificateur reproductif par la méthode de Löwenheim.

Théorème 5.5 *(Méthode de Löwenheim). Soit* $\underline{i'} = (i'_1, \ldots, i'_m) \in A^m$ *une solution particulière de*

$$t(x_1, \ldots, x_m) =^? 0.$$

La substitution σ *définie par*

$$\sigma = \bigcup_{k=1}^{m} \{x_k \mapsto d(t(x_1, \ldots, x_m), x_k, i'_k)\}$$

est un unificateur principal de $t =^? 0$.

Preuve : On montre que la substitution σ est un unificateur reproductif.

Si $t(\underline{i}) = 0$ alors

$$\forall k \in \{1, \ldots, m\},\ x_k\sigma(\underline{i}) = d(t(\underline{i}), i_k, i'_k) = i_k.$$

Si $t(\underline{i}) \neq 0$ alors

$$\forall k \in \{1, \ldots, m\},\ x_k\sigma(\underline{i}) = d(t(\underline{i}), i_k, i'_k) = i'_k.$$

Par conséquent,

$$\forall \underline{i} \in A^m,\ t(\sigma(\underline{i}), \ldots, \sigma(\underline{i})) = 0,$$

et σ est un unificateur principal. □

Exemple 5.6 *(Suite de l'Exemple 5.5). L'unification de* $x_1 *_3 x_2 =^? 0$ *par la méthode de Löwenheim donne*

$$\{x_1 \mapsto C_0(x_1 *_3 x_2) \cdot x_1, x_2 \mapsto C_0(x_1 *_3 x_2) \cdot x_2\}$$

avec la solution particulière $(0, 0)$. *On remarquera que l'interprétation de* $*_3$ *n'est nécessaire que pour le calcul de cette solution particulière.*

L'unification par ces deux méthodes ne simplifie pas vraiment le problème au sens où les mêmes variables sont réintroduites. L'algorithme peut éventuellement s'appliquer à d'autres types de contraintes mais il faut qu'elles soient préalablement transformées en un problème de filtrage $t =^? 0$. L'algorithme que nous allons voir permet de traiter les contraintes en général.

5.5 Un résolveur de contraintes

L'algorithme exposé maintenant ne repose pas sur les méthodes de Boole et Löwenheim. Cet algorithme, proposé pour la première fois par W. Büttner et al. [BES+90], a l'avantage d'introduire aussi peu de variables que possible. La technique de preuve utilisée ici est plus directe que celle donnée dans [BES+90] et elle permet de s'appliquer directement à des contraintes quelconques et non plus seulement équationnelles.

Définition 5.8 *(Langage primal). Le* langage de contraintes primal $CL_{AF}[\overline{\Sigma}, \mathcal{X}]$ *est défini par la signature* $\overline{\Sigma} = (\{s_*\}, \overline{\mathcal{F}}, \mathcal{P})$, *la* $\overline{\mathcal{F}}$*-algèbre* $\overline{\mathcal{A}}$ *et un ensemble de relations* $\mathcal{P}_A$ *sur* A.

Dans ce langage de contraintes primal, Nous allons montrer qu'un ensemble complet minimal de solutions symboliques contient au plus un élément qu'on appellera solution principale.

Les contraintes auquels nous nous intéresserons plus particulièrement sont les équations, les diséquations et les inéquations. Par exemple si $\overline{\mathcal{F}} = \{0, 1, C_0, +, \cdot\}$, $\mathcal{P} = \{=^?, \neq^?, \leq^?\}$ et $\mathcal{X} = \{v, w, x, y, z\}$ alors $(x =^? z \cdot (z + 1))$, $(x + y \neq^? v \cdot w)$, $(x \cdot y \leq^? z + 1)$ sont des contraintes atomiques de ce langage de contraintes primal.

Comme les algèbres considérées sont finies, l'ensembles des solutions $Sol_A(c)_{|\mathcal{V}(c)}$ d'une contrainte c est facilement calculable: il suffit de tester toutes les valeurs possibles et de ne garder que celles qui satisfont la contrainte. On s'intéresse plus particulièrement à une représentation symbolique de cet ensemble fini de solutions. Pour analyser ce problème, nous allons caractériser une solution symbolique σ grâce à une application surjective de l'ensemble des assignations $ASS_A^{\mathcal{V}(c\sigma)}$ de nouvelles variables vers les assignations solutions $Sol_A(c)_{|\mathcal{V}(c)}$.

Etant donné une contrainte c, une substitution σ définit une application

$$\sigma_c : ASS_A^{\mathcal{V}(c\sigma)} \mapsto ASS_A^{\mathcal{V}(c)}$$

qui associe à tout $\alpha \in ASS_A^{\mathcal{V}(c\sigma)}$ un assignation définie par

$$\forall x \in \mathcal{V}(c),\ \sigma_c(\alpha)(x) = \underline{\alpha}(x\sigma).$$

Cette relation s'étend aisément par induction structurelle aux termes formés sur $\mathcal{V}(c)$:

$$\forall t \in T(\overline{\mathcal{F}}, \mathcal{V}(c)),\ \underline{\sigma_c(\alpha)}(t) = \underline{\alpha}(t\sigma).$$

On note $\mathcal{I}(\sigma_c)$ le rang de cette application σ_c, $\mathcal{I}(\sigma_c) = \{\sigma_c(\alpha) | \alpha \in ASS_A^{\mathcal{V}(c\sigma)}\}$.

Exemple 5.7 *Dans l'algèbre de Boole, considérons l'équation* $c = (x =^? x + y)$ *et la substitution* $\sigma = \{y \mapsto x\}$. σ_c *associe à* $(x \mapsto 0)$ *l'assignation* $(x \mapsto 0)(y \mapsto 0)$ *et à* $(x \mapsto 1)$ *l'assignation* $(x \mapsto 1)(y \mapsto 1)$.

Le résultat suivant réduit la recherche d'une solution symbolique à une condition nécessaire et suffisante sur σ_c.

Proposition 5.5 *Une substitution* σ *est une solution symbolique d'une contrainte* c *si et seulement si* $\mathcal{I}(\sigma_c) \subseteq Sol_A(c)_{|\mathcal{V}(c)}$.

Preuve : Il suffit de considérer une contrainte atomique $c = p(t_1, \ldots, t_m)$, où p est un prédicat (ou $=$). Une substitution σ est une solution symbolique si et seulement si

$$\begin{array}{rl} & \forall \alpha,\ (\underline{\alpha}(\sigma(t_1)), \ldots, \underline{\alpha}(\sigma(t_m))) \in p_A \\ \iff & \forall \alpha,\ (\underline{\sigma_c(\alpha)}(t_1), \ldots, \underline{\sigma_c(\alpha)}(t_m)) \in p_A \\ \iff & \mathcal{I}(\sigma_c) \subseteq Sol_A(c)_{|\mathcal{V}(c)} \end{array}$$

□

En particulier, une contrainte c n'a pas de solution symbolique si $Sol_A(c)$ est vide.

La richesse algébrique des algèbres primales permet d'établir un lien entre le préordre de subsomption et l'inclusion des assignations schématisées.

Proposition 5.6 *Soient σ et σ' deux substitutions et c une contrainte. Alors*

$$\sigma \leq_{AF}^{\mathcal{V}(c)} \sigma' \text{ si et seulement si } \mathcal{I}(\sigma'_c) \subseteq \mathcal{I}(\sigma_c).$$

Preuve : ($\Leftarrow$) Une assignation de $\mathcal{V}(c\sigma')$ vers A est notée ici α'. Par hypothèse, nous avons

$$\forall \alpha' \exists \alpha,\ \forall x \in \mathcal{V}(c), \underline{\alpha'}(x\sigma') = \sigma'_c(\alpha')(x) = \sigma_c(\alpha)(x) = \underline{\alpha}(x\sigma).$$

Il existe donc

$$u : ASS_A^{\mathcal{V}(c\sigma')} \rightarrow ASS_A^{\mathcal{V}(c\sigma)}$$

tel que

$$\forall \alpha'\ \forall x \in \mathcal{V}(c),\ \underline{\alpha'}(x\sigma') = \underline{u(\alpha')}(x\sigma).$$

Soit μ la substitution

$$\{x \mapsto \sum_{\{\alpha' : \mathcal{V}(c\sigma') \rightarrow A\}} \prod \alpha'(\mathcal{V}(c\sigma')) \cdot [u(\alpha')(x)]\}_{x \in \mathcal{V}(c\sigma)}.$$

D'après l'interprétation des opérateurs de $\overline{\mathcal{F}}$, on a $\mu_{c\sigma} = u$. Par conséquent,

$$\forall \alpha'\ \forall x \in \mathcal{V}(c),\ \underline{\alpha'}(x\sigma\mu) = \underline{\mu_{c\sigma}(\alpha')}(x\sigma) = \underline{u(\alpha')}(x\sigma) = \underline{\alpha'}(x\sigma').$$

Ou encore, dans la théorie équationelle AF:

$$\forall x \in \mathcal{V}(c),\ x\sigma' =_{AF} x\sigma\mu.$$

($\Rightarrow$) La notation est analogue à la première partie de la preuve. On peut supposer sans perte de généralité que $\mathcal{V}(c\sigma\mu) = \mathcal{V}(c\sigma')$.

$$\begin{aligned} \forall \alpha'\ \forall x \in \mathcal{V}(c),\ \sigma'_c(\alpha')(x) &= \underline{\alpha'}(x\sigma') \\ &= \underline{\alpha'}(x\sigma\mu) \\ &= \underline{\mu_{c\sigma}(\alpha')}(x\sigma) \\ &= \sigma_c(\mu_{c\sigma}(\alpha'))(x). \end{aligned}$$

C'est-à dire $\mathcal{I}(\sigma'_c) \subseteq \mathcal{I}(\sigma_c)$. □

Corollaire 5.2 *S'il existe une substitution σ telle que $\mathcal{I}(\sigma_c) = Sol_A(c)_{|\mathcal{V}(c)}$, alors σ est une solution principale de c.*

Le problème est maintenant de prouver l'existence d'une telle substitution σ. Pour se faire, il suffit de construire une application σ_c de l'ensemble des assignations de nouvelles variables Y (introduit pour exprimer les assignations $\alpha : \mathcal{V}(c\sigma) \mapsto A$) vers les assignations

des variables de $\mathcal{V}(c)$. Le nombre de nouvelles variables Y sera choisi aussi petit que possible mais doit satisfaire la condition

$$n^{|Y|} \geq |Sol_A(c)_{|\mathcal{V}(c)}|.$$

En effet, puisque σ_c est une application, nous avons

$$|ASS_A^Y| \geq |\mathcal{I}(\sigma_c)| = |Sol_A(c)_{|\mathcal{V}(c)}|.$$

De plus $|ASS_A^Y| = |A|^{|Y|}$ avec $|A| = n$. Dans le pire cas, le cardinal de Y est égal à celui de $\mathcal{V}(c)$ et le plus facile alors est de conserver l'ensemble des solutions (modulo un renommage) et d'envoyer les non-solutions sur une solution particulière. Mais, de toute façon, n' importe quelle surjection de ASS_A^Y vers $Sol_A(c)_{|\mathcal{V}(c)}$ peut être utilisée pour σ_c.

Exemple 5.8 *Dans l'algèbre de Boole, considérons l'équation* $e = (x + yz =^? xyz)$*, où le symbole* $\cdot$ *est omis. Une assignation (par exemple* $\beta = (x \mapsto 0)(y \mapsto 0)(z \mapsto 0)$*) est notée de façon abusive par son atome (*$\overline{x}\ \overline{y}\ \overline{z}$ *pour* β*)). Le lecteur pourra vérifier que* $Sol(e) = \{\overline{x}\ \overline{y}\ \overline{z}, \overline{x}\ \overline{y}\, z, \overline{x}\, y\, \overline{z}, xyz\}$.
Il est nécessaire d'introduire deux variables y_1 *et* y_2*, puisque* $2^{|Y|} \geq 4$ *implique* $|Y| = 2$ *comme plus petite possibilité. On peut choisir* σ_e *comme suit:*

$$\sigma_e(\overline{y_1}\ \overline{y_2}) = \overline{x}\ \overline{y}\ \overline{z} \quad \sigma_e(\overline{y_1}\, y_2) = \overline{x}\ \overline{y}\, z \quad \sigma_e(y_1\, \overline{y_2}) = \overline{x}\, y\, \overline{z} \quad \sigma_e(y_1 y_2) = xyz$$

La forme canonique permet de construire la substitution σ correspondant à l'application σ_c.

Théorème 5.6 *Soit c une contrainte, Y un ensemble fini de nouvelles variables disjoint de* $\mathcal{V}(c)$ *et* σ_c *une application de* ASS_A^Y *vers* $ASS_A^{\mathcal{V}(c)}$ *tels que* $\mathcal{I}(\sigma_c) = Sol_A(c)_{|\mathcal{V}(c)}$*. La substitution*

$$\sigma = \{x \mapsto \sum_{\{\alpha: Y \to A\}} \prod \alpha(Y) \cdot [\sigma_c(\alpha)(x)]\}_{x \in \mathcal{V}(c)}$$

est une solution principale de c.

Preuve : Par construction et pour tout $x \in \mathcal{V}(c)$, $\underline{\alpha}(x\sigma) = \sigma_c(\alpha)(x)$. □

Exemple 5.9 *(Suite de l'Exemple 5.8:* $e = (x+yz =^? xyz)$*).* $\sigma_e(\alpha)(x) = 1$ *si* α *correspond à l'atome* $y_1 y_2$*,* $\sigma_e(\alpha)(y) = 1$ *si* α *est* $y_1\, \overline{y_2}$ *ou* $y_1 y_2$*,* $\sigma_e(\alpha)(z) = 1$ *si* α *est* $\overline{y_1}\, y_2$ *ou* $y_1 y_2$*. Après simplification, on obtient*

$$mgs(c) = \{x \mapsto y_1 y_2, y \mapsto y_1, z \mapsto y_2\}.$$

L'unification booléenne permet d'apporter une aide à la vérification de circuits. L'exemple suivant est tiré de [DSvH87].

Exemple 5.10 *Le circuit CROSS de la figure 5.2 est composé exclusivement de portes élémentaires ET, OU, NON. Ce circuit peut donc être directement spécifié par un ensemble d'équations dans l'algèbre de Boole:*

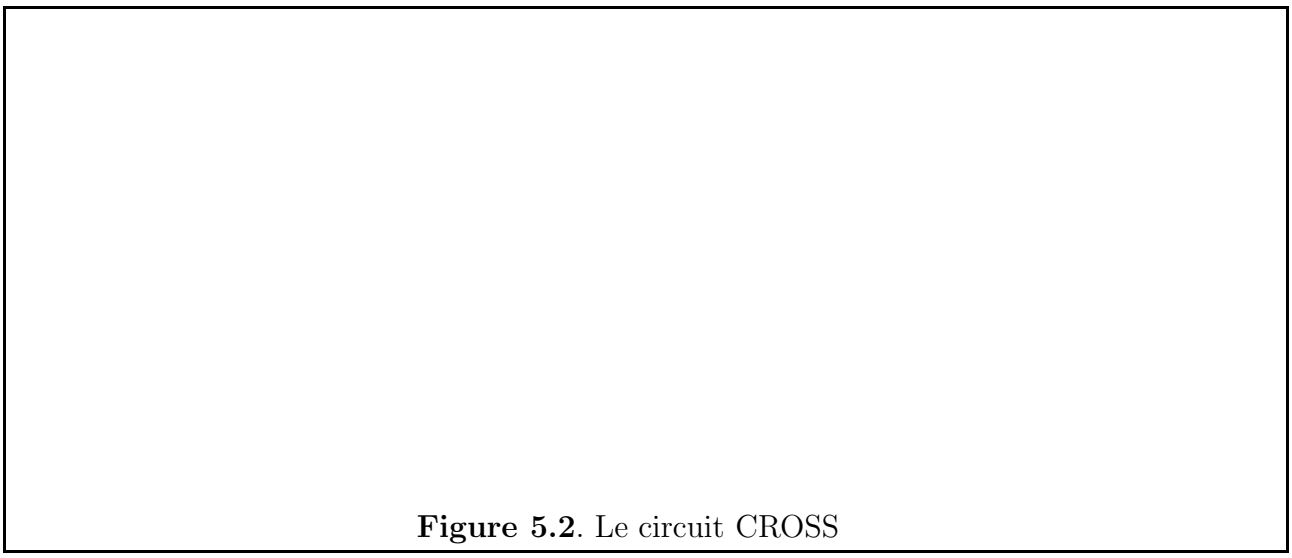

Figure 5.2. Le circuit CROSS

$$\begin{cases} G & =^? & \overline{X} \\ E & =^? & \overline{Y} \\ F & =^? & X \cdot Y \\ I & =^? & G + F \\ H & =^? & E + F \\ K & =^? & \overline{I} \\ J & =^? & \overline{H} \\ M & =^? & K + F \\ L & =^? & J + F \\ O & =^? & L + M \\ P & =^? & \overline{K} \\ N & =^? & \overline{J} \\ A & =^? & P \cdot O \\ B & =^? & N \cdot O \end{cases}$$

L'algorithme de résolution appliqué à ce système d'équations retourne pour solution principale

$$\{X \mapsto Y_1, Y \mapsto Y_2, A \mapsto Y_2, B \mapsto Y_1\}.$$

On peut en conclure que le but de ce circuit est d'échanger les entrées. Nous verrons plus tard comment modifier l'algorithme afin d'exprimer les sorties A, B en fonction des entrées en considérant X, Y comme des constantes.

L'algorithme de résolution ne s'applique pas uniquement à des équations mais aussi à des diséquations.

Exemple 5.11 *Dans l'algèbre de Boole, considérons la diséquation $d = (x \cdot (y + z) \neq^? x \cdot \overline{y})$. Le lecteur pourra vérifier que l'ensemble des solutions est $Sol(d) = \{x\,\overline{y}\,\overline{z}, xy\,\overline{z}, xyz\}$. Comme $|Sol(d)| = 3$, la condition $2^{|Y|} \geq 3$ implique $|Y| = 2$ pour plus petite possibilité.*

Nous allons donc introduire deux nouvelles variables y_1 et y_2. L'application σ_d peut être construite de la façon suivante:

$$\sigma_d(\overline{y_1}\,\overline{y_2}) = x\,\overline{y}\,\overline{z} \quad \sigma_d(\overline{y_1}\,y_2) = xy\,\overline{z} \quad \sigma_d(y_1\,\overline{y_2}) = xyz \quad \sigma_d(y_1y_2) = xyz$$

La solution principale correspondante est

$$\{x \mapsto \mathbf{1}, y \mapsto \overline{y_1}\,y_2 + y_1, z \mapsto y_1\}.$$

L'algorithme s'applique encore à des inéquations.

Exemple 5.12 *Toujours dans l'algèbre de Boole dans laquelle on définit $0 < 1$, considérons l'inéquation $i = (x + y \leq^? \overline{x})$. Le lecteur pourra vérifier que l'ensemble des solutions est $Sol(i) = \{x\,\overline{y}, xy\}$.*
Comme $|Sol(i)| = 2$, la condition $2^{|Y|} \geq 2$ implique $|Y| = 1$ pour plus petite possibilité. Nous allons donc introduire une unique nouvelle variable y_1. L'application σ_i peut être fixée de la façon suivante:

$$\sigma_i(\overline{y_1}) = x\,\overline{y} \quad \sigma_i(\overline{y_1}) = xy$$

La solution principale correspondante est

$$\{x \mapsto \mathbf{1}, y \mapsto \overline{y_1}\}.$$

L'inégalité

$$1 + \overline{y_1} \leq y_1$$

est effectivement valide dans l'algèbre de Boole.

Le résolveur symbolique s'applique à toute forme de contrainte c. La mise sous forme équationnelle d'une contrainte quelconque [BES+90, Nip90] devient alors superflue. La seule hypothèse est de savoir calculer l'ensemble fini des solutions $Sol_A(c)_{|\mathcal{V}(c)}$. Un domaine infini peut aussi rentrer dans ce cadre pourvu qu'il n'y ait toujours qu'un ensemble fini de valeurs prises par les variables. C'est ce que nous allons voir avec la structure des pseudo-booléens qui mélange les opérateurs sur les entiers et l'algèbre de Boole, les variables prenant uniquement des valeurs sur $\{0, 1\}$.

5.6 Les pseudo-booléens

Une algèbre primale est par définition mono-sortée. Un terme s'interprète par une fonction dont le domaine et le codomaine sont identiques et n'importe quelle fonction est l'interprétation d'un terme. Cette notion peut être généralisée à des algèbres multi-sortées et plus particulièrement ordo-sortées. Dans ce cadre, un terme bien formé s'interprète par une fonction dont le domaine fini est différent du codomaine infini. Nous allons nous intéresser à titre d'exemple aux fonctions pseudo-booléennes $f : \{0,1\}^m \to \mathcal{Z}$ où $\mathcal{Z}$ désigne l'ensemble des entiers relatifs. La structure algébrique ordo-sortée introduite par A. Bockmayr [Boc91] et qui est reprise ici possède deux sortes, l'une *Bool* interprétée par $\{0, 1\}$ et l'autre *Int* par $\mathcal{Z}$. Chacune de ces deux sortes est munie des opérateurs usuels. L'inclusion de sorte $Bool \leq Int$ permet de construire dans une certaine mesure des termes hétérogènes qui s'avèrent utile pour représenter n'importe quelle fonction pseudo-booléenne.

Définition 5.9 *Considérons la signature ordo-sortée* $\Sigma_{PB} = (\{Bool, Int\}, \mathcal{F}_{BA} \cup \mathcal{F}_{Int}, \mathcal{P}_{PB})$ *avec* $Bool \leq Int$ *telle que les symboles de fonctions* $\mathcal{F}_{BA} = \{\mathbf{0}, \mathbf{1}, \vee, \wedge, \bar{}\}$ *soient munis des profils*

$$\begin{aligned} \mathbf{0}, \mathbf{1} &:\rightarrow Bool \\ \bar{} &: Bool \rightarrow Bool \\ \vee, \wedge &: Bool \times Bool \rightarrow Bool \end{aligned}$$

et les symboles de fonction $\mathcal{F}_{Int} = \{-, +, *\}$ *soient munis des profils*

$$\begin{aligned} - &: Int \rightarrow Int \\ +, * &: Int \times Int \rightarrow Int \end{aligned}$$

Soit $\mathcal{PB}$ *une* Σ_{PB}*-structure telle que*

- $(Bool)_{\mathcal{PB}} = \{0, 1\}$,
- $(Int)_{\mathcal{PB}} = \mathbf{Z}$,
- $(\mathcal{F}_{BA})_{\mathcal{PB}} = (\mathcal{F}_{BA})_{\mathcal{B}}$,
- $(\mathcal{F}_{Int})_{\mathcal{PB}} = (\mathcal{F}_{Int})_{\mathcal{Z}}$,

où $\mathcal{B}$ *est la* $\mathcal{F}_{BA}$*-algèbre de Boole et* $\mathcal{Z}$ *est la* $\mathcal{F}_{Int}$*-algèbre interprétant respectivement* $-, +, *$ *par l'opposé, l'addition et la multiplication sur* $\mathcal{Z}$*, les entiers relatifs.*

La structure $\mathcal{PB}$ et un ensemble de variables $\mathcal{X} = \mathcal{X}_{Bool} \cup \mathcal{X}_{Int}$ définissent un langage de contraintes dans lequel nous allons chercher à résoudre symboliquement certaines contraintes.

Définition 5.10 *Un terme* pseudo-booléen *est un terme de* $T(\Sigma_{PB}, \mathcal{X}_{Bool})$*. Une contrainte* pseudo-booléenne *est une contrainte dont les termes sont pseudo-booléens.*

Nous allons nous restreindre à la résolutions de contraintes pseudo-booléennes c qui ont la particularité d'avoir un ensemble fini d'assignations solutions $Sol_{\mathcal{PB}}(c)_{|\mathcal{V}(c)}$.

Les assignations des variables booléennes par les valeurs $\{0, 1\}$, et les substitutions de ces variables booléennes par des termes nécessairement booléens font que la démarche suivie pour la résolution de contraintes dans l'algèbre de Boole est encore valable dans cette algèbre ordo-sortée. Cela mène au résultat suivant, identique à un renommage près à celui vu au théorème 5.6.

Notation: Le monôme construit sur l'assignation $\alpha : V \rightarrow \{0, 1\}$ est ici désigné par $\bigwedge \alpha(V)$ et vaut

$$\begin{array}{lll} \bigwedge \alpha(V) = \top & \text{si} & V = \emptyset \\ \bigwedge \alpha(V) = \bar{x} \wedge (\bigwedge \alpha(V \backslash \{x\})) & \text{si} & V \neq \emptyset \text{ et } \alpha(x) = 0 \\ \bigwedge \alpha(V) = x \wedge (\bigwedge \alpha(V \backslash \{x\})) & \text{si} & V \neq \emptyset \text{ et } \alpha(x) = 1 \end{array}$$

Théorème 5.7 *Soit* c *une contrainte pseudo-booléenne,* Y *un ensemble fini de nouvelles variables booléennes disjoint de* $\mathcal{V}(c)$ *et* σ_c *une application de* $ASS^{Y}_{\{0,1\}}$ *vers* $ASS^{\mathcal{V}(c)}_{\{0,1\}}$ *tels que* $\mathcal{I}(\sigma_c) = Sol_{\mathcal{PB}}(c)_{|\mathcal{V}(c)}$*. La substitution*

$$\sigma = \{x \mapsto \bigvee_{\{\alpha : Y \rightarrow \{0,1\}\}} \bigwedge \alpha(Y) \wedge [\sigma_c(\alpha)(x)]\}_{x \in \mathcal{V}(c)}$$

est une solution principale de c *dans* $\mathcal{PB}$*.*

Le problème de la construction d'une solution principale à partir de l'ensemble des solutions $Sol_{\mathcal{PB}}(c)_{|\mathcal{V}(c)}$ est identique à celui vu dans l'algèbre de Boole car les variables sont de sorte *Bool*: ces variables s'évaluent dans $\{0,1\}$ et sont substituées par des termes de sorte *Bool* interprétés par des fonctions booléennes.

La structure $\mathcal{PB}$ offre donc la possibilité de combiner la résolution symbolique de contraintes booléennes et l'arithmétique qui elle doit être intégrée au calcul des solutions $Sol_{\mathcal{PB}}(c)$.

Cette structure permet aussi de représenter toute fonction pseudo-booléenne par un terme pseudo-booléen puisque toute fonction $f : \{0,1\}^m \to \mathcal{Z}$ et $V = \{x_1, \ldots, x_m\}$ est l'interprétation fonctionnelle d'un terme à m variables $V = \{x_1, \ldots, x_m\}$,

$$\sum\!\!\!\!\!\!\sum_{\alpha : V \to \{0,1\}} f(\alpha(x_1), \ldots, \alpha(x_m)) \bigwedge \alpha(V),$$

où $\sum\!\!\!\!\!\!\sum$ désigne l'itération de $+\!\!\!+$, et avec par convention

$$\sum\!\!\!\!\!\!\sum_{i=1}^{m} t = m\ t.$$

On notera que cette propriété n'a pas été utilisée jusqu'à présent pour construire une solution principale.

Autrement dit, tout terme possède une forme canonique ayant même interprétation fonctionnelle.

Lemme 5.5 *Un terme $t \in T(\Sigma_{PB}, \mathcal{X}_{Bool})$ est égal dans $\mathcal{PB}$ à sa* forme canonique*:*

$$t =_{\mathcal{PB}} \sum\!\!\!\!\!\!\sum_{\alpha : \mathcal{V}(t) \to \{0,1\}} \underline{\alpha}(t) \bigwedge \alpha(\mathcal{V}(t)).$$

Ce terme bien formé est constitué d'opérateurs sur les booléens et d'autres sur les entiers. Les opérateurs booléens $\wedge$ et $^-$ peuvent s'exprimer en fonction d'opérateurs sur les entiers. En effet,

$$X : Bool \wedge Y : Bool =_{\mathcal{PB}} X : Bool * Y : Bool,$$

$$\overline{X : Bool} =_{\mathcal{PB}} \mathbf{1} - X : Bool.$$

Toute terme pseudo-booléen t s'écrit en définitive sous la forme d'une *expression polynomiale* $t\downarrow$,

$$\sum\!\!\!\!\!\!\sum_{W \subseteq V} c_W \prod_{w \in W} w,$$

où les c_W désignent des coefficient sur $\mathcal{Z}$ et $\prod$ l'itération de $*$.

Exemple 5.13 *La fonction $f : \{0,1\}^2 \to \mathcal{Z}$ définie par la table de vérité*

X	Y	$f(X,Y)$
0	0	2
0	1	3
1	0	4
1	1	1

a pour forme canonique

$$2\overline{X} \wedge \overline{Y} + 3\overline{X} \wedge Y + 4X \wedge \overline{Y} + X \wedge Y = 2(1-X)(1-Y) + 3(1-X)Y + 4X(1-Y) + XY$$

et pour expression polynomiale $2 + 2X + Y - 4XY$.

Nous allons exhiber une présentation équationnelle ordo-sortée PB permettant de transformer tout terme pseudo-booléen en sa forme polynomiale. Elle n'est constituée que d'égalités valides dans $\mathcal{PB}$.

Définition 5.11 *Soit PB l'ensemble fini d'axiomes ordo-sortés sur $\mathcal{T}(\Sigma_{PB}, \mathcal{X})$ obtenus par la réunion des axiomes BA dont les variables sont supposées de sorte $Bool$ et des axiomes donnés en figure 5.3. Tous les axiomes de PB sont universellement quantifiés sur $\mathcal{X}$.*

$$\begin{array}{rcl}
x : Int + (y : Int + z : Int) & = & (x : Int + y : Int) + z : Int \\
x : Int + \mathbf{0} & = & x : Int \\
x : Int + (-x : Int) & = & \mathbf{0} \\
x : Int + y : Int & = & y : Int + x : Int \\
x : Int * (y : Int * z : Int) & = & (x : Int * y : Int) * z : Int \\
x : Int * \mathbf{1} & = & x : Int \\
x : Int * y : Int & = & y : Int * x : Int \\
x : Int * (y : Int + z : Int) & = & (x : Int * y : Int) + (x : Int * z : Int) \\
X : Bool * X : Bool & = & X : Bool \\
X : Bool \wedge Y : Bool & = & X : Bool * Y : Bool \\
\overline{X : Bool} & = & \mathbf{1} - X : Bool \\
X : Bool \vee Y : Bool & = & X : Bool + Y : Bool - X : Bool * Y : Bool
\end{array}$$

Figure 5.3. La présentation équationnelle ordo-sortée $PB \backslash BA$

La démarche suivie par A. Bockmayr [Boc91] est de réécrire systématiquement les termes avec symboles booléens grâce au système de réécriture

$$R_{Bool \to Int} = \left\{ \begin{array}{rcl}
X : Bool \wedge Y : Bool & \to & X : Bool * Y : Bool \\
\overline{X : Bool} & \to & \mathbf{1} - X : Bool \\
X : Bool \vee Y : Bool & \to & X : Bool + Y : Bool - X : Bool * Y : Bool
\end{array} \right.$$

dont les égalités figurent dans PB. La forme normale suivant $R_{Bool \to Int}$ d'un terme booléen est de sorte Int. Cela pose problème car la forme normale d'une solution symbolique de sorte $Bool$ n'est plus une substitution bien formée au sens habituel. La définition d'un unificateur est modifiée dans [Boc91] pour qu'une substitution σ de sorte $Bool$ soit un unificateur de $s =^? t$ si $(s\sigma) \downarrow_{R_{Bool \to Int}} = (t\sigma) \downarrow_{R_{Bool \to Int}}$ est valide dans $\mathcal{PB}$.

Nous avons préféré inclure les règles du sytème de réécriture $R_{Bool \to Int}$ comme axiomes de la présentation PB. Ce choix permet d'éviter l'adaptation du concept d'unificateur à ce contexte particulier.

Le système de réécriture $R_{Bool \to Int}$ permet de montrer des propositions analogues à celles mises en évidence dans les algèbres finies et énoncées dans les deux propositions suivantes.

Proposition 5.7 *Les Σ_{PB}-algèbres $\mathcal{PB}$ et $\mathcal{T}(\Sigma_{PB}, \mathcal{X})/ =_{PB}$ ont le même ensemble de théorèmes équationnels pseudo-booléens: pour toute égalité $(s = t)$ de termes pseudo-booléens, $\mathcal{PB} \models s = t$ si et seulement si $s =_{PB} t$.*

Preuve : ($\Rightarrow$: une égalité valide dans $\mathcal{PB}$ est une PB-égalité). Le remplacement d'égal par égal modulo PB permet de transformer les termes s et t en leurs expressions polynomiales $s\downarrow$ et $t\downarrow$ respectives: $s =_{PB} s\downarrow$ et $t =_{PB} t\downarrow$. Par hypothèse, nous avons $s =_{\mathcal{PB}} t$ et donc $s\downarrow = t\downarrow$. Ce qui permet de conclure que $s =_{PB} t$.

($\Leftarrow$) Il suffit de vérifier que tous les axiomes de la présentation PB sont valides dans $\mathcal{PB}$. □

Proposition 5.8 *Les Σ_{PB}-algèbres $\mathcal{PB}$ et $\mathcal{T}(\Sigma_{PB})/ =_{PB}$ sont isomorphes.*

Preuve : Soit h l'unique homomorphisme de $\mathcal{T}(\Sigma_{PB})$ vers $\mathcal{PB}$. D'après la proposition 5.7, nous savons que

$$\forall t \in T(\Sigma_{PB}),\ t =_{PB} t\downarrow = \overbrace{\mathbf{1}+\ldots+\mathbf{1}}^{h(t) \text{ fois}} \text{ si } h(t) \neq 0, t =_{PB} t\downarrow = \mathbf{0} \text{ sinon.}$$

En quotientant $\mathcal{T}(\Sigma_{PB})$ par $=_{PB}$ on obtient une bijection car

- si $h(s) = h(t)$ alors $s =_{PB} t$,
- si $h(s) \neq h(t)$ alors $\mathcal{PB} \not\models s\downarrow = t\downarrow$ et $s\downarrow \neq_{PB} t\downarrow$, grâce à la proposition 5.7. Donc $s \neq_{PB} t$.

□

Corollaire 5.3 *La présentation équationnelle ordo-sortée (Σ_{PB}, PB) est ω-complète sur les pseudo-booléens, i.e. les algèbres $\mathcal{T}(\Sigma_{PB}, \mathcal{X})/ =_{PB}$ et $\mathcal{T}(\Sigma_{PB})/ =_{PB}$ (isomorphe à $\mathcal{PB}$) ont les mêmes théorèmes équationnels pseudo-booléens.*

L'unification dans la théorie équationnelle ordo-sortée PB de contraintes équationnelles pseudo-booléennes est donc unitaire.

5.7 Les pseudo-finis

La notion de pseudo-booléens peut être directement étendue à une algèbre primale $\overline{\mathcal{A}}$. La structure algébrique ainsi définie est appelée pseudo-finis.

Définition 5.12 *Considérons la signature ordo-sortée $\Sigma_{PF} = (\{Prim, Int\}, \overline{\mathcal{F}} \cup \mathcal{F}_{Int}, \mathcal{P}_{PF})$ avec $Prim \leq Int$ telle que les symboles de fonction $\overline{\mathcal{F}}$ soient munis des profils*

$$\begin{aligned} &[0], \ldots, [n-1] :\rightarrow Prim \\ &C_0, \ldots, C_{n-1} : Prim \rightarrow Prim \\ &+, \cdot : Prim \times Prim \rightarrow Prim \end{aligned}$$

Soit $\mathcal{PF}$ une Σ_{PF}-structure telle que

- $(Prim)_{\mathcal{PF}} = A$,
- $(Int)_{\mathcal{PF}} = \mathbf{Z}$,
- $(\overline{\mathcal{F}})_{\mathcal{PF}} = (\overline{\mathcal{F}})_{\overline{\mathcal{A}}}$,
- $(\mathcal{F}_{Int})_{\mathcal{PF}} = (\mathcal{F}_{Int})_{\mathcal{Z}}$,

où $\mathcal{Z}$ est la $\mathcal{F}_{Int}$-algèbre interprétant respectivement $-, +, *$ par l'opposé, l'addition et la multiplication sur $\mathcal{Z}$, les entiers relatifs.

La structure $\mathcal{PF}$ et un ensemble de variables $\mathcal{X} = \mathcal{X}_{Prim} \cup \mathcal{X}_{Int}$ définissent un langage de contraintes. Comme précédemment pour les pseudo-booléens, nous allons nous limiter à certaines contraintes.

Définition 5.13 *Un terme* pseudo-fini *est un terme de $T(\Sigma_{PF}, \mathcal{X}_{Prim})$. Une contrainte* pseudo-finie *est une contrainte dont les termes sont exclusivement pseudo-finis.*

On s'intéressera exclusivement aux contraintes pseudo-finies c car elles ont un ensemble fini de solutions $Sol_{\mathcal{PF}}(c)_{|\mathcal{V}(c)}$.

Le résultat suivant s'inspire directement du théorème 5.6. La preuve en est évidente puisque les assignations à considérer sont celles de $ASS_A^{\mathcal{V}(c)}$.

Théorème 5.8 *Soit c une contrainte pseudo-finie, Y un ensemble fini de nouvelles variables de sorte Prim disjoint de $\mathcal{V}(c)$ et σ_c une application de ASS_A^Y vers $ASS_A^{\mathcal{V}(c)}$ tels que $\mathcal{I}(\sigma_c) = Sol_{\mathcal{PF}}(c)_{|\mathcal{V}(c)}$. La substitution*

$$\sigma = \{x \mapsto \sum_{\{\alpha: Y \to A\}} \prod \alpha(Y) \cdot [\sigma_c(\alpha)(x)]\}_{x \in \mathcal{V}(c)}$$

est une solution principale de c dans $\mathcal{PF}$.

L'intérêt principal du résolveur est de minimiser le nombre de variables introduites.

Exemple 5.14 *Voici un petit jeu de réflexion. Le but est de compléter la phrase ci-dessous, en remplaçant les "..." par un seul chiffre (de 0 à 9), afin d'obtenir une phrase cohérente.*

"Dans cette phrase, il y a
... fois le chiffre 0,
... fois le chiffre 1,
... fois le chiffre 2,
... fois le chiffre 3,
... fois le chiffre 4,
... fois le chiffre 5,
... fois le chiffre 6,
... fois le chiffre 7,
... fois le chiffre 8,
... fois le chiffre 9."

Ce problème peut être exprimé par le système d'équations

$$\begin{aligned} X_0 &=^? 1+C_0(X_0)\cdot 1+C_0(X_1)\cdot 1+\cdots+C_0(X_9)\cdot 1 \\ &\cdots \\ X_9 &=^? 1+C_9(X_0)\cdot 1+C_9(X_1)\cdot 1+\cdots+C_9(X_9)\cdot 1 \end{aligned}$$

dans la structure $\mathcal{PF}$ dont la sorte Prim s'interprète par l'ensemble des chiffres allant de 0 *à* 9. *Les variables $X_0, X_1, \ldots, X_9$ sont de sorte Prim et correspondent aux chiffres qui doivent se substituer aux "..." de chaque ligne. L'équation*

$$X_1 =^? 1+C_1(X_0)\cdot 1+C_1(X_1)\cdot 1+\cdots+C_1(X_9)\cdot 1$$

possède 40951 *solutions et sa résolution permet de passer d'un problème de* 10 *variables à un plus simple ne contenant plus que* 5 *variables.*

Cette structure algébrique permet de représenter par des termes l'ensemble des fonctions $f : A^m \to \mathcal{Z}$. Soit $V = \{x_1, \ldots, x_m\}$ un ensemble de m variables. L'interprétation fonctionnelle du terme

$$\sum\!\!\!\!\!\!\sum_{\alpha:\mathcal{V}(t)\to\{0,1\}} f(\alpha(x_1), \ldots \alpha(x_m)) \ (\prod \alpha(\mathcal{V}(t)) \cdot [1]).$$

est la fonction f.

A nouveau, nous allons montrer une présentation équationnelle permettant d'obtenir par remplacement d'égal par égal une forme canonique pour tout terme t. Cette forme normale correspond à la décomposition de son interprétation fonctionnelle. Cette présentation contient les axiomes AF, les axiomes des anneaux et certaines égalités de liaison entre ces deux théories.

Définition 5.14 *Soit PF l'ensemble fini d'axiomes ordo-sortés sur $\mathcal{T}(\Sigma_{PF}, \mathcal{X})$ obtenus par la réunion des axiomes AF dont les variables sont supposées de sorte Prim et des axiomes donnés en figure 5.4. Tous les axiomes de PF sont universellement quantifiés sur $\mathcal{X}$.*

Lemme 5.6 *Un terme t de $T(\Sigma_{PF}, \mathcal{X}_{Prim})$ est égal modulo PF à sa* forme canonique*:*

$$t =_{PF} \sum\!\!\!\!\!\!\sum_{\alpha:\mathcal{V}(t)\to A} \underline{\alpha}(t)(\prod \alpha(\mathcal{V}(t)) \cdot [1]).$$

Ce lemme conduit à un résultat comparable à ceux énoncés pour les algèbres primales puis pour les pseudo-booléens.

Corollaire 5.4 *La présentation équationnelle ordo-sortée (Σ_{PF}, PF) est ω-complète sur les pseudo-finis, i.e. les algèbres $\mathcal{T}(\Sigma_{PF}, \mathcal{X})/ =_{PF}$ et $\mathcal{T}(\Sigma_{PF})/ =_{PF}$ (isomorphe à $\mathcal{PF}$) ont les mêmes théorèmes équationnels pseudo-finis.*

$$
\begin{array}{rcl}
x : Int + (y : Int + z : Int) & = & (x : Int + y : Int) + z : Int \\
x : Int + [0] & = & x : Int \\
x : Int + (-x : Int) & = & [0] \\
x : Int + y : Int & = & y : Int + x : Int \\
x : Int * (y : Int * z : Int) & = & (x : Int * y : Int) * z : Int \\
x : Int * [1] & = & x : Int \\
x : Int * y : Int & = & y : Int * x : Int \\
x : Int * (y : Int + z : Int) & = & (x : Int * y : Int) + (x : Int * z : Int) \\
\forall f \in \overline{\mathcal{F}}_p,\ f(X_1 : Prim, \ldots, X_p : Prim) & = & \sum_{(i_1,\ldots,i_p)\in A^p} f(i_1, \ldots, i_p) \\
 & & (C_{i_1}(X_1 : Prim) \cdots C_{i_p}(X_p : Prim) \cdot [1]) \\
(X : Prim \cdot [1]) * (Y : Prim \cdot [1]) & = & X : Prim \cdot Y : Prim \cdot [1]
\end{array}
$$

Figure 5.4. La présentation équationnelle ordo-sortée $PF \backslash AF$

5.8 Le mélange des pseudo-booléens et pseudo-finis

Deux instances du même résolveur de contraintes dans une algèbre finie doivent être utilisées lorsque par exemple les sortes *Bool* et *Prim* figurent dans une même structure. Il n'y a pas d'échange entre les résolveurs puisque chacun s'occupe de ses variables.

Définition 5.15 *Considérons la signature $\Sigma_{PBF} = \Sigma_{PB} \cup \Sigma_{PF}$ et la Σ_{PBF}-structure $\mathcal{PBF}$ définie par l'union de la Σ_{PB}-structure $\mathcal{PB}$ et de la Σ_{PF}-structure $\mathcal{PF}$.*

On notera que l'union des structures est ici définie sans ambiguité puisque les symboles de $\Sigma_{PB} \cap \Sigma_{PF}$ sont par définition interprétés de façon identique dans $\mathcal{PB}$ et $\mathcal{PF}$. La Σ_{PBF}-structure $\mathcal{PBF}$ et un ensemble de variables $\mathcal{X} = \mathcal{X}_{Bool} \cup \mathcal{X}_{Prim} \cup \mathcal{X}_{Int}$ définissent un langage de contraintes.

Exemple 5.15 *Il s'agit du problème connu sous le nom $SEND + MORE = MONEY$. Le problème original consiste à rendre l'addition juste en attribuant aux lettres des valeurs entre 0 et 9, avec la contrainte supplémentaire que deux lettres différentes aient des valeurs différentes. En écrivant cette addition avec les retenues, on obtient*

$$
\begin{array}{cccccc}
 & R_4 & R_3 & R_2 & R_1 & \\
 & & S & E & N & D \\
+ & & M & O & R & E \\
\hline
 & M & O & N & E & Y
\end{array}
$$

où les variables de retenues sont de sorte Bool et les autres variables de sorte Prim, avec Prim s'interprétant par l'ensemble des chiffres allant de 0 *à* 9*.*

On va s'intéresser à résoudre de façon symbolique les contraintes pseudo-combinées, toujours dans l'intention de faire décroitre le nombre de variables exprimant les contraintes.

Définition 5.16 *Un terme de* $T(\Sigma_{PBF}, \mathcal{X}_{Bool} \cup \mathcal{X}_{Prim})$ *est dit* pseudo-combiné. *Une contrainte formée de termes pseudo-combinés est* pseudo-combinée.

Une substitution σ bien formée, de domaine inclus dans $\mathcal{X}_{Bool} \cup \mathcal{X}_{Prim}$, définit une application

$$\sigma_c : ASS_{\{0,1\}}^{\mathcal{V}(c\sigma)_{Bool}} \times ASS_A^{\mathcal{V}(c\sigma)_{Prim}} \to ASS_{\{0,1\}}^{\mathcal{V}(c)_{Bool}} \times ASS_A^{\mathcal{V}(c)_{Prim}}.$$

La substitution σ est solution symbolique d'une contrainte c pseudo-combinée si et seulement si

$$\mathcal{I}(\sigma_c) \subseteq Sol_{\mathcal{PBF}}(c)_{|\mathcal{V}(c)}.$$

Les résultats vus pour les algèbres primales s'appliquent à la fois pour la sorte *Bool* et la sorte *Prim* et conduisent au théorème suivant.

Théorème 5.9 *Soient c une contrainte pseudo-combinée, Y un ensemble fini de nouvelles variables de sorte Bool disjoint de* $\mathcal{V}(c)$, *Y′ un ensemble fini de nouvelles variables de sorte Prim disjoint de* $\mathcal{V}(c)$ *et* σ_c *une application de*

$$ASS_{\{0,1\}}^{Y} \times ASS_A^{Y'} \to ASS_{\{0,1\}}^{\mathcal{V}(c)_{Bool}} \times ASS_A^{\mathcal{V}(c)_{Prim}}$$

tels que

$$\mathcal{I}(\sigma_c) = Sol_{\mathcal{PBF}}(c)_{|\mathcal{V}(c)}.$$

La substitution

$$\sigma = \{x \mapsto \bigvee_{\{\alpha : Y \cup Y' \to \{0,1\} \cup A\}} \bigwedge \alpha_{Bool}(Y) \wedge [\sigma_c(\alpha)(x)]\}_{x \in \mathcal{V}(c)_{Bool}}$$

$$\cup \{x \mapsto \sum_{\{\alpha : Y \cup Y' \to \{0,1\} \cup A\}} \prod \alpha_{Prim}(Y') \cdot [\sigma_c(\alpha)(x)]\}_{x \in \mathcal{V}(c)_{Prim}}$$

est une solution principale de c dans $\mathcal{PBF}$.

Cet algorithme de résolution de contraintes dans $\mathcal{PBF}$ fournit un algorithme d'unification dans la théorie équationnelle obtenue par mélange de PB et PF.

Lemme 5.7 *Un terme t de* $T(\Sigma_{PBF}, \mathcal{X}_{Bool} \cup \mathcal{X}_{Prim})$ *est égal modulo* $PB \cup PF$ *à sa* forme canonique*:*

$$t =_{PB \cup PF} \sum\!\!\!\!\!\!\sum_{\alpha : \mathcal{V}(t) \to Bool_{\mathcal{PBF}} \cup Prim_{\mathcal{PBF}}} \underline{\alpha}(t) \ (\bigwedge \alpha_{Bool}(\mathcal{V}(t)_{Bool})) * (\prod \alpha_{Prim}(\mathcal{V}(t)_{Prim}) \cdot [1]).$$

Corollaire 5.5 *La présentation équationnelle ordo-sortée* $(\Sigma_{PBF}, PB \cup PF)$ *est* ω*-complète sur les termes pseudo-combinés, i.e. les algèbres* $\mathcal{T}(\Sigma_{PBF}, \mathcal{X}) / =_{PB \cup PF}$ *et* $\mathcal{T}(\Sigma_{PBF}) / =_{PB \cup PF}$ *(isomorphe à* $\mathcal{PBF}$*) ont les mêmes théorèmes équationnels pseudo-combinés.*

Le mélange des théories équationnelles PB et PF est remarquable. Ces deux théories ordo-sortées sont non disjointes et il est pourtant possible d'obtenir un algorithme d'unification dans l'union $PB \cup PF$ à partir de ceux existants dans PB et PF. L'unification dans l'union $PB \cup PF$ est unitaire comme elle l'est dans PB et PF.

La sorte commune Int a les mêmes propriétés dans chacune des deux structures ce qui explique en partie le phénomène. En outre, la construction de termes mixtes est sérieusement limitée. Il est interdit par exemple de construire dans ce cadre ordo-sorté un terme uniquement formé de symboles $\mathcal{F}_{BA}$ et $\overline{\mathcal{F}}$ alors que pour certaines interprétations, $Bool_{\mathcal{PBF}} \subseteq Prim_{\mathcal{PBF}}$.

Nous verrons dans le prochain chapitre comment combiner deux algèbres finies avec une intersection des domaines non vide sans passer par un intermédiaire, les entiers en l'occurrence, qui favorise leur "pseudo" combinaison.

6
Combinaison de résolveurs de contraintes

Ce chapitre est consacré à la définition d'un langage de contraintes combiné pour lequel un résolveur est obtenu par combinaison de résolveurs de contraintes symboliques dans des langages formés sur des signatures disjointes ou plus généralement ayant des constantes partagées. Un cas plus général de partage de symboles imposerait de trop fortes restrictions sur les langages composants. L'objectif est cette fois d'étendre les principes de combinaison à des contraintes non équationnelles et à des structures qui ne soient pas nécessairement basées sur des théories équationnelles. La construction d'un tel langage de contraintes passe par le choix d'une structure qui est une extension conservative des structures composantes. Il va s'agir d'une algèbre de termes quotientée par une relation de congruence obtenue en mélangeant les égalités valides dans chaque structure composante. L'interprétation des prédicats est définie grâce à l'abstraction des sous-termes étrangers. Mais cette abstraction ne sera effectuée que sur des termes préalablement normalisés par une relation de réécriture convergente de même expressivité que l'égalité dans la structure combinée.

Les principales règles de combinaison vont donc s'inspirer de celles déjà vues dans le cas particulier de l'unification. Il faudra par exemple supposer l'existence d'un résolveur de contraintes suffisamment spécialisé pour permettre de résoudre des contraintes suivant une certaine restriction (linéaire).

Afin d'illustrer la notion de langage combiné sur le résolveur de contraintes dans les algèbres primales vu au chapitre 5, nous allons d'abord devoir étendre ce résolveur pour qu'il puisse tenir compte d'une restriction. L'algorithme correspondant est obtenu en deux phases bien distinctes, une première phase de résolution proprement dite suivi d'une phase de prise en compte de la restriction par l'utilisation d'un algorithme d'élimination de constante que nous détaillerons.

Nous verrons ensuite comment combiner ce résolveur de contraintes avec un algorithme d'unification en montrant comment se ramener à un problème de combinaison d'algorithmes d'unification grâce à l'existence d'une présentation équationnelle AF de l'égalité dans une algèbre primale $\overline{\mathcal{A}}$.

On détaillera finalement des applications au langage combiné:

Le cas le plus simple de mélange avec une théorie équationnelle est celui qui consiste à étendre l'algèbre primale par des symboles libres que l'on voudrait interpréter par des fonctions sur le domaine de cette algèbre. Nous préciserons les liens existants entre le

langage combiné et la classe de tous les enrichissements possibles de l'algèbre primale sur son domaine.

Dans le même ordre d'idée, la combinaison de résolveurs de contraintes dans deux algèbres finies permet d'obtenir un résolveur de contraintes symboliques dans une structure proche de celle liée au langage combiné mais qui est, elle, définie plus simplement par l'union des domaines et interprétations. Cette approche n'est intéressante que lorsque les domaines respectifs ont une intersection non vide car les contraintes hétérogènes ont alors une solution. Les constantes représentant les éléments de l'intersection des domaines sont dans ce cas partagées.

6.1 Un langage de contraintes combiné

Un modèle pour la combinaison de deux langages de contraintes doit au mieux préserver la validité des contraintes pures de chaque langage pour qu'on puisse ensuite résoudre des contraintes pures dans les langages respectifs. L'existence d'un tel modèle dépend fortement des symboles de sortes, fonctions ou prédicats partagés par les deux signatures correspondantes. Intuitivement, il serait naturel de vouloir prendre pour modèle la structure formée de l'union des interprétations.

Définition 6.1 *Soient* $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1, \mathcal{P}_1)$, $\Sigma_2 = (\mathcal{S}_2, \mathcal{F}_2, \mathcal{S}_2)$ *deux signatures du premier ordre et* $\Sigma_1 \cup \Sigma_2 = (\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{P}_1 \cup \mathcal{P}_2)$ *l'union des signatures. On note* $\mathcal{A}_1 \cup \mathcal{A}_2$ *la* $\Sigma_1 \cup \Sigma_2$*-structure définie par:*

- $(\mathcal{S}_1 \cup \mathcal{S}_2)_{\mathcal{A}_1 \cup \mathcal{A}_2} = (\mathcal{S}_1)_{\mathcal{A}_1} \cup (\mathcal{S}_2)_{\mathcal{A}_2}$,
- $(\mathcal{F}_1 \cup \mathcal{F}_2)_{\mathcal{A}_1 \cup \mathcal{A}_2} = (\mathcal{F}_1)_{\mathcal{A}_1} \cup (\mathcal{F}_2)_{\mathcal{A}_2}$,
- $(\mathcal{P}_1 \cup \mathcal{P}_2)_{\mathcal{A}_1 \cup \mathcal{A}_2} = (\mathcal{P}_1)_{\mathcal{A}_1} \cup (\mathcal{P}_2)_{\mathcal{A}_2}$.

La structure $\mathcal{A}_1 \cup \mathcal{A}_2$ n'est cependant définie que si les symboles partagés ont la même interprétation dans les deux sous-structures $\mathcal{A}_1$ et $\mathcal{A}_2$. Lorsque c'est le cas, une contrainte i-pure est valide dans $\mathcal{A}_1 \cup \mathcal{A}_2$ si et seulement si elle l'est dans $\mathcal{A}_i$. Dans cette forme de combinaison, les ensemble des sortes $\mathcal{S}_1$ et $\mathcal{S}_2$ peuvent être disjoints même si les domaines d'interprétation ont une intersection non vide, i.e. $(\mathcal{S}_1)_{\mathcal{A}_1} \cap (\mathcal{S}_2)_{\mathcal{A}_2} \neq \emptyset$, auquel cas un terme hétérogène peut avoir une solution dans $\mathcal{A}_1 \cup \mathcal{A}_2$. Nous nous intéresserons plus particulièrement au cours de ce chapitre à des algèbres finies ayant une intersection des domaines non vide.

Le mélange de théories équationnelles $(\mathcal{F}_1, E_1)$ et $(\mathcal{F}_2, E_2)$ mono-sortées et partageant la même sorte s_* ne rentre pas dans ce cadre puisque $=_{E_1 \cup E_2} \neq =_{E_1} \cup =_{E_2}$. Nous avons pourtant vu aux chapitres précédents que sous certaines conditions comme par exemple les signatures disjointes, il était possible là aussi de préserver les égalités de $=_{E_1}$ et $=_{E_2}$ et plus encore de pouvoir obtenir un algorithme d'unification dans $E_1 \cup E_2$ en combinant ceux existants dans E_1 et E_2.

Nous allons maintenant chercher à construire sur le même principe un langage de contraintes combiné avec conservation des propriétés atomiques de deux structures $\mathcal{A}_1$ et $\mathcal{A}_2$ mono-sortées et partageant la même sorte s_*. Il n'est pas surprenant d'avoir à considérer dans ce cas la classe des modèles validant les égalités de $\mathcal{T}h(\mathcal{A}_1)$ et celles de $\mathcal{T}h(\mathcal{A}_2)$ et plus précisement l'algèbre libre sur un ensemble de variables $\mathcal{X}$.

Définition 6.2 *Soient* $\Sigma_1 = (\{s_*\}, \mathcal{F}_1, \mathcal{P}_1)$ *et* $\Sigma_2 = (\{s_*\}, \mathcal{F}_2, \mathcal{S}_2)$ *deux signatures du premier ordre. Le* langage combiné $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$ *est défini par la structure* $\mathcal{C}$ *qui est l'algèbre de termes quotient* $\mathcal{T}(\{s_*\}, \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{X})/ =_{E_1 \cup E_2}$ *où* $E_1 = \mathcal{T}h(\mathcal{A}_1)$, $E_2 = \mathcal{T}h(\mathcal{A}_2)$ *et* $\mathcal{X}$ *est un ensemble de variables de sorte* s_*. *On note* $=_{\mathcal{C}}$ *la relation* $=_{E_1 \cup E_2}$.

Cette définition est volontairement incomplète et sera précisée en temps utile. Il faudra en particulier définir une interprétation pour les prédicats.

Nous utiliserons comme représentants canoniques des classes d'équivalence de $=_{E_1 \cup E_2}$ les formes normales suivant le système de réécriture combiné R introduit aux chapitres 2 et 3. Soient $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ et $>$ un ordre de simplification total sur les termes (clos) $\mathcal{T}(\mathcal{F} \cup \mathcal{X})$. Rappelons que l'ensemble des instances hétérogènes orientées des E_i-égalités est noté $E_i^{>}$. On note que la construction de R qui est faite précédemment ne suppose pas que $=_{E_1}$ ou $=_{E_2}$ soit finiment engendrée.

6.1.1 Hypothèses

Les conditions à satisfaire par l'égalité dans les structures composantes ne sont pas restrictives. On suppose que seules les constantes sont partagées. S'autoriser d'autres symboles partagés n'est pas réaliste dans ce contexte car cela implique de fortes restrictions sur ces symboles et donc sur l'égalité dans chaque composante, qui rappelons-le, doit être quelconque.

Hypothèse 6.1

1. *Les signatures* $\Sigma_1 = (\{s_*\}, \mathcal{F}_1, \mathcal{P}_1)$ *et* $\Sigma_2 = (\{s_*\}, \mathcal{F}_2, \mathcal{P}_2)$ *sont finies.*
2. $\mathcal{P}_1 \cap \mathcal{P}_2 = \{=^?\}$.
3. $\mathcal{F}_1 \cap \mathcal{F}_2 = SC$ *est un ensemble de constantes.*
4. *Les* Σ_1*-structure* $\mathcal{A}_1$ *et* Σ_2*-structure* $\mathcal{A}_2$ *sont consistantes.*
5. *Pour chaque structure* $\mathcal{A}_i$ $(i = 1, 2)$,

$$\forall a, b \in \mathcal{F}_i, a \neq b,\ (a)_{\mathcal{A}_i} \neq (b)_{\mathcal{A}_i}.$$

En conséquence, les théories $E_1 = \mathcal{T}h(\mathcal{A}_1)$ et $E_2 = \mathcal{T}h(\mathcal{A}_2)$ sont consistantes et il n'y a pas de E_i-égalité entre constantes dont seulement certaines sont partagées par E_1 et E_2. Nous nous sommes donc replacés dans les conditions vues au chapitre 3 pour la combinaison d'algorithmes d'unification dans le mélange de théories équationnelles partageant des constantes uniquement.

Rappelons seulement qu'il ne peut y avoir de constante partagée en tête des membres gauches des règles du système de réécriture combiné R. Une constante partagée est donc un terme irréductible.

6.1.2 Abstraction

Une contrainte atomique $p(t_1, \ldots, t_n)$ hétérogène est décomposée en une conjonction de contraintes atomiques par l'introduction de nouvelles equations résolues de la forme $x =^? t$ où t est sous terme étranger et x la variable qui le remplace.

Au niveau des preuves des théorèmes équationnels, nous avons déjà évoqué le besoin d'un outil analogue permettant de représenter chaque classe de termes hétérogènes équivalents par une unique variable.

Définition 6.3 *Une* abstraction par variables *est une bijection π entre l'ensemble des termes R-normalisés $T\downarrow_R = \{u\downarrow_R | u \in T(\mathcal{F}\cup\mathcal{X})$ et $u\downarrow_R \in T(\mathcal{F}\cup\mathcal{X})\backslash(\mathcal{X}\cup SC)\}$ et un sous-ensemble de variables de $\mathcal{X}$.*

Un terme partagé, nécessairement une variable ou une constante, n'a pas à être abstrait.

Le terme t^{π_i}, appelé *i-abstraction* du terme t, est défini de façon inductive comme suit:

- si $t = a \in SC$ alors $t^{\pi_i} = a$,
- si $t = x \in \mathcal{X}$ alors $t^{\pi_i} = x$,
- si $t = f(s_1,\ldots,s_p)$ et $f \in \mathcal{F}_i$ alors $t^{\pi_i} = f(s_1^{\pi_i},\ldots,s_p^{\pi_i})$
  sinon si $t\downarrow_R \notin (\mathcal{X}\cup SC)$ alors $t^{\pi_i} = \pi(t\downarrow_R)$ sinon $t^{\pi_i} = t\downarrow_R$.

Exemple 6.1 *Considérons les théories équationnelles $E_1 = \{x\times\top = x\}$, $E_2 = \{x+\top = \top\}$ et le terme hétérogène $t = (y\times(y+\top))\times\top$. t^{π_1} est $(y\times\top)\times\top$ puisque $(y+\top)\downarrow_R = \top$. t^{π_2} est y puisque $t\downarrow_R = y$.*

On rappelle que l'application réciproque π^{-1} de π n'est pas à proprement parler une substitution car elle a un domaine potentiellement infini mais on utilisera systématiquement sa restriction à un ensemble fini de variables. La substitution σ^{π_i} désigne la i-abstraction de σ definie par $x\sigma^{\pi_i} = \{x \mapsto (x\sigma)^{\pi_i} | x \in \mathcal{D}om(\sigma)\}$. Nous utiliserons dans la suite le fait que t^{π_i} est un terme i-pur, que $(t\sigma)^{\pi_i} = t\sigma^{\pi_i}$ si t est i-pur et finalement que $\sigma^{\pi_i} \leq^{\mathcal{X}}_{\emptyset} \sigma$ si σ est R-normalisée.

6.1.3 Résolution dans une composante

Intéressons-nous tout d'abord au cas des contraintes équationnelles en adaptant les résultats vus pour la combinaison d'algorithmes d'unification dans les théories équationnelles.

Lemme 6.1 *Soit t un terme i-pur et σ une substitution R-normalisée. Alors*

$$((t\sigma)\downarrow_R)^{\pi_i} =_{\mathcal{A}_i} t\sigma^{\pi_i}.$$

Preuve : D'après le Lemme 2.12, nous savons que $(t\sigma)\downarrow_R = (t\sigma)\downarrow_{E_i^>}$ et donc

$$((t\sigma)\downarrow_R)^{\pi_i} =_{E_i} t\sigma^{\pi_i}$$

avec, par définition, les équivalences suivantes:

$$((t\sigma)\downarrow_R)^{\pi_i} =_{E_i} t\sigma^{\pi_i} \Leftrightarrow ((t\sigma)\downarrow_R)^{\pi_i} =_{\mathcal{T}h(\mathcal{A}_i)} t\sigma^{\pi_i} \Leftrightarrow ((t\sigma)\downarrow_R)^{\pi_i} =_{\mathcal{A}_i} t\sigma^{\pi_i}.$$

□

Proposition 6.1 *Soient s et t deux termes i-purs et σ une substitution R-normalisée. Alors*

$$s\sigma =_{\mathcal{C}} t\sigma \Leftrightarrow s\sigma^{\pi_i} =_{\mathcal{A}_i} t\sigma^{\pi_i}.$$

Preuve : D'après le Lemme 6.1, $t\sigma^{\pi_i} =_{\mathcal{A}_i} ((t\sigma) \downarrow_R)^{\pi_i}$ et $t\sigma^{\pi_i} =_{\mathcal{A}_i} ((s\sigma) \downarrow_R)^{\pi_i}$ avec $s\sigma =_{E_1 \cup E_2} t\sigma$ si et seulement si $(s\sigma) \downarrow_R = (t\sigma) \downarrow_R$. □

L'interprétation d'un prédicat p de $\mathcal{P}_i$ différent de l'égalité est d'abord définie sur l'ensemble des termes $\mathcal{T}(\mathcal{F}, \mathcal{X})$ puis on montrera la compatibilité vis-à-vis de la relation d'équivalence $=_{\mathcal{C}}$.

Définition 6.4 *(Extension des prédicats). Soient p un prédicat n-aire de $\mathcal{P}_i$, $t_1, \ldots, t_n$ des termes de $\mathcal{T}(\mathcal{F} \cup \mathcal{X})$. L'interprétation $p_{\mathcal{C}}$ de p est la relation définie par:*
$p_{\mathcal{C}}(t_1, \ldots, t_n)$ est vrai ssi $\mathcal{A}_i \models p((t_1 \downarrow_R)^{\pi_i}, \ldots, (t_n \downarrow_R)^{\pi_i})$.

Cette définition est compatible avec la relation d'équivalence $=_{\mathcal{C}}$. En effet si $t_k =_{E_1 \cup E_2} s_k$ alors leur forme normale suivant R est identique et donc $(t_k \downarrow_R)^{\pi_i} =_{\mathcal{A}_i} (s_k \downarrow_R)^{\pi_i}$ puisque $(t_k \downarrow_R)^{\pi_i} = (s_k \downarrow_R)^{\pi_i}$.

Proposition 6.2 *Soit p un prédicat n-aire de $\mathcal{P}_i$.*
$\mathcal{C} \models p(t_1, \ldots, t_n)$ si et seulement si $\mathcal{A}_i \models p((t_1 \downarrow_R)^{\pi_i}, \ldots, (t_n \downarrow_R)^{\pi_i})$.

Preuve : On montre tout d'abord que $(t \downarrow_R)^{\pi_i} \leq_{\mathcal{A}_i} (t\sigma \downarrow_R)^{\pi_i}$ si σ est une substitution R-normalisée et t un i-terme. Considérons les termes v et u obtenus en remplaçant les sous-termes étrangers par leurs R-formes normales. D'après le Lemme 6.1, nous avons $v^{\pi_i} =_{\mathcal{A}_i} (t \downarrow_R)^{\pi_i}$ et $u^{\pi_i} =_{\mathcal{A}_i} (t\sigma \downarrow_R)^{\pi_i}$. Le terme u^{π_i} est une instance de v^{π_i}, i.e. $u^{\pi_i} = v^{\pi_i}\phi$, où ϕ est définie comme suit:

- $s^{\pi_i}\phi = (s\sigma \downarrow_R)^{\pi_i}$ si $s \in AST(v)$,
- $x\phi = x\sigma^{\pi_i}$ si $x \in \mathcal{V}_i(v)$.

Par conséquent, $(t \downarrow_R)^{\pi_i} =_{\mathcal{A}_i} v^{\pi_i} \leq u^{\pi_i} =_{\mathcal{A}_i} (t\sigma \downarrow_R)^{\pi_i}$.

Maintenant, si $p_{\mathcal{C}}(t_1, \ldots, t_n)$ est vrai alors $p((t_1 \downarrow_R)^{\pi_i}, \ldots, (t_n \downarrow_R)^{\pi_i})$ est valide dans $\mathcal{A}_i$ ainsi que $p((t_1\sigma \downarrow_R)^{\pi_i}, \ldots, (t_n\sigma \downarrow_R)^{\pi_i})$ et donc $p_{\mathcal{C}}(t_1\sigma, \ldots, t_n\sigma)$ est encore vrai.

Pour la réciproque, il suffit de considérer la substitution identité qui est R-normalisée. □

Exemple 6.2 *Dans le langage obtenu par combinaison de l'algèbre de Boole et de la théorie équationnelle $\{f(x) = f(x)\}$, les contraintes suivantes sont valides: $f(x) + \overline{f(x)} = \mathbf{1}$, $f(x) \neq \overline{f(x)}$, $f(x) + \mathbf{1} \geq \mathbf{1}$ etc...*

L'interprétation $\mathcal{C}$ définie ci-dessus à l'avantage de préserver exactement la validité des contraintes pures de chaque langage et rien que celles-ci.

Proposition 6.3 *Si c_i est une contrainte atomique i-pure de $CL_{\mathcal{A}_i}[\Sigma_i, \mathcal{X}]$ alors*

$$\mathcal{C} \models c_i \Leftrightarrow \mathcal{A}_i \models c_i.$$

Preuve : Si $t_1, \ldots, t_n$ sont i-purs alors $(t_k \downarrow_R)^{\pi_i} =_{\mathcal{A}_i} t_k$ pour $k = 1, \ldots, n$ et donc

$$\mathcal{A}_i \models p(t_1, \ldots, t_n) \Leftrightarrow \mathcal{A}_i \models p((t_1 \downarrow_R)^{\pi_i}, \ldots, (t_n \downarrow_R)^{\pi_i}),$$

c'est-à-dire $\mathcal{C} \models p(t_1, \ldots, t_n)$ si et seulement si $\mathcal{A}_i \models p(t_1, \ldots, t_n)$. □

La satisfaisabilité des contraintes pures est préservée pour cette interprétation des prédicats: si c_i est une contrainte i-pure, alors à une solution dans $\mathcal{C}$ correspond une solution symbolique dans $\mathcal{A}_i$.

Proposition 6.4 *Soient $p \in \mathcal{P}_i$, $p(t_1, \ldots, t_n)$ une contrainte atomique i-pure et σ une substitution R-normalisée. Alors*

$$\mathcal{C} \models p(t_1\sigma, \ldots, t_n\sigma) \Leftrightarrow \mathcal{A}_i \models p(t_1\sigma^{\pi_i}, \ldots, t_n\sigma^{\pi_i}).$$

Preuve : Par définition, $p(t_1\sigma, \ldots, t_n\sigma)$ est vrai dans $\mathcal{C}$ ssi $p(((t_1\sigma) \downarrow_R)^{\pi_i}, \ldots, ((t_n\sigma) \downarrow_R)^{\pi_i})$ est valide dans $\mathcal{A}_i$. A nouveau d'après le Lemme 6.1, on a $((t_k\sigma) \downarrow_R)^{\pi_i} =_{\mathcal{A}_i} t_k\sigma^{\pi_i}$ et donc $p(((t_1\sigma) \downarrow_R)^{\pi_i}, \ldots, ((t_n\sigma) \downarrow_R)^{\pi_i})$ est valide dans $\mathcal{A}_i$ si et seulement si $p(t_1\sigma^{\pi_i}, \ldots, t_n\sigma^{\pi_i})$ est valide $\mathcal{A}_i$. □

En conséquence, un ensemble complet de solutions symboliques $CSS_{\mathcal{A}_i}(c)$ est encore un $CSS_{\mathcal{C}}(c)$ si c est une contrainte i-pure, ce qui signifie qu'un résolveur de contraintes symboliques dans $CL_{\mathcal{A}_i}[\Sigma_i, \mathcal{X}]$ peut-être utilisé de façon correcte et complète. C'est ce résultat qui justifie notre démarche et permet de voir $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$ comme "le langage combiné".

Dans notre approche, une diséquation i-pure $s \neq_i^? t$ peut être résolue dans $\mathcal{A}_i$ sans perte de complétude si $\neq_i$ est vue comme un prédicat. Mais l'interprétation dans $\mathcal{C}$ qui est faite du prédicat $\neq_i$ ne correspond pas à la négation de $=_{\mathcal{C}}$ mais à l'extension de $\neq_i$ au domaine de $\mathcal{C}$. On notera que dans ce contexte, chaque langage de contraintes composant doit posséder son propre prédicat de diségalité. Cette approche constructive de la négation permet de réutiliser un algorithme de disunification dans un langage composant comme ce qui est fait pour l'unification.

6.1.4 Combinaison des solutions

Le problème de la combinaison des solutions fournies par chaque langage composant se pose à nouveau. En effet, nous savons qu'un ensemble complet de solutions symboliques $CSS_{\mathcal{A}_i}(c_i)$ est un $CSS_{\mathcal{C}}(c_i)$ pour $i = 1, 2$ si $c_1 \wedge c_2$ est la conjonction de contraintes pures équivalente à la contrainte initiale c. Mais cela ne suffit pas pour trouver l'ensemble complet $CSS_{\mathcal{C}}(c_1 \wedge c_2)$. Comme pour la combinaison d'algorithmes d'unification, on montre qu'un ensemble complet de solutions $CSS_{\mathcal{C}}(c_1 \wedge c_2)$ est obtenu par combinaison de solutions de $SS_{\mathcal{A}_1}(c_1)$ et $SS_{\mathcal{A}_2}(c_2)$ satisfaisant une même restriction linéaire. Cette notion s'étend parfaitement à des contraintes non équationnelles.

Définition 6.5 *Soient $<$ un ordre linéaire sur $V_1 \oplus V_2$ l'union disjointe de deux ensembles de variables et c_i une contrainte de $CL_{\mathcal{A}_i}[\Sigma_i, \mathcal{X}]$ telle que $\mathcal{V}(c_i) \subseteq V_1 \oplus V_2$. Une* solution symbolique suivant la restriction linéaire $<$ *est une solution symbolique σ telle que*

- $\forall x_j \in V_j$, $x_j\sigma = x_j$,

- $\forall x_j \in V_j,\ \forall x_i \in V_i,\ x_j \notin x_i\sigma$ *si* $x_i < x_j$.

L'ensemble de ces solutions symboliques est noté $SS^{<}_{\mathcal{A}_i}(c_i, V_j)$.

Définition 6.6 *Un ensemble complet de substitutions est un* ensemble complet de solutions suivant la restriction linéaire $<$ *d'une* $CL_{\mathcal{A}_i}[\Sigma_i, \mathcal{X}]$*-contrainte* c_i*, désigné par* $CSS^{<}_{\mathcal{A}_i}(c_i, V_j)$*, si*

1. *(Protection des variables)* $\forall \sigma \in CSS^{<}_{\mathcal{A}_i}(c_i, V_j),\ \mathcal{D}om(\sigma) \cap \mathcal{VR}an(\sigma) = \emptyset$.
2. *(Correction)* $CSS^{<}_{\mathcal{A}_i}(c_i, V_j) \subseteq SS^{<}_{\mathcal{A}_i}(c_i, V_j)$.
3. *(Complétude)* $\forall \phi \in SS^{<}_{\mathcal{A}_i}(c_i, V_j),\ \exists \sigma \in CSS^{<}_{\mathcal{A}_i}(c_i, V_j),\ \sigma \leq^{\mathcal{V}(c_i)}_{\mathcal{A}_i} \phi$.

Un ensemble complet est minimal *si deux substitutions de cet ensemble ne peuvent être comparées par* $\leq^{\mathcal{V}(c_i)}_{\mathcal{A}_i}$*. Lorque un ensemble complet minimal est un singleton, celui-ci est noté* $mgs^{<}_{\mathcal{A}_i}(c_i, V_j)$.

Le résultat suivant est analogue à celui donné par F. Baader et K. Schulz pour la combinaison d'algorithmes d'unification. On s'intéresse d'abord à un résultat de décidabilité.

On montre dans un premier temps que pour toute solution de $c_1 \wedge c_2$ dans $\mathcal{C}$ il existe une identification, une instanciation par des constantes partagées et enfin une restriction linéaire pour lesquelles il existe une solution dans chaque langage composant.

Proposition 6.5 *La résolution de contraintes dans* $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$ *est décidable si la résolution de contraintes avec restriction linéaire est décidable dans les langages* $CL_{\mathcal{A}_1}[\Sigma_1, \mathcal{X}]$ *et* $CL_{\mathcal{A}_2}[\Sigma_2, \mathcal{X}]$.

Preuve : Soit σ une solution R-normalisée d'une conjonction de contraintes atomiques $c_1 \wedge c_2$. On peut supposer sans perte de généralité qu'il n'existe pas deux variables x, y dans le problème tel que $x\sigma = y\sigma$, sinon elles doivent d'abord être identifiées. L'abstraction par variables π peut être alors définie comme une bijection telle que pour les variables x du problème on a $\pi(x\sigma) = x$ si $x\sigma \notin \mathcal{X} \cup SC$. Ainsi, $x\sigma$ est abstrait par x lorsque $x\sigma$ est sous-terme étranger. L'ordre linéaire $<$ est choisi tel que $x < y$ si $x\sigma$ est sous-terme strict de $y\sigma$.

On considère les substitutions σ_1 et σ_2 définies par

$$\sigma_i = \{x \mapsto x\sigma^{\pi_i}\}_{|x\sigma(\epsilon) \in \mathcal{F}_i \setminus SC},$$

et l'instanciation par des constantes

$$\rho = \{x \mapsto x\sigma\}_{|x\sigma \in SC}.$$

Par hypothèse, on a $\sigma \in SS_{\mathcal{C}}(c_1 \wedge c_2)$ et donc $\sigma^{\pi_1} \in SS_{\mathcal{A}_1}(c_1)$ et $\sigma^{\pi_2} \in SS_{\mathcal{A}_2}(c_2)$ ou encore $\sigma_1 \in SS_{\mathcal{A}_1}(c_1\rho)$ et $\sigma_2 \in SS_{\mathcal{A}_2}(c_2\rho)$, d'après les notations introduites. Chaque variable n'est instanciée que par une seule des deux substitutions, σ_1 ou σ_2. La restriction linéaire est définie sur $V_1 \oplus V_2$ avec $V_i = \mathcal{D}om(\sigma_i)$. Il faut maintenant s'assurer que les substitutions σ_1 et σ_2 satisfont la même restriction $<$. Si le terme $x\sigma_i$ $(i = 1, 2)$ contient une variable y instanciée dans σ_j $(j \neq i)$ alors montrons que $x \not< y$. Par construction, si y est une variable de $x\sigma_i$ alors le sous terme étranger $y\sigma$ est sous terme de $x\sigma$. Il est sous terme strict car $x\sigma$ et $y\sigma$ sont différents puisque les variables x et y sont non identifiées. Donc $y < x$. En conclusion $\sigma_1 \in SS^{<}_{\mathcal{A}_1}(c_1\rho, V_2)$ et $\sigma_2 \in SS^{<}_{\mathcal{A}_2}(c_2\rho, V_1)$. □

Nous n'avons pour l'instant mis en évidence qu'un résultat de complétude. Réciproquement, deux solutions dans chaque langage de contraintes doivent être combinées pour donner une solution dans le langage combiné.

La prochaine section est consacrée à la construction effective d'un ensemble complet de solutions symboliques dans le langage combiné à partir des ensembles complets calculés par chaque résolveur de contraintes.

6.1.5 Solutions combinées

Une solution combinée correspond simplement à la transformation d'une forme séquentiellement résolue en une forme résolue par remplacements successifs.

Définition 6.7 *Soient c_1 et c_2 deux contraintes respectivement 1-pure et 2-pure. La* solution combinée $\sigma_1 \odot \sigma_2$ *de c_1 et c_2 obtenue à partir de $\sigma_1 \in SS^{<}_{\mathcal{A}_1}(c_1, V_2)$ et $\sigma_2 \in SS^{<}_{\mathcal{A}_2}(c_2, V_1)$ est définie comme suit: si x une variable de V_i et $\{y_1, \ldots y_n\}$ l'ensemble des variables de V_j $(i \neq j)$ inférieures à x par rapport à $<$, alors $x\sigma = x\sigma_i[y_k \hookleftarrow y_k\sigma]_{k=1,\ldots,n}$.*

Proposition 6.6 *Une solution combinée $\sigma_1 \odot \sigma_2$ de c_1 et c_2 est solution symbolique de $c_1 \wedge c_2$ dans $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$.*

Preuve : Par définition nous avons $\sigma_1 \in SS_{\mathcal{A}_1}(c_1)$ et $\sigma_2 \in SS_{\mathcal{A}_2}(c_2)$ et donc $\sigma_1 \in SS_{\mathcal{C}}(c_1)$ et $\sigma_2 \in SS_{\mathcal{C}}(c_2)$. Ensuite, par construction $\sigma_i \leq_{\mathcal{C}}^{\mathcal{V}(c_1 \wedge c_2)} \sigma_1 \odot \sigma_2$. N'importe quelle instance d'une solution σ_i de c_i est une solution de c_i. Ainsi $\sigma_1 \odot \sigma_2 \in SS_{\mathcal{C}}(c_1)$ et $\sigma_1 \odot \sigma_2 \in SS_{\mathcal{C}}(c_2)$, c'est-à-dire $\sigma_1 \odot \sigma_2 \in SS_{\mathcal{C}}(c_1 \wedge c_2)$. □

La proposition 2.20 introduite au chapitre 2 s'applique en fait à des solutions symboliques de n'importe quelles contraintes. Elle stipule qu'un ensemble complet de solutions combinées est obtenu en combinant des ensembles complets de solutions. C'est ce qui nous permet d'obtenir aussi un algorithme de résolution.

Théorème 6.1 *La résolution de contraintes dans $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$ est finitaire (resp. décidable) si la résolution de contraintes avec restriction linéaire est finitaire (resp. décidable) dans les langages $CL_{\mathcal{A}_1}[\Sigma_1, \mathcal{X}]$ et $CL_{\mathcal{A}_2}[\Sigma_2, \mathcal{X}]$.*

L'algorithme est basé sur la proposition suivante:

Proposition 6.7 *L'ensemble des substitutions $\xi\rho(\sigma_1 \odot \sigma_2)$ tel que*

- $\xi \in ID_{\mathcal{V}(c_1 \wedge c_2)}$,
- $\rho \in SUBS^{SC}_{\mathcal{V}((c_1 \wedge c_2)\xi)}$,
- $<$ *est un ordre linéaire sur* $V_1 \oplus V_2 = \mathcal{V}((c_1 \wedge c_2)\xi\rho)$,
- $\sigma_1 \in CSS^{<}_{\mathcal{A}_1}(c_1\xi\rho, V_2)$ *et* $\sigma_2 \in CSS^{<}_{\mathcal{A}_2}(c_2\xi\rho, V_1)$,

est un $CSS_{\mathcal{C}}(c_1 \wedge c_2)$.

Preuve : Nous avons vu pour la proposition 6.5 qu'une solution σ R-normalisée est une instance d'une solution combinée:

$$\sigma = (\sigma_1 \odot \sigma_2)\pi^{-1}$$

où $\sigma_1 \in SS^{<}_{\mathcal{A}_1}(c_1\rho)$ et $\sigma_2 \in SS^{<}_{\mathcal{A}_2}(c_2\rho)$, avec $\sigma_1, \sigma_2, \rho, <$ définis en fonction de σ et c_1, c_2 identifiées en fonction de σ.

D'après la proposition 2.20, il existe $\sigma'_1 \in CSS^{<}_{\mathcal{A}_1}(c_1\rho)$ et $\sigma'_2 \in CSS^{<}_{\mathcal{A}_2}(c_2\rho)$ telles que

$$\sigma'_1 \odot \sigma'_2 \leq^{V_1 \oplus V_2}_{\mathcal{C}} \sigma,$$

où $V_1 \oplus V_2 = \mathcal{V}(c_1\rho \wedge c_2\rho)$. En considérant maintenant toutes les variables de $\mathcal{V}(c_1 \wedge c_2)$, on a

$$(\sigma'_1 \odot \sigma'_2)\rho \leq^{\mathcal{V}(c_1 \wedge c_2)}_{\mathcal{C}} \sigma$$

puisque σ et ρ sont égales sur les variables $\mathcal{V}(c_1 \wedge c_2)\backslash(V_1 \oplus V_2)$. □

L'algorithme correspondant permet de combiner soit des algorithmes de résolution, soit des algorithmes de décision.

6.1.6 Règles pour la combinaison

Les règles de transformation sont données en figure 6.1. Il y a principalement deux étapes appelées Purification et Combinaison qui produisent une forme séquentiellement résolue. Une troisième étape consiste à appliquer la règle de Remplacement jusqu'à obtenir une forme résolue. L'application séquentielle de ces trois étapes termine évidement et retourne un ensemble complet de $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$-solutions symboliques.

L'étape déterministe Purification transforme une contrainte c en conjonction de contraintes pures $c_1 \wedge c_2$ chacune satisfaisable dans un langage composant.

L'étape indéterministe Combinaison est consacrée à la résolution dans chaque composante et à la combination des solutions. Les paramètres de cette étape sont

- Une conjonction $c_1 \wedge c_2$ formée de $CL_{\mathcal{A}_i}[\Sigma_i, \mathcal{X}]$-contraintes c_i $(i = 1, 2)$.
- Un ensemble de variables V_1 instanciées dans $CL_{\mathcal{A}_1}[\Sigma_1, \mathcal{X}]$ et gelées dans $CL_{\mathcal{A}_2}[\Sigma_2, \mathcal{X}]$.
- Un ensemble de variables V_2 instanciées dans $CL_{\mathcal{A}_2}[\Sigma_2, \mathcal{X}]$ et gelées dans $CL_{\mathcal{A}_1}[\Sigma_1, \mathcal{X}]$.
- Un ordre linéaire sur l'union disjointe de V_1 et V_2 qui représente l'ordre d'"Occur-Check" de la forme séquentiellement résolue souhaitée.

On rappelle que $\hat{\sigma}$ désigne la forme résolue associée à toute substitution σ idempotente.

6.2 Extension du résolveur de contraintes dans les algèbres finies

Nous allons d'abord préciser comment il est possible de repositionner simplement le problème dans le cadre de la combinaison d'algorithmes d'unification lorsque les composants sont un langage primal $CL_{AF}[\overline{\Sigma}, \mathcal{X}]$ et un langage équationnel $CL_{E_2}[\Sigma_2, \mathcal{X}]$.

1. Purification

   Appliquer tant que possible les règles suivantes:

   Abstraction Variable(Contrainte)

   $$\frac{c \wedge p(t_1, \ldots, t_m)}{\exists x : c \wedge p(t_1, \ldots, t_k[x]_\omega, \ldots, t_m) \wedge x =^? t_{k|\omega}}$$

   $$\textbf{si} \begin{cases} p \in \mathcal{P}_i, \\ \omega \in AlienPos(t_k) \text{ si } t_k(\epsilon) \in \mathcal{F}_i, \omega = \epsilon \text{ si } t_k(\epsilon) \in \mathcal{F}_j, j \neq i, \\ x \text{ est une nouvelle variable} \end{cases}$$

   Abstraction Variable(Equation)

   $$\frac{c \wedge s =^? t}{\exists x : c \wedge s =^? t[x]_\omega \wedge x =^? t_{|\omega}} \quad \textbf{si} \begin{cases} \omega \in AlienPos(t), \\ x \text{ est une nouvelle variable} \end{cases}$$

   Equation Impure

   $$\frac{c \wedge s =^? t}{\exists x : c \wedge x =^? s \wedge x =^? t} \quad \textbf{si} \begin{cases} s \in T(\mathcal{F}_1, \mathcal{X}) \backslash \mathcal{X},\ t \in T(\mathcal{F}_2, \mathcal{X}) \backslash \mathcal{X} \\ x \text{ est une nouvelle variable} \end{cases}$$

2. Combinaison

   Considérer

   - toute identification $\xi \in ID_{\mathcal{V}(c_1 \wedge c_2)}$,
   - toute instanciation $\rho \in SUBS^{SC}_{\mathcal{V}((c_1 \wedge c_2)\xi)}$,
   - tout ordre linéaire $<$ sur $V_1 \oplus V_2 = \mathcal{V}((c_1 \wedge c_2)\xi\rho)$

   et appliquer la règle suivante

   Resolution

   $$\frac{(c_1 \wedge c_2)}{(\hat{\rho} \wedge \hat{\xi} \wedge \hat{\sigma}_1 \wedge \hat{\sigma}_2)} \quad \textbf{si } \sigma_i \in CSS^{<}_{\mathcal{A}_i}(c_i \xi \rho, V_j)$$

3. Solution combinée

   Appliquer tant que possible la règle suivante

   Remplacement

   $$\frac{c \wedge x =^? t}{\{x \mapsto t\}(c) \wedge x =^? t} \quad \textbf{si } x \in \mathcal{V}(c) \text{ et } t \notin \mathcal{X}$$

Figure 6.1. Résolution de contraintes dans $CL_{\mathcal{C}}[\Sigma_1 \cup \Sigma_2, \mathcal{X}]$

Dans ce contexte bien particulier, une contrainte atomique primale c est équivalente dans $\overline{\mathcal{A}}$ à une conjonction d'équations $\widehat{mgs}(c)$. Ces deux contraintes ont donc même ensemble complet de solutions symboliques dans $\mathcal{C}$.

La purification étendue aux contraintes non équationnelles transforme la contrainte initiale en une conjonction de contraintes atomiques pures dans une composante du langage. Celles qui sont pures dans le langage primal sont ensuite transformées en équations. En définitive, on obtient une contrainte c' telle que $CSS_{\mathcal{C}}(c) = CSU_{AF \cup E_2}(c')$. Il s'agit donc dans ce cas d'un problème de combinaison d'algorithmes d'unification dans les théories équationnelles AF et E_2.

Pour résoudre ce problème de combinaison il faut disposer d'un algorithme d'unification avec restriction linéaire pour le langage primal et le langage équationnel. Nous allons nous intéresser plus généralement à la résolution de contraintes avec restriction linéaire dans un langage primal. Il faut pouvoir

1. considérer comme constantes les variables instanciées dans l'autre théorie E_2,
2. tenir compte d'une restriction linéaire en éliminant certaines constantes des termes solutions.

Cette décomposition est celle utilisée couramment pour la combinaison d'algorithmes d'unification (et non d'unifiabilité). Elle s'étend parfaitement à la résolution de contraintes symboliques et plus précisément à un langage primal où une contrainte atomique primale est équivalente à une contrainte équationnelle.

Ces deux phases distinctes nécessitent pour chaque théorie équationnelle des outils spécifiques qui sont l'unification avec constantes et l'élimination de constante. Ces outils sont obtenus pour la théorie AF en transposant le problème dans l'algèbre $\overline{\mathcal{A}}$ puisque l'égalité dans AF est celle dans $\overline{\mathcal{A}}$. Le résolveur de contraintes dans le langage $CL_{AF}[\overline{\Sigma}, \mathcal{X}]$ doit être spécialisé pour que les solutions symboliques n'instancient pas les variables dites *gelées*, qui sont considérées comme constantes dans la théorie équationnelle AF. D'autre part, les solutions ne doivent pas contenir certaines variables gelées, vues dans la théorie équationnelle AF comme des constantes à éliminer dans les termes solutions. Nous verrons que l'élimination de variable gelée peut s'effectuer par unification dans un langage primal. Ainsi une conjonction de problèmes d'élimination est simplement une contrainte équationnelle. C'est la raison pour laquelle nous nous limiterons à l'étude d'un problème élémentaire, à savoir l'élimination d'une seule variable gelée dans un seul terme.

6.2.1 Résolution de contraintes avec variables gelées

Le résolveur de contraintes dans $CL_{AF}[\overline{\Sigma}, \mathcal{X}]$ doit être spécialisé pour éviter l'instanciation de variables gelées.

Un ensemble complet de solutions d'une contrainte c par rapport à un ensemble de variables gelées $\mathcal{M} \in \mathcal{V}(c)$ est défini en imposant qu'une solution σ n'instancie pas les variables de $\mathcal{M}$: $\forall \sigma \in CSS(c, \mathcal{M}),\ \mathcal{D}om(\sigma) \subseteq \mathcal{V}(c) \backslash \mathcal{M}$. Lorsqu'un ensemble complet minimal $CSS(c, \mathcal{M})$ contient au plus un élément alors celui-ci sera noté $mgs(c, \mathcal{M})$.

Lorsqu'une substitution σ n'instancie pas les variables gelées $\mathcal{M}$ dans une contrainte c, ces variables apparaissent encore dans la contrainte $c\sigma$ et l'application σ_c entre assignations laisse inchangée les valeurs assignées aux variables gelées $\mathcal{M}$:

$$\forall \alpha\ \forall x \in \mathcal{M},\ \sigma_c(\alpha)(x) = \underline{\alpha}(x\sigma) = \alpha(x).$$

Nous allons voir comment s'intègre cette condition supplémentaire à celles données dans le cas sans variable gelée.

Cette condition signifie que les assignations δ des variables gelées $\mathcal{M}$ sont inchangées par l'application σ_c entre assignations et qu'il suffira de s'intéresser aux variables non gelées. Etant donnée une assignation $\delta \in ASS_A^{\mathcal{M}}$ de variables gelées, on note $\boldsymbol{\delta}$ la substitution obtenue en remplaçant la valeur $\delta(x) \in A$ par la constante correspondante $[\delta(x)] \in \overline{\mathcal{F}}$. La condition que les variables $\mathcal{M}$ ne sont pas instanciées par σ s'écrit

$$\mathcal{I}(\sigma_c) = \delta \cup \mathcal{I}(\sigma_{c\boldsymbol{\delta}}).$$

Les propositions suivantes sont des conséquences directes de cette décomposition appliquée aux résultats du chapitre précédent.

Proposition 6.8 *Si σ est une substitution n'instanciant pas les variables gelées $\mathcal{M}$ alors σ est une solution symbolique de c si et seulement si*

$$\forall \delta \in ASS_A^{\mathcal{M}},\ \mathcal{I}(\sigma_{c\boldsymbol{\delta}}) \subseteq Sol_A(c\boldsymbol{\delta})_{|\mathcal{V}(c\boldsymbol{\delta})}$$

L'équivalence entre préordre de subsomption des substitutions σ et σ' et inclusion des rangs de σ_c et σ'_c peut être raffinée lorsque σ et σ' gèlent les variables de $\mathcal{M}$.

Proposition 6.9 *Soit σ et σ' deux substitutions n'instanciant pas les variables gelées $\mathcal{M}$. Alors $\sigma \leq_{AF}^{\mathcal{V}(c)} \sigma'$ ssi $\forall \delta \in ASS_A^{\mathcal{M}},\ \mathcal{I}(\sigma_{c'\boldsymbol{\delta}}) \subseteq \mathcal{I}(\sigma_{c\boldsymbol{\delta}})$.*

Corollaire 6.1 *S'il existe une substitution σ n'instanciant pas les variables gelées $\mathcal{M}$ telle que $\forall \delta \in ASS_A^{\mathcal{M}},\ \mathcal{I}(\sigma_{c\boldsymbol{\delta}}) = Sol_A(c\boldsymbol{\delta})_{|\mathcal{V}(c\boldsymbol{\delta})}$, alors σ est une solution principale de c.*

La résolution d'une contrainte c par rapport à un ensemble de variables gelées $\mathcal{M}$ s'effectue en considérant les sous problèmes obtenus en projetant de toutes les façons possibles les variables gelées sur les valeurs du domaine. En effet, une solution symbolique σ est solution principale de c par rapport à un ensemble de variable gelées $\mathcal{M}$ si et seulement si pour toute substitution $\boldsymbol{\delta}$ des variables gelées $\mathcal{M}$, $\sigma\boldsymbol{\delta}$ est solution principale de $c\boldsymbol{\delta}$.

Théorème 6.2 *Soit c une contrainte et $\mathcal{M}$ un ensemble de variables inclus dans $\mathcal{V}(c)$. La substitution*

$$\{x \mapsto \sum_{\delta} \prod \delta(\mathcal{M}) \cdot mgs(c\boldsymbol{\delta})(x)\}_{x \in \mathcal{V}(c) \setminus \mathcal{M}}$$

est une solution principale de c par rapport à l'ensemble de variables gelées $\mathcal{M}$.

Preuve : Cette substitution n'instancie pas les variables de $\mathcal{M}$ et vérifie la condition du Corollaire 6.1. □

Exemple 6.3 *(Suite de l'exemple 5.8: $c = (x + yz =^? xyz)$). Supposons $\mathcal{M} = \{x\}$. Il faut considérer $c0 = (\mathbf{0} + yz =^? \mathbf{0}yz)$ où $Sol(c0) = \{\,\overline{y}\,\overline{z}\,, \overline{y}\,z, y\,\overline{z}\,\}$ et $c1 = (\mathbf{1} + yz =^? \mathbf{1}yz)$ où $Sol(c1) = \{yz\}$. Ainsi $mgs(c1) = \{y \mapsto \mathbf{1}, z \mapsto \mathbf{1}\}$ et l'application*

$$\sigma_{c0}(\,\overline{y_1}\ \overline{y_2}\,) = \overline{y}\ \overline{z} \quad \sigma_{c0}(\,\overline{y_1}\,y_2) = \overline{y}\,z \quad \sigma_{c0}(y_1\,\overline{y_2}\,) = y\,\overline{z} \quad \sigma_{c0}(y_1y_2) = y\,\overline{z}$$

produit $mgs(c0) = \{y \mapsto y_1, z \mapsto \overline{y_1}\,y_2\}$.
Les termes $\overline{x} \cdot mgs(c0)(y) + x \cdot mgs(c1)(y)$ et $\overline{x} \cdot mgs(c0)(z) + x \cdot mgs(c1)(z)$ sont respectivement associés à y et z: $mgs(c, \mathcal{M}) = \{y \mapsto \overline{x}\,y_1 + x, z \mapsto \overline{x}\ \overline{y_1}\,y_2 + x\}$.

6.2.2 Un algorithme d'élimination de variable gelée

Nous considérons à présent le problème des cycles composés

$$y_1 =^? t_1[x_1] \wedge x_1 =^? s_1[y_2] \wedge \cdots y_n =^? t_n[x_n] \wedge x_n =^? s_n[y_1]$$

de termes non variables purs alternativement dans chaque langage de contraintes. Ces cycles peuvent avoir des solutions dans le langage de contraintes combiné. Pour les trouver il faut calculer les bonnes instances qui transforment le cycle composé en une forme séquentiellement résolue. C'est l'objet d'un algorithme d'élimination de variable gelée.

Définition 6.8 *Un terme u est une instance d'un terme t suivant un ensemble de variables $\mathcal{M}$ s'il existe une substitution σ telle que*

- $\mathcal{D}om(\sigma) \cap \mathcal{M} = \emptyset$.
- $t\sigma =_{AF} u$.

Le préordre de subsomption ainsi défini sera noté $\leq_{AF}^{\overline{\mathcal{M}}}$.

Définition 6.9 *Un terme u* élimine *x dans t suivant un ensemble de variables gelées $\mathcal{M}$ si $x \notin u, x \notin \mathcal{M}$ et $t \leq_{AF}^{\overline{\mathcal{M}\cup\{x\}}} u$. L'ensemble de ces termes est noté $STE(x,t,\mathcal{M})$. La substitution σ est un* éliminateur *de x dans t. L'ensemble des éliminateurs de x dans t est noté $SE(x,t,\mathcal{M})$.*

Exemple 6.4 *Considérons le terme booléen $t = x + y$. La substitution $\phi = \{y \mapsto \overline{x}\}$ est un éliminateur de x dans t. Le terme $\mathbf{1}$ élimine x dans t. La substitution $\sigma = \{y \mapsto \overline{x} + z\}$ est aussi un éliminateur mais est plus général que ϕ.*

Définition 6.10 *Un ensemble de termes est un* ensemble complet de termes éliminant *x* dans *t suivant les variables gelées $\mathcal{M}$, noté par $CSTE(x,t,\mathcal{M})$, si*

1. $CSTE(x,t,\mathcal{M}) \subseteq STE(x,t,\mathcal{M})$.
2. $\forall u \in STE(x,t,\mathcal{M})\ \exists s \in CSTE(x,t,\mathcal{M}),\ s \leq_{AF}^{\overline{\mathcal{M}}} u$.

Un $CSTE(x,t,\mathcal{M})$ est minimal *si deux termes de $CSTE(x,t,\mathcal{M})$ ne peuvent pas être comparés par $\leq_{AF}^{\overline{\mathcal{M}\cup\{x\}}}$. Lorsqu'un tel ensemble est au plus un singleton, $mgte(x,t,\mathcal{M})$ désigne cet unique élément.*

Un terme u éliminant x dans t prend des valeurs de t notées

$$Val(t) = \{\underline{\alpha}(t) \mid \alpha \in ASS_A\},$$

et en particulier celles indépendantes des valeurs prises par x,

$$Common(x,t) = \bigcap_{i \in A} Val(t\{x \mapsto [i]\}).$$

Il y a à nouveau équivalence entre l'inclusion $\subseteq$ sur l'ensemble des valeurs prises par les termes et le préordre de subsomption $\leq_{AF}$ sur les termes.

Proposition 6.10 *Soient t et u deux termes.*

$$t \leq_{AF}^{\overline{\mathcal{M}}} u \text{ si et seulement si } \forall \delta \in ASS_A^{\mathcal{M}},\ Val(u\boldsymbol{\delta}) \subseteq Val(t\boldsymbol{\delta}).$$

En particulier, si $\mathcal{M} = \emptyset$ alors on a $t \leq_{AF} u$ si et seulement $Val(u) \subseteq Val(t)$.

Preuve : La preuve est basée sur l'application de la proposition 5.6 aux substitutions $\sigma = \{z \mapsto t\}$, $\sigma' = \{z \mapsto u\}$ et à la contrainte triviale

$$c = \bigwedge_{x \in \mathcal{V}(t) \cup \mathcal{V}(u) \cup \{z\}} x =^? x.$$

La variable z est supposée ne pas apparaitre dans s et t. Comme σ et σ' n'instancient pas les variables gelées $\mathcal{M}$, il existe une substitution μ telle que

$$\forall x \in \mathcal{V}(c),\ x\sigma = x\sigma'\mu$$

si et seulement si

$$\forall \delta \in ASS_A^{\mathcal{M}},\ \mathcal{I}({\sigma'}_{c\boldsymbol{\delta}}) \subseteq \mathcal{I}(\sigma_{c\boldsymbol{\delta}}),$$

cette condition étant équivalente à

$$\forall \delta \in ASS_A^{\mathcal{M}},\ Val(u\boldsymbol{\delta}) \subseteq Val(t\boldsymbol{\delta}).$$

On peut supposer sans perte de généralité que μ n'instancie pas les variables de $\mathcal{M}$ puisque

$$\forall x \in \mathcal{M},\ x\mu =_{AF} x.$$

□

Proposition 6.11 *Si u est un terme éliminant x dans t suivant un ensemble de variables gelées $\mathcal{M}$ alors*

$$\forall \delta \in ASS_A^{\mathcal{M}}, Val(u\boldsymbol{\delta}) \subseteq Common(x, t\boldsymbol{\delta}).$$

Preuve : Le terme u est une instance de t suivant un ensemble de variables gelées $\mathcal{M} \cup \{x\}$ avec $x \notin u$.

$$t \leq_{AF}^{\overline{\mathcal{M} \cup \{x\}}} u \Leftrightarrow \forall i \in A, \forall \delta \in ASS_A^{\mathcal{M}},\ Val(u\boldsymbol{\delta}\{x \mapsto [i]\}) \subseteq Val(t\boldsymbol{\delta}\{x \mapsto [i]\}).$$

D'autre part, les valeurs prises par u ne dépendent pas des valeurs de x puisque $x \notin u$. Ainsi

$$Val(u\boldsymbol{\delta}) = Val(u\boldsymbol{\delta}\{x \mapsto [0]\}) = \ldots = Val(u\boldsymbol{\delta}\{x \mapsto [n-1]\}).$$

En regroupant, on obtient

$$Val(u\boldsymbol{\delta}) \subseteq \bigcap_{i \in A} Val(x, t\boldsymbol{\delta}\{x \mapsto [i]\}) = Common(x, t\boldsymbol{\delta}).$$

□

Il suffit ensuite d'exhiber un terme u ne satisfaisant pas seulement l'inclusion mais l'égalité: c'est un terme principal éliminant x dans t.

Le cas avec variables gelées est à nouveau basé sur celui sans variable gelée.

Théorème 6.3 *Le problème qui consiste à trouver un ensemble complet de termes éliminant x dans t est unitaire dans les algèbres finies.*
• *Si $\mathcal{M} = \emptyset$ alors $mgte(x, t, \emptyset) = \sum_{i \in A} C_i(v) \cdot [S(i)]$, où v est une nouvelle variable ($v \notin \mathcal{V}(t)$) et $S : A \mapsto Common(x, t)$ une application surjective.*
• *Si $\mathcal{M} \neq \emptyset$ alors $mgte(x, t, \mathcal{M}) = \sum_{\delta} \prod \delta(\mathcal{M}) \cdot mgte(x, t\boldsymbol{\delta}, \emptyset)$.*

Exemple 6.5 *Si t est un terme booléen sans variable ($\mathcal{M} = \emptyset$) non équivalent à x, alors on peut choisir $mgte(x, t, \emptyset) = v$ lorsque $Common(x, t) = \{0, 1\}$, où $mgte(x, t, \emptyset) = \mathbf{0}$ (resp. $\mathbf{1}$) lorsque $Common(x, t) = \{0\}$ (resp. $\{1\}$).*

La résolution de contraintes avec variables gelées fournit un éliminateur principal: c'est la solution principale d'un problème de filtrage ($t \leq^?_{AF} u$) où u est le terme principal éliminant x dans t.

Théorème 6.4 *Le problème d'élimination de variable gelée est unitaire dans les algèbres finies. Un éliminateur principal de x dans t est $mgs(t =^? mgte(x, t, \mathcal{M}), \mathcal{M} \cup \{x, v\})$ où $\mathcal{M} \cup \{v\}$ sont les variables dans $mgte(x, t, \mathcal{M})$.*

Preuve : On note c le problème de filtrage à résoudre et σ sa solution principale. Si σ' désigne un éliminateur de x dans t suivant un ensemble de variables gelées $\mathcal{M}$, alors il existe un terme dans $STE(x, t, \mathcal{M})$ et par conséquent,

$$\forall \delta \in ASS_A^{\mathcal{M} \cup \{x,v\}},\ \mathcal{I}(\sigma'_{c\boldsymbol{\delta}}) \subseteq \{\beta \mid \underline{\beta}(t\boldsymbol{\delta}) \in Common(x, t\boldsymbol{\delta})\} = \mathcal{I}(\sigma_{c\boldsymbol{\delta}}).$$

La substitution σ' est donc une instance de σ. La substitution μ telle que

$$\forall x \in \mathcal{V}(c), x\sigma' =_{AF} x\sigma\mu$$

n'instancie pas les variables de $\mathcal{M} \cup \{x, v\}$ puisque $\mathcal{D}om(\sigma) \cap \{\mathcal{M} \cup \{x, v\}\} = \emptyset$ par construction et $\mathcal{D}om(\sigma') \cap \{\mathcal{M} \cup \{x, v\}\} = \emptyset$ par hypothèse. □

Exemple 6.6 *Soit t un terme booléen $\overline{x}\,(\,\overline{y} + y\,\overline{z}\,) + xyz$. Comme $Common(x, t) = \{0, 1\}$, le terme éliminant est une variable v. Il s'agit ensuite de considérer $c = (t =^? v)$ où x et v sont gelées, c'est-à-dire:*
$c00 = \{x \mapsto \mathbf{0}, v \mapsto \mathbf{0}\}(c)$ *avec* $Sol(c00) = \{yz\}$.
$c01 = \{x \mapsto \mathbf{0}, v \mapsto \mathbf{1}\}(c)$ *avec* $Sol(c01) = \{\,\overline{y}\,\overline{z}\,, \overline{y}\,z, y\,\overline{z}\,\}$.
$c10 = \{x \mapsto \mathbf{1}, v \mapsto \mathbf{0}\}(c)$ *avec* $Sol(c10) = \{\,\overline{y}\,\overline{z}\,, \overline{y}\,z, y\,\overline{z}\,\}$.
$c11 = \{x \mapsto \mathbf{1}, v \mapsto \mathbf{1}\}(c)$ *avec* $Sol(c11) = \{yz\}$.
Ainsi $mgs(c00) = mgs(c11) = \{y \mapsto \mathbf{1}, z \mapsto \mathbf{1}\}$ *et* $mgs(c01) = mgs(c10) = \{y \mapsto y_1, z \mapsto \overline{y_1}\,y_2\}$ *grâce à l'application*

$$\sigma_{c01}(\,\overline{y_1}\,\overline{y_2}\,) = \overline{y}\,\overline{z} \quad \sigma_{c01}(\,\overline{y_1}\,y_2) = \overline{y}\,z \quad \sigma_{c01}(y_1\,\overline{y_2}\,) = y\,\overline{z} \quad \sigma_{c01}(y_1y_2) = y\,\overline{z}$$

Finalement,

$$mgs(c, \{x, v\}) = \{y \mapsto\ \overline{x}\ \overline{v}\ +\ \overline{x}\,vy_1 + x\,\overline{v}\,y_1 + xv, z \mapsto\ \overline{x}\ \overline{v}\ +\ \overline{x}\,v\,\overline{y_1}\,y_2 + x\,\overline{v}\ \overline{y_1}\,y_2 + xv\}.$$

6.2.3 Résolution de contraintes avec restriction linéaire

L'algorithme proposé s'inspire fortement de la construction générale qui utilise d'abord un résolveur de contraintes avec variables gelées puis un algorithme d'élimination de variable gelée pour tenir compte de la restriction linéaire. Les particularités de la résolution de contraintes dans le langage primal font qu'une solution principale est obtenue par composition de deux substitutions correspondant à ces deux étapes bien distinctes. Une conjonction de problèmes d'élimination est une conjonction de filtre-équations

$$t[x] \leq^?_{AF} u$$

où u est le terme éliminant x dans t. Il est supposé qu'une nouvelle variable permettant de construire un terme éliminant ne peut figurer dans deux termes éliminants différents. Elle est implicitement gelée puisque elle apparait dans un membre droit d'une filtre-équation.

Proposition 6.12 *Une solution principale de la contrainte c suivant la restriction linéaire $<$ sur $\mathcal{V}(c) = V \oplus \mathcal{M}$ est la composée de deux substitution $\sigma\psi$ telle que*

1. $\sigma \in mgs(c, \mathcal{M})$,
2. $\psi \in mgs(\bigwedge_{\{(x,y)\in\mathcal{M}\times V | x<y\}} x\sigma \leq^? mgte(y, x\sigma, \mathcal{M}\backslash\{y\}), \mathcal{M})$.

La prise en compte de la restriction linéaire nécessite de savoir résoudre une conjonction de problèmes d'élimination d'une variable gelée dans un terme. Savoir résoudre un problème d'élimination ne permet pas en général de résoudre une conjonction de problèmes. En effet ce problème n'est pas stable par instanciation. Il n'est pas possible de d'abord résoudre un problème d'élimination puis de reporter cette solution dans le problème suivant qui risque de l'instancier sans qu'elle reste une solution. Une démarche incrémentale ne peut donc pas être entreprise en général. Le fait de pouvoir transposer un problème d'élimination en une équation est donc de la plus grande importance. Cela permet de voir une conjonction de problèmes d'élimination comme une conjonction d'équations ou plus exactement de filtre-équations. Il n'est pas nécessaire de résoudre une équation une par une si l'on dispose d'une méthode plus efficace de résolution de systèmes d'équations.

En particulier, nous avons à considérer des sous-systèmes de filtre-équations de la forme

$$\bigwedge_{\{y\in\mathcal{M} \mid x<y\}} x\sigma \leq^?_{AF} mgte(y, x\sigma, \mathcal{M}\backslash\{y\}).$$

Soit $\mathcal{M}_x$ l'ensemble des variables gelées supérieures à x pour l'ordre linéaire $<$. Pour chacune de ces variables gelées, un terme éliminant est créé, alors qu'un seul suffirait pour représenter l'ensemble de ces termes éliminants. Ce terme est construit sur l'ensemble des valeurs

$$\bigcap_{y\in\mathcal{M}_x} Common(y, x\sigma\boldsymbol{\delta}),$$

et ne nécessite l'introduction que d'une seule variable afin de schématiser cet ensemble de valeurs. L'étude de la résolution d'un système d'équations permet donc de mettre en évidence une optimisation de l'élimination de plusieurs variables gelées dans un seul terme. S'intéresser directement à l'élimination de plusieurs variables dans un terme aurait alourdi inutilement la notation. Grâce à ce regroupement des problèmes d'élimination,

il suffit d'introduire une nouvelle variable comme doublure de chaque variable non gelée sauf pour la variable maximale par rapport à $<$ qui, elle, ne contient pas de variable gelée à éliminer. L'ensemble

$$\bigcap_{y \in \mathcal{M}_x} Common(y, x\sigma\boldsymbol{\delta})$$

peut aussi être un singleton, auquel cas la valeur correspondante est directement utilisée comme terme éliminant.

Le système de problèmes d'élimination à résoudre pour le second point de la proposition 6.12 est donc en définitive une conjonction de sous-systèmes qui se réduisent simplement en une seule filtre-équation, une pour chaque variable $x \in V$.

L'idée est ensuite de résoudre ce système de filtre-équations de façon incrémentale. On commence par choisir l'élément minimal x de manière indéterministe et en ne fixant pas *a priori* l'ordre $<$ sur les autres variables. L'ensemble $\mathcal{M}_x$ est quand même défini et permet d'éliminer toutes les variables supérieures à x. La solution est ensuite reportée dans la contrainte équationnelle sous forme résolue. Ce qui a été fait pour rendre x minimal ne sera plus à refaire, la variable étant complètement déconnectée des autres variables. En effet, le terme solution associé à cette variable ne changera plus au cours du processus d'élimination, il est gelé. Ce processus est réitéré sur les variables restantes et un nouvel élément minimal doit être fixé. L'élimination se poursuit jusqu'à ce qu'il n'y ait plus de variable non ordonnée. On s'arrête finalement avec une substitution respectant la restriction linéaire qui s'est construite au fur et à mesure.

Corollaire 6.2 *La résolution de contraintes avec restriction linéaire est unitaire dans un langage primal.*

Nous savions que la résolution de contraintes avec variables gelées est unitaire mais cela ne suffisait pas pour combiner un résolveur de contraintes dans un langage primal avec un algorithme d'unification dans un langage équationnel. Le résolveur de contraintes dont nous disposons maintenant est prêt à être combiné avec un autre algorithme d'unification pourvu que ce dernier soit lui aussi suffisamment puissant.

6.2.4 Résolution de contraintes pseudo-booléennes et pseudo-finis avec restriction

Le problème de la résolution avec restriction (linéaire) de contraintes pseudo-booléennes ou plus généralement pseudo-finies n'a pas été traité jusqu'à présent.

Nous montrons qu'il est possible de résoudre des contraintes ayant des variables de sorte Int lorsque celles-ci sont toujours membres d'une équation. En particulier, il est possible d'avoir une équation $x : Int =^? t$. La question est alors de savoir comment éliminer une variable (nécessairement de sorte $Bool$) dans t. C'est une question essentielle si l'on veut combiner $\mathcal{PF}$ ou simplement $\mathcal{PB}$ à une autre structure ordo-sortée (par exemple une théorie équationnelle), problème évoqué au chapitre suivant.

Définition 6.11 *Une contrainte pseudo-booléenne (resp. pseudo-finie)* étendue *est une contrainte dont les membres des équations sont des termes pseudo-booléens (resp. pseudo-finis) ou des variables de sorte Int.*

Un algorithme de résolution de contraintes pseudo-finies étendues avec restriction est obtenu en utilisant d'abord un algorithme de résolution de contraintes avec variables gelées puis un algorithme d'élimination de variable gelée dans un terme pseudo-fini.

Précisons comment s'enchaînent ces algorithmes.

- L'algorithme de résolution est utilisé pour les contraintes pseudo-finies sans considérer les équations résolues $x : Int =^? t$.
- La restriction doit être ensuite prise en compte. Comme un problème de filtrage $t \leq^? x : Int$ n'a pas de solution dans $\mathcal{PF}$, il suffit de savoir éliminer une variable (forcément de sorte $Prim$) de t.

On note au passage que l'identification de variables de sorte Int dans une contrainte pseudo-finie étendue produit une contrainte pseudo-finie étendue.

Lemme 6.2 *L'élimination de variable gelée dans un terme pseudo-booléen (resp. pseudo-fini) est équivalente à un problème de filtrage dans les pseudo-booléens (resp. pseudo-finis).*

Preuve : Considérons le cas simple où il n'y a pas d'autre variable gelée dans t que x, celle à éliminer. Comme nous l'avons vu précédemment, le problème d'élimination de x dans t est équivalent à un problème de filtrage $t \leq^? u$. La seule différence avec les algèbres primales se situe dans le nombre de variables à introduire dans le terme u. Celui-ci est construit d'après l'ensemble des valeurs prises par t indépendamment de celles de x qui est noté $Common(x, t)$. L'ensemble fini des valeurs prises par t, noté $Val(t)$, est inclus strictement dans $\mathcal{Z}$ mais n'est pas inclus dans A. L'ensemble $Common(x, t)$ est défini par $Common(x, t) = \cap_{i \in A} Val(t\{x \mapsto [i]\})$. Le terme u est construit grâce à une surjection S de $A^m \to Common(x, t)$. Le nombre m de variables vérifie nécessairement $|A|^m > |Common(x, t)|$. On le choisira minimal.

$$u = \sum_{(i_1,\ldots,i_m) \in A^m} C_{i_1}(v_1) \cdots C_{i_m}(v_m) \cdot [S(\vec{i})].$$

La construction du terme éliminant u se généralise de la même facon que dans le cas des algèbres primales pour autoriser d'autres variables gelées dans t. □

On peut noter qu'il a été possible de ramener le problème de l'élimination de variable gelée à un problème de filtrage parce que toute fonction pseudo-finie peut s'exprimer par un terme.

Corollaire 6.3 *La résolution symbolique avec restriction des contraintes pseudo-booléennes étendues (resp. pseudo-finies étendues) est unitaire.*

6.2.5 Combinaison avec une théorie équationnelle

L'existence d'une présentation équationnelle AF ω-complète dont l'algèbre initiale est $\overline{\mathcal{A}}$ offre une justification au mélange avec une autre présentation équationnelle définissant par exemple des propriétés simples sur d'autres symboles comme par exemple la commutativité, l'associativité-commutativité ou simplement des symboles sans propriétés dits libres. Les outils nécessaires pour résoudre l'unification dans le mélange le sont aussi

pour combiner plus généralement un résolveur de contraintes dans un langage primal avec un algorithme d'unification dans un langage équationnel. On utilise alors pleinement le résolveur de contraintes dans le langage primal sans se limiter aux seules équations.

W. Büttner [BES+90] avait montré que l'unification dans les algèbre finies (enrichies) est unitaire. Dans le même contexte, nous pouvons maintenant conclure que l'unification dans les algèbres finies en présence de symboles libres est finitaire. Ce résultat s'étend à la résolution de contraintes dans le modèle de combinaison choisi et à d'autres théories comme par exemple la commutativité ou l'associativité-commutativité.

Proposition 6.13 *La résolution de contraintes dans les algèbres finies en présence de symboles libres, commutatifs, associatifs-commutatifs est finitaire.*

Preuve : L'unification avec restriction linéaire est équivalente à l'unification élémentaire si les théories sont régulières et non effondrantes. Nous savons en particulier que l'unification dans la théorie vide est unitaire et que l'unification modulo la commutativité ou l'associativité-commutativité est finitaire. □

Un certain nombre de restrictions linéaires n'ont pas à être considérées car le filtrage sur une constante et l'élimination de constante n'ont pas de solution dans les théories régulières et non effondrantes. La démarche suivie dans ce cas est de construire une restriction linéaire progressivement et seulement lorsque elle s'avère nécessaire, c'est-à-dire en cas de conflit ou de cycle entre théories.

Exemple 6.7 *Considérons la combinaison d'une algèbre primale 3 avec les symboles de fonctions* $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, C_0, C_1, C_2, +, \cdot\}$ *et la théorie définie par les deux symboles de fonctions* $\{a, f\}$ *et l'axiome de commutativité* $C = \{f(x, y) = f(y, x)\}$*, La conjonction d'équations pures*

$$(y =^? v \cdot w \wedge x =^? z \cdot (z + \mathbf{1})) \wedge (f(v, x) =^? f(a, f(x, y)))$$

est résolue comme suit:

On résout d'abord la seconde équation dans la théorie C, pour obtenir

$$(y =^? v \cdot w \wedge x =^? z \cdot (z + \mathbf{1})) \wedge (x =^? a \wedge v =^? f(x, y)).$$

La variable x est maintenant instanciée dans la théorie Commutative. Donc x doit être gelée dans 3 et l'équation correspondante résolue. Nous obtenons

$$(y =^? v \cdot w \wedge z =^? x) \wedge (x =^? a \wedge v =^? f(x, y)).$$

Il existe encore un cycle composé

$$v =^? f(x, y) \wedge y =^? v \cdot w$$

pouvant être cassé en posant la restriction linéaire $y < v$ *avec v instanciée dans C et gelée dans 3. La variable v est éliminée dans* $v \cdot w$*: on obtient* $w =^? C_0(v) \cdot z'$ *et* $y =^? \mathbf{0}$ *où* z' *est une nouvelle variable apparaissant au cours du processus d'élimination.*

Finalement, la contrainte équationnelle est en forme résolue

$$(z =^? x \wedge w =^? C_0(v) \cdot z' \wedge y =^? \mathbf{0}) \wedge (x =^? a \wedge v =^? f(x, y)).$$

La solution correspondante est $\{x \mapsto a, v \mapsto f(a, \mathbf{0}), y \mapsto \mathbf{0}, z \mapsto a, w \mapsto C_0(f(a, \mathbf{0})) \cdot z'\}$.

Il est remarquable que le mélange des deux théories unitaires que sont la théorie vide et AF est une théorie seulement finitaire comme le montre l'exemple suivant.

Exemple 6.8 *Considérons une équation booléenne avec un symbole libre f:*

$$x =^? f(x) + \overline{f(y)} + f(\mathbf{0}).$$

Un ensemble complet minimal de solutions de cette équation est formé des **deux** *substitutions $\{x \mapsto \mathbf{1}, y \mapsto \mathbf{1}\}$ et $\{x \mapsto \mathbf{1}, y \mapsto \mathbf{0}\}$.*

L'unification avec restriction linéaire est unitaire dans AF alors que l'unification avec symboles libres est seulement finitaire. Cela n'empêche pas ces deux notions d'être équivalentes au sens où un algorithme d'unification avec restriction linéaire fournit un algorithme d'unification avec symboles libres et réciproquement [BS92]. Mais la construction d'un algorithme d'unification avec symboles libres se fait au prix de nombreuses étapes indéterministes puisque il faut *a priori* considérer toutes les identifications et toutes les restrictions linéaires.

L'algorithme de combinaison s'applique aussi à des diséquations en supposant que le symbole de diségalité est un prédicat du langage primal. Prenons l'exemple d'une diséquation primale élémentaire contenant exclusivement des symboles de fonctions étrangers.

Exemple 6.9 *Considérons la diséquation $x \neq^? f(y)$. Cette diséquation est purifiée pour obtenir:*

$$x \neq^? z \wedge z =^? f(y).$$

La résolution de $x \neq^? z$ dans le langage de Boole renvoie la solution $x =^? \overline{z}$. La conjonction $x =^? \overline{z} \wedge z =^? f(y)$ est en forme séquentiellement résolue dont la solution est la substitution

$$\{x \mapsto \overline{f(y)}\}.$$

Cette solution est correcte pour toute interprétation de f dans l'algèbre de Boole.

L'exemple suivant nécessite l'utilisation de l'algorithme d'élimination.

Exemple 6.10 *Considérons la diséquation*

$$f(y) \cdot y \neq^? \overline{f(y)} \cdot y + f(y) \cdot \overline{y}$$

dans l'algèbre de Boole en présence d'un symbole libre f. Après purification, on obtient les deux contraintes pures

$$(x \cdot y \neq^? \overline{x} \cdot y + x \cdot \overline{y}) \wedge (x =^? f(y)).$$

La résolution de la diséquation dans l'algèbre de Boole introduit un cycle composé

$$(y =^? \overline{x} + x \cdot y_1) \wedge (x =^? f(y)).$$

Pour le casser, on pose la restriction linéaire $y < x$ où x est instanciée dans la théorie vide et y dans l'algèbre de Boole. Le terme éliminant x dans le terme associée à y vaut $\mathbf{1}$ *et donc $\{y =^? \mathbf{1}\}$.*

Une autre possibilité serait d'instancier y dans la théorie vide et donc de la geler dans l'algèbre de Boole. Mais l'équation $y =^? \overline{x} + x \cdot y_1$ n'a pas de solution lorsque x et y sont gelées. Le dernier recours est alors d'identifier x et y mais dans ce cas, c'est l'autre équation $x =^? f(x)$ qui n'a pas de solution dans la théorie vide.

Par conséquent, l'unique solution est $\{y \mapsto \mathbf{1}\}$. Vérifions que cette substitution est solution:

$$\begin{aligned} f(\mathbf{1}) \cdot \mathbf{1} &\neq \overline{f(\mathbf{1})} \cdot \mathbf{1} + f(\mathbf{1}) \cdot \overline{\mathbf{1}} \\ &\neq \overline{f(\mathbf{1})} + f(\mathbf{1}) \cdot \mathbf{0} \\ &\neq \overline{f(\mathbf{1})} \end{aligned}$$

Cette dernière diséquation est valide dans toutes les interprétations et en particulier celles dont le domaine reste celui de l'algèbre de Boole.

6.2.6 Interprétation de la combinaison avec symboles libres

Le mélange d'une algèbre finie avec des symboles ayant certaines propriétés abstraites, comme la commutativité, permet de résoudre des problèmes génériques, en faisant abstraction de toutes les autres propriétés. Il serait tentant d'interpréter les symboles libres par des fonctions sur le seul domaine A fini. Cela peut se révéler dangereux car il n'existe qu'un nombre fini d'interprétations possibles.

Exemple 6.11 *L'égalité $f(f(f(x))) =^? f(x)$ est valide pour toute interprétation de f sur le domaine de l'algèbre de Boole, que ce soit les fonctions constantes 0 ou 1, l'identité ou la négation $\overline{x}$. L'algorithme de combinaison fait appel à l'unification dans la théorie vide qui ne trouve pas de solution.*

L'algorithme de combinaison n'est donc pas complet pour le langage de contraintes augmenté de symboles libres mais dont le seul domaine d'interprétation est A, celui de l'algèbre finie. Pour préserver la complétude mais aussi le domaine d'interprétation, il faudrait considérer la classe des langages de contraintes dont les domaines d'interprétation sont toutes les puissances A^m $(m \geq 1)$ de A.

Définition 6.12 *Soit $\mathcal{A}$ une $\Sigma = (\{s_*\}, \mathcal{F}, \mathcal{P})$-structure et $\mathcal{F}_\emptyset$ un ensemble de symboles de fonctions disjoint de $\mathcal{F}$. On note $\mathcal{A}^\infty$ la classe de toutes les $\Sigma' = (\mathcal{S}, \mathcal{F} \cup \mathcal{F}_\emptyset, \mathcal{P})$-structures $\mathcal{A}^m$ pour $m \geq 1$ vérifiant:*

- $(s_*)_{\mathcal{A}^m} = A^m$.
- $\forall f \in \mathcal{F}, \forall (a_1, \ldots, a_q) \in (A^m)^q$,

$$f_{\mathcal{A}^m}(a_1, \ldots, a_q) = (f_{\mathcal{A}}((a_1)_{|1}, \ldots, (a_q)_{|1}), \ldots, f_{\mathcal{A}}((a_1)_{|m}, \ldots, (a_q)_{|m}))$$

- $\forall p \in \mathcal{P}, \forall (a_1, \ldots, a_q) \in (A^m)^q$,

$$p_{\mathcal{A}^m}(a_1, \ldots, a_q) = p_{\mathcal{A}}((a_1)_{|1}, \ldots, (a_q)_{|1}) \wedge \ldots \wedge p_{\mathcal{A}}((a_1)_{|m}, \ldots, (a_q)_{|m}).$$

L'objectif du paragraphe est de mettre en évidence le lien existant entre la validité dans le langage de contraintes combiné $CL_{AF\cup\emptyset}[\overline{\Sigma}', \mathcal{X}]$ où les symboles de $\overline{\Sigma}' \setminus \overline{\Sigma}$ sont libres et le langage de contraintes $CL_{\overline{\mathcal{A}}^\infty}[\overline{\Sigma}', \mathcal{X}]$ dont les interprétations sont basées sur $\overline{\mathcal{A}}$. Nous allons principalement montrer que l'égalité dans $\overline{\mathcal{A}}^\infty$ est celle dans $AF \cup \emptyset$.

Le mélange avec symboles libres est très particulier puisque une égalité entre termes hétérogènes se projette directement sur une égalité dans AF:

$$\forall s, t \in \mathcal{T}(\overline{\mathcal{F}} \cup \mathcal{F}_\emptyset, \mathcal{X}),\ s =_{AF\cup\emptyset} t \Leftrightarrow s^{\pi_1} =_{AF} t^{\pi_1}.$$

Cette équivalence permet de montrer le Lemme suivant par récurrence sur la hauteur de théories qui est, rappelons-le, grossièrement le nombre maximum de couches dans un terme.

Lemme 6.3 *Soit h un entier positif. Il existe une interprétation $\overline{\mathcal{A}}^m$ telle que un ensemble fini de termes non $AF \cup \emptyset$-égaux deux à deux et de hauteur de théories égale à h s'interprètent différement dans $\overline{\mathcal{A}}^m$.*

Preuve : Si $h = 0$ alors les termes sont purs. S'ils le sont dans AF on prend $m = 1$. L'entier m est fonction de la taille des termes lorsqu'ils sont purs dans la théorie vide.

Soit un ensemble de termes de hauteur de théories h. On suppose le Lemme vrai pour les sous termes étrangers de hauteur de théories $h-1$. Les symboles de tête libres (i.e. dans $\mathcal{F}_\emptyset$) de termes de l'ensemble peuvent être considérées distinctes deux à deux, sinon on peut décomposer les termes et le problème. Trouver une interprétation pour distinguer les termes à symboles de tête libres distinctes ne pose pas de difficulté. Nous allons maintenant nous intéresser à un couple (s, t) de termes dont l'un au moins a son symbole de tête dans la théorie AF. Par hypothèse nous avons $s^{\pi_1} \neq_{AF} t^{\pi_1}$ et il existe m et $\overline{\mathcal{A}}^m$ interprétant différement les sous termes étrangers non $AF\cup\emptyset$-équivalents. Il est ensuite facile de construire une interprétation $\overline{\mathcal{A}}^{m+1}$ interprétant différement s et t. □

Proposition 6.14 *Soient $\mathcal{F}_\emptyset$ un ensemble de symboles de fonctions disjoint de $\overline{\mathcal{F}}$ et c une $CL_{AF\cup\emptyset}[\overline{\Sigma}', \mathcal{X}]$-contrainte atomique. Alors*

$$CL_{\overline{\mathcal{A}}^\infty}[\overline{\Sigma}', \mathcal{X}] \models c \Leftrightarrow CL_{AF\cup\emptyset}[\overline{\Sigma}', \mathcal{X}] \models c.$$

Preuve : Remarquons dans un premier temps que $t =_{AF\cup\emptyset} t \downarrow_R$ et $t^{\pi_1} =_{AF} (t \downarrow_R)^{\pi_1}$. Par conséquent une contrainte atomique $p((t_1 \downarrow_R)^{\pi_1}, \ldots, (t_n \downarrow_R)^{\pi_1})$ est valide dans $\overline{\mathcal{A}}$ si et seulement si $p(t_1^{\pi_1}, \ldots, t_n^{\pi_1})$ est valide dans $\overline{\mathcal{A}}$.

On utilise ensuite la définition de l'interprétation standard d'un langage combiné pour montrer que

- Si $p(t_1, \ldots, t_n)$ est valide dans $AF \cup \emptyset$ alors $p(t_1^{\pi_1}, \ldots, t_n^{\pi_1})$ est valide dans $\overline{\mathcal{A}}$ et $\overline{\mathcal{A}}^\infty$ et donc par instanciation $p(t_1, \ldots, t_n)$ est valide dans $\overline{\mathcal{A}}^\infty$.
- Si $p(t_1, \ldots, t_n)$ n'est pas valide dans $AF \cup \emptyset$ alors $p(t_1^{\pi_1}, \ldots, t_n^{\pi_1})$ n'est pas valide dans $\overline{\mathcal{A}}$. On peut donc construire une interprétation de $\overline{\mathcal{A}}^\infty$ à partir de l'interprétation fournie par l'application du Lemme 6.3 à l'ensemble des sous termes étrangers de $\{t_1, \ldots, t_n\}$. Cette interprétation ne satisfait donc pas la contrainte atomique $p(t_1, \ldots, t_n)$.

□

Cette proposition permet de conclure que l'algorithme de résolution de contraintes dans la combinaison d'une algèbres finie et de symboles libres est aussi un algorithme de résolution symbolique pour une classe de langages de contraintes dont les domaines sont uniquement basés sur celui de l'algèbre finie.

Corollaire 6.4 *La résolution de contraintes symboliques est finitaire dans* $CL_{\overline{\mathcal{A}}^{\infty}}[\overline{\Sigma}', \mathcal{X}]$.

L'interprétation des symboles libres sur le seul domaine A est donc trop approximatif. C'est l'ensemble des puissances de A qu'il s'agit de considérer.

6.2.7 Combinaison de deux algèbres finies non disjointes

La combinaison de deux langages de contraintes primaux

$$CL_{AF_1}[(\{s_*\}, \overline{\mathcal{F}}_1, \mathcal{P}_1), \mathcal{X}] \text{ et } CL_{AF_2}[(\{s_*\}, \overline{\mathcal{F}}_2, \mathcal{P}_2), \mathcal{X}]$$

est envisageable dans le même contexte. Les algèbres finies $\overline{\mathcal{A}}_1$ et $\overline{\mathcal{A}}_2$ sont les algèbres initiales des présentations ω-complètes $(\overline{\mathcal{F}}_1, AF_1)$ et $(\overline{\mathcal{F}}_2, AF_2)$. Nous nous intéresserons principalement aux algèbres finies $\overline{\mathcal{A}}_1$ et $\overline{\mathcal{A}}_2$ dont les domaines A_1 et A_2 ont une intersection non vide. Prenons par exemple:

- La $(\{Bool\}, \overline{\mathcal{F}}_1)$-algèbre $\overline{\mathcal{A}}_1$ avec $\overline{\mathcal{F}}_1 = \{\mathbf{0}, \mathbf{1}, \vee, \wedge, ^-\}$ interprété par les opérateurs sur le domaine $\{0, 1\}$.
- La $(\{Prim\}, \overline{\mathcal{F}}_2)$-algèbre $\overline{\mathcal{A}}_2$ avec $\overline{\mathcal{F}}_2 = \{\mathbf{0}, \mathbf{1}, \mathbf{2}, +, \cdot, C_0, C_1, C_2\}$ interprété comme dans la définition 5.4 sur le domaine $\{0, 1, 2\}$.

Les domaines de ces deux algèbres finies ont une intersection non vide, en l'occurrence $\{0, 1\}$. Les présentations équationnelles partagent les constantes correspondantes $\{\mathbf{0}, \mathbf{1}\}$.

Ce partage des seules constantes permet d'établir un lien entre les solutions du langage combiné dont les sortes de chaque langage composant sont disjointes

$$CL_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}[(\mathcal{S}_1 \cup \mathcal{S}_2, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}]$$

(avec par exemple $\mathcal{S}_1 = \{Bool\}$ et $\mathcal{S}_2 = \{Prim\}$) et les solutions symboliques du langage combiné

$$CL_{AF_1 \cup AF_2}[(\{s_*\}, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}].$$

En effet, un résolveur de contraintes dans

$$CL_{AF_1 \cup AF_2}[(\{s_*\}, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}]$$

fournit un résolveur de contraintes **symbolique** dans

$$CL_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}[(\mathcal{S}_1 \cup \mathcal{S}_2, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}].$$

La proposition suivante précise la relation entre solutions et solutions symboliques.

Proposition 6.15 *Soit c une $CL_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}[(\mathcal{S}_1 \cup \mathcal{S}_2, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}]$-contrainte. Pour toute solution $\alpha \in Sol_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}(c)$ il existe une substitution $\sigma \in CSS_{AF_1 \cup AF_2}(c)$ telle que*

$$\forall x \in \mathcal{V}(c),\ \alpha(x) = \underline{\alpha}(x\sigma).$$

Preuve : Dans ce qui suit, une substitution σ est dite *par valeurs* si $\mathcal{R}an(\sigma)$ est un ensemble de constantes. Toute substitution par valeurs correspond à la restriction d'une assignation dans $\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2$ à un ensemble fini de variables, et réciproquement.

On peut supposer que c est une conjonction de contraintes atomiques pures $c_1 \wedge c_2$ car l'étape de purification préserve l'ensemble des solutions dans les deux langages de contraintes. Si $\underline{\alpha}(c)$ est vrai dans $\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2$ alors la substitution par valeurs définie comme suit:

$$\forall x \in \mathcal{V}(c),\ x\phi = [\underline{\alpha}(x)]$$

est une solution de c dans $AF_1 \cup AF_2$. Il existe donc une restriction linéaire sur $V_1 \oplus V_2 = \mathcal{V}(c_1 \wedge c_2)$ et une solution combinée $\sigma = \sigma_1 \odot \sigma_2$ telle que $\sigma_1 \leq^{V_1}_{AF_1} \phi$ et $\sigma_2 \leq^{V_2}_{AF_2} \phi$. Soient γ_1, γ_2 les substitutions correspondantes:

$$\forall x \in V_i,\ x\phi =_{AF_i} x\sigma_i\gamma_i.$$

Comme ϕ est une substitution par valeurs, nous pouvons supposer sans perte de généralité que γ_i est encore une substitution par valeurs définie sur $\mathcal{VR}an(\sigma_i) \backslash V_i$ avec $\mathcal{R}an(\gamma_i) \subseteq (\mathcal{F}_i)_0$, sinon on peut toujours clore γ_i pour qu'elle le devienne. En combinant les solutions, nous obtenons

$$\forall x \in \mathcal{V}(c),\ x\phi =_{AF_1 \cup AF_2} x\sigma\gamma_1\gamma_2.$$

Les variables de $\mathcal{VR}an(\sigma_i) \cap \mathcal{D}om(\sigma_j)$, $(i \neq j)$ sont des variables identifiées à celles de $\mathcal{V}(c_1) \cap \mathcal{V}(c_2)$ et qui sont donc instanciées par ϕ avec des constantes de SC. Ainsi $\gamma_1\gamma_2$ correspond à une solution α de $x\sigma$ dans $\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2$. Ensuite ϕ est définie sur $\mathcal{D}om(\gamma_1) \cup \mathcal{D}om(\gamma_2) = \mathcal{V}(c\sigma) \backslash \mathcal{V}(c)$ comme étant égale à $\gamma_1\gamma_2$. La substitution ϕ étendue aux nouvelles variables, $\mathcal{D}om(\phi) = \mathcal{V}(c) \cup \mathcal{V}(c\sigma)$, correspond finalement à une solution $\alpha \in Sol_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}(c)$ qui satisfait

$$\forall x \in \mathcal{V}(c),\ \alpha(x) = \underline{\alpha}(x\sigma).$$

□

Réciproquement, si $\sigma \in CSS_{AF_1 \cup AF_2}(c)$ et α est une assignation des variables telle que $\underline{\alpha}(c\sigma)$ est vrai dans $\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2$, alors $\alpha \in Sol_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}(c)$.

L'algorithme de résolution dans

$$CL_{AF_1 \cup AF_2}[(\{s_*\}, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}]$$

permet d'obtenir une schématisation de l'ensemble des solutions d'un autre langage combiné qui, lui, préserve les domaines de chaque algèbre finie composante. L'algorithme dont nous disposons pour ce dernier a la même fonctionnalité que les algorithmes qui le composent. Il permet de synthétiser l'ensemble des solutions par valeurs. Son intérêt est nul

lorsque la contrainte à résoudre est insatisfaisable. Le problème de la satisfaisabilité est en effet tout aussi évident dans le langage combiné

$$CL_{\overline{\mathcal{A}}_1 \cup \overline{\mathcal{A}}_2}[(\mathcal{S}_1 \cup \mathcal{S}_2, \overline{\mathcal{F}}_1 \cup \overline{\mathcal{F}}_2, \mathcal{P}_1 \cup \mathcal{P}_2), \mathcal{X}]$$

que dans les langages composants puisque les domaines d'interprétations restent finis. Ce problème ne nécessite pas un algorithme de représentation compacte des solutions.

L'algorithme de combinaison tel qu'il est décrit est trop indéterministe pour espérer une bonne représentation des solutions. Toutes les restrictions linéaires, toutes les instanciations par des constantes partagées et finalement toutes les identifications doivent être appliquées. Nous allons déterminer quels sont les cas qui peuvent être évités sans perte de complétude.

L'idée générale est de fixer d'abord les restrictions linéaires qui sont plus ou moins incontournables et de n'envisager l'instanciation par des constantes partagées ou l'identification de variables qu'en cas d'extrême nécessité et ce afin de préserver un ensemble complet de solutions. Cette optimisation est valable pour d'autres mélanges de théories équationnelles mais peut être plus facilement mise en oeuvre pour $AF_1 \cup AF_2$, d'autant que AF_1 et AF_2 sont deux théories ayant le même comportement. Pour toutes les deux, le filtrage est décidable mais pas n'importe comment puisque il correspond d'après la proposition 5.6 à l'inclusion des assignations schématisées par les substitutions.

Choix de la restriction linéaire

Le choix que nous faisons est franchement radical. Il consiste à voir les variables partagées comme des constantes libres dans les deux théories. On empêche ainsi tout risque d'introduction de cycle composé. La notion même de restriction linéaire n'a pas d'interêt dans ce cas dénaturé. C'est pourtant un cas de figure fréquent car dans notre contexte une solution principale avec variables gelées, si elle existe, est encore solution principale.

Proposition 6.16 *Soit $\mathcal{M}$ l'ensemble des variables partagées dans c_1 et c_2, i.e. $\mathcal{M} = \mathcal{V}(c_1) \cap \mathcal{V}(c_2)$. Si les substitutions $\sigma_1 = mgs_{AF_1}(c_1, \mathcal{M})$ et $\sigma_2 = mgs_{AF_2}(c_2, \mathcal{M})$ existent alors la solution composée $\sigma = \sigma_1\sigma_2$ est une solution principale de $c_1 \wedge c_2$ dans $AF_1 \cup AF_2$.*

Preuve : Par construction, la contrainte équationnelle c_i est équivalent à $\widehat{mgs(c_i, \mathcal{M})}$ dans AF_i si $mgs(c_i, \mathcal{M})$ existe. Donc

$$CSS_{AF_1 \cup AF_2}(\widehat{mgs(c_1, \mathcal{M})} \wedge \widehat{mgs(c_2, \mathcal{M})})$$

est un $CSS_{AF_1 \cup AF_2}(c_1 \wedge c_2)$, où $\widehat{mgs(c_1, \mathcal{M})} \wedge \widehat{mgs(c_2, \mathcal{M})}$ est en forme séquentiellement résolue. □

L'unification dans le mélange $AF_1 \cup AF_2$ est donc unitaire pour une large classe de contraintes équationnelles.

Exemple 6.12 *Soient $c_1 = (x \vee v =^? v)$ et $c_2 = (v =^? y \cdot (C_1(v) + C_2(v)))$. Les algorithmes d'unification avec v gelée dans $\overline{\mathcal{A}}_1$ et $\overline{\mathcal{A}}_2$ donnent respectivement*

$$\{x \mapsto v \wedge y_1\},$$

$$\{y \mapsto (C_0(v) \cdot C_1(y_2) + C_1(v)) \cdot \mathbf{1} + C_0(v) \cdot C_2(y_2) + C_2(v)\}$$

dont la composition est un unificateur principal dans le mélange.

Instanciation par des constantes partagées

L'algorithme de combinaison force l'instanciation par des constantes partagées de manière complètement indéterministe. Cette approche va à l'encontre de la schématisation par des solutions symboliques que nous souhaitons obtenir. Ce genre d'instanciation ne sera effectué que si nécessaire. Une instanciation est rendue nécessaire lorsque un problème n'a pas de solution ou lorsque l'élimination de variable gelée ne préserve pas l'ensemble des solutions, ce qui peut se produire car cette dernière opération consiste justement à ne sélectionner que certaines solutions particulières.

Le critère suivant est motivé par la facilité de décider du filtrage dans les algèbre finies.

Proposition 6.17 *Soit $<$ une restriction linéaire sur $V_1 \oplus V_2 = \mathcal{V}(c_1 \wedge c_2)$ et ρ une substitution par des constantes partagées. Si*

$$mgs^{<}_{AF_1}(c_1, V_2)\rho_{|V_2} \leq^{V_1}_{AF_1} mgs^{<}_{AF_1}(c_1\rho_{|V_2} \wedge \hat{\rho}_{|V_1}, V_2),$$

$$mgs^{<}_{AF_2}(c_2, V_1)\rho_{|V_1} \leq^{V_2}_{AF_2} mgs^{<}_{AF_2}(c_2\rho_{|V_1} \wedge \hat{\rho}_{|V_2}, V_1),$$

alors la substitution ρ n'a pas à être considérée à l'étape de Combinaison.

Preuve : On note σ' la solution combinée de $\sigma'_1 = mgs^{<}_{AF_1}(c_1\rho_{|V_2} \wedge \hat{\rho}_{|V_1}, V_2)$ et $\sigma'_2 = mgs^{<}_{AF_2}(c_2\rho_{|V_1} \wedge \hat{\rho}_{|V_2}, V_1)$, et σ la solution combinée de $\sigma_1 = mgs^{<}_{AF_1}(c_1, V_2)$ et $\sigma_2 = mgs^{<}_{AF_2}(c_2, V_1)$. Il s'agit de montrer que $\sigma = \sigma_1 \odot \sigma_2 \leq^{V_1 \cup V_2}_{AF_1 \cup AF_2} \sigma'_1 \odot \sigma'_2 = \sigma'$. Par hypothèse nous savons qu'il existe deux substitutions γ_1, γ_2 telles que

$$\sigma'_i =^{V_i}_{AF_i} \sigma_i\rho_{|V_j}\gamma_i$$

pour $i, j = 1, 2$ et $i \neq j$. Les hypothèses faites sur γ_1 et γ_2 et qu'on rappelle sont les mêmes que celles de la proposition 2.20. Les substitutions n'instancient pas les variables de $V_1 \oplus V_2$: $\mathcal{D}om(\gamma_1) \cap (V_1 \oplus V_2) = \emptyset$ et $\mathcal{D}om(\gamma_2) \cap (V_1 \oplus V_2) = \emptyset$. On supposera sans perte de généralité que γ_1 (resp. γ_2) est idempotente et instancie des variables introduites par l'unification dans AF_1 (resp. AF_2), c'est-à-dire $\gamma_1\gamma_1 = \gamma_1$, $\gamma_2\gamma_2 = \gamma_2$ et $\mathcal{D}om(\gamma_1) \cap \mathcal{D}om(\gamma_2) = \emptyset$.

La preuve se fait par induction noétherienne sur l'ordre $<$. Soit x une variable de V_i et $\{y_1, \ldots, y_n\}$ l'ensemble des variables de V_j inférieures à x par rapport à $<$. L'hypothèse d'induction est

$$y\sigma' =_{AF_1 \cup AF_2} y\sigma\gamma_1\gamma_2$$

pour $y < x$. Cette hypothèse est vérifiée pour la variable minimale z de V_m avec $m = 1, 2$ car par définition

$$z\sigma' = z\sigma'_m =_{AF_m} z\sigma_m\gamma_m = z\sigma_m\gamma_1\gamma_2 = z\sigma\gamma_1\gamma_2.$$

Pour la variable x, la définition inductive d'une solution combinée est

$$\begin{array}{rcl} x\sigma' & = & x\sigma'_i[y_k \hookleftarrow y_k\sigma']_{k=1,\ldots,n} \\ & =_{AF_1 \cup AF_2} & x\sigma_i\rho_{|V_j}\gamma_i[y_k \hookleftarrow y_k\sigma']_{k=1,\ldots,n} \\ & =_{AF_1 \cup AF_2} & x\sigma_i\gamma_i[y_k \hookleftarrow y_k\sigma']_{k=1,\ldots,n} \end{array}$$

car $y_k\sigma' =_{AF_1 \cup AF_2} c$ si $y_k\rho_{|V_j} = c \in SC$. Ensuite

$$\begin{array}{rcl} x\sigma' & = & x\sigma_i\gamma_i[y_k \hookleftarrow y_k\sigma']_{k=1,\ldots,n} \\ & =_{AF_1 \cup AF_2} & x\sigma_i\gamma_i[y_k \hookleftarrow y_k\sigma\gamma_1\gamma_2]_{k=1,\ldots,n} \\ & = & (x\sigma_i[y_k \hookleftarrow y_k\sigma]_{k=1,\ldots,n})\gamma_1\gamma_2 \\ & = & x\sigma\gamma_1\gamma_2. \end{array}$$

Nous avons donc montré que $y\sigma' =_{AF_1 \cup AF_2} y\sigma\gamma_1\gamma_2$ pour $y \leq x$. □

Nous nous intéressons maintenant aux causes empêchant le critère de la proposition 6.17 d'être toujours applicable. L'unification avec variables gelées dans AF_1 et AF_2 vérifie en effet

$$mgs_{AF_1}(c_1\rho_{|V_2} \wedge \hat{\rho}_{|V_1}, V_2) =_{AF_1}^{V_1} mgs_{AF_1}(c_1, V_2)\rho_{|V_2}$$

et

$$mgs_{AF_2}(c_2\rho_{|V_1} \wedge \hat{\rho}_{|V_2}, V_1) =_{AF_2}^{V_2} mgs_{AF_2}(c_2, V_1)\rho_{|V_1}.$$

si $mgs_{AF_1}(c_1, V_2)$ et $mgs_{AF_2}(c_2, V_1)$ existent. Mais cette propriété n'est malheureusement pas préservée en passant à la résolution de contraintes avec restriction linéaire. La cause en est justement la phase d'élimination qui peut se faire avec perte de solutions.

Exemple 6.13 *Soient les contraintes* $c_1 = (x =^? \overline{y}\, v \vee y)$, $c_2 = (y =^? C_2(x))$ *avec* $V_1 = \{x\}, V_2 = \{y\}$ *et* $x < y$. *Le terme éliminant* y *dans* $\overline{y}\, v \vee y$ *est* $\mathbf{1}$. *La contrainte à résoudre pour tenir compte de la restriction est*

$$(x =^? (\overline{y}\, v \vee y) =^? \mathbf{1})$$

dans $\mathcal{B}$. *Cela n'empêche pas ensuite d'avoir à instancier* x *par* $\mathbf{0}$ *et de résoudre*

$$(x =^? (\overline{y}\, v \vee y) =^? \mathbf{0}).$$

La solution obtenue ne sera évidemment pas une instance de la précédente.

Le critère de la proposition 6.17 peut être testé sans construire explicitement la substitution.

Proposition 6.18 *Soit* $<$ *une restriction linéaire sur* $\mathcal{V}(c) = V \oplus \mathcal{M}$ *et* $\rho \in SUBS_{\mathcal{M}}^{SC}$.

$$mgs_{AF}^{<}(c, \mathcal{M})\rho \leq_{AF}^{\mathcal{V}(c)} mgs_{AF}(c\rho, \mathcal{M})$$

si et seulement si

$$Common(y, t\rho_{|\mathcal{M}\setminus\{x\}}) \subseteq Common(y, t)$$

pour tout problème d'élimination de y *dans* t *à considérer durant le calcul de* $mgs_{AF}^{<}(c, \mathcal{M})$.

Preuve : D'après la proposition 5.6. □

Pour ensuite appliquer le critère de la proposition 6.17, il suffit de remarquer que $mgs_{AF_i}^{<}(c_i\rho_{|V_j}, V_j) \leq_{AF_i}^{V_i} mgs_{AF_i}^{<}(c_i\rho_{V_j} \wedge \hat{\rho}_{V_i}, V_j)$.

Ce critère est même satisfait *a priori* pour tout $\rho \in SUBS_{\mathcal{M}}^{SC}$ lorsque les ensembles $Common(y, t)$ calculés durant la phase d'élimination ne sont pas seulement inclus dans A mais égaux à A. Dans ce cas un $mgs_{AF}^{<}(c, \mathcal{M})$ est un $mgs_{AF}(c, \mathcal{M})$.

Identification des variables

Une proposition analogue à celle pour les constantes partagées est valable pour les identifications.

Proposition 6.19 *Soit $<$ une restriction linéaire sur $V_1 \oplus V_2 = \mathcal{V}(c_1 \wedge c_2)$ et ξ une identification compatible avec $<$. Si*

$$mgs^{<}_{AF_1}(c_1, V_2)\xi_{|V_2} \leq^{V_1}_{AF_1} mgs^{<}_{AF_1}(c_1\xi_{|V_2} \wedge \hat{\xi}_{|V_1}, V_2),$$

$$mgs^{<}_{AF_2}(c_2, V_1)\xi_{|V_1} \leq^{V_2}_{AF_2} mgs^{<}_{AF_2}(c_2\xi_{|V_1} \wedge \hat{\xi}_{|V_2}, V_1),$$

alors l'identification ξ n'a pas à être considérée à l'étape de Combinaison.

Preuve : La preuve est quasiment identique à celle de la proposition 6.17. Elle se fait à nouveau par induction nœthérienne sur $<$. La seule différence se situe dans la construction de la solution combinée σ' de $\sigma'_1 = mgs^{<}_{AF_1}(c_1\xi_{|V_2} \wedge \hat{\xi}_{|V_1})$ et $\sigma'_2 = mgs^{<}_{AF_2}(c_2\xi_{|V_1} \wedge \hat{\xi}_{|V_2})$ où l'on utilise cette fois le fait que $y_{k_1}\sigma' = y_{k_2}\sigma'$ si $y_{k_1}\xi_{|V_j} = y_{k_2}\xi_{|V_j}$ pour $y_{k_1}, y_{k_2} \in V_j$. □

Il faut à nouveau remarquer que la condition d'application de la proposition 6.19 n'est pas toujours satisfaite pour l'unification avec restriction linéaire, alors qu'elle l'est pour l'unification avec variables gelées puisque

$$mgs_{AF_1}(c_1\xi_{|V_2} \wedge \hat{\xi}_{|V_1}, V_2) =^{V_1}_{AF_1} mgs_{AF_1}(c_1, V_2)\xi_{|V_2}$$

et

$$mgs_{AF_2}(c_2\xi_{|V_1} \wedge \hat{\xi}_{|V_2}, V_1) =^{V_2}_{AF_2} mgs_{AF_2}(c_2, V_1)\xi_{|V_1}.$$

si $mgs_{AF_1}(c_1, V_2)$ et $mgs_{AF_2}(c_2, V_1)$ existent.

L'élimination ne préserve pas obligatoirement l'ensemble des solutions. Les unificateurs obtenus par identification ne sont donc pas toujours des instances de ceux obtenus sans identification.

Exemple 6.14 *Soient les contraintes $c_1 = (x =^? t[u, y, z])$, $c_2 = (y =^? C_2(x)) \wedge (z =^? C_2(v))$ avec $V_1 = \{x\}$, $V_2 = \{y, z\}$ et $x < y < z$. Le terme booléen t est*

$$\overline{y}\ \overline{z}\, u \vee\ \overline{y}\, z \vee y\,\overline{z}\ \vee yz\,\overline{u}\,.$$

Le terme éliminant $\{y, z\}$ dans t est $\mathbf{1}$. L'unificateur principal suivant la restriction $<$ instanciera donc x par $\mathbf{1}$. Considérons maintenant l'identification $\{z \mapsto y\}$. Le terme éliminant y dans $t\{z \mapsto y\}$ est une variable. L'unificateur principal de la contrainte identifiée ne sera donc pas une instance du précédent.

La proposition 5.6 permet de raisonner directement sur les valeurs prises par les termes sans avoir à construire de substitutions.

Proposition 6.20 *Soit $<$ une restriction linéaire sur $\mathcal{V}(c) = V \oplus \mathcal{M}$ et $\xi \in ID_{\mathcal{M}}$ telle que $\forall x, y\ \mathcal{M}, x\xi < y\xi$ si $x < y$.*

$$mgs^{<}_{AF}(c, \mathcal{M})\xi \leq^{\mathcal{V}(c)}_{AF} mgs^{<}_{AF}(c\xi, \mathcal{M})$$

si et seulement si

$$Common(y\xi, t\xi) \subseteq Common(y, t)$$

pour tout problème d'élimination de y dans t à considérer durant le calcul de $mgs^{<}_{AF}(c, \mathcal{M})$.

On peut en conclure que le critère de la proposition 6.19 est satisfait *a priori* lorsque les termes éliminants à construire durant la phase d'élimination le sont grâce à des surjections sur A, c'est-à-dire lorsque le $mgs^{<}_{AF}(c, \mathcal{M})$ est encore un $mgs_{AF}(c, \mathcal{M})$.

Il est remarquable cependant que l'unification avec et sans restriction linéaire bien qu'elles soient toutes les deux unitaires sont donc fondamentalement différentes même pour des algèbres finies. Les deux phases que sont résolution puis élimination n'ont pas les mêmes bonnes propriétés de conservation des solutions et il est bon qu'elles soient distinguées.

Dans la première, l'identification (resp. l'instanciation par constantes partagées) n'est nécessaire que lorsque l'une des deux contraintes n'a pas de solution.

Dans la seconde, l'identification (resp. l'instanciation par constantes partagées) permet éventuellement de trouver d'autres solutions.

Application

L'intégration d'algorithmes d'unification dans des structures algébriques spécifiques a permis d'augmenter l'expressivité de Prolog. L'algorithme d'unification booléenne est par exemple utilisé dans CHIP [BS87]. L'algorithme d'unification dans les algèbres finies sert aussi de support à un Prolog étendu [BES+90, Tid88] dont les applications sont notamment la vérification symbolique de circuits électroniques. Les outils dont nous disposons permettent désormais d'intégrer une ou plusieurs algèbres finies et le domaine de Herbrand dans un seul et même Prolog. Les algorithmes d'unification sous-jacents font alors appel aux techniques de combinaison lorsque les problèmes le nécessitent.

La vérification de circuits électroniques s'effectue par exemple dans [BES+90] à différents niveaux d'abstraction.

- L'algèbre de Boole $\mathcal{B}$. Une porte $nor(A, B, X)$ où A, B sont les entrées et X la sortie peut être spécifiée par l'équation

$$\{X =^{?}_{\mathcal{B}} \overline{A \vee B}\}.$$

- L'algèbre de Heyting $\mathcal{H}$ à 7 éléments $\{Z, \tilde{0}, \tilde{1}, \tilde{U}, 0, 1, U\}$ est aussi un treillis dont les opérateurs

$$\{\mathbf{Z}, \tilde{\mathbf{0}}, \tilde{\mathbf{1}}, \tilde{\mathbf{U}}, \mathbf{0}, \mathbf{1}, \mathbf{U}, +, \cdot, C_Z, \ldots, C_U\})$$

  sont définis de façon usuelle. Cette algèbre est utilisée pour modéliser des circuits électroniques dans [BES+90, Tid88]. Une porte $nand(A, B, X)$ est spécifiée dans la technologie nMOS par le système d'équations

$$\left\{ \begin{array}{lll} X & =^{?}_{\mathcal{H}} & \tilde{\mathbf{1}} + O \qquad \S1 \\ T & =^{?}_{\mathcal{H}} & C_1(A) \cdot X + C_0(A) \cdot P + (C_U(A) + C_Z(A) + C_{\tilde{U}}(A)) \cdot Q \qquad \S2 \\ \mathbf{0} & =^{?}_{\mathcal{H}} & C_1(B) \cdot T + C_0(B) \cdot R + (C_U(B) + C_Z(B) + C_{\tilde{U}}(B)) \cdot S \qquad \S3 \end{array} \right\}$$

Ces deux niveaux de spécification pourraient être mélangés dans un Prolog intégrant l'unification dans la combinaison des deux algèbres finies. Une intersection des domaines non vide est essentielle pour introduire un cadre de combinaison algébrique dans lequel s'utilise l'algorithme de combinaison avec constantes partagées. Cette forme de combinaison n'engendre pas un nouveau domaine de calcul. Mais l'hypothèse de signatures

non disjointes est requise dans ce cas alors que jusqu'à présent mieux valait supposer le contraire. Les constantes partagées apparaissent donc comme un bon compromis pour adopter des techniques de combinaison sans pour autant se placer sur un nouveau domaine de calcul.

7

Résolution de contraintes incrémentale avec des structures prédéfinies

Nous avons vu aux chapitres précédents comment les principes de combinaison permettent de construire systématiquement de nouveaux algorithmes. Un autre mécanisme de construction systématique de procédures de résolution de contraintes est fourni par la notion de surréduction. A l'origine, ce mécanisme [Hul80] a été utilisé pour résoudre des équations dans des théories présentées par des systèmes de réécriture convergents. La surréduction est une généralisation de la réécriture qui consiste à remplacer le filtrage par l'unification. Les variables d'un terme surréduit sont instanciées grâce à l'unification et cette instanciation fournit une solution à une équation $s=^?t$ lorsque s et t se surréduisent en deux termes unifiables.

La surréduction a ensuite été étendue à des systèmes de réécriture modulo une théorie équationnelle E [JKK83]. Ce processus de surréduction présente l'inconvénient de terminer rarement et d'être relativement inefficace lorsqu'un problème de E-unification a un grand nombre de solutions (voire une infinité de solutions) auquel cas il produit alors de multiples dérivations de surréduction modulo E. L'idée de la surréduction contrainte est de ne pas résoudre un problème d'unification mais plutôt de le garder autant que possible sous la forme d'une contrainte à satisfaire. Nous allons nous placer dans le cadre de la surréduction contrainte en nous limitant à des contraintes dans une structure prédéfinie pour laquelle on suppose disposer d'un résolveur de contraintes. On considèrera comme exemples privilégiés ceux traités dans les chapitres précédents.

Afin d'illustrer les notions de réécriture et surréduction contraintes, considérons un exemple élémentaire adapté de [AB94], qui consiste à axiomatiser le plus grand commun diviseur grâce à la structure prédéfinie $\mathcal{N}$ des entiers de sorte Nat munie de l'addition $+$, de son élément neutre 0 et de la relation d'ordre $>$. On enrichit cette signature de base en ajoutant un nouvel opérateur $g : Nat, Nat \rightarrow Nat$:

$$\begin{array}{rcl} g(x:Nat, 0) & \rightarrow & x:Nat \parallel \top \\ g(x:Nat + y:Nat, y:Nat) & \rightarrow & g(x:Nat, y:Nat) \parallel y:Nat > 0 \\ g(x:Nat, y:Nat) & = & g(y:Nat, x:Nat) \end{array}$$

On note qu'une règle de réécriture est dotée d'un partie contrainte dont l'objet est de

déterminer les instances admissibles pour son application. On s'autorise aussi des axiomes dont l'application n'est pas contrainte.

La réécriture est effectuée dans une structure qui est une extension conservative de $\mathcal{N}$ et qui intègre le nouveau symbole g et sa propriété d'être commutatif. Cette structure est plus précisément une algèbre de termes quotientée par une relation de congruence $\sim_{\mathcal{C}}$ prenant en compte la $\mathcal{N}$-égalité et l'égalité engendrée par $C = \{g(X,Y) = g(Y,X)\}$. La réécriture s'applique ensuite modulo $\sim_{\mathcal{C}}$ et permet par exemple de calculer la valeur de g avec pour arguments $(g(2,1)+1, g(0,1))$:

$$g(g(2,1)+1,g(0,1))(\sim_{\mathcal{C}} g(g(2,1)+1,g(1,0)) \rightarrow_{R,\mathcal{C}} g(g(2,1)+1,1) \rightarrow_{R,\mathcal{C}}$$

$$g(g(2,1),1)(\sim_{\mathcal{C}} g(g(1+1,1),1)) \rightarrow_{R,\mathcal{C}} g(g(1,1),1)(\sim_{\mathcal{C}} g(g(0+1,1),1)) \rightarrow_{R,\mathcal{C}}$$

$$g(g(0,1),1)(\sim_{\mathcal{C}} g(g(1,0),1)) \rightarrow_{R,\mathcal{C}} g(1,1)(\sim_{\mathcal{C}} g(0+1,1)) \rightarrow_{R,\mathcal{C}} g(0,1)(\sim_{\mathcal{C}} g(1,0)) \rightarrow_{R,\mathcal{C}} 1.$$

On peut remarquer que la relation de réécriture $\rightarrow_{R,\mathcal{C}}$ ne termine pas si l'on supprime la partie contrainte de la deuxième règle car

$$g(x:Nat,0)(\sim_{\mathcal{C}} g(x:Nat+0,0)) \rightarrow_{R,\mathcal{C}} g(x:Nat,0).$$

La surréduction permet ensuite d'utiliser les règles de réécriture contraintes pour construire une procédure de résolution de contraintes sur la signature enrichie à partir d'un algorithme de résolution de contraintes modulo $\sim_{\mathcal{C}}$. Mais pour obtenir une procédure complète, il faut pouvoir tester au préalable la convergence de la relation $\rightarrow_{R,\mathcal{C}}$. On met ainsi en évidence la nécessité d'introduire un langage de contraintes hétérogènes formées à la fois de symboles prédéfinis et de symboles ajoutés par l'enrichissement.

La première partie du chapitre est consacrée à préciser les différentes notions de mélanges avec une structure prédéfinie. Ces différentes notions correspondent au souci de pouvoir intégrer les structures étudiées jusqu'à présent. Le langage de contraintes combiné qui en résulte s'inspire fortement de celui vu au chapitre précédent mais on l'adaptera pour traiter des structures à plusieurs sortes et pour permettre la définition d'une relation de réécriture sur des classes d'équivalence de termes hétérogènes.

On construit ensuite un langage de contraintes associé à une spécification basée sur une structure prédéfinie. La sémantique opérationnelle d'une telle spécification est une relation de réécriture contrainte dont la confluence locale peut être vérifiée par la convergence de paires critiques à la condition de s'imposer une certaine restriction syntaxique sur la forme des contraintes.

La surréduction contrainte peut elle aussi être définie dans ce contexte et permet de résoudre des contraintes dans le langage associé à la spécification de façon correcte et complète lorsque la relation de réécriture est supposée convergente. Elle fournit en cela une autre façon de construire incrémentalement un algorithme de résolution de contraintes non nécessairement équationnelles.

7.1 Enrichissement d'une structure prédéfinie

Afin d'être le plus général possible, nous allons désormais considérer des structures prédéfinies à plusieurs sortes et plus précisément à sortes ordonnées.

Les hypothèses syntaxiques et sémantiques que nous serons amenés à faire sont celles communément admises dans le cadre de la logique équationnelle ordo-sortée [SNGM89].

Uns structure prédéfinie n'en est pas pour autant quelconque. Sa signature est supposée régulière, i.e. tout terme a une unique plus petite sorte. La structure doit, comme toujours, être engendrée par les termes. Par exemple, la structure des entiers est engendrée par un symbole s interprété comme la fonction successeur et par 0 mais rien n'empêche d'enrichir la signature par une infinité de constantes $0, 1, 2, 3, \ldots$

Définition 7.1 *Une Σ_0-structure ordo-sortée $\mathcal{A}$ est* prédéfinie *si*

- $\Sigma_0 = (\mathcal{S}_0, \mathcal{F}_0, \mathcal{P}_0)$ *est une signature régulière.*
- $\mathcal{A}$ *est une structure engendrée par les termes.*

On note $L_{\mathcal{A}}$ le langage de contraintes de *base* défini par la Σ_0-structure prédéfinie $\mathcal{A}$ et un ensemble de variables $\mathcal{X}_0$ de sorte dans $\mathcal{S}_0$. Les termes et contraintes du langage $L_{\mathcal{A}}$ seront désormais appelés termes et contraintes de *base*. Les symboles de Σ_0 sont *prédéfinis*.

Dans une spécification basée sur une structure prédéfinie, on étend cette structure par la définition de nouveaux symboles. La signature Σ_0 est donc enrichie par de nouveaux symboles de sortes et de fonctions pour aboutir à une signature $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{P}, \leq)$ telle que $\mathcal{S} \supseteq \mathcal{S}_0$, $\mathcal{F} \supseteq \mathcal{F}_0$ et $\mathcal{P} = \mathcal{P}_0$. Soient $\mathcal{F}_1 = \mathcal{F} \backslash \mathcal{F}_0$, $\mathcal{S}_1$ l'ensemble des sortes dans les profils de $\mathcal{F}_1$ et $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1, \emptyset)$.

Les signatures Σ_0 et Σ_1 ont donc par construction des ensembles de symboles de fonctions disjoints.

L'ensemble $\mathcal{X}_0$ des variables de base est lui aussi étendu en un ensemble de variables $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1$ par l'adjonction d'un ensemble de variables $\mathcal{X}_1$ de sortes dans $\mathcal{S}_1 \backslash \mathcal{S}_0$.

Le problème que nous allons considérer est celui du mélange de deux théories équationnelles à sortes ordonnées (Σ_0, E_0) et (Σ_1, E_1) dont l'une, E_0, est l'ensemble des égalités $\mathcal{T}h(\mathcal{A})$ valides dans $\mathcal{A}$ et l'autre, E_1, est un ensemble de Σ_1-axiomes.

Le problème de combinaison traité par A. Boudet [Bou92] peut être vu comme un cas particulier de mélange de théories partageant des symboles de fonctions: un même symbole peut par exemple être associatif dans l'une des théories et être associatif-commutatif dans l'autre théorie. Le problème abordé au cours de ce chapitre est fondamentalement différent: les seuls symboles partagés seront cette fois les symboles de sortes et non les symboles de fonctions.

Bien que l'hypothèse des ensembles de symboles de fonctions disjoints soit primordiale pour construire une extension conservative à E_0 et E_1, elle ne suffit pas pour autant à généraliser le cas mono-sorté. Il faut encore supposer que l'enrichissement est construit de façon hiérarchique [BGW92], c'est-à-dire que les nouvelles sortes sont supérieures à celles de base.

Hypothèse 7.1

- Σ_0, Σ_1 *sont des signatures régulières.*
- *Les sortes de $\mathcal{S}_1 \backslash \mathcal{S}_0$ sont supérieures ou incomparables à celles de $\mathcal{S}_0$.*
- $\mathcal{F}_0 \cap \mathcal{F}_1 = \emptyset$.
- $\mathcal{P}_1 = \emptyset$.

On notera que la signature Σ_0 n'est pas enrichie par de nouveaux prédicats.
Précisons quelques cas d'enrichissements remarquables.

- Le cas sans intérêt où $\mathcal{S}_0 \cap \mathcal{S}_1 = \emptyset$ et les sortes de $\mathcal{S}_0$ et $\mathcal{S}_1$ sont incomparables. Il n'existe alors aucun terme bien formé pouvant être construit à la fois avec des symboles de $\mathcal{F}_0$ et $\mathcal{F}_1$.

- Le cas dégénéré où $\mathcal{S}_1 \cap \mathcal{S}_0 = \emptyset$ mais avec des sortes de $\mathcal{S}_0$ et $\mathcal{S}_1$ comparables. Modifions l'exemple en introduction, où g a le profil $g : Term, Term \to Term$ avec $Nat \leq Term$. Le terme $g(x * y, x + 1)$ est bien formé mais il n'existe pas de terme de sorte Nat ayant un sous-terme étranger de sorte $Term$, et donc pas d'égalité $g(x*y, x+1)+g(x, y) = g(x, y)+g(x*y, x+1)$. La propriété que $+$ soit commutatif dans $\mathcal{N}$ ne s'étend pas à la sorte $Term$.

- Le cas mono-sorté: $\mathcal{S}_0 = \mathcal{S}_1 = \{s_*\}$ pour lequel la construction d'un langage de contraintes combiné a déjà été étudiée au chapitre 6.

- Un cas plus intéressant est celui du *mélange avec partage d'une seule sorte visible* s_* qui est considérée comme maximale pour Σ_0 et minimale pour Σ_1. Le terme $g(x * y, y) + 1$ est bien formé si $g : Nat, Nat \to Nat$, où Nat est la seule sorte partagée. C'est par défaut ce cas que nous privilégions dans ce qui suit même s'il n'est pas complètement général. Il englobe toutefois le cas plus simple précédent.

- Il est parfois souhaitable que la sorte maximale de la structure prédéfinie ne soit pas celle visible du reste de l'enrichissement. Prenons par exemple la structure des pseudo-booléens dont la sorte maximale est Int mais dans laquelle on ne sait pas résoudre de façon symbolique les contraintes dont les variables sont justement de cette sorte. C'est au contraire la sorte $Bool$ qui devrait être visible de l'extérieur. Cet état de fait nous a poussé à étudier le cas général autorisant des sortes partagées sans restriction. En contrepartie, ce cas exige des hypothèses supplémentaires d'ordre sémantique pour permettre de généraliser la construction d'un système de réécriture combiné.

Les nouveaux symboles peuvent posséder des propriétés élémentaires exprimées sous forme d'axiomes comme la commutativité ou l'associativité-commutativité. Nous supposerons dorénavant que (Σ_1, E_1) est une théorie équationnelle régulière et non effondrante. De plus, (Σ_1, E_1) satisfait une propriété de préservation des sortes par E_1-égalité qui permet en général [GKK90, Kir88] de passer d'un cadre mono-sorté à un cadre ordo-sorté.

Définition 7.2 *Soit Σ une signature régulière. Une théorie équationnelle ordo-sortée engendrée par* (Σ, E) préserve les sortes *si*

$$\forall t, t' \in \mathcal{T}(\Sigma, \mathcal{X}),\ t =_E t' \Rightarrow ls(t) = ls(t').$$

Hypothèse 7.2 *(Σ_1, E_1) est une théorie équationnelle ordo-sortée régulière, non effondrante et préservant les sortes.*

Cette hypothèse est nécessaire pour s'assurer de la terminaison de la réécriture modulo la congruence sur les termes hétérogènes que nous allons définir et qui formalise celle présentée en introduction.

7.2 Le mélange de théories partageant une seule sorte

Nous allons montrer comment l'hypothèse du mélange avec une seule sorte partagée s_* permet de construire un système de réécriture combiné analogue à celui du cas mono-sorté.

7.2.1 Hypothèses

On suppose l'existence d'une sorte s_* de Σ_0 qui s'interprète par le domaine de la structure prédéfinie. Cette sorte s_* peut donc être considérée comme maximale dans $\mathcal{S}_0$. Les nouvelles sortes de Σ_1 sont toutes supérieures à la seule sorte *visible* de Σ_0.

Hypothèse 7.3

- $\mathcal{S}_0 \cap \mathcal{S}_1 \subseteq \{s_*\}$,
- s_* *est sorte maximale pour* $\mathcal{S}_0$,
- s_* *est sorte minimale pour* $\mathcal{S}_1$.

7.2.2 Système de réécriture combiné

Les méthodes de combinaison vues jusqu'à présent s'appuyent fortement sur la connaissance d'une relation de réécriture convergente ayant la même expressivité que l'égalité dans le mélange des théories équationnelles. Nous allons construire une telle relation mais cette fois sur les termes ordo-sortés. L'ensemble des règles $R_c = E_0^> \cup E_1^>$ est défini exactement comme dans le cas mono-sorté. La seule différence se situe dans le fait que $E_i^>$ est l'ensemble des instances orientées de E_i-égalités *ordo-sortées*. Par contre, ces règles ne peuvent pas s'appliquer n'importe comment, au risque de réécrire un terme bien formé en un autre qui ne l'est plus. Pour éviter le problème, on se limite à réécrire à certaines positions, dites de ruptures, qui sont intuitivement celles où s'effectuent les changements de théories. La définition précise est donnée au chapitre 1.

Définition 7.3 *La relation* $\rightarrow_{E_i^>}$ *est définie par:* $t \rightarrow_{E_i^>} t'$ *si*
$\exists l \rightarrow r \in E_i^>$, $\exists \omega \in RPos(t)$ *t.q.* $t_{|\omega} = l$ *et* $t' = t[r]_\omega$. *La relation* $\rightarrow_{R_c}$ *est* $\rightarrow_{E_0^>} \cup \rightarrow_{E_1^>}$.

On vérifie ensuite que $\rightarrow_{R_c}$ simule effectivement l'égalité $=_{E_0 \cup E_1}$.

Proposition 7.1 *Si* $t, t' \in T(\Sigma, \mathcal{X})$ *alors*

$$t =_{E_0 \cup E_1} t' \iff t \stackrel{*}{\longleftrightarrow}_{R_c} t'.$$

Preuve : ($\Leftarrow$) Si le pas de réécriture s'effectue au sommet, alors $t \rightarrow_{R_c,\epsilon} t'$ implique trivialement $t =_{E_0 \cup E_1} t'$.

Considérons maintenant les autres cas.

Soient $t \rightarrow_{E_1^>} t'$, une position $\omega \neq \epsilon \in RPos(t)$ et une règle $l\psi \rightarrow r\psi \in E_1^>$ telles que $t' = t[\omega \hookleftarrow r\psi]$. Le sous-terme étranger $t_{|\omega}$ est forcément de sorte s_* car $\mathcal{S}_0 \cap \mathcal{S}_1 = \{s_*\}$. La plus petite sorte de $t_{|\omega}$ ne peut être inférieure à s_* car

s_* est minimale dans $\mathcal{S}_1$. Donc $ls(t_{|\omega}) = s_*$ et $t[\omega \hookleftarrow x : s_*] \in T(\Sigma, \mathcal{X})$. Ensuite $ls(t_{|\omega}) = ls(r\psi) = ls(l\psi) = s_*$ car (Σ_1, E_1) préserve les sortes. Par conséquent, $t' \in T(\Sigma, \mathcal{X})$ et $t =_{E_0 \cup E_1} t'$.

Soient $t \rightarrow_{E_0^>} t'$, une position $\omega \neq \epsilon \in RPos(t)$ et une règle $l\psi \rightarrow r\psi \in E_0^>$ telles que $t' = t[\omega \hookleftarrow r\psi]$. Le sous-terme étranger $t_{|\omega}$, avec $t(\omega)$ un symbole de Σ_0, est donc de sorte s_*. A nouveau, $t[\omega \hookleftarrow x : s_*] \in T(\Sigma, \mathcal{X})$ car $\mathcal{S}_0 \cap \mathcal{S}_1 = \{s_*\}$. Le terme $r\psi$ étant lui aussi de sorte s_*, on a $t' \in T(\Sigma, \mathcal{X})$ et $t =_{E_0 \cup E_1} t'$.

($\Rightarrow$) On montre que $t \longleftrightarrow_{E_0 \cup E_1} t'$ implique $t \longleftrightarrow_{R_c} t'$. Supposons $t_{|\lambda} = g\psi$, $t' = t[\lambda \hookleftarrow d\psi]$ et $g = d \in E_i$. Il existe une position $\omega \in RPos(t)$ et un terme u tels que $t' = t'[u[d\psi]]_\omega$ et $u[g] =_{E_i} u[d]$. Le couple $(u[g]\psi, u[d]\psi)$ s'oriente forcément en une règle de $E_i^>$ parce que l'ordre de simplification est total sur $T(\Sigma \cup \mathcal{X})$. □

Les relations $\rightarrow_{E_0^>}$, $\rightarrow_{E_1^>}$ et $\rightarrow_{R_c}$ convergent pour les mêmes raisons que celles évoquées dans le cas mono-sorté.

L'existence d'un système de réécriture combiné convergent sur $T(\Sigma, \mathcal{X})$ permet de généraliser sans difficulté les résultats du cas mono-sorté.

Les sous-termes étrangers sont forcément de sorte s_* et sont donc abstraits par des variables de sorte s_*.

Définition 7.4 *Une* abstraction par variables *est une bijection π entre l'ensemble des termes non variables R_c-normalisés $T\downarrow_{R_c} = \{u \downarrow_{R_c} | u \in T(\Sigma \cup \mathcal{X})$ et $u \downarrow_{R_c} \in T(\Sigma \cup \mathcal{X}) \backslash \mathcal{X}\}$ et un sous-ensemble de variables de $\mathcal{X}$ tel que $\pi(u \downarrow_{R_c}) = x : s_*$.*

La i-abstraction t^{π_i} d'un terme t est un terme bien formé.

Une équation i-pure peut être résolue de façon correcte et complète. La preuve qui établit cette propriété essentielle est rigoureusement identique à celle du cas mono-sorté.

Le système de réécriture, ou plus exactement la 0-abstraction sert à définir l'interprétation de $\mathcal{P}_0$ dans le langage $L_{E_0 \cup E_1}$ dont le domaine d'interprétation est la théorie équationnelle $=_{E_0 \cup E_1}$. Nous avons montré que la résolution s'effectue ensuite de façon analogue au cas équationnel. Cependant, la définition de l'interprétation de $\mathcal{P}_0$ peut être simplifiée en tenant compte du fait que E_1 est non effondrante. En conséquence, $s =_{E_0 \cup E_1} t$ est équivalent à $s^{\pi_0} =_{E_0} t^{\pi_0}$, c'est-à-dire à $s^{\pi_0} =_{\mathcal{A}} t^{\pi_0}$, et la validité de $p(t_1^{\pi_0}, \ldots, t_n^{\pi_0})$ dans $\mathcal{A}$ est équivalent à celle de $p((t_1 \downarrow_{R_c})^{\pi_0}, \ldots, (t_n \downarrow_{R_c})^{\pi_0})$ dans $\mathcal{A}$. Mais quoi qu'il en soit, le système de réécriture R_c est toujours utilisé même s'il est caché dans la 0-abstraction.

7.3 Le mélange de théories ordo-sortées

Au lieu de se restreindre à une seule sorte partagée et maximale pour la structure prédéfinie, on veut maintenant étudier les résultats précédents et la construction d'un système de réécriture combiné dans le cas du mélange de théories équationnelles à sortes ordonnées satisfaisant les mêmes hypothèses.

7.3.1 Hypothèses

On suppose que $\Sigma = \Sigma_0 \cup \Sigma_1$ est une signature régulière ayant un ensemble fini de sortes. Les théories (Σ_0, E_0) et (Σ_1, E_1) vérifient que toute classe d'équivalence modulo $=_{E_i}$ contient un terme de plus petite sorte minimale.

Définition 7.5 *Soit Σ une signature régulière. Une théorie équationnelle ordo-sortée (Σ, E) est* sémantiquement régulière *si*

$$\forall t, t' \in \mathcal{T}(\Sigma, \mathcal{X}), t =_E t',\ \exists t'' \in \mathcal{T}(\Sigma, \mathcal{X}),\ t =_E t'' =_E t'\ t.q.\ ls(t) \geq ls(t'') \leq ls(t').$$

Une théorie équationnelle préservant les sortes est de façon évidente sémantiquement régulière.

Exemple 7.1 *Les structures $\mathcal{PB}$ des pseudo-booléens et $\mathcal{PF}$ des pseudo-finis sont sémantiquement régulières car l'ordre sur les sortes est total: $Bool \leq Int$ pour $\mathcal{PB}$ et $Prim \leq Int$ pour $\mathcal{PF}$.*

L'hypothèse de régularité sémantique faite sur (Σ_0, E_0) et (Σ_1, E_1) permet de construire une relation de réécriture combinée qui sans cela n'aurait pas pu être convergente.

7.3.2 Système de réécriture combiné

Le système de réécriture utilise un ordre de simplification $>_{simp}$ sur $T(\Sigma \cup \mathcal{X})$ tel que les variables $\mathcal{X}$ soient inférieures aux autres termes par $>_{simp}$. Cela évite d'avoir à introduire de nouveaux sous-termes étrangers par réécriture.

Définition 7.6 *Soit E_i un ensemble de Σ_i-axiomes ($i = 0, 1$). On note $E_i^>$ l'ensemble des règles $l\sigma \to r\sigma$ telles que*

- *$l, r \in T(\Sigma_i, \mathcal{X})$ et $l =_{E_i} r$,*
- *σ est une Σ-substitution vérifiant $\forall x \in \mathcal{X},\ ls(x\sigma) = ls(x)$ et $\forall x \in \mathcal{V}(r) \backslash \mathcal{V}(l), x\sigma \in \mathcal{X}$,*
- *$ls(l\sigma) > ls(r\sigma)$ ou $ls(l\sigma) = ls(r\sigma)$ et $l\sigma >_{simp} r\sigma$.*

On notera que la condition $ls(x\sigma) = x$ est justifiée par le fait que l'on considère les instances de toutes les E_i-égalités. Ainsi l'instanciation d'une E_i-égalité par une spécialisation [GKK90] est encore une E_i-égalité. La condition $x\sigma \in \mathcal{X}$ pour les variables x n'apparaissant pas dans le membre gauche d'une E_i-égalité est justifiée par le fait que les variables sont inférieures aux autres termes par $>_{simp}$.

La solution de facilité pour s'assurer de la terminaison serait de supposer l'existence d'un ordre de simplification compatible avec la décroissance des sortes. Nous allons voir que cela n'est pas nécessaire parce qu'il suffit d'appliquer les règles aux positions de rupture en réutilisant la réécriture de la définition 7.3.

Proposition 7.2 *Si $t, t' \in T(\Sigma, \mathcal{X})$ alors*

$$t =_{E_0 \cup E_1} t' \iff t \overset{*}{\longleftrightarrow}_{R_c} t'.$$

Preuve : On montre que si $t \longleftrightarrow_{E_0 \cup E_1} t'$ alors $t \overset{*}{\longleftrightarrow}_{R_c} t'$. Supposons $t_{|\lambda} = g\psi$, $t' = t[\lambda \hookleftarrow d\psi]$ et $(g = d) \in E_i$. Il existe une position $\omega \in RPos(t)$ et un terme u tels que $t' = t'[u[d\psi]]_\omega$ et $u[g] =_{E_i} u[d]$. La substitution ψ est la composition $\mu\sigma$ d'un renommage μ et d'une substitution σ telle que $ls(x\sigma) = ls(x)$ pour tout $x \in \mathcal{X}$. En posant $l = u[g]\mu$, $r = u[d]\mu$, on a $l =_{E_i} r$. Par hypothèse de régularité sémantique, il existe q tel que $l =_{E_i} q =_{E_i} r$ avec $ls(l) \geq ls(q) \leq ls(r)$. En appliquant σ, on a toujours $ls(l\sigma) \geq ls(q\sigma) \leq ls(r\sigma)$. Donc $t \longleftrightarrow_{R_c,\omega,l\sigma \longleftrightarrow q\sigma} \longleftrightarrow_{R_c,\omega,q\sigma \longleftrightarrow r\sigma} t'$.

La réciproque est évidente. □

Proposition 7.3 *Les relations* $\rightarrow_{E_0^>}, \rightarrow_{E_1^>}$ *et* $\rightarrow_{R_c}$ *sont convergentes.*

Preuve : Terminaison: on considère la mesure de complexité associée à un terme t et définie par $MC(t) = (MS(t), t)$ où:

- $MS(t) = \bigcup_{\{t_{|\omega} \mid \omega \in RPos(t)\}} ls(t_{|\omega})$, un multiensemble sur $\mathcal{S}$;
- t le terme ;

et l'extension lexicographique $(>_{mult}, >_{simp})$ de l'extension multi-ensemble $>_{mult}$ de $>$ bien fondé sur $\mathcal{S}$ et de $>_{simp}$ un ordre bien fondé sur $T(\Sigma \cup \mathcal{X})$.

Soit $t \rightarrow_{R_c, l\sigma \rightarrow r\sigma} t'$. Supposons d'abord que le nombre de sous termes étrangers ne diminue pas. Si $ls(l\sigma) > ls(r\sigma)$ alors $MS(t) >_{mult} MS(t')$. Si $ls(l\sigma) = ls(r\sigma)$ alors $MS(t) = MS(t')$ mais $t >_{simp} t'$. Supposons maintenant que le membre droit de la règle appliquée est un sous-terme étranger du membre gauche. Dans ce cas, $\{t_{|\omega} \mid \omega \in RPos(t)\} \supset \{t'_{|\omega'} \mid \omega' \in RPos(t')\}$ et $MS(t) >_{mult} MS(t')$. En définitive, $MC(t)\,(>_{mult}, >_{simp})\,MC(t')$ si $t \rightarrow_{R_c} t'$.

Confluence: considérons le cas non trivial d'une superposition $t' \leftarrow_{E_i^>, \omega} t \rightarrow_{E_i^>, \epsilon} t''$ à une position critique. Il existe alors $p =_{E_i} q$ et σ tel que $t' = p\sigma, t'' = q\sigma$ avec $ls(p\sigma) = ls(p)$ et $ls(q\sigma) = ls(q)$. Plusieurs cas peuvent ensuite se produire:

Si $ls(p) = ls(q)$ alors $t' \longleftrightarrow_{E_i^>} t''$.
Si $ls(p) > ls(q)$ alors $t' \rightarrow_{E_i^>} t''$.
Si $ls(p) < ls(q)$ alors $t' \leftarrow_{E_i^>} t''$.
Si $ls(p)$ et $ls(q)$ sont incomparables, alors, par hypothèse, il existe r tel que $p =_{E_i} r =_{E_i} q$ avec $ls(p) > ls(r) < ls(q)$ et donc $t' \rightarrow_{E_i^>} r\sigma \leftarrow_{E_i^>} t''$. En résumé,

$$t' \xrightarrow{*}_{E_i^>} \circ \xleftarrow{*}_{E_i^>} t''.$$

Finalement, une superposition $t' \leftarrow_{E_i^>} t \rightarrow_{E_j^>} t'', i \neq j$ correspond à une superposition à une position variable puisque les signatures Σ_0 et Σ_1 sont disjointes. Cette superposition est similaire au cas mono-sorté car les relations $\rightarrow_{E_0^>}$ et $\rightarrow_{E_1^>}$ font décroître les sortes. □

Les propositions détaillées dans ce qui suit étendent celles vues dans le cas mono-sorté.

On montre tout d'abord que $(\Sigma_0 \cup \Sigma_1, E_0 \cup E_1)$ est une extension conservative de (Σ_0, E_0) et (Σ_1, E_1).

Proposition 7.4 *Si* t, t' *sont des termes* i*-purs alors*

$$t =_{E_0 \cup E_1} t' \Longleftrightarrow t =_{E_i} t'.$$

Preuve : $(\Leftarrow)$ $=_{E_i} \subseteq =_{E_0 \cup E_1}$.

$(\Rightarrow)$ Par construction, $t =_{E_0 \cup E_1} t' \Leftrightarrow t \downarrow_{E_i^>} = t' \downarrow_{E_i^>}$. Si t (ou t') un terme i-pur est réductible par $\rightarrow_{E_i^>}$ alors il existe une règle $l\sigma \rightarrow r\sigma \in E_i^>$ n'introduisant pas de sous-termes étrangers, i.e. $\mathcal{R}an(\sigma) \subseteq \mathcal{T}(\Sigma_i, \mathcal{X})$, tel que $t[l\sigma] \rightarrow_{E_i^>} t[r\sigma]$. Donc $l\sigma =_{E_i} r\sigma$, $ls(l\sigma) \geq ls(r\sigma)$ et $t[l\sigma] =_{E_i} t[r\sigma]$ avec $t[r\sigma]$ i-pur. On prouve ensuite par induction nœthérienne sur $\rightarrow_{E_i^>}$ que $t =_{E_i} t \downarrow_{E_i^>}$. □

Corollaire 7.1 $(\Sigma_0 \cup \Sigma_1, E_0 \cup E_1)$ *est une extension conservative sémantiquement régulière de* (Σ_0, E_0) *et* (Σ_1, E_1).

Preuve : $(\Sigma_0 \cup \Sigma_1, E_0 \cup E_1)$ est sémantiquement régulière puisque

$$\forall t, t' \in \mathcal{T}(\Sigma, \mathcal{X}),\ t =_{E_0 \cup E_1} t \downarrow_{R_c} =_{E_0 \cup E_1} t' \text{ t.q. } ls(t) \geq ls(t \downarrow_{R_c}) \leq ls(t').$$

$(\Sigma_0 \cup \Sigma_1, E_0 \cup E_1)$ est une extension conservative de (Σ_0, E_0) et (Σ_1, E_1), d'après la proposition précédente. □

$(\Sigma_0 \cup \Sigma_1, E_0 \cup E_1)$ a donc la même propriété que celle requise initialement pour (Σ_0, E_0) et (Σ_1, E_1).

La relation $\rightarrow_{R_c}$ permet ensuite de définir l'abstraction de façon très naturelle puisqu'un terme en forme normale est abstrait par une variable dont la sorte est la plus petite sorte du terme.

Définition 7.7 *Une* abstraction par variables *est une bijection* π *entre l'ensemble des termes non variables* R_c*-normalisés* $T \downarrow_{R_c} = \{u \downarrow_{R_c} | u \in T(\Sigma \cup \mathcal{X})$ *et* $u \downarrow_{R_c} \in T(\Sigma \cup \mathcal{X}) \backslash \mathcal{X}\}$ *et un sous-ensemble de variables de* $\mathcal{X}$ *tel que* $\pi(u \downarrow_{R_c}) = x : s$ *si* $ls(u \downarrow_{R_c}) = s$.

La i-abstraction d'un terme R_c-normalisé remplace chaque sous-terme étranger u par une variable dont la sorte s est la plus petite sorte de $u \downarrow_{R_c}$ avec $ls(u) \geq s$ puisque $\rightarrow_{R_c}$ fait décroître les sortes. La i-abstraction d'un terme reste donc un terme bien formé. Mais comme dans le cas mono-sorté, la i-abstraction ne sera pratiquement appliquée que sur des termes ou des substitutions R_c-normalisés.

Les résultats vus précédemment concernant la résolution dans une composante se généralisent immédiatement. La 0-abstraction est ensuite utilisée afin de définir le langage de contraintes $L_{E_0 \cup E_1}$ pour lequel un algorithme de résolution se dérive directement de celui vu dans le cas mono-sorté.

7.4 Langage combiné

La construction du langage combiné que nous entreprenons est valable pour les deux formes de mélanges décrits jusqu'à présent. Mais contrairement à ce que l'on pourrait imaginer, la structure qui va permettre de définir le langage combiné n'est pas tout-à-fait un mélange de théories équationnelles même si elle s'en inspire fortement. La difficulté se situe dans la possible non régularité des égalités de $\mathcal{T}h(\mathcal{A})$. L'objectif principal étant d'introduire une réécriture modulo des classes de termes hétérogènes, il est impossible d'avoir une égalité non régulière sans faire apparaitre un cycle.

Exemple 7.2 *En reprenant l'exemple donné en introduction avec la structure des entiers* $\mathcal{N}$*, on a:* $0 =_{\mathcal{N}} 0 * g(0,0) \rightarrow_R 0 * 0 =_{\mathcal{N}} 0$ *car* $g(0,0) \rightarrow_R 0$*. Il y a donc un cycle, c'est-à-dire que la relation* $\rightarrow_{R/\mathcal{N}}$ *simulant la réécriture modulo* $=_{\mathcal{N}}$ *et définie par* $=_{\mathcal{N}} \circ \rightarrow_R \circ =_{\mathcal{N}}$ *ne termine pas.*

Le problème est dû à la non régularité des égalités de $\mathcal{T}h(\mathcal{A})$ qui permet d'engendrer de nouveaux sous-termes étrangers. Mais il est difficilement acceptable de supposer quoi que ce soit sur la forme des égalités de $E_0 = \mathcal{T}h(\mathcal{A})$, et en particulier la régularité. Cela serait contraire au principe de théorie prédéfinie qui peut être *a priori* quelconque.

7.4.1 Construction de l'interprétation

La solution que nous proposons consiste à utiliser une restriction de $=_{E_0 \cup E_1}$ préservant la théorie du symbole de tête ainsi que le nombre de couches non prédéfinies.

Définition 7.8 *Le nombre de* 1*-couches d'un terme* t *est défini inductivement par:*

- $nc_1(x) = 0$,
- $nc_1(t) = 1 + \sum_{s \in AST(t)} nc_1(s)$ *si* $t(\epsilon) \in \mathcal{F}_1$,
- $nc_1(t) = \sum_{s \in AST(t)} nc_1(s)$ *si* $t(\epsilon) \notin \mathcal{F}_1$.

La relation de congruence $\sim_{\mathcal{C}}$ est la restriction de $=_{E_0 \cup E_1}$ préservant la théorie du symbole de tête et le nombre de 1-couches. Nous en donnons la définition suivante:

Définition 7.9 *La relation* $\sim_{\mathcal{C}}$ *est définie par:* $s \sim_{\mathcal{C}} t$ *si* $s =_{E_0 \cup E_1} t$ *et*

- $s(\epsilon), t(\epsilon) \in \mathcal{F}_i \cup \mathcal{X}_i$
- *et* $nc_1(s) = nc_1(t)$

La relation $\sim_{\mathcal{C}}$ est clairement une relation de congruence.
Détaillons quelques situations remarquables:

- Si s et t sont des termes de base alors $s \sim_{\mathcal{C}} t$ si $s =_{E_0 \cup E_1} t$, qui est encore équivalent à $s =_{\mathcal{A}} t$.
- Un 1-terme ne peut pas être égal modulo $\sim_{\mathcal{C}}$ à un 0-terme. En particulier, $0 \not\sim_{\mathcal{C}} g(0,0)$.
- $\sim_{\mathcal{C}}$ et $=_{E_0 \cup E_1}$ sont identiques lorsque E_0 et E_1 sont régulières et non effondrantes.
- Dans le cas où $\mathcal{S}_1 \cap \mathcal{S}_0 = \emptyset$ avec des sortes de $\mathcal{S}_0$ inférieures à celles de $\mathcal{S}_1$, on a $s \sim_{\mathcal{C}} t$ si $s =_{E_0 \cup E_1} t$ car $nc_1(s) = nc_1(t)$.

La relation de réécriture introduite au cours de ce chapitre a pour objet de réécrire un 1-terme modulo $\sim_{\mathcal{C}}$. Il est donc essentiel de ne pas introduire de nouveaux 1-termes qui seraient autant de nouvelles réécritures potentielles. Mais il faut encore pouvoir s'assurer de la préservation de la plus petite sorte d'un 1-terme par égalité modulo $\sim_{\mathcal{C}}$.

Définition 7.10 *La relation* $\sim_{\mathcal{C}}$ préserve la sorte d'un terme à symbole de tête non prédéfini *si pour tout* 1*-terme* t*,*

$$t \sim_{\mathcal{C}} t' \Rightarrow ls(t) = ls(t').$$

Par définition, un 1-terme ne peut être égal modulo $\sim_{\mathcal{C}}$ qu'à un autre 1-terme.

Lorsque le mélange s'effectue au travers d'une seule sorte visible s_*, on a $ls(t) = ls(t \downarrow_{R_c})$ si t est un 1-terme et donc $ls(t) = ls(t \downarrow_{R_c}) = ls(t' \downarrow_{R_c}) = ls(t')$ si $t \sim_{\mathcal{C}} t'$.

Dans le cas plus général, nous supposerons que:

Hypothèse 7.4 *La relation* $\sim_{\mathcal{C}}$ *préserve la sorte d'un terme à symbole de tête non prédéfini.*

Pour assurer cette hypothèse dans le cas général, il suffit par exemple que les symboles de fonctions dans Σ_1 ne soient pas surchargés de plusieurs profils.

Définition 7.11 *Le langage de contraintes combiné $L_{\mathcal{C}}$ est défini par la Σ-structure, l'algèbre de termes $\mathcal{T}(\Sigma, \mathcal{X})/\sim_{\mathcal{C}}$ et l'ensemble des variables $\mathcal{X}$.*

L'interprétation des prédicats dans le langage combiné $L_{\mathcal{C}}$ est identique à celle prise pour $L_{E_0 \cup E_1}$ dans les sections précédentes.

La relation de congruence $\sim_{\mathcal{C}}$ n'étant pas toujours stable par instanciation, nous allons nous restreindre aux substitutions dont l'application préserve l'égalité modulo $\sim_{\mathcal{C}}$. Il s'agit d'instancier les variables par des termes qui ne risquent pas de changer le nombre de 1-couches. En particulier, les variables de base sont instanciées par des termes de base et non par des termes hétérogènes.

Définition 7.12 *Un terme de* base *appartient à $\mathcal{T}(\Sigma_0, \mathcal{X}_0)$. Une substitution* admissible *est la composée de deux substitutions $\mu_1\mu_0$ telles que*

- $\mathcal{D}om(\mu_1) \subseteq \mathcal{X}_1$ *et* $\mathcal{R}an(\mu_1) \subseteq \mathcal{T}(\Sigma_1, \mathcal{X})$,
- $\mathcal{D}om(\mu_0) \subseteq \mathcal{X}_0$ *et* $\mathcal{R}an(\mu_0) \subseteq \mathcal{T}(\Sigma_0, \mathcal{X}_0)$.

L'ensemble des Σ-substitutions admissibles est noté $SUBST_0$.

Avec cette définition, si μ est une solution admissible alors les termes de $\mathcal{R}an(\mu)$ n'ont qu'une 1-couche au plus. La restriction aux variables de base d'une substitution admissible est en particulier une substitution de base.

Proposition 7.5 *Si σ et ϕ sont deux substitutions admissibles alors*

$$\sigma \sim_{\mathcal{C}} \phi \Longleftrightarrow \sigma =_{E_0 \cup E_1} \phi.$$

Preuve : Si E_1 est une théorie non effondrante, alors un 1-terme s vérifiant $nc_1(s) = 1$ est forcément égal par $=_{E_0 \cup E_1}$ à un autre 1-terme t vérifiant $nc_1(t) = 1$. □

Une autre propriété de ces substitutions est de conserver l'égalité modulo $\sim_{\mathcal{C}}$ par instanciation.

Lemme 7.1 *Si t est un terme non variable et σ une substitution admissible alors $nc_1(t\sigma) = nc_1(t)$.*

Preuve : Une substitution admissible instancie une variable de $\mathcal{X}_0$ par un terme de base, ce qui ne crée pas une nouvelle 1-couche. Cette substitution instancie aussi une variable de $\mathcal{X}_1$ qui se trouve nécessairement sous un symbole de fonction dans $\mathcal{F}_1$ par un terme vérifiant $nc_1(t) \leq 1$ ce qui ne crée pas non plus une nouvelle 1-couche. □

Proposition 7.6 $\forall s, t \in \mathcal{T}(\Sigma, \mathcal{X}),\ s \sim_{\mathcal{C}} t \Longleftrightarrow \forall \sigma \in SUBST_0,\ s\sigma \sim_{\mathcal{C}} t\sigma.$

Preuve : Si s et t sont des i-termes alors $nc_1(s\sigma) = nc_1(t\sigma)$ lorsque σ est une substitution admissible.

Si l'un des deux termes s ou t est une variable alors il s'agit nécessairement de deux termes de base. □

Il existe une infinité de substitutions admissibles qui sont solutions d'une contrainte, qu'elle soit équationnelle ou non. Ces solutions peuvent être représentées par un ensemble complet de solutions admissibles ordonnées suivant l'ordre de subsumption $\leq_{\mathcal{C}}^{*V}$ défini par $\sigma \leq_{\mathcal{C}}^{*V} \phi$ s'il existe $\mu \in SUBST_0$ telle que $\forall x \in V,\ x\phi =_{E_0 \cup E_1} x\sigma\mu$.

Définition 7.13 *L'ensemble des* solutions admissibles *d'une $L_{\mathcal{C}}$-contrainte c est*

$$SS^*_{\mathcal{C}}(c) = SS_{\mathcal{C}}(c) \cap SUBST_0.$$

Un ensemble complet de solutions admissibles d'une $L_{\mathcal{C}}$-contrainte c est noté $CSS^*_{\mathcal{C}}(c)$.

L'objectif est maintenant de déterminer un algorithme de résolution de contraintes dans $L_{\mathcal{C}}$ qui calcule un ensemble complet de solutions admissibles à partir d'un algorithme d'unification dans C et d'un algorithme de résolution de contraintes dans $L_{\mathcal{A}}$.

7.4.2 Algorithme de résolution

La démarche adoptée consiste à d'abord transformer un problème dans $L_{\mathcal{C}}$ en un autre dans $L_{E_0 \cup E_1}$ puis à déterminer comment en extraire les solutions admissibles. Les principes de combinaison vont devoir être adaptés pour tenir compte de cette restriction supplémentaire sur la forme des solutions souhaitées.

On montre dans un premier temps comment se ramener facilement à une contrainte dans $L_{E_0 \cup E_1}$ par unification, entre eux, des sous-termes étrangers.

Proposition 7.7 *Soient s et t des termes et σ une substitution admissible.*

1. *Si $s(\epsilon), t(\epsilon) \in \mathcal{F}_i$ et $nc_1(s) = nc_1(t)$ alors $s\sigma \sim_{\mathcal{C}} t\sigma \iff s\sigma =_{E_0 \cup E_1} t\sigma$.*
2. *Si $s \in \mathcal{X}$ et t un terme tel que $nc_1(t) \leq 1$, alors $s\sigma \sim_{\mathcal{C}} t\sigma \iff s\sigma =_{E_0 \cup E_1} t\sigma$.*
3. *Dans les cas contraires (cas symétriques mis à part), il n'existe pas de substitution admissible σ telle que $s\sigma \sim_{\mathcal{C}} t\sigma$.*

Preuve : C'est une conséquence immédiate de la définition 7.9 et du fait que E_1 soit non effondrante pour le point 2. □

Corollaire 7.2 *Pour tout $L_{\mathcal{C}}$-contrainte d, il existe une contrainte c telle que*

$$SS^*_{\mathcal{C}}(d) = SS^*_{E_0 \cup E_1}(c),$$

où $SS^*_{E_0 \cup E_1}(c) = SS_{E_0 \cup E_1}(c) \cap SUBST_0$.

On se place désormais dans le langage de contraintes $L_{E_0 \cup E_1}$.

Comme dans le cas mono-sorté, une contrainte c est transformée en une conjonction $c_0 \wedge c_1$ de contraintes respectivement 0-pure et 1-pure par l'introduction de nouvelles variables qui sont existentiellement quantifiées et peuvent être instanciées par des termes complètement hétérogènes.

La restriction aux substitutions admissibles permet de limiter le non-déterminisme inhérent à la combinaison des solutions.

Définition 7.14 *Soient $<$ un ordre linéaire sur $V_0 \oplus V_1 = \mathcal{V}(c_0 \wedge c_1)$ et V un sous-ensemble de $\mathcal{V}(c_0 \wedge c_1)$. Une solution combinée $\sigma_0 \odot \sigma_1 \in SS^{<}_{E_0}(c_0, V_1) \odot SS^{<}_{E_1}(c_1, V_0)$ est* élémentaire *pour V si*

- $V \cap \mathcal{X}_0 \subseteq V_0$.
- $\forall v_0 \in V \cap \mathcal{X}_0,\ \forall v_1 \in V_1,\ v_0 < v_1$.
- *Pour toutes variables $v_0 \in V \cap \mathcal{X}_0$ et $v_1 \in V_1 \cap (V \cap \mathcal{X}_1)$, il n'existe pas $w_0 \in V_0, w_1 \in V_1$ telles que*
$$v_0 < w_1 < w_0 < v_1.$$
- *Pour toutes variables $v_0 \in V \cap \mathcal{X}_0$ et $w_0 \in V_0 \cap (V \cap \mathcal{X}_1)$, il n'existe pas $w_1 \in V_1$ telle que*
$$v_0 < w_1 < w_0.$$

Les solutions admissibles de c sont obtenues à partir des solutions de $c_0 \wedge c_1$ élémentaires pour $\mathcal{V}(c)$.

Proposition 7.8 *(Correction) La restriction à $\mathcal{V}(c)$ d'une solution combinée $\sigma_0 \odot \sigma_1 \in SS^{<}_{E_0}(c_0, V_1) \odot SS^{<}_{E_1}(c_1, V_0)$ élémentaire pour $\mathcal{V}(c)$ est une solution admissible de la contrainte c.*

Preuve : Il est clair qu'une solution combinée élémentaire pour $\mathcal{V}(c)$ est une solution de c. Reste à montrer qu'il s'agit d'une solution admissible. Par hypothèse, les variables de $\mathcal{V}(c) \cap \mathcal{X}_0$ sont instanciées dans E_0 par des termes ne pouvant contenir de sous-termes étrangers car ces variables sont minimales pour $<$. Il s'agit donc de termes de base. Les autres variables de $\mathcal{V}(c)$ s'instancient dans E_1 par des termes ne pouvant contenir que des termes de base pour sous-termes étrangers. La substitution $(\sigma_0 \odot \sigma_1)_{|\mathcal{V}(c)}$ est donc une solution admissible de c. □

Il faut encore s'assurer qu'une solution combinée élémentaire pour $\mathcal{V}(c)$ est une instance admissible d'une solution combinée construite à l'aide de solutions principales suivant la même restriction linéaire $<$.

Proposition 7.9 *Pour toute solution combinée $\phi \in SS_{E_0}(c_0, V_1) \odot SS_{E_1}(c_1, V_0)$ élémentaire pour V, il existe une solution combinée $\sigma \in CSS^{<}_{E_0}(c_0, V_1) \odot CSS^{<}_{E_1}(c_1, V_0)$ élémentaire pour V telle que $\sigma \leq^{*V}_{\mathcal{C}} \phi$.*

Preuve : Nous avons vu au chapitre 2 que si $\sigma_0 \leq^{V_0}_{E_0} \phi_0$ et $\sigma_1 \leq^{V_1}_{E_1} \phi_1$ alors $(\phi_0 \odot \phi_1) =^{V_0 \oplus V_1}_{E_0 \cup E_1} (\sigma_0 \odot \sigma_1)\mu_0\mu_1$ où $\sigma_0 =_{E_0} \phi_0\mu_0$ et $\sigma_1 =_{E_1} \phi_1\mu_1$. Les substitutions μ_0 et μ_1 ont des domaines disjoints, leur composition ou union est une substitution admissible. □

Toute solution admissible en forme normale suivant R_c est l'instance d'une solution combinée dont la restriction linéaire est déterminée d'après l'ordre sous-terme.

On traite d'abord du cas simple ne nécessitant pas d'identification.

Proposition 7.10 *(Complétude). Soit σ une solution R_c-normalisée dont les termes de $\mathcal{R}an(\sigma)$ sont des termes non-variables et distincts deux à deux. Si σ est une solution de $c_0 \wedge c_1$ telle que $\sigma_{|V} \in SUBST_0$, alors il existe une solution combinée $\sigma_0 \odot \sigma_1 \in CSS^{<}_{E_0}(c_0, V_1) \odot CSS^{<}_{E_1}(c_1, V_0)$ élémentaire pour V telle que $\sigma_0 \odot \sigma_1 \leq^{*V}_{\mathcal{C}} \sigma$.*

Preuve : Soit σ une solution admissible de $c_0 \wedge c_1$ en forme normale suivant R_c. Nous avons vu dans le cas mono-sorté comment une substitution en forme normale permet de définir une restriction linéaire $<$ sur $V_0 \oplus V_1$. Soit $x_0 \in V \cap \mathcal{X}_0$. Puisque $x_0\sigma(\epsilon) \in \mathcal{F}_0$, on a $x_0 \in V_0$ par définition de V_0.

Supposons qu'il existe une variable $v_1 \in V_1$ telle que $v_1 < x_0 \in \mathcal{X}_0$. Par définition de $<$, $v_1\sigma$ avec $x_1\sigma(\epsilon) \in \mathcal{F}_1$ est sous-terme de $x_0\sigma$, ce qui est absurde puisque $x_0\sigma \in \mathcal{T}(\Sigma_0, \mathcal{X}_0)$.

Soit $x \in V \cap \mathcal{X}_1$ telle que $x_0 < x$. S'il existe une ou des variables ne respectant pas les conditions de la définition 7.14 sur $<$, alors le terme $x\sigma$ contient un 0-terme qui n'est pas un terme de base, ce qui est absurde puisque σ est supposée être une substitution admissible.

La substitution σ est égale à $(\sigma^{\pi_0} \odot \sigma^{\pi_1})\pi^{-1}$ avec $\sigma^{\pi_0} \odot \sigma^{\pi_1} \in SS^{<}_{E_0}(c_0, V_1) \odot SS^{<}_{E_1}(c_1, V_0)$, où π^{-1} consiste à remplacer éventuellement des variables de $\mathcal{X}_0$ par des termes de base dans $\mathcal{R}an((\sigma^{\pi_0} \odot \sigma^{\pi_1})_{|V \cap \mathcal{X}_1})$, ce qui signifie que $\sigma^{\pi_0} \odot \sigma^{\pi_1} \leq^{*V}_{\mathcal{C}} \sigma$. D'après la proposition précédente, il existe $\sigma_0 \odot \sigma_1 \in CSS^{<}_{E_0}(c_0, V_1) \odot CSS^{<}_{E_1}(c_1, V_0)$ telle que $\sigma_0 \odot \sigma_1 \leq^{*V}_{\mathcal{C}} \sigma^{\pi_0} \odot \sigma^{\pi_1} \leq^{*V}_{\mathcal{C}} \sigma$. □

Finalement, pour tenir compte de l'ensemble des solutions admissibles R_c-normalisées, il suffit de considérer toutes les identifications sur les variables de $\mathcal{V}(c_0 \wedge c_1)$.

Proposition 7.11 *L'ensemble des substitutions $\xi(\sigma_0 \odot \sigma_1)$ telles que*

- $\xi \in ID_{\mathcal{V}(c_0 \wedge c_1)}$,
- $<$ *est un ordre linéaire sur* $V_0 \oplus V_1 = \mathcal{V}(c_0 \wedge c_1)\xi$,
- $\sigma_0 \odot \sigma_1 \in CSS^{<}_{E_0}(c_0\xi, V_1) \odot CSS^{<}_{E_1}(c_1\xi, V_0)$ *est une solution combinée élémentaire pour* $\mathcal{V}(c)\xi$,

est un $CSS^{*}_{\mathcal{C}}(c)$.

Preuve : Remarquons tout d'abord que la R_c-forme normale d'une solution admissible reste une substitution admissible. Puis on identifie les variables des contraintes afin de satisfaire la condition d'application de la proposition précédente et prouver ainsi la complétude. □

7.5 Spécification basée sur une structure prédéfinie

Une spécification équationnelle basée sur une structure prédéfinie est formée d'égalités qui sont contraintes dans la structure prédéfinie.

Définition 7.15 *Soit $\mathcal{A}$ une $\Sigma_0 = (\mathcal{S}_0, \mathcal{F}_0, \mathcal{P}_0)$-structure prédéfinie et Σ une signature ordo-sortée. Une égalité* contrainte dans une Σ_0-structure prédéfinie $\mathcal{A}$ *est un couple formé d'une égalité et d'une contrainte, noté $(l = r \parallel c)$, où $l, r \in \mathcal{T}(\Sigma, \mathcal{X})$ et c est une contrainte de $L_\mathcal{A}$.*

On peut remarquer que deux signatures sont utilisées, l'une pour la partie égalité et l'autre pour la partie contrainte.

Définition 7.16 *Une* spécification basée sur une structure prédéfinie $\mathcal{A}$ *est un quadruplet $(\Sigma, E, C, L_\mathcal{A})$ tel que*

- *Σ est un enrichissement d'une signature ordo-sortée Σ_0,*
- *$L_\mathcal{A}$ est un langage de contraintes s'interprétant dans la Σ_0-structure $\mathcal{A}$,*
- *E est un ensemble d'égalités sur $\mathcal{T}(\Sigma, \mathcal{X})$ contraintes dans la Σ_0-structure prédéfinie $\mathcal{A}$,*
- *C est un ensemble d'égalités sur $\mathcal{T}(\Sigma, \mathcal{X})$.*

La structure prédéfinie est une Σ_0-structure alors que E est un ensemble d'égalités formées sur un enrichissement de Σ_0. La structure prédéfinie est donc insuffisante pour donner une interprétation à une spécification $(\Sigma, E, C, L_\mathcal{A})$. Il s'agit de déterminer une extension conservative qui tienne compte des nouveaux symboles. On utilise en l'occurrence la structure du langage combiné $L_\mathcal{C}$.

7.5.1 Construction d'une interprétation

La structure associée à une spécification $(\Sigma, E, C, L_\mathcal{A})$ est une algèbre de termes quotientée par une relation de congruence prenant en compte à la fois la $\mathcal{A}$-égalité, la C-égalité et la schématisation des égalités E contraintes dans $L_\mathcal{A}$.

On reprend la notation introduite dans [KKR90]. L'ensemble des égalités schématisées par une égalité contrainte $(l = r \parallel c)$ est

$$\mathcal{S}_\mathcal{A}(l = r \parallel c) = \{l\sigma = r\sigma \mid \sigma \in SUBST_0, \sigma_{|\mathcal{X}_0} \in SS_\mathcal{A}(c)\}.$$

L'ensemble des règles de réécriture schématisées par une règle de réécriture contrainte $(l \to r \parallel c)$ est de manière identique

$$\mathcal{S}_\mathcal{A}(l \to r \parallel c) = \{l\sigma \to r\sigma \mid \sigma \in SUBST_0, \sigma_{|\mathcal{X}_0} \in SS_\mathcal{A}(c)\}.$$

Les définitions qui suivent sont valables pour les égalités contraintes et les règles de réécriture contraintes.

L'ensemble des instances schématisées par un ensemble E d'égalités contraintes est l'union des égalités schématisées

$$\mathcal{S}_{\mathcal{A}}(E) = \bigcup_{(l=r \parallel c)\in E} \mathcal{S}_{\mathcal{A}}(l = r \parallel c).$$

L'interprétation standard d'une spécification basée sur une structure prédéfinie est une congruence contenant $\sim_{\mathcal{C}}$ et celle engendrée par $\mathcal{S}_{\mathcal{A}}(E)$.

Définition 7.17 L'interprétation standard $\mathcal{M}$ *d'une spécification* $(\Sigma, E, C, L_{\mathcal{A}})$ *est l'algèbre de termes*

$$\mathcal{T}(\Sigma, \mathcal{X})/(\sim_{\mathcal{C}} \cup \sim_{\mathcal{S}_{\mathcal{A}}(E)})^*,$$

où $\sim_{\mathcal{S}_{\mathcal{A}}(E)}$ *est la plus petite relation de congruence sur* $\mathcal{T}(\Sigma, \mathcal{X})$ *contenant* $\mathcal{S}_{\mathcal{A}}(E)$*. On note* $\sim_{\mathcal{M}}$ *la relation* $(\sim_{\mathcal{C}} \cup \sim_{\mathcal{S}_{\mathcal{A}}(E)})^*$

L'objectif est de substituer complètement le langage de contraintes combiné $L_{\mathcal{C}}$ au langage de base $L_{\mathcal{A}}$. C'est pourquoi nous allons introduire une schématisation des égalités contraintes dans le langage combiné qui va s'avérer équivalente à celle fixée dans le langage de base si l'on se restreint aux substitutions admissibles. Le langage de base $L_{\mathcal{A}}$ ne sera donc plus directement utilisé.

La schématisation d'une égalité contrainte dans la structure $\mathcal{C}$ est définie par

$$\mathcal{S}_{\mathcal{C}}(l = r \parallel c) = \{l\sigma = r\sigma \mid \sigma \in SS^*_{\mathcal{C}}(c)\}$$

et $\mathcal{S}_{\mathcal{C}}(E) = \bigcup_{(l=r \parallel c)\in E} \mathcal{S}_{\mathcal{C}}(l = r \parallel c)$. Une substitution $\sigma \in SS^*_{\mathcal{C}}(c)$ est simplement dite solution de c modulo $\sim_{\mathcal{C}}$ car seules les substitutions admissibles seront désormais considérées.

Nous allons nous limiter au cas où l'ensemble E est orienté en un ensemble R de règles de réécriture contraintes.

Définition 7.18 *Le langage de contraintes* $L_{R,\mathcal{C}}$ *associé à la spécification* $(\Sigma, R, C, L_{\mathcal{C}})$ *est défini par la signature* Σ*, l'interprétation standard* $\mathcal{M}$ *et l'ensemble des variables* $\mathcal{X}$*.*

Nous verrons ultérieurement comment s'interprètent les prédicats dans le langage $L_{R,\mathcal{C}}$ grâce à la relation de réécriture engendrée par R, de même expressivité que $\sim_{\mathcal{M}}$ et qui peut être vue comme une réécriture des classes d'équivalence de $\sim_{\mathcal{C}}$.

7.5.2 Réécriture contrainte

L'égalité dans le langage de contraintes $L_{R,\mathcal{C}}$ est simulée par une relation de réécriture contrainte modulo $\sim_{\mathcal{C}}$.

Définition 7.19 *La relation* $\rightarrow_{R,\mathcal{C}}$ *est définie sur* $T(\Sigma, \mathcal{X})$ *par:* $t \rightarrow_{R,\mathcal{C}} t'$ *si* $\exists (g \rightarrow d \parallel c) \in R$, $\exists \sigma$ *t.q.*

$$t_{|\omega} \sim_{\mathcal{C}} g\sigma$$
$$\sigma \in SS^*_{\mathcal{C}}(c)$$
$$t' = t[\omega \hookleftarrow d\sigma]$$

La relation $\rightarrow_{R,\mathcal{C}}$ est stable par instanciation admissible.

Si l'on suppose que les règles de R ne contiennent que des contraintes c de base, alors la relation $\sim_{R,\mathcal{C}}= (\sim_{\mathcal{C}} \cup \longleftrightarrow_{R,\mathcal{C}})^*$ coïncide avec $\sim_{\mathcal{M}}$.

Proposition 7.12 *Les relations $\sim_{R,\mathcal{C}}=\sim_{\mathcal{M}}$ sont identiques si R ne contient que des contraintes de base.*

Preuve : Si c est une contrainte de $L_{\mathcal{A}}$ alors $\{\sigma \in SUBST_0 \mid \sigma_{|\mathcal{X}_0} \in SS_{\mathcal{A}}(c)\} = SS^*_{\mathcal{C}}(c)$ et donc $\mathcal{S}_{\mathcal{A}}(R) = \mathcal{S}_{\mathcal{C}}(R)$. □

Nous verrons plus loin que le système de réécriture R doit aussi vérifier que **tout terme de base est irréductible**. Pour ce faire, il suffit que C soit un ensemble de Σ_1-axiomes non effondrants et que chaque membre gauche de R ait un symbole de tête non prédéfini. Ces conditions sont suffisantes pour préserver la structure prédéfinie et signifient qu'un symbole prédéfini est un constructeur pour R et qu'un axiome effondrant doit être orienté en une règle de réécriture contrainte.

Définition 7.20 *Le système de réécriture R* termine modulo $\sim_{\mathcal{C}}$ *si la relation $\rightarrow_{R/\mathcal{C}}$ définie par $\sim_{\mathcal{C}} \circ \rightarrow_{R,\mathcal{C}} \circ \sim_{\mathcal{C}}$ termine. La relation $\rightarrow_{R,\mathcal{C}}$ est* convergente modulo $\sim_{\mathcal{C}}$ *si la relation $\rightarrow_{R/\mathcal{C}}$ termine et si*

$$\overset{*}{\longleftrightarrow}_{R,\mathcal{C}} \subseteq \overset{*}{\longrightarrow}_{R/\mathcal{C}} \circ \sim_{\mathcal{C}} \circ \overset{*}{\longleftarrow}_{R/\mathcal{C}} .$$

Pour la relation de réécriture que nous venons d'introduire, les définitions de confluence, cohérence, confluence et cohérence locale sont définies de façon standard [JK86].

Théorème 7.1 *Soit R un système de réécriture avec contraintes de base. Si R termine modulo $\sim_{\mathcal{C}}$ alors les propriétés suivantes sont équivalentes:*

1. *$\rightarrow_{R,\mathcal{C}}$ est convergente modulo $\sim_{\mathcal{C}}$,*
2. *$\rightarrow_{R,\mathcal{C}}$ est confluente et cohérente modulo $\sim_{\mathcal{C}}$,*
3. *$\rightarrow_{R,\mathcal{C}}$ est localement confluente et localement cohérente modulo $\sim_{\mathcal{C}}$,*
4. *$\forall t, t' \in T(\Sigma, \mathcal{X}),\ t \sim_{R,\mathcal{C}} t' \Leftrightarrow t \downarrow_{R,\mathcal{C}} \sim_{\mathcal{C}} t' \downarrow_{R,\mathcal{C}}$.*

Preuve : Similaire à celle du théorème 1.3. □

7.5.3 Propriété de Church-Rosser

La confluence peut être obtenue par le calcul de paires critiques et d'extensions. Le langage de contraintes $L_{\mathcal{C}}$ présente l'avantage d'être défini par une Σ-structure et non pas par une Σ_0-structure comme l'était $L_{\mathcal{A}}$. Une égalité (resp. une règle de réécriture) $(l = r \parallel c)$ (resp. $(l \rightarrow r \parallel c)$) est formée à partir de termes l, r et d'une contrainte c construits sur la même signature Σ. C'est ce qui va permettre de définir les notion des paires critiques et extensions comme étant des égalités contraintes et de se placer dans le même cadre que celui introduit dans [KKR90].

Définition 7.21 *Une paire critique contrainte de* $(g \to d \parallel c')$ *et* $(l \to r \parallel c)$ *est l'égalité contrainte*

$$d = g[r]_\omega \parallel c \wedge c' \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$$

où $c \wedge c' \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$ *est satisfaisable.*

Définition 7.22 *Une extension contrainte de* $(l \to r \parallel c)$ *par rapport à* $(g = d) \in C$ *est*

$$g[l]_\omega = g[r]_\omega \parallel c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$$

où $c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$ *est satisfaisable.*

Les ensembles de paires critiques et extensions contraintes d'un ensemble de règles R sont notés respectivement $CCP(R)$ et $CCE(R)$.

Lemme 7.2 *Si* $t_1 \leftarrow^{\omega, l\to r \parallel c}_{R,\mathcal{C}} g\sigma \longleftrightarrow^{\epsilon}_{C} d\sigma$ *ou* $t_1 \leftarrow^{\omega, l\to r \parallel c}_{R,\mathcal{C}} g\sigma \to^{\epsilon, g\to d \parallel c'}_{R} d\sigma$, *avec* ω *une position non variable de* g, *alors il existe une extension contrainte ou une paire critique contrainte* $(p = q \parallel S)$ *telle que* σ *est une solution de* S *modulo* $\sim_{\mathcal{C}}$ *et* $p\sigma \sim_{\mathcal{C}} d\sigma$, $q\sigma \sim_{\mathcal{C}} t_1$.

Pour faire converger ces pics, il suffit que les paires critiques et extensions convergent.

Définition 7.23 *Une paire critique ou une extension contrainte* $(p = q \parallel S)$ *est convergente modulo* $\sim_{\mathcal{C}}$ *si* $\forall \sigma$ *solution de* S *modulo* $\sim_{\mathcal{C}}$,

$$p\sigma \xrightarrow{*}_{R,\mathcal{C}} u \sim_{\mathcal{C}} v \xleftarrow{*}_{R,\mathcal{C}} q\sigma$$

Un système de réécriture doit respecter la structure prédéfinie.

Définition 7.24 R *est un* système de réécriture avec contraintes de base *si* $\forall (l \to r \parallel c) \in R$,

- $\mathcal{V}(l) \supseteq \mathcal{V}(r)$,
- $\forall \sigma \in SS^*_{\mathcal{C}}(c), ls(l\sigma) \geq ls(r\sigma)$,
- l *est un* 1*-terme,*
- $\mathcal{V}(c) \subseteq \mathcal{X}_0$.

La condition syntaxique sur la forme des règles de R empêche un terme de base de se réécrire en un terme ayant un symbole de tête non prédéfini.

Corollaire 7.3 *Un terme de base est irréductible par* $\to_{R/\mathcal{C}}$.

Preuve : Supposons un terme base t réductible par $\to_{R,\mathcal{C}}$. Il existe donc $(g \to d \parallel c)$ de R et une position ω telles que $t_{|\omega} \sim_{\mathcal{C}} g\sigma$. Puisque C est un ensemble d'axiomes non effondrants, $(t_{|\omega})^{\pi_0} =_{E_0} (g\sigma)^{\pi_0}$. Le terme g ayant un symbole de $\mathcal{F}_1$ en tête, $(g\sigma)^{\pi_0}$ est une variable qui n'apparait pas dans $(t_{|\omega})^{\pi_0} = t_{|\omega}$ un terme de base. Ceci contredit l'hypothèse que $E_0 = \mathcal{T}h(\mathcal{A})$ est une théorie consistante.

Un terme de base est encore irréductible par $\to_{R/\mathcal{C}}$ car un terme de base ne peut être égal modulo $\sim_{\mathcal{C}}$ qu'à un autre terme de base. □

Le fait que $\sim_{\mathcal{C}}$ préserve la sorte d'un 1-terme permet aussi de s'assurer d'une propriété essentielle pour pouvoir généraliser les propriétés de la réécriture mono-sortée.

Proposition 7.13 *Si $t \to_{R,\mathcal{C}} t'$ alors $ls(t) \geq ls(t')$.*

Preuve : Si t est réductible par $\to_{R,\mathcal{C}}$, alors il existe $(g \to d \parallel c)$ de R et une position ω telles que $t_{|\omega} \sim_{\mathcal{C}} g\sigma$ et $g\sigma$ est un 1-terme. Par hypothèse, $ls(t_{|\omega}) = ls(g\sigma) \geq ls(r\sigma)$. Donc $ls(t) \geq ls(t[\omega \hookleftarrow d\sigma])$. □

Toutes les contraintes des règles de R ne contiennent que des variables de $\mathcal{X}_0$ qui s'instancient nécessairement par des termes irréductibles pour $\to_{R,\mathcal{C}}$. C'est ce qui permet de faire converger un pic à une position variable lorsque celle-ci est contrainte, c'est-à-dire dans $\mathcal{X}_0$. La condition imposant que les symboles de tête des règles ne sont pas prédéfinis sert à limiter la construction des extensions aux seuls axiomes de C, sans avoir à considérer ceux de $\mathcal{T}h(\mathcal{A})$.

Théorème 7.2 *Soit R un système de réécriture avec contraintes de base. Toutes les paires critiques et extensions contraintes sont convergentes modulo $\sim_{\mathcal{C}}$ si et seulement si $\to_{R,\mathcal{C}}$ est localement confluent et cohérent modulo $\sim_{\mathcal{C}}$.*

Preuve : ($\Leftarrow$) Evident.

($\Rightarrow$) Supposons $t_1 \leftarrow^{\omega}_{R,\mathcal{C}} t \longleftrightarrow^{\epsilon, g\alpha \longleftrightarrow d\alpha}_{R_c} t_2$.

Si $\omega \in \mathcal{VP}os(g)$ alors c'est un cas de superposition variable similaire au cas mono-sorté car $\to_{R,\mathcal{C}}$ est sorte décroissante. On a donc

$$t_1 \xrightarrow{*}_{R,\mathcal{C}} t_1\alpha' \longleftrightarrow_{R_c} t_2\alpha' \xleftarrow{*}_{R,\mathcal{C}} t_2,$$

où la substitution α' est définie comme d'habitude.

Si $\omega \notin \mathcal{VP}os(g)$ alors $t(\epsilon) \in \mathcal{F}_1$.

- Si $t \longleftrightarrow_{E_0^>} t_2$ alors t est sous-terme étranger de t_2 et $t_1 \xleftarrow{*}_{R,\mathcal{C}} t_2$.
- Si $t \longleftrightarrow_{E_1^>} t_2$ alors $t \xleftrightarrow{*}_{C} t_2$. Il existe ainsi un terme t'_1 pour lequel $t_1 \leftarrow^{\omega}_{R,\mathcal{C}} t \longleftrightarrow_{C} t'_1 \sim_{\mathcal{C}} t_2$ et (t_1, t'_1) est une instance d'une extension contrainte $(p = q \parallel S)$ convergente par hypothèse:

$$t_1 \sim_{\mathcal{C}} p\sigma \xrightarrow{*}_{R,\mathcal{C}} u \sim_{\mathcal{C}} v \xleftarrow{*}_{R,\mathcal{C}} q\sigma \sim_{\mathcal{C}} t'_1 \sim_{\mathcal{C}} t_2.$$

Supposons $t_1 \leftarrow^{\omega}_{R,\mathcal{C}} t \to^{\epsilon}_{R} t_2$. On distingue trois cas de superposition:

- position disjointe: les réductions commutent.
- position variable: si c'est une variable de $\mathcal{X}_0$ alors un terme de base serait réductible, ce qui est absurde par définition de R. Sinon la variable n'apparait pas dans la contrainte et l'on trouve alors une preuve par réécriture comme habituellement dans le cas mono-sorté ou ordo-sorté avec hypothèse de décroissance des sortes.

- position critique: s'il existe une position de rupture entre ϵ et ω, alors cela correspond à un cas de superposition variable. Dans le cas contraire, t_1 et t_2 sont des instances d'une paire critique contrainte $(p = q \parallel S)$ convergente par hypothèse:

$$t_1 \sim_{\mathcal{C}} p\sigma \xrightarrow{*}_{R,\mathcal{C}} u \sim_{\mathcal{C}} v \xleftarrow{*}_{R,\mathcal{C}} q\sigma \sim_{\mathcal{C}} t_2.$$

□

Une paire critique convergente pour toute solution de $CSS^*_{\mathcal{C}}(c)$ l'est aussi pour toute solution de $SS^*_{\mathcal{C}}(c)$. Lorsque la résolution de contraintes dans $\sim_{\mathcal{C}}$ est finitaire, cela permet de n'avoir à tester qu'un ensemble fini de paires critiques non contraintes. Cette remarque est une motivation au troisième point du théorème suivant:

Théorème 7.3 *L'égalité contrainte $(p = q \parallel S)$ est convergente si*

- *S est insatisfaisable modulo $\sim_{\mathcal{C}}$,*
- *ou $p \sim_{\mathcal{C}} q$,*
- *ou $\forall \theta \in CSS^*_{\mathcal{C}}(S)$, $(p\theta = q\theta \parallel \top)$ est convergente*
- *ou $g \rightarrow_{R,\mathcal{C}} g'$ et $(g' = d \parallel S)$ est convergente.*

Les règle de transformation pour tester la confluence et la cohérence modulo $\sim_{\mathcal{C}}$ sont données en figure 7.1. La correction de ces règles est établie par le théorème suivant:

Théorème 7.4 *S'il existe une dérivation finie par ces règles:*

$$(\emptyset, R) \vdash (CE_1, R) \vdash \ldots \vdash (CE_i, R)$$

telle que $CE_i = \emptyset$, $CCP(R) \cup CCE(R) \subseteq \bigcup_{0 \leq j \leq i} CE_j$ alors R est localement confluent et cohérent modulo $\sim_{\mathcal{C}}$.

Preuve : Si R n'est pas localement confluent et cohérent, alors d'après le théorème 7.2, il existe une paire critique ou extension non convergente mise dans $\bigcup_{0 \leq j \leq i} CE_j$. La règle Supprimer ne peut s'appliquer sur une égalité non convergente, sinon il y a contradiction d'après le théorème 7.1. L'application des autres règles sur une égalité non convergente engendre une égalité non convergente, sinon il y a contradiction d'après le théorème 7.1. Par conséquent, on montre par induction sur $j \leq i$ que CE_j est non vide, ce qui est en contradiction avec l'hypothèse. □

On remarquera que le test de convergence de $\rightarrow_{R,\mathcal{C}}$ est encore conditionné par celui de la terminaison de R modulo $\sim_{\mathcal{C}}$, problème évoqué dans la section suivante.

Théorème 7.5 *Soit R un système de règles de réécriture avec contraintes de base. Si la relation $\rightarrow_{R,\mathcal{C}}$ est convergente modulo $\sim_{\mathcal{C}}$, alors $\mathcal{M}$ est un enrichissement consistant de $\mathcal{A}$.*

Preuve : Considérons deux éléments a, a' de $\mathcal{A}$ qui sont égaux modulo $\sim_{R,\mathcal{C}}$. Alors

$$a \xrightarrow{*}_{R,\mathcal{C}} s \sim_{\mathcal{C}} w \xleftarrow{*}_{R,\mathcal{C}} a'$$

Aucune règle de R ni égalité de C ne peut s'appliquer sur a ou a', donc $a =_{\mathcal{A}} a'$. □

Deduire
$CE, CR \cup \{(g \to d \parallel c'), (l \to r \parallel c)\}$
$\mapsto\!\!\!\!\!\rightarrow$
$CE \cup \{g[r]_\omega = d \parallel c \wedge c' \wedge (g_{|\omega} =^?_{\mathcal{C}} l)\}$,
$CR \cup \{(g \to d \parallel c'), (l \to r \parallel c)\}$
si $c \wedge c' \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$ satisfaisable
Etendre
$CE, CR \cup \{(l \to r \parallel c)\}$
$\mapsto\!\!\!\!\!\rightarrow$
$CE \cup \{g[l]_\omega = g[r] \parallel c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)\}$,
$CR \cup \{(l \to r \parallel c)\}$
si $(g = d) \in C$ et $c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$ satisfaisable
Simplifier
$CE \cup \{(g = d \parallel S)\}, CR$
$\mapsto\!\!\!\!\!\rightarrow$
$CE \cup \{(g' = d \parallel S)\}, CR$
si $g \to_{R,\mathcal{C}} g'$
Supprimer
$CE \cup \{(p = q \parallel S)\}, CR$
$\mapsto\!\!\!\!\!\rightarrow$
CE, CR
si $p =_{\mathcal{C}} q$ ou S insatisfaisable
Propager
$CE \cup \{(p = q \parallel S \wedge \hat{\theta})\}, CR$
$\mapsto\!\!\!\!\!\rightarrow$
$CE \cup \{(p\theta = q\theta \parallel S)\}, CR$
si $\theta \in \mathcal{D}om(\theta) \cap \mathcal{V}(S) = \emptyset$ et $\theta \in SUBST_0$

Figure 7.1. Test de confluence et cohérence modulo $\sim_{\mathcal{C}}$

L'étape suivante pour obtenir une procédure de complétion serait d'orienter les paires critiques produites par Deduire et les extensions produites par Etendre. A ce stade, nous sommes confrontés à plusieurs problèmes:

- Une paire critique $(p = q \parallel S)$ vérifiant $p(\epsilon), q(\epsilon) \in \mathcal{F}_0$ ne peut pas être orientée d'après la définition 7.24. Ce cas de figure ne se présente que lors d'une superposition en ϵ.
- La contrainte S d'une paire critique $(p = q \parallel S)$ doit être résolue pour que les variables de $\mathcal{V}(S) \not\subseteq \mathcal{X}_0$ puissent instancier $(p = q)$ et donc ne plus figurer dans la partie contrainte.

- Enfin, pour une égalité contrainte $(p = q \parallel S)$ vérifiant $p(\epsilon) \in \mathcal{F}_1$ et $\mathcal{V}(S) \subseteq \mathcal{X}_0$, il faut encore trouver un ordre de réduction. Ce problème est évoqué dans la section suivante.

7.5.4 Terminaison

La structure choisie comme interprétation standard d'une spécification $(\Sigma, R, C, L_{\mathcal{A}})$ ne facilite pas le problème de terminaison.

Pour que la relation $\rightarrow_{R,\mathcal{C}}$ termine, il suffit de trouver un ordre de réduction $>_{\mathcal{C}}$ compatible avec $\sim_{\mathcal{C}}$, stable par instanciation de base et par contexte pour lequel $l > r$ si $(l \rightarrow r \parallel c) \in R$.

Une première idée consiste naturellement à essayer d'étendre sur la structure $\mathcal{C}$ un ordre nœthérien défini sur $\mathcal{A}$ en posant $l >_{\mathcal{C}} r$ si $l^{\pi_0} >_{\mathcal{A}} r^{\pi_0}$. Cette extension a l'avantage à première vue de rester un ordre nœthérien. Malheureusement, cet ordre ne permet pas d'orienter des règles de réécriture dont chaque membre gauche l a une tête dans $\mathcal{F}_1$ qui, une fois 0-abstrait, est égal à une variable. Pour un ordre nœthérien $>_{\mathcal{A}}$ et $\mathcal{A}$ une structure consistante, une variable est forcément un terme minimal. En conclusion, $>_{\mathcal{C}}$ ne permet d'orienter aucune des règles d'un système de réécriture avec contraintes de base. C'est un ordre trop sémantique.

Par contre, on peut tester la terminaison par un critère plus syntaxique basé par exemple sur le nombre de 1-couches $nc_1(t)$ d'un terme t. Cette mesure est compatible par définition avec $\sim_{\mathcal{C}}$: $nc_1(s) = nc_1(t)$ si $s \sim_{\mathcal{C}} t$. Lorsque les règles de R font décroître le nombre de 1-couches, alors $nc_1(s) > nc_1(t)$ si $s \rightarrow_{R,\mathcal{C}} t$ et l'on montre ainsi que $\rightarrow_{R/\mathcal{C}}$ termine. C'est en particulier le cas lorsque les membres droits des règles de R sont des termes de base. Ce critère est cependant très syntaxique.

Si l'on cherche une méthode plus générale, il est clair qu'il ne faut pas se ramener uniquement à la structure prédéfinie $\mathcal{A}$ comme nous avons tenté de le faire mais essayer de combiner un ordre compatible avec $=_{\mathcal{T}h(\mathcal{A})}$ et un ordre compatible avec $=_C$. Mais la difficulté du problème de modularité dans le contexte de la terminaison ne permet pas à l'heure actuelle de dégager des idées générales qui puissent être appliquées au problème de terminaison d'un système de réécriture avec contraintes de base.

A titre d'exemple, prenons toutefois le cas élémentaire des listes de booléens où la congruence $\sim_{\mathcal{C}}$ est suffisamment pauvre pour ne pas poser de problème dans la recherche d'un ordre nœthérien qui lui est compatible.

Exemple 7.3 *Soient* Σ_1 *l'ensemble des symboles* vide, cons, tete, queue *munis des profils suivants:*

$$\begin{array}{l} vide :\rightarrow List \\ cons : Bool, List \rightarrow List \\ tete : List \rightarrow Bool \\ queue : List \rightarrow List \end{array}$$

l'ensemble des axiomes

$$L = \left\{ \begin{array}{lcl} tete(cons(b : Bool, l : List)) & = & b : Bool \\ queue(cons(b : Bool, l : List)) & = & l : List \end{array} \right.$$

et le langage de contraintes engendré par la structure des booléens.

Les égalités contraintes s'orientent évidemment de la gauche vers la droite, sans avoir à se soucier de la compatibilité avec $\sim_{\mathcal{C}}$ car chaque terme n'est égal qu'à une instance de lui-même.

7.6 Surréduction contrainte

L'objectif de cette section est d'appliquer des techniques de surréduction à la résolution de contrainte dans le langage $L_{R,\mathcal{C}}$. L'intérêt de la méthode réside dans son incrémentalité puisqu'elle permet à partir d'un algorithme de résolution de contraintes dans $L_{\mathcal{C}}$ d'obtenir une procédure de résolution dans $L_{R,\mathcal{C}}$.

7.6.1 Interprétation des prédicats

Comme nous l'avons fait en son temps pour $L_{\mathcal{C}}$, il s'agit de déterminer une interprétation des prédicats de $\mathcal{P}_0$ pour $L_{R,\mathcal{C}}$.

En supposant R convergent modulo $\sim_{\mathcal{C}}$, on ramène le problème de la validité modulo $\sim_{R,\mathcal{C}}$ à celui de la validité modulo $\sim_{\mathcal{C}}$ par mise en forme normale suivant $\rightarrow_{R,\mathcal{C}}$.

Définition 7.25 *Si $p \in \mathcal{P}_0$ alors l'interprétation $p_{\mathcal{M}}$ de p est la relation définie par*

$$\forall t_1, \ldots, t_n \in T(\Sigma, \mathcal{X}),\ p_{\mathcal{M}}(t_1, \ldots, t_n) \text{ est vrai ssi } L_{\mathcal{C}} \models p(t_1 \downarrow_{R,\mathcal{C}}, \ldots, t_n \downarrow_{R,\mathcal{C}}).$$

Cette définition est évidemment compatible avec $\sim_{\mathcal{M}} = \sim_{R,\mathcal{C}}$ puisque $\rightarrow_{R,\mathcal{C}}$ est convergente.

On montre ensuite qu'une contrainte $p^?(t_1, \ldots, t_n)$ valide dans $L_{\mathcal{C}}$ est encore valide dans $L_{R,\mathcal{C}}$. C'est vrai de façon évidente pour une contrainte équationnelle puisque $\sim_{\mathcal{C}} \subseteq \sim_{R,\mathcal{C}}$. Pour les autres contraintes, nous avons besoin d'un lemme stipulant que l'égalité modulo $=_{E_0 \cup E_1}$ est préservée par normalisation suivant $\rightarrow_{R,\mathcal{C}}$.

Lemme 7.3 *Pour tous termes $s, t \in T(\Sigma, \mathcal{X})$, si $s =_{E_0 \cup E_1} t$ alors $s \downarrow_{R,\mathcal{C}} =_{E_0 \cup E_1} t \downarrow_{R,\mathcal{C}}$.*

Preuve : On utilise le fait que si $t_1 \xleftarrow{*}_{R,\mathcal{C}} t \longleftrightarrow_{R_c} t_2$ alors $t_1 \xrightarrow{*}_{R,\mathcal{C}} \circ \xleftrightarrow{*}_{R_c} \circ \xleftarrow{*}_{R,\mathcal{C}} t_2$.
Deux cas sont à considérer:

- Si $t_1 \xleftarrow{*}_{R,\mathcal{C}} t \longleftrightarrow_{E_0^>} t_2$ alors $t_1 \xrightarrow{*}_{R,\mathcal{C}} \circ \longleftrightarrow_{E_0^>} \circ \xleftarrow{*}_{R,\mathcal{C}} t_2$ car lorsque $\longleftrightarrow_{E_0^>}$ s'applique à position $t(\omega) \in \mathcal{F}_1$, alors $t_{|\omega}$ est nécessairement sous-terme de t_2 et $t_1 \xleftarrow{*}_{R,\mathcal{C}} t_2$.
- Si $t_1 \xleftarrow{*}_{R,\mathcal{C}} t \longleftrightarrow_{E_1^>} t_2$ alors $t_1 \xleftarrow{*}_{R,\mathcal{C}} t \sim_{\mathcal{C}} t_2$ car $\longleftrightarrow_{E_1^>} \subseteq \sim_{\mathcal{C}}$, puis $t_1 \xrightarrow{*}_{R,\mathcal{C}} \circ \sim_{\mathcal{C}} \circ \xleftarrow{*}_{R,\mathcal{C}} t_2$ par cohérence de $\sim_{\mathcal{C}}$ et finalement $t_1 \xrightarrow{*}_{R,\mathcal{C}} \circ \xleftrightarrow{*}_{R_c} \circ \xleftarrow{*}_{R,\mathcal{C}} t_2$ car $\sim_{\mathcal{C}} \subseteq \xleftrightarrow{*}_{R_c}$.

□

Proposition 7.14 *Soit $p \in \mathcal{P}_0$. $L_{R,\mathcal{C}} \models p(t_1, \ldots, t_n) \Leftrightarrow L_{\mathcal{C}} \models p(t_1 \downarrow_{R,\mathcal{C}}, \ldots, t_n \downarrow_{R,\mathcal{C}})$.*

Preuve : Il s'agit de montrer que si $p_{\mathcal{M}}(t_1, \ldots, t_n)$ est vrai alors $p_{\mathcal{M}}(t_1\sigma, \ldots, t_n\sigma)$ est encore vrai pour n'importe quelle substitution σ normalisée suivant $\rightarrow_{R,\mathcal{C}}$ (admissible ou non). Nous avons $(t\sigma) \downarrow_{R,\mathcal{C}} \sim_{\mathcal{C}} (t \downarrow_{R,\mathcal{C}} \sigma) \downarrow_{R,\mathcal{C}}$ puis

$$(t\sigma \downarrow_{R,\mathcal{C}})^{\pi_0} =_{\mathcal{A}} ((t \downarrow_{R,\mathcal{C}} \sigma) \downarrow_{R,\mathcal{C}})^{\pi_0} =_{\mathcal{A}} (((t \downarrow_{R,\mathcal{C}} \sigma) \downarrow_{R_c}) \downarrow_{R,\mathcal{C}})^{\pi_0}$$

d'après le lemme 7.3, et

$$\begin{array}{rcl} (t\sigma \downarrow_{R,\mathcal{C}})^{\pi_0} & =_{\mathcal{A}} & (((t \downarrow_{R,\mathcal{C}} \sigma) \downarrow_{R_c}) \downarrow_{R,\mathcal{C}})^{\pi_0} \\ & = & ((t \downarrow_{R,\mathcal{C}} \sigma) \downarrow_{R_c})^{\pi_0} (\pi^{-1} \downarrow_{R,\mathcal{C}})^{\pi_0} \\ & =_{\mathcal{A}} & (t \downarrow_{R,\mathcal{C}} \sigma)^{\pi_0} (\pi^{-1} \downarrow_{R,\mathcal{C}})^{\pi_0} \end{array}$$

D'autre part,

$$(t\sigma)^{\pi_0} = t^{\pi_0}\sigma^{\pi_0}\{s^{\pi_0} \mapsto (s\sigma)^{\pi_0}\}_{s^{\pi_0} \in \mathcal{V}(t^{\pi_0})}$$

et par conséquent,

$$(t_i\sigma \downarrow_{R,\mathcal{C}})^{\pi_0} =_{\mathcal{A}} (t_i \downarrow_{R,\mathcal{C}})^{\pi_0} \sigma^{\pi_0} \xi (\pi^{-1} \downarrow_{R,\mathcal{C}})^{\pi_0}$$

où $\xi = \{s^{\pi_0} \mapsto (s\sigma)^{\pi_0}\}_{s^{\pi_0} \in \cup_i^n \mathcal{V}(t_i^{\pi_0})}$.

On peut en conclure que

$$L_{\mathcal{A}} \models p((t_1\sigma \downarrow_{R,\mathcal{C}})^{\pi_0}, \ldots, (t_n\sigma \downarrow_{R,\mathcal{C}})^{\pi_0}) \text{ si } L_{\mathcal{A}} \models p((t_1 \downarrow_{R,\mathcal{C}})^{\pi_0}, \ldots, (t_n \downarrow_{R,\mathcal{C}})^{\pi_0})$$

c'est-à-dire

$$L_{\mathcal{C}} \models p(t_1\sigma \downarrow_{R,\mathcal{C}}, \ldots, t_n\sigma \downarrow_{R,\mathcal{C}}) \text{ si } L_{\mathcal{C}} \models p(t_1 \downarrow_{R,\mathcal{C}}, \ldots, t_n \downarrow_{R,\mathcal{C}}).$$

□

Proposition 7.15 *Soit $p \in \mathcal{P}_0$. Si $L_{\mathcal{C}} \models p(t_1, \ldots, t_n)$ alors $L_{R,\mathcal{C}} \models p(t_1, \ldots, t_n)$.*

Preuve : D'après le lemme 7.3, $t \downarrow_{R,\mathcal{C}} =_{E_0 \cup E_1} (t \downarrow_{R_c}) \downarrow_{R,\mathcal{C}}$ et donc

$$(t \downarrow_{R,\mathcal{C}})^{\pi_0} =_{\mathcal{A}} ((t \downarrow_{R_c}) \downarrow_{R,\mathcal{C}})^{\pi_0} = (t \downarrow_{R_c})^{\pi_0} (\pi^{-1} \downarrow_{R,\mathcal{C}})^{\pi_0} =_{\mathcal{A}} t^{\pi_0} (\pi^{-1} \downarrow_{R,\mathcal{C}})^{\pi_0}$$

car E_1 est non-effondrante. Par conséquent, si $L_{\mathcal{C}} \models p(t_1, \ldots, t_n)$ alors $L_{\mathcal{A}} \models p(t_1^{\pi_0}, \ldots, t_n^{\pi_0})$, $L_{\mathcal{A}} \models p((t_1 \downarrow_{R,\mathcal{C}})^{\pi_0}, \ldots, (t_n \downarrow_{R,\mathcal{C}})^{\pi_0})$, $L_{\mathcal{C}} \models p(t_1 \downarrow_{R,\mathcal{C}}, \ldots, t_n \downarrow_{R,\mathcal{C}})$ et $L_{R,\mathcal{C}} \models p(t_1, \ldots, t_n)$. □

7.6.2 Correction

La surréduction contrainte est définie sur des formules contraintes appelées *buts*, de la forme $(\exists X, p^?(t_1, \ldots, t_n) \parallel S)$ où X est un ensemble de variables existentiellement quantifiées, $p^?(t_1, \ldots, t_n)$ une contrainte atomique telle que $p \in \mathcal{P}_0$ et S une contrainte de $L_{\mathcal{C}}$.

Définition 7.26 *L'ensemble des solutions modulo $\sim_{R,\mathcal{C}}$ d'un but $G = (\exists X, p^?(t_1, \ldots, t_n) \parallel S)$ est défini par*

$$SOL_{R,\mathcal{C}}(G) = \{\sigma_{|\mathcal{V}(G)\setminus X} \mid \sigma \in SS^*_{\mathcal{C}}(S) \text{ et } L_{R,\mathcal{C}} \models p(t_1\sigma, \ldots, t_n\sigma)\}.$$

L'ensemble des solutions modulo $\sim_{\mathcal{C}}$ d'un but $G = (\exists X, p^?(t_1, \ldots, t_n) \parallel S)$ est défini par

$$SOL_{\mathcal{C}}(G) = \{\sigma_{|\mathcal{V}(G)\setminus X} \mid \sigma \in SS^*_{\mathcal{C}}(S \wedge p^?(t_1, \ldots, t_n))\}.$$

Proposition 7.16 *$SOL_{\mathcal{C}}(G) \subseteq SOL_{R,\mathcal{C}}(G)$.*

Preuve : D'après la proposition 7.15. □

Nous allons voir que toute solution modulo $\sim_{\mathcal{C}}$ d'un but G' obtenu par surréduction est encore une solution modulo $\sim_{R,\mathcal{C}}$ de G.

Par souci de commodité, on se restreint aux contraintes élémentaires dont le prédicat est binaire, comme par exemple $=, \neq, <$.

Définition 7.27 *Le but $(\exists X, p^?(g, d) \parallel S)$ est* surréduit *avec la règle $(l \to r \parallel c)$ en*

$$(\exists X \cup \mathcal{V}(l \to r \parallel c), \quad p^?(g[r]_\omega, d) \parallel S \wedge c \wedge (g_{|\omega} =^?_{\mathcal{C}} l))$$

si $S \wedge c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$ est satisfaisable.

La relation entre surréduction et réécriture dans ce contexte est exprimée par le résultat suivant.

Proposition 7.17 *Si le but $(\exists X, p^?(g, d) \parallel S)$ est surréduit avec $(l \to r \parallel c)$ en $(\exists X \cup \mathcal{V}(l \to r \parallel c), \ p^?(g[r]_\omega, d) \parallel S')$, où $S' = S \wedge c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$, alors pour toute solution α de S' modulo $\sim_{\mathcal{C}}$, $g\alpha \to_{R,\mathcal{C}} g[r]\alpha$ avec la règle $(l \to r \parallel c)$.*

Preuve : Puisque α est solution de S', $g_{|\omega}\alpha \sim_{\mathcal{C}} l\alpha$ et α est solution de c modulo $\sim_{\mathcal{C}}$. □

Proposition 7.18 *Soit $G = (\exists X, p^?(g, d) \parallel S)$ et $G' = (\exists X', p^?(g', d') \parallel S')$ t.q. G est surréduit avec $(l \to r \parallel c)$ en G'. Alors $SOL_{R,\mathcal{C}}(G') \subseteq SOL_{R,\mathcal{C}}(G)$.*

Preuve : On peut supposer sans perte de généralité que
$G' = (\exists X \cup \mathcal{V}(l \to r \parallel c), p^?(g[r]_\omega, d) \parallel S \wedge c \wedge (g_{|\omega} =^?_{\mathcal{C}} l))$. Toute solution de $SOL_{R,\mathcal{C}}(G')$ est la restriction à $\mathcal{V}(G')\setminus X' = \mathcal{V}(G)\setminus X$ d'une substitution α telle que $\alpha \in SS^*_{\mathcal{C}}(S')$ et $L_{R,\mathcal{C}} \models p(g'\alpha, d\alpha)$. La substitution α est aussi une solution de S modulo $\sim_{\mathcal{C}}$ puisque $S' = S \wedge c \wedge (g_{|\omega} =^?_{\mathcal{C}} l)$. Ensuite, comme $\alpha \in SS^*_{\mathcal{C}}(c)$, $g\alpha \to_{R,\mathcal{C}} g'\alpha$. Donc $g\alpha \downarrow_{R,\mathcal{C}} = g'\alpha \downarrow_{R,\mathcal{C}}$ et $L_{R,\mathcal{C}} \models p(g\alpha, d\alpha)$.

Pour le prédicat d'égalité, nous n'avons pas à supposer la convergence de R modulo $\sim_{\mathcal{C}}$ puisque dans ce cas $g\alpha \to_{R,\mathcal{C}} g'\alpha \sim_{R,\mathcal{C}} d\alpha$. □

La proposition suivante prouve la correction de la surréduction.

Proposition 7.19 *S'il existe une dérivation par surréduction*

$$G_0 = (\exists \emptyset, \ p^?(g_0, d_0) \parallel \top) \rightsquigarrow G_1 \rightsquigarrow \ldots \rightsquigarrow G_n = (\exists X_n, p^?(g_n, d_n) \parallel S_n)$$

alors $SOL_{\mathcal{C}}(G_n) \subseteq SOL_{R,\mathcal{C}}(G_0)$.

Preuve : D'après les propositions 7.16 et 7.18. □

7.6.3 Complétude

Il s'agit de prouver que pour toute solution $\sigma \in SOL_{R,\mathcal{C}}(\exists\emptyset, p^?(g_0, d_0) \parallel \top)$, il existe une dérivation par surréduction

$$G_0 = (\exists\emptyset,\ p^?(g_0, d_0) \parallel \top) \rightsquigarrow G_1 \rightsquigarrow \ldots \rightsquigarrow G_n = (\exists X_n, p^?(g_n, d_n) \parallel S_n)$$

et une substitution θ telle que $\sigma =_{\mathcal{C}}^{\mathcal{V}(G_0)} \theta$ et $\theta \in SOL_{\mathcal{C}}(G_n)$.

Lemme 7.4 *Soit $G_i = (\exists X_i, p^?(g_i, d_i), S_i)$ un but et μ_i une substitution définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_i)$. Si $\mu_i \in SS^*_{\mathcal{C}}(S_i)$ et $p^?(g_i, d_i)\mu_i$ réductible par $\rightarrow_{R,\mathcal{C}}$, alors il existe une étape de réduction $(p^?(g_i, d_i))\mu_i \rightarrow_{R,\mathcal{C}} p^?(u, v)$, une étape de surréduction*

$$G_i = (\exists X_i, p^?(g_i, d_i), S_i) \rightsquigarrow G_{i+1} = (\exists X_{i+1}, p^?(g_{i+1}, d_{i+1}), S_{i+1})$$

et une substitution normalisée μ_{i+1} définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_{i+1})$, telle que $\mu_{i+1} \in SS^*_{\mathcal{C}}(S_{i+1})$, $p^?(g_{i+1}, d_{i+1})\mu_{i+1} \sim_{\mathcal{C}} p^?(u, v)$ et $\mu_{i+1} =^{\mathcal{V}(G_i)\setminus X_i} \mu_i$

Preuve : Si $(p^?(g_i, d_i))\mu_i \rightarrow_{R,\mathcal{C}} p^?(u, v)$, il existe une position ω dans $g_i\mu_i$ (ou $d_i\mu_i$), une règle $(l \rightarrow r \parallel c) \in R$, et une substitution α telle que $(g_i\mu_i)_{|\omega} \sim_{\mathcal{C}} l\alpha$, $\alpha \in SS^*_{\mathcal{C}}(c)$, $u = (g_i\mu_i)[r\alpha]_\omega$. Puisque μ_i est normalisée pour $\rightarrow_{R,\mathcal{C}}$, on a $(g_i\mu_i)_{|\omega} = g_{i|\omega}\mu_i$. Donc $g_{i|\omega}$ et l sont unifiables modulo $\sim_{\mathcal{C}}$ et $c \wedge g_{i|\omega} =^?_{\mathcal{C}} l$ est satisfaisable modulo $\sim_{\mathcal{C}}$.

Il existe une stratégie de réduction de l'intérieur vers l'extérieur ("innermost") à filtrage normalisant et l'on peut donc supposer sans perte de généralité que α est définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(l \rightarrow r \parallel c)$, avec $(\mathcal{V}(G_i) \cup \mathcal{VR}an(\mu_i)) \cap \mathcal{V}(l \rightarrow r \parallel c) = \emptyset$. La substitution μ_{i+1} définie sur $\mathcal{V}(G_{i+1})$ par $\mu_{i+1} =^{\mathcal{V}(G_i)} \mu_i$, $\mu_{i+1} =^{\mathcal{V}(l=r \parallel c)} \alpha$ est normalisée pour $\rightarrow_{R,\mathcal{C}}$. Donc $\mu_{i+1} = \mu_i\alpha \in SS^*_{\mathcal{C}}(S_{i+1})$ et $(p^?(g_i[r]_\omega, d_i))\mu_{i+1} \sim_{\mathcal{C}} p^?(u, v)$. □

Lemme 7.5 *Soit $G_i = (\exists X_i, p^?(g_i, d_i), S_i)$ un but et μ_i une substitution définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_i)$. Si $\mu_i \in SS^*_{\mathcal{C}}(S_i)$ et $(p^?(g_i, d_i))\mu_i$ réductible par $\rightarrow_{R/\mathcal{C}}$, alors il existe une dérivation par réduction*

$$(p^?(g_i, d_i))\mu_i \xrightarrow{+}_{R/\mathcal{C}} p^?(s, t)$$

avec s, t irréductibles par $\rightarrow_{R/\mathcal{C}}$, une dérivation par surréduction

$$G_i = (\exists X_i,\ p^?(g_i, d_i) \parallel S_i) \rightsquigarrow G_{i+1} \rightsquigarrow \ldots \rightsquigarrow G_k = (\exists X_k, p^?(g_k, d_k) \parallel S_k)$$

et une substitution normalisée μ_k définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_k)$, telle que $\mu_k \in SS^*_{\mathcal{C}}(S_k)$, $p^?(g_k, d_k)\mu_k \sim_{\mathcal{C}} p^?(s, t)$ et $\mu_k =^{\mathcal{V}(G_i)\setminus X_i} \mu_i$.

Preuve : Par induction nœthérienne sur $\rightarrow_{R/\mathcal{C}}$.

Si $p^?(g_i, d_i)\mu_i$ est réductible pour $\rightarrow_{R/\mathcal{C}}$ alors il l'est aussi pour $\rightarrow_{R,\mathcal{C}}$ car $\rightarrow_{R,\mathcal{C}}$ est cohérent avec $\sim_{\mathcal{C}}$ et $\rightarrow_{R/\mathcal{C}}$ termine. Donc $p^?(g_i, d_i)\mu_i$ est réductible par $\rightarrow_{R,\mathcal{C}}$ et d'après le lemme 7.4, il existe une étape de réduction $p^?(g_i, d_i)\mu_i \rightarrow_{R,\mathcal{C}} p^?(u, v)$, une étape de surréduction $G_i \rightsquigarrow G_{i+1}$ et une substitution μ_{i+1} définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_{i+1})$ telle que $\mu_{i+1} \in SS^*_{\mathcal{C}}(S_{i+1}), \mu_{i+1} =^{\mathcal{V}(G_i)\setminus X_i} \mu_i$ et $p^?(g_i, d_i)\mu_i \rightarrow_{R,\mathcal{C}} p^?(u, v) \sim_{\mathcal{C}} p^?(g_{i+1}, d_{i+1})\mu_{i+1}$. Si $p^?(u, v)$ est irréductible par $\rightarrow_{R/\mathcal{C}}$ alors on prend $k =$

$i+1$. Sinon $p^?(g_{i+1}, d_{i+1})\mu_{i+1}$ est réductible par $\rightarrow_{R/\mathcal{C}}$ et par hypothèse d'induction, il existe une dérivation par réduction

$$p^?(g_{i+1}, d_{i+1})\mu_{i+1} \xrightarrow{*}_{R/\mathcal{C}} p^?(s', t')$$

avec s', t' irréductibles par $\rightarrow_{R/\mathcal{C}}$, une dérivation par surréduction

$$G_{i+1} \leadsto \ldots \leadsto G_k = (\exists X_k, p^?(g_k, d_k) \parallel S_k),$$

et une substitution μ_k définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_k)$ telle que $\mu_k \in SS^*_{\mathcal{C}}(S_k), p^?(g_k, d_k)\mu_k \sim_{\mathcal{C}} p^?(s', t')$ et $\mu_k =^{\mathcal{V}(G_{i+1})\setminus X_{i+1}} \mu_{i+1}$. Puisque $\mathcal{V}(G_{i+1})\setminus X_{i+1} = \mathcal{V}(G_i)\setminus X_i$ et $\mu_{i+1} =^{\mathcal{V}(G_i)\setminus X_i} \mu_i$, on a $\mu_k =^{\mathcal{V}(G_i)\setminus X_i} \mu_i$. □

Théorème 7.6 *Soient $G_0 = (\exists\emptyset, p^?(g_0, d_0), \top)$ un but et μ_0 une substitution définie et normalisée pour $\rightarrow_{R,\mathcal{C}}$ sur $\mathcal{V}(G_0)$. Si $\mu_0 \in SOL_{R,\mathcal{C}}(G_0)$, alors il existe une dérivation par surréduction*

$$G_0 = (\exists\emptyset,\ p^?(g_0, d_0) \parallel \top) \leadsto G_1 \leadsto \ldots \leadsto G_n = (\exists X_n, p^?(g_n, d_n) \parallel S_n)$$

telle que $\mu_0 \in SOL_{\mathcal{C}}(G_n)$.

Preuve : Si $p^?(g_0, d_0)\mu_0$ est irréductible par $\rightarrow_{R,\mathcal{C}}$ alors $L_{R,\mathcal{C}} \models p(g_0, d_0)\mu_0$ implique $L_{\mathcal{C}} \models p(g_0, d_0)\mu_0$ par définition de $L_{R,\mathcal{C}}$.

Si $p^?(g_0, d_0)\mu_0$ est réductible par $\rightarrow_{R,\mathcal{C}}$ alors il l'est aussi par $\rightarrow_{R/\mathcal{C}}$. D'après le lemme 7.5, il existe une dérivation par réduction

$$p^?(g_0, d_0)\mu_0 \xrightarrow{*}_{R/\mathcal{C}} p^?(s, t)$$

avec s, t irréductibles par $\rightarrow_{R/\mathcal{C}}$, une étape de surréduction

$$G_0 = (\exists\emptyset,\ p^?(g_0, d_0) \parallel \top) \leadsto G_1 \leadsto \ldots \leadsto G_n = (\exists X_n, p^?(g_n, d_n) \parallel S_n)$$

et une substitution normalisée μ_n telle que $\mu_n \in SS^*_{\mathcal{C}}(S_n)$, $p^?(g_n, d_n)\mu_n \sim_{\mathcal{C}} p^?(s, t)$ et $\mu_0 =^{\mathcal{V}(G_0)} \mu_n$. Les termes s et t sont normalisés pour $\rightarrow_{R,\mathcal{C}}$ et $L_{R,\mathcal{C}} \models p(s, t)$ par hypothèse sur μ_0. Par conséquent, $L_{\mathcal{C}} \models p(s, t) \sim_{\mathcal{C}} p(g_n, d_n)\mu_n$ et $\mu_0 = (\mu_n)_{|\mathcal{V}(G_n)\setminus X_n} \in SOL_{\mathcal{C}}(G_n)$. □

La surréduction fournit donc une procédure complète de résolution de contraintes pour le langage $L_{R,\mathcal{C}}$.

Exemple 7.4 *Considérons le langage de contraintes $L_{\mathcal{B}}$ engendré par la Σ_0-structure $\mathcal{B}$ des booléens avec*

$$\Sigma_0 = (\{Bool\}, \{\wedge, \vee : Bool, Bool \rightarrow Bool, {}^- : Bool \rightarrow Bool, 0, 1 :\rightarrow Bool\}, \{=, \neq, <\})$$

et l'ensemble $\mathcal{X}_0$ des variables de base $\{x, y, z, v : Bool, \ldots\}$. Considérons la spécification (incomplète) $(\Sigma, R, AC(xor), L_{\mathcal{B}})$ où:

$$R = \begin{cases} xor(x, x) \rightarrow 0 \\ xor(x, 1) \rightarrow \overline{x} \end{cases}$$

On cherche à résoudre le but

$$G_0 = (\exists\emptyset,\ xor(xor(s,t),0) \neq^? xor(1,v) \parallel \top)$$

où s, t sont des termes de base. Par surréduction de G_0, on obtient

$$\begin{aligned} G_0 &\rightsquigarrow (\exists x,\ xor(0,0) \neq^? xor(1,v) \parallel xor(s,t)=^? xor(x,x)) \\ &\rightsquigarrow (\exists x,y,\ 0 \neq^? xor(1,v) \parallel xor(s,t)=^? xor(x,x) \wedge xor(0,0)=^? xor(y,y)) \\ &\rightsquigarrow G_3 = (\exists x,y,z,\ 0 \neq^? \overline{z} \parallel xor(s,t)=^? xor(x,x) \wedge xor(0,0)=^? xor(y,y) \\ &\quad \wedge xor(1,v)=^? xor(z,1)) \end{aligned}$$

tel que

$$\begin{aligned} SOL_{\mathcal{C}}(G_3) &= SS^*_{\mathcal{C}}(\exists x,y,z,\ 0 \neq^? \overline{z} \wedge xor(s,t)=^? xor(x,x) \wedge xor(0,0)=^? xor(y,y) \\ &\quad \wedge xor(1,v)=^? xor(z,1)) \\ &= SS^*_{\mathcal{C}}(\exists x,y,z,\ 0 \neq^? \overline{z} \wedge x=^? s \wedge s=^? t \wedge y=^? 0 \wedge v=^? z) \\ &= SS^*_{\mathcal{C}}(0 \neq^? \overline{v} \wedge s=^? t) \\ &= SS_{\mathcal{B}}(0 \neq^? \overline{v} \wedge s=^? t) \end{aligned}$$

car v, s, t sont ici des termes de base. La solution principale de $v =^?_{\mathcal{B}} 0 \wedge s =^?_{\mathcal{B}} t$ est solution principale du but G_0 car G_3 est le seul but se dérivant de G_0 par surréduction contrainte.

Conclusion

Nous nous sommes consacrés au cours de ce travail à l'étude de la combinaison des résolutions de contraintes dans la double intention d'étendre les techniques connues pour l'unification et de les appliquer au cadre plus général de la démonstration automatique avec contraintes. Nos travaux fournissent en cela des algorithmes puissants permettant d'étendre la combinaison à des mélanges de théories équationnelles non disjointes mais aussi à des problèmes spécifiques comme le filtrage, le problème du mot ou l'élimination de constante. Nous proposons plus généralement un modèle pour la combinaison de résolveurs de contraintes symboliques qui s'applique à la combinaison de structures concrètes non disjointes ou à l'intégration d'une structure concrète au raisonnement par contraintes. Il résulte de ce dernier point une procédure de résolution héritant des deux formes d'incrémentalité que sont surréduction et combinaison.

Apports

L'étude du mélange des théories non disjointes nous a conduit à introduire la notion de théories décomposables. Il s'agit de l'union de théories équationnelles non nécessairement disjointes pouvant se représenter par un système de réécriture combiné, convergent et qui est formé de règles dont la tête à elle seule détermine l'appartenance à l'une ou l'autre des théories composantes. Une théorie décomposable est alors une extension conservative de ses composantes pour laquelle même la résolution est préservée. C'est l'une des propriétés essentielles dont il faut s'assurer pour réutiliser un algorithme d'unification disponible dans l'une des composantes.

Nous présentons deux algorithmes d'unification dans des théories décomposables:

- Un algorithme de résolution uniquement qui suppose l'existence d'un ensemble fini de contextes partagés en tête des égalités.
- Un algorithme de décision et résolution qui suppose l'existence d'un entier, appelé facteur de partage, correspondant au nombre de symboles partagés qu'il est nécessaire de faire apparaitre dans les égalités.

Ces hypothèses abstraites permettent de résoudre le problème important de la combinaison des solutions qui vient se greffer à celui de la construction d'une extension conservative. A l'intersection de ces hypothèses se trouve le cas du mélange de théories partageant les constantes uniquement [Rin92] qui est sûrement celui qui comprend le plus grand nombre de théories.

L'introduction des théories partiellement linéaires a été rendue nécessaire par la spécialisation des techniques de combinaison à des problèmes spécifiques comme le filtrage. La spécificité d'un problème n'est pas en général préservée par les transformations préalables à la réutilisation des algorithmes de chaque composante. Il est fort probable alors d'avoir à considérer des problèmes qui ne sont plus du ressort des algorithmes spécialisés. L'intérêt des théories partiellement linéaires est justement de pouvoir laisser tomber sans perte de généralité les problèmes qui ne peuvent pas être résolus par les algorithmes spécialisés disponibles. Cette classe de théories est intéressante à plus d'un titre et en particulier pour le filtrage et l'élimination de constante qui peuvent s'effectuer sans faire appel à l'unification. L'approche qui consiste à d'abord résoudre chaque problème pur séparément puis à régler les éventuels conflits inter-théories se trouve alors pleinement justifiée.

Les résultats concernant le filtrage dans le mélange de théories partiellement linéaires disjointes s'obtiennent grâce à un algorithme de combinaison du filtrage étendu, un problème composé de filtre-équations et équations résolues. L'algorithme de combinaison que nous proposons s'applique au problème de décision, ce que ne permettait pas l'algorithme de combinaison du filtrage donné par T. Nipkow dans les théories régulières.

L'étude de la combinaison du filtrage a directement pour conséquence d'avoir à s'intéresser à la combinaison du problème du mot. Le principe de décision est le même que celui utilisé par réécriture à ceci près qu'on remplace la notion trop théorique de forme normale suivant le système de réécriture combiné par celle, plus pratique car calculable, de forme réduite par couches. Un problème du mot hétérogène se résout par le calcul des formes réduites par couches suivi d'un test d'égalité dans la théorie de tête grâce à l'abstraction des sous-termes étrangers eux-aussi en forme réduite par couches.

La notion de forme réduite par couches est étendue aux théories décomposables mais sa construction utilise non plus seulement les algorithmes de décision du problème du mot mais aussi les algorithme de filtrage de chaque théorie composante. Les résultats concernant problème du mot et filtrage s'étendent en conséquence aux théories décomposables. Il est remarquable cependant que le filtrage dans les théories décomposables régulières et non effondrantes n'ait pas besoin de l'existence d'un facteur de partage car le problème de la combinaison des solutions se réduit dans ce cas à l'utilisation des algorithmes de décision du problème du mot. Il s'agit là encore d'un résultat plus fort que celui valable pour l'unification.

Après l'étude de ces problèmes équationnels spécifiques, nous avons cherché à définir un modèle pour la combinaison d'algorithmes résolvant de manière symbolique d'autres contraintes que les simples contraintes équationnelles. Dans la pratique, plusieurs résolveurs de contraintes sont en effet amenés à coexister dans un même contexte (langage de programmation ou démonstrateur de théorèmes). Il est alors naturel de vouloir les combiner afin de les faire résoudre des problèmes éventuellement hétérogènes et donc plus généraux que ceux auxquels ils étaient initialement destinés. Nous montrons que la résolution de contraintes symbolique dans le langage combiné préservant la validité des formules atomiques de chaque composante s'effectue par combinaison des résolveurs de contraintes avec restriction linéaire. Nous avons montré comment prendre en compte cette restriction linéaire dans le cas des algèbres primales grâce à un algorithme d'élimination de constante obtenu par extension d'un résolveur de contraintes [KR92] dans diverses structures à domaine fini: algèbres finies ou booléennes, pseudo-booléens ou plus généralement

pseudo-finis. Cet algorithme d'élimination utilise l'algorithme de résolution de contraintes intégré au logiciel UNIF dont l'une des caractéristiques est de représenter les termes par des graphes de décision [Bry86] pour lesquels les nœuds sont des variables et les arcs sont étiquetés par des valeurs du domaine. L'objet de la résolution consiste alors à représenter une contrainte sous la forme la plus compacte possible en minimisant le nombres de variables. Cela permet ensuite d'améliorer sensiblement la résolution incrémentale des contraintes.

Nous disposons donc de résolveurs de contraintes qui peuvent se combiner pour s'appliquer au mélange de deux algèbres finies ayant une intersection non vide grâce aux constantes partagées.

Une autre application est celle qui consiste à intégrer une structure prédéfinie au sein d'une spécification équationnelle introduisant propriétés et nouveaux symboles à l'aide de règles contraintes ou axiomes. Une procédure hiérarchisée de résolution de contraintes est présentée pour une extension conservative de la structure prédéfinie. Sa principale caractéristique est d'être fondée à la fois sur des techniques de surréduction et sur des techniques de combinaison dans le cadre de la construction incrémentale d'algorithmes de résolution de contraintes.

Cette application ouvre la voie à l'utilisation concrète de la combinaison en programmation logique.

Perspectives

Les algorithmes d'unification et filtrage que nous proposons s'appliquent à un large éventail de théories équationnelles. En contre-partie, il n'est pas possible à l'heure actuelle de décider si une théorie est décomposable ou partiellement linéaire, si elle est bornée ou admet un facteur de partage. Ce travail doit s'effectuer à la main avant de songer à combiner les algorithmes d'unification ou de filtrage. Il serait donc intéressant de trouver des critères suffisants mais décidables permettant de conclure qu'une théorie satisfait l'une des propriétés requises par nos algorithmes de combinaison. Dans le même ordre idée, on peut se demander:

- Dans quelle mesure l'union de deux théories est-elle décomposable?

  Nous savons déjà que l'union de théories disjointes, ou faiblement partagées, ou partageant seulement des constantes "irréductibles" est décomposable. La question qui se pose maintenant est de savoir si l'ordre de réduction sur les termes (clos) hétérogènes dont nous avons besoin peut être construit de façon modulaire à partir d'ordres de réduction fournis par chaque théorie composante.

- Sous quelles conditions les notions de théorie bornée ou de facteur de partage sont-elles modulaires?

  Un élément de réponse a déjà été apporté en montrant que l'existence d'un facteur de partage est modulaire pour les théories faiblement partagées.

- Existe-t-il des critères simples sur les axiomes d'une théorie, autres que la linéarité ou la régularité avec non-effondrement, permettant de s'assurer qu'une théorie est partiellement linéaire?

L'hypothèse de décomposabilité bien que pouvant capturer une forme de mélange non disjoint impose cependant une sérieuse restriction sur les symboles partagés puisqu'ils doivent être ω-libres. Les problèmes équationnels composés des seuls symboles partagés sont alors résolus dans la théorie vide. Il semble possible d'autoriser d'autres théories que la théorie vide, ce qui suppose que les symboles partagés aient la même propriété dans les deux théories composantes. L'ordre de simplification total sur les termes clos hétérogènes doit être en conséquence défini modulo la théorie partagée, celle vérifiée par les symboles partagés. Nous pensons plus particulièrement aux symboles partagés commutatifs ou associatifs-commutatifs.

La construction d'une extension conservative préservant les résolutions de chaque théorie doit encore pouvoir être réalisée d'une manière totalement différente, basée, elle, sur des critères concernant les paires critiques. Mais il reste alors le problème de la combinaison des solutions qui est en principe disjoint de celui de la construction d'une extension conservative, mais qui n'est pas pour autant à sous-estimer.

Les théories partiellement linéaires semblent être les seules théories adaptées à la combinaison du filtrage et de la décision du filtrage. Nous montrons d'ailleurs un résultat qui va dans ce sens puisque l'on prouve que l'appel à un algorithme d'unification ne peut être évité pour les théories qui ne sont pas partiellement linéaires. Il existe pourtant un algorithme de combinaison du filtrage pour les théories régulières, qui ne sont pas toujours des théories partiellement linéaires. Il n'est pas étonnant au vu du chapitre 2 d'obtenir des algorithmes de combinaison fondamentalement différents qui dépendent fortement des approches par résolution ou décision qui furent prises. Celui proposé par T. Nipkow repose entièrement sur une propriété valable uniquement pour le filtrage dans les théories régulières et qui consiste à n'engendrer que des solutions closes. Cet algorithme ne peut donc être, lui non plus, étendu à une plus large classe de théories.

La combinaison des résolveurs de contraintes non symboliques mais retournant uniquement les solutions closes est encore un problème ouvert. F. Baader et K. Schulz ont montré récemment dans le cadre de la disunification que le problème de l'existence d'une solution close ne se prêtait pas à la modularité, puisque l'existence d'une solution close hétérogène ne pouvait pas se décider grâce à la combinaison usuelle des algorithmes de décision du même problème dans les théories composantes. Ainsi, la combinaison des résolveurs de contraintes dans les algèbres initiales $\mathcal{T}(\mathcal{F}_1)/=_{E_1}$ et $\mathcal{T}(\mathcal{F}_2)/=_{E_2}$ à la manière de ce qui est fait dans les algèbres libres ne permet pas d'obtenir un résolveur de contraintes dans $\mathcal{T}(\mathcal{F}_1 \cup \mathcal{F}_2)/=_{E_1 \cup E_2}$. Un nouveau modèle reste donc à définir.

Un autre problème restant encore ouvert est celui de savoir s'il existe une méthode permettant d'obtenir directement et pour n'importe quelle théorie, un algorithme d'unification générale (avec symboles libres) à partir d'un algorithme d'unification avec constantes. Nous pensons et espérons, comme tous ceux qui se sont intéressés au problème de combinaison, que cette méthode n'existe pas. Dans le cas contraire, toutes les méthodes de combinaison connues à ce jour concernant les théories quelconques deviendraient obsolètes. Les premiers éléments de réponse à ce sujet concernant la décidabilité d'une seule équation (au lieu d'un système) laissent à penser que l'on peut dormir tranquille.

Sur un plan pratique et à plus court terme, les algorithmes décrits au cours de ce document doivent être incorporés à une bibliothèque de résolutions de contraintes et devenir ainsi un outil de résolution pour des structures hiérarchisées complexes. Les similitudes entre les différents algorithmes de combinaison, puisque relevant des mêmes principes, per-

mettent là aussi de pratiquer une démarche très modulaire lors de l'implémentation. C'est ce que nous avons commencé à faire pour la bibliothèque de résolutions de contraintes UNIF et que nous pensons achever dans un avenir proche.

Bibliographie

[AB94] Avenhaus (J.) et Becker (K.). – Operational specifications with built-in's. *In : Proceedings of STACS-94*. – 1994. to appear.

[Baa89] Baader (F.). – Unification in commutative theories. *Journal of Symbolic Computation*, vol. 8, n° 5, 1989, pp. 479–498.

[BCL87] Ben Cherifa (A.) et Lescanne (P.). – Termination of rewriting systems by polynomial interpretations and its implementation. *Science of Computer Programming*, vol. 9, n° 2, octobre 1987, pp. 137–160.

[BDH86] Bachmair (L.), Dershowitz (N.) et Hsiang (J.). – Orderings for equational proofs. *In : Proceedings 1st IEEE Symposium on Logic in Computer Science, Cambridge (Mass., USA)*. pp. 346–357. – IEEE, 1986.

[BES+90] Büttner (W.), Estenfeld (K.), Schmid (R.), Schneider (H.-A.) et Tiden (E.). – Symbolic constraint handling through unification in finite algebras. *Applicable Algebra in Engineering, Communication and Computation*, vol. 1, n° 2, 1990, pp. 97–118.

[BGW92] Bachmair (L.), Ganzinger (H.) et Waldmann (U.). – Theorem proving for hierarchic first-order theories. *In : Proceedings 3rd International Conference on Algebraic and Logic Programming, Volterra (Italy)*, éd. par Kirchner (H.) et Levi (G.). pp. 420–434. – Springer-Verlag, septembre 1992.

[BHSS90] Bürckert (H.-J.), Herold (A.) et Schmidt-Schauß (M.). – On equational theories, unification and (un)decidability. *In : Unification*, éd. par Kirchner (C.), pp. 69–116. – London, Academic Press, 1990.

[Bir35] Birkhoff (G.). – On the structure of abstract algebras. *Proceedings Cambridge Phil. Soc.*, vol. 31, 1935, pp. 433–454.

[BJSS90] Boudet (A.), Jouannaud (J.-P.) et Schmidt-Schauß (M.). – Unification in boolean rings and abelian groups. *In : Unification*, éd. par Kirchner (C.), pp. 267–296. – London, Academic Press, 1990.

[BM70] Birkhoff (G.) et MacLane (S.). – *Algèbre*. – Paris, Gauthier–Villard, 1970, *Cahiers Scientifiques*, volume I et II. Traduit de l'anglais par J. Weil.

[Boc91] Bockmayr (A.). – *Logic Programming with Pseudo-Boolean Constraints.* – Rapport technique n° MPI-I-91-227, Im Stadtwald, W6600 Saarbrücken, Germany, Max-Planck-Institut für Informatik, avril/décembre 1991.

[Boo47] Boole (G.). – *The Mathematical Analysis of Logic.* – New York, Macmillan, 1847. Reprinted: B. Blackwell, London, England, 1948.

[Bou90a] Boudet (A.). – *Unification dans les mélanges de théories équationelles.* – Orsay (France), Thèse de Doctorat d'Université, Université de Paris-Sud, février 1990.

[Bou90b] Boudet (A.). – Unification in a combination of equational theories: An efficient algorithm. *In : Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, éd. par Stickel (M. E.). – Springer-Verlag, juillet 1990.

[Bou92] Boudet (A.). – Unification in order-sorted algebras with overloading. *In : Proceedings 11th International Conference on Automated Deduction, Saratoga Springs (N.Y., USA)*, pp. 193–207. – 1992.

[BP85] Bachmair (L.) et Plaisted (D.). – Associative path orderings. *In : Proceedings 1st Conference on Rewriting Techniques and Applications, Dijon (France).* – Springer-Verlag, 1985.

[Bry86] Bryant (R. E.). – Graph-based algorithms for boolean function manipulation. *IEEE Transactions on computers*, vol. C-35, n° 8, août 1986, pp. 677–691.

[BS87] Büttner (W.) et Simonis (H.). – Embedding boolean expressions into logic programming. *Journal of Symbolic Computation*, vol. 4, n° 2, 1987, pp. 191–205.

[BS92] Baader (F.) et Schulz (K.). – Unification in the union of disjoint equational theories: Combining decision procedures. *In : Proceedings 11th International Conference on Automated Deduction, Saratoga Springs (N.Y., USA)*, pp. 50–65. – 1992.

[BS93] Baader (F.) et Schulz (K.). – Combination techniques and decision problems for disunification. *In : Proceedings 5th Conference on Rewriting Techniques and Applications, Montreal (Canada)*, éd. par Kirchner (C.). pp. 301–315. – Springer-Verlag, juin 1993.

[Bür87] Bürckert (H.-J.). – *Solving Disequations in Equational Theories.* – Rapport technique n° SEKI-Report SR-87-15, Universität Kaiserslautern, 1987.

[Bür90] Bürckert (H.-J.). – Matching — A special case of unification? *In : Unification*, éd. par Kirchner (C.), pp. 125–138. – London, Academic Press, 1990.

[Büt88] Büttner (W.). – Unification in finite algebras is unitary. *In : Proceedings 9th International Conference on Automated Deduction, Argonne (Ill., USA).* pp. 368–205. – Springer-Verlag, 1988.

[Büt89] Büttner (W.). – Implementing complex domains of application in an extended prolog system. *International Journal of General Systems*, vol. 15, 1989, pp. 129–139.

[CB83] Corbin (J.) et Bidoit (M.). – A rehabilitation of Robinson's unification algorithm. *In : Proceedings of 1983 IFIP Congress*, éd. par Pavon (R.). pp. 909–914. – Elsevier Science Publishers B. V. (North-Holland), 1983.

[CHJ92] Comon (H.), Haberstrau (M.) et Jouannaud (J.-P.). – Decidable problems in shallow equational theories (extended abstract). *In : Proceedings, Seventh Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, pp. 255–265. – Santa Cruz, California, 22–25 juin 1992.

[CL88] Comon (H.) et Lescanne (P.). – *Equational problems and disunification.* – Rapport technique n° 88-R-026, Centre de Recherche en Informatique de Nancy, 1988. To appear in the Journal of Symbolic Computation, special issue on unification, 89.

[Col90] Colmerauer (A.). – An introduction to Prolog III. *Communications of the Association for Computing Machinery*, vol. 33, n° 7, 1990, pp. 69–90.

[Com88] Comon (H.). – *Unification et disunification. Théories et applications.* – Thèse de Doctorat d'Université, Institut Polytechnique de Grenoble (France), 1988.

[Com91] Comon (H.). – Disunification: a survey. *In : Computational Logic. Essays in honor of Alan Robinson*, éd. par Lassez (J.-L.) et Plotkin (G.), chap. 9, pp. 322–359. – Cambridge (MA, USA), MIT Press, 1991.

[Dau89] Dauchet (M.). – Simulation of Turing machines by a left-linear rewrite rule. *In : Proceedings 3rd Conference on Rewriting Techniques and Applications, Chapel Hill (N.C., USA)*, éd. par Dershowitz (N.). pp. 109–120. – Springer-Verlag, avril 1989.

[Der82] Dershowitz (N.). – Orderings for term-rewriting systems. *Theoretical Computer Science*, vol. 17, 1982, pp. 279–301.

[Der87] Dershowitz (N.). – Termination of rewriting. *Journal of Symbolic Computation*, vol. 3, n° 1 & 2, 1987, pp. 69–116.

[DJ90a] Dershowitz (N.) et Jouannaud (J.-P.). – Rewrite systems. *In : Handbook of Theoretical Computer Science*, éd. par van Leeuwen (J.). – Elsevier Science Publishers B. V. (North-Holland), 1990.

[DJ90b] Dougherty (D.) et Johann (P.). – An improved general E-unification method. *In : Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, éd. par Stickel (M. E.). pp. 261–275. – Springer-Verlag, juillet 1990.

[DK93] Domenjoud (E.) et Klay (F.). – Shallow AC theories. *In : Proceedings of the 2nd CCL Workshop, La Escala (Spain)*, éd. par Orejas (F.). – septembre 1993.

[DS78] Downey (P.) et Sethi (R.). – Assignment commands with array references. *Journal of the Association for Computing Machinery*, vol. 25, n° 4, 1978.

[DSvH87] Dincbas (M.), Simonis (H.) et van Hentenryck (P.). – Extending equation solving and constraint handling in logic programming. *In : Proceedings of the Colloquium on Resolution of Equations in Algebraic Structures*. – Austin (Texas), mai 1987.

[DvHS+88] Dincbas (M.), van Hentenryck (P.), Simonis (H.), Aggoun (A.), Graf (T.) et Berthier (F.). – The constraint logic programming language CHIP. *In : Proceedings of the International Conference on Fifth Generation Computer Systems*. pp. 693–702. – Institute for New Generation Computer Technology, 1988.

[Fay79] Fay (M.). – First order unification in equational theories. *In : Proceedings 4th Workshop on Automated Deduction, Austin (Tex., USA)*, pp. 161–167. – 1979.

[FH83] Fages (F.) et Huet (G.). – Unification and matching in equational theories. *In : Proceedings Fifth Colloquium on Automata, Algebra and Programming, L'Aquila (Italy)*. – Springer-Verlag, 1983.

[Fos53] Foster (A. L.). – Generalized "boolean" theory of universal algebras. *Math. Zeitschr.*, vol. Bd. 59, 1953, pp. 191–199.

[GKK90] Gnaedig (I.), Kirchner (C.) et Kirchner (H.). – Equational completion in order-sorted algebras. *Theoretical Computer Science*, vol. 72, 1990, pp. 169–202.

[GS89] Gallier (J.) et Snyder (W.). – Complete sets of transformations for general E-unification. *Theoretical Computer Science*, vol. 67, n° 2-3, octobre 1989, pp. 203–260.

[HD83] Hsiang (J.) et Dershowitz (N.). – Rewrite methods for clausal and non-clausal theorem proving. *In : Proceedings of 10th International Colloquium on Automata, Languages and Programming*. pp. 331–346. – Barcelona (Spain), 1983.

[Her87] Herold (A.). – *Combination of Unification Algorithms in Equational Theories*. – Thèse de PhD, Universität Kaiserslautern (Germany), 1987.

[Hue80] Huet (G.). – Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the Association for Computing Machinery*, vol. 27, n° 4, octobre 1980, pp. 797–821. – Preliminary version in 18th Symposium on Foundations of Computer Science, IEEE, 1977.

[Hul80] Hullot (J.-M.). – *Compilation de Formes Canoniques dans les Théories équationelles*. – Orsay (France), Thèse de Doctorat de Troisième Cycle, Université de Paris Sud, 1980.

[JK86] Jouannaud (J.-P.) et Kirchner (H.). – Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, vol. 15, n° 4, 1986, pp. 1155–1194. – Preliminary version in Proceedings 11th ACM Symposium on Principles of Programming Languages, Salt Lake City (USA), 1984.

[JK91] Jouannaud (J.-P.) et Kirchner (C.). – Solving equations in abstract algebras: a rule-based survey of unification. *In : Computational Logic. Essays in honor of Alan Robinson*, éd. par Lassez (J.-L.) et Plotkin (G.), chap. 8, pp. 257–321. – Cambridge (MA, USA), MIT Press, 1991.

[JKK83] Jouannaud (J.-P.), Kirchner (C.) et Kirchner (H.). – Incremental construction of unification algorithms in equational theories. *In : Proceedings International Colloquium on Automata, Languages and Programming, Barcelona (Spain)*. pp. 361–373. – Springer-Verlag, 1983.

[JL87] Jaffar (J.) et Lassez (J.-L.). – Constraint logic programming. *In : Proceedings of the 14th Annual ACM Symposium on Principles Of Programming Languages, Munich (Germany)*, pp. 111–119. – 1987.

[JLR82] Jouannaud (J.-P.), Lescanne (P.) et Reinig (F.). – Recursive decomposition ordering. *In : Formal Description of Programming Concepts 2*, éd. par Bjørner (D.). pp. 331–348. – Garmisch-Partenkirchen, Germany, 1982.

[KB70] Knuth (D. E.) et Bendix (P. B.). – Simple word problems in universal algebras. *In : Computational Problems in Abstract Algebra*, éd. par Leech (J.), pp. 263–297. – Oxford, Pergamon Press, 1970.

[Kir85a] Kirchner (C.). – *Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories équationnelles.* – Thèse de Doctorat d'Etat, Université de Nancy 1, 1985.

[Kir85b] Kirchner (H.). – *Preuves par complétion dans les variétés d'algèbres.* – Thèse de Doctorat d'Etat, Université de Nancy 1, 1985.

[Kir88] Kirchner (C.). – Order-sorted equational unification. – Presented at the fifth International Conference on Logic Programming (Seattle, USA), août 1988. Also as rapport de recherche INRIA 954, Dec. 88.

[Kir89] Kirchner (C.). – From unification in combination of equational theories to a new AC-unification algorithm. *In : Resolution of Equations in Algebraic Structures, Volume 2: Rewriting Techniques*, éd. par Aït-Kaci (H.) et Nivat (M.). pp. 171–210. – Academic Press, 1989.

[KK86] Kirchner (C.) et Kirchner (H.). – Reveur-3: Implementation of a general completion procedure parametrized by built-in theories and strategies. *Science of Computer Programming*, vol. 20, n° 8, 1986, pp. 69–86.

[KKR90] Kirchner (C.), Kirchner (H.) et Rusinowitch (M.). – Deduction with symbolic constraints. *Revue d'Intelligence Artificielle*, vol. 4, n° 3, 1990, pp. 9–52. – Special issue on Automatic Deduction.

[KL80] Kamin (S.) et Lévy (J.-J.). – Attempts for generalizing the recursive path ordering. – 1980. Unpublished manuscript.

[KO90] Kurihara (M.) et Ohuchi (A.). – Modularity of simple termination of term rewriting systems. *Journal of IPS Japan*, vol. 31, n° 5, 1990, pp. 633–642.

[KO92] Kurihara (M.) et Ohuchi (A.). – Modularity of simple termination of term rewriting systems with shared constructors. *Theoretical Computer Science*, vol. 103, n° 2, 1992, pp. 273–282.

[KR92] Kirchner (H.) et Ringeissen (C.). – A constraint solver in finite algebras and its combination with unification algorithms. *In : Proc. Joint International Conference and Symposium on Logic Programming*, éd. par Apt (K.). pp. 225–239. – MIT Press, 1992.

[Kru60] Kruskal (J. B.). – Well-quasi ordering, the tree theorem and Vazsonyi's conjecture. *Trans. Amer. Math. Soc.*, vol. 95, 1960, pp. 210–225.

[Lan75] Lankford (D. S.). – *Canonical inference*. – Rapport technique, Louisiana Tech. University, 1975.

[Lan77] Lankford (D. S.). – *Some approaches to equality for computational logic: A survey and assessment*. – Memo n° ATP-36, Austin (Texas, USA), Automatic Theorem Proving Project, University of Texas, 1977.

[LBB84] Lankford (D. S.), Butler (G.) et Brady (B.). – Abelian group unification algorithms for elementary terms. *In : Automated Theorem Proving: After 25 Years*, éd. par Bledsoe (W.) et Loveland (W.). – AMS, 1984.

[Löw08] Löwenheim (L.). – Uber das auflösungsproblem im logischen klassenkalkül. *Sitzungsberg. Berl. Math. Gesell*, vol. 7, 1908, pp. 89–94.

[LS76] Livesey (M.) et Siekmann (J.). – *Unification of Bags and Sets*. – Rapport technique, Institut für Informatik I, Universität Karlsruhe, 1976.

[Mid89] Middeldorp (A.). – A sufficient condition for the termination of the direct sum of term rewriting systems. *In : Proceedings 4th IEEE Symposium on Logic in Computer Science, Pacific Grove*, pp. 396–401. – 1989.

[MM82] Martelli (A.) et Montanari (U.). – An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, vol. 4, n° 2, 1982, pp. 258–282.

[MN86] Martin (U.) et Nipkow (T.). – Unification in boolean rings. *In : Proceedings 8th International Conference on Automated Deduction, Oxford (UK)*, éd. par Siekmann (J.). pp. 506–513. – Springer-Verlag, 1986.

[MN88] Martin (U.) et Nipkow (T.). – Unification in boolean rings. *Journal of Automated Reasoning*, vol. 4, 1988, pp. 381–396.

[MN89] Martin (U.) et Nipkow (T.). – Boolean unification — the story so far. *Journal of Symbolic Computation*, vol. 7, n° 3 & 4, 1989, pp. 275–294. – Special issue on unification. Part one.

[MT91] Middeldorp (A.) et Toyama (Y.). – Completeness of combinations of constructor systems. *In : Proceedings 4th Conference on Rewriting Techniques and Applications, Como (Italy)*, éd. par Book (R. V.). pp. 188–199. – Springer-Verlag, avril 1991.

[Nel81] Nelson (G.). – *Techniques for Program Verification.* – Rapport technique n° CS-81-10, Xerox Palo Research Center California USA, 1981.

[New42] Newman (M. H. A.). – On theories with a combinatorial definition of equivalence. *In : Annals of Math*, pp. 223–243. – 1942.

[Nip88] Nipkow (T.). – Unification in primal algebras. *In : Proceedings of the 13th Colloquium on Trees in Algebra and Programming*, éd. par Dauchet (M.) et Nivat (M.). pp. 117–131. – Nancy (France), 1988.

[Nip90] Nipkow (T.). – Unification in primal algebras, their powers and their varieties. *Journal of the Association for Computing Machinery*, vol. 37, n° 1, octobre 1990, pp. 742–776.

[Nip91] Nipkow (T.). – Combining matching algorithms: The regular case. *Journal of Symbolic Computation*, 1991, pp. 633–653.

[NO79] Nelson (C. G.) et Oppen (D. C.). – Simplifications by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, vol. 1, n° 2, 1979.

[NO90] Narendran (P.) et Otto (F.). – Some results on equational unification. *In : Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, éd. par Stickel (M. E.), pp. 276–291. – juillet 1990.

[NR93] Narendran (P.) et Rusinowitch (M.). – The unifiability problem in ground AC theories. *In : Proceedings 8th IEEE Symposium on Logic in Computer Science, Montréal (Québec, Canada).* pp. 364–370. – IEEE Computer Society Press, juin 1993.

[Nut90] Nutt (W.). – Unification in monoidal theories. *In : Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, éd. par Stickel (M. E.). pp. 618–632. – Springer-Verlag, juillet 1990.

[Opp80] Oppen (D. C.). – Convexity, complexity and combination of theories. *Theoretical Computer Science*, 1980. – Preliminary version in Proceedings 4th Workshop on Automated Deduction, Austin (Tex., USA).

[Pla78] Plaisted (D.). – A recursively defined ordering for proving termination of term rewriting sytems. *Dept. of Computer Science Report 78-943*, 1978.

[Plo72] Plotkin (G.). – Building-in equational theories. *Machine Intelligence*, vol. 7, 1972, pp. 73–90.

[PS81] Peterson (G.) et Stickel (M. E.). – Complete sets of reductions for some equational theories. *Journal of the Association for Computing Machinery*, vol. 28, 1981, pp. 233–264.

[Rin92] Ringeissen (C.). – Unification in a combination of equational theories with shared constants and its application to primal algebras. *In : Proceedings of the 1st International Conference on Logic Programming and Automated Reasoning, St. Petersburg (Russia)*. pp. 261–272. – Springer-Verlag, 1992.

[Rob65] Robinson (J. A.). – A machine-oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery*, vol. 12, 1965, pp. 23–41.

[RT90] Ridoux (O.) et Tonneau (H.). – Une mise en œuvre de l'unification d'expressions booléennes. *In : Actes de SPLT'90, Trégastel*. – CNET, 1990.

[Rud74] Rudeanu (S.). – *Boolean functions and equations*. – North-Holland, 1974.

[Rus87] Rusinowitch (M.). – On termination of the direct sum of term rewriting systems. *Information Processing Letters*, vol. 26, n° 2, 1987, pp. 65–70.

[SA91] Socher-Ambrosius (R.). – Boolean algebra admits no convergent term rewriting system. *In : Proceedings 4th Conference on Rewriting Techniques and Applications, Como (Italy)*, éd. par Book (R. V.). pp. 264–274. – Springer-Verlag, avril 1991.

[Sho79] Shostak (R.). – A practical decision procedure for arithmetic with function symbols. *Journal of the Association for Computing Machinery*, vol. 26, n° 2, 1979, pp. 351–360.

[Sho84] Shostak (R.). – Deciding combination of theories. *Journal of the Association for Computing Machinery*, vol. 31, n° 1, 1984, pp. 1–12.

[Sie90] Siekmann (J.). – Unification theory. *In : Unification*, éd. par Kirchner (C.), pp. 1–68. – London, Academic Press, 1990.

[SJ77] Suzuki (N.) et Jefferson (D.). – Verification decidability of presburger array programs. *In : Proceedings of the Conference on Theoretical Computer Science*. – University of Waterloo, 1977.

[Smo89] Smolka (G.). – *Logic Programming over Polymorphically Order-Sorted Types*. – Thèse de PhD, FB Informatik, Universität Kaiserslautern, Germany, 1989.

[SNGM89] Smolka (G.), Nutt (W.), Goguen (J. A.) et Meseguer (J.). – Order-sorted equational computation. *In : Resolution of Equations in Algebraic Structures, Volume 2: Rewriting Techniques*, éd. par Aït-Kaci (H.) et Nivat (M.). pp. 297–367. – Academic Press, 1989.

[SS89] Schmidt-Schauß (M.). – Combination of unification algorithms. *Journal of Symbolic Computation*, vol. 8, n° 1 & 2, 1989, pp. 51–100. – Special issue on unification. Part two.

[Sti75] Stickel (M. E.). – A complete unification algorithm for associative-commutative functions. *In : Proceedings 4th International Joint Conference on Artificial Intelligence, Tbilissi (USSR)*, pp. 71–76. – 1975.

[Sti85] Stickel (M.). – Automated deduction by theory resolution. *Journal of Automated Reasoning*, vol. 1, n° 4, 1985, pp. 285–289.

[Sza79] Szabó (P.). – *The Undecidability of the DA-Unification Problem*. – Rapport technique, University Karlsruhe, 1979.

[Sza82] Szabó (P.). – *Unifikationstheorie erster Ordnung*. – Thèse de PhD, Universität Karlsruhe, 1982.

[Tid86] Tidèn (E.). – *First-order unification in combinations of equational theories*. – Stockholm, Thèse de PhD, The Royal Institute of Technology, 1986.

[Tid88] Tidén (E.). – Symbolic verification of switch-level circuits using a prolog enhanced with unification in finite algebras. *In : The fusion of hardware design and verification*, éd. par Milne (G.). IFIP WG 10.2, pp. 465–485. – North-Holland, 1988.

[Toy86] Toyama (Y.). – Counterexamples to termination for the direct sum of term rewriting systems. *Information Processing Letters*, vol. 25, n° 3, mai 1986, pp. 141–143.

[Toy87] Toyama (Y.). – On the church-rosser property for the direct sum of term rewritig systens. *Journal of the Association for Computing Machinery*, vol. 34, n° 1, janvier 1987, pp. 128–143.

[Yel87] Yelick (K.). – Unification in combinations of collapse-free regular theories. *Journal of Symbolic Computation*, vol. 3, n° 1 & 2, avril 1987, pp. 153–182.

Liste des Figures

Index