Université Henri Poincaré – Nancy 1
Faculté des Sciences

CRIN – CNRS
INRIA Lorraine

[UFR STMIA]
École Doctorale IAE + M
Département de Formation Doctorale en Informatique

Déduction Automatique avec Contraintes Symboliques dans les Théories Equationnelles

THÈSE

présentée et soutenue publiquement le 18 novembre 1994

pour l'obtention du

Doctorat de l'Université Henri Poincaré – Nancy 1

(Spécialité Informatique)

par

Laurent Vigneron

Composition du jury :

Jean-Pierre JOUANNAUD *(Président)*
Ricardo CAFERRA
Hélène KIRCHNER
Michaël RUSINOWITCH
René SCHOTT
Wayne SNYDER

Rapporteurs :

Philippe JORRAND
René SCHOTT
Wayne SNYDER

Je tiens à exprimer toute ma gratitude à Michaël Rusinowitch pour avoir guidé mes premiers pas dans ce domaine difficile qu'est la déduction automatique, et qui m'était complètement inconnu. Par sa compétence, sa disponibilité, sa patience et sa sympathie, il a su faire naître en moi une passion pour la recherche qui n'est pas un feu de paille.

Je souhaite également remercier les personnes qui ont accepté de participer à mon jury :

> *Jean-Pierre Jouannaud m'a fait l'honneur de présider le jury et a montré un grand intérêt pour les travaux effectués.*
>
> *Wayne Snyder m'a fait l'honneur d'être rapporteur de cette thèse et n'a pas hésité à venir spécialement de Boston pour participer à la soutenance. Les discussions que nous avons eues ainsi que ses remarques sur le document m'ont été très précieuses et le resteront pour l'avenir.*
>
> *Ricardo Caferra a très gentiment accepté de s'intéresser à ce travail et de rédiger un rapport, à une époque où il était très occupé. Je l'en remercie ainsi que pour la profonde sympathie qu'il m'a témoigné. Ses commentaires sur le document m'ont été très utiles. Je remercie également Philippe Jorrand d'avoir accepté de co-signer ce rapport.*
>
> *René Schott a effectué, en tant que rapporteur interne, une lecture très attentive de ce document. Ses remarques m'ont permis d'effectuer de nettes améliorations, et les discussions que nous avons eues m'ont fait découvrir de nouvelles applications pour les travaux effectués.*
>
> *Hélène Kirchner a accepté d'examiner ce document. Je lui en suis très reconnaissant, ainsi que pour les excellents conseils qu'elle m'a donnés durant ces trois dernières années.*

Cette thèse s'étant déroulée dans les équipes EURECA puis PROTHEO, je tiens à remercier tous les membres de ces équipes. En particulier, les conseils de Claude Kirchner m'ont été très utiles pour mener à bien mes travaux.

Pour qu'un travail soit correctement réalisé, il est nécessaire de le faire dans une bonne ambiance. J'en profite donc pour dire un grand merci au groupe de joyeux thésards du CRIN, qui m'a aidé à concilier travail et détente.

Je remercie enfin le personnel de l'IUT informatique de l'Université Nancy 2 pour la profonde sympathie qu'il m'a témoigné durant les trois années de monitorat que j'y ai effectuées.

Résumé

Nous proposons dans ce document des techniques de déduction automatique pour effectuer des preuves de propriétés par réfutation dans la logique du premier ordre. Ces preuves sont réalisées modulo un ensemble d'axiomes formant une théorie régulière E.

Les systèmes d'inférence définis sont fondés sur les règles de résolution et de paramodulation. Nous proposons plusieurs stratégies de sélection de clauses (ordonnée ou positive), et de remplacement (paramodulation ou superposition). Nos règles d'inférence évitent la combinatoire de l'usage des axiomes de E, par un calcul de *contextes de remplacements* et un *algorithme d'unification modulo E*. En plus de ces règles d'inférence, nous définissons des règles de simplification permettant d'éliminer les clauses redondantes. Nous démontrons la complétude réfutationnelle de ces systèmes grâce à une extension de la technique des arbres sémantiques transfinis.

Pour le cas des théories associatives et commutatives, nous montrons que l'on peut éviter de résoudre les problèmes d'unification. Seul le test d'existence d'une solution est nécessaire. Ainsi, tout problème d'unification est ajouté à la clause déduite sous la forme d'une contrainte, de même que toute autre condition non résolue pour appliquer l'inférence. Cette stratégie contrainte simule la stratégie basique et n'engendre qu'une seule clause par étape d'inférence, au lieu d'autant de clauses que de solutions au problème d'unification rencontré.

Nous décrivons le logiciel `DATAC`, implantant les résultats présentés dans ce document, pour le cas des théories associatives et commutatives.

Mots-clés: déduction automatique, théorie équationnelle, résolution, paramodulation, réécriture, contraintes symboliques, stratégie basique, arbres sémantiques.

Abstract

In this document, we propose techniques of automated deduction for proving properties by refutation in first order logic. These proofs are done modulo a set of axioms forming a regular theory E.

The inference systems defined are based on resolution and paramodulation rules. We propose several strategies for selecting clauses (ordered or positive) and for applying replacements (paramodulation or superposition). Our inference rules avoid the combinatory of a direct use of axioms of E by a calculus of *replacements contexts* and an *algorithm for unification modulo E*. In addition to these inference rules, we define simplification rules which permit the deletion of redundant clauses. These systems are proved refutationally complete by an extension of the transfinite semantic trees technique.

For associative and commutative theories, we show that we can avoid to solve unification problems. We only have to test the existence of a solution. Each unification problem is added to the deduced clause as a constraint, and each unsolved condition for applying the deduction is added as a constraint too. This constrained strategy simulates the basic strategy and generates only one clause by deduction step, instead of as many as solutions of the unification problem.

We describe the system `DATAC`, implementing the results presented in this document, for associative and commutative theories.

Keywords: automated deduction, equational theories, resolution, paramodulation, term rewriting, symbolic constraints, basic strategy, semantic trees.

Table des matières

Introduction

La déduction automatique est un rêve très ancien des mathématiciens, qui consiste à vouloir mécaniser le raisonnement humain. Dans ce but, des règles de déduction (ou règles d'inférence) ont été mises au point, comme le *modus ponens*. Ces règles permettent entre autres de montrer de manière constructive la consistance de spécifications.

Cette volonté d'effectuer automatiquement des déductions apparaît dans de très nombreux domaines. En calcul formel, pour calculer les bases de Gröbner par exemple, la tendance est d'utiliser des outils automatisant les techniques de réécriture de polynômes ainsi que les calculs d'idéaux. Cet intérêt pour la déduction automatique concerne également la validation de programmes ou la programmation logique. D'autre part, ces domaines manipulent très souvent des fonctions ayant des propriétés équationnelles, telles que l'associativité et la commutativité. Il est donc intéressant de raisonner modulo ces propriétés. Nous présentons justement dans ce document des techniques de déduction modulo une théorie équationnelle, permettant d'effectuer des preuves par réfutation.

Nos travaux sur la déduction automatique, comme beaucoup dans ce domaine, reposent sur les résultats logiques fondamentaux introduits par J. Herbrand dans sa thèse [Her30]. Citons une conséquence de ses travaux :

> *Si S est un ensemble fini de clauses, alors S est insatisfaisable si et seulement si il existe un ensemble fini d'instances closes de S formant un système insatisfaisable.*

Ce théorème est un fondement essentiel de nombreuses techniques de preuve réfutationnelles ; il a permis de faire évoluer les définitions de systèmes de règles d'inférence, avec le souci permanent d'essayer de diminuer l'espace de recherche.

Résolution et Unification

La première étape significative a été franchie par J.A. Robinson [Rob65], lorsqu'il a proposé la règle de *résolution*, qui représente une règle de la logique classique, mais a l'avantage d'être simple. Elle est fondée sur un *algorithme d'unification*, qui résout les différences entre deux atomes de manière à les rendre identiques. La règle de résolution binaire est définie par

$$\frac{A_1 \vee D_1 \qquad \neg A_2 \vee D_2}{D_1\sigma \vee D_2\sigma}$$

si les atomes A_1 et A_2 sont égaux lorsque la substitution σ leur est appliquée.
Cette règle de résolution, combinée avec une règle de *factorisation* éliminant les occurrences

multiples d'un atome dans une clause, est réfutationnellement complète, c'est-à-dire engendre une contradiction de tout ensemble incohérent.

Egalité

Par souci de concision et d'efficacité, des règles d'inférence dérivées ont été introduites (J.R. Slagle [Sla67]). Une seule application de règle correspond à l'enchaînement de plusieurs étapes de déduction. Par exemple, la théorie de l'égalité est classiquement exprimée par les propriétés de réflexivité, symétrie, transitivité et réflexivité fonctionnelle. La présence de ces axiomes augmente énormément le nombre de déductions possibles, dont beaucoup sont redondantes. G.A. Robinson et L.T. Wos [RW69] ont introduit une règle appelée *paramodulation* qui, combinée avec la résolution et les axiomes de réflexivité fonctionnelle, forme un système de déduction complet par réfutation. Cette règle de paramodulation consiste à remplacer un terme par un terme égal. Elle est définie par :

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2[l_2] \vee D_2}{L_2[r_1]\sigma \vee D_1\sigma \vee D_2\sigma}$$

si la substitution σ est un unificateur des termes l_1 et l_2. Nous dirons alors que la paramodulation a été appliquée dans les termes l_2.
D. Brand [Bra75] a montré que seul l'axiome de réflexivité $(x \simeq x)$ est nécessaire pour la complétude. Une amélioration très importante a été montrée par G.E. Peterson [Pet83] : il est inutile d'appliquer une paramodulation dans une variable.

Par la suite, M. Rusinowitch [Rus89] et J. Hsiang [HR91] ont défini une technique de preuve dans les arbres sémantiques transfinis pour montrer la complétude réfutationnelle des règles dites de factorisation, résolution et paramodulation ordonnées, qui consiste à restreindre les inférences aux seuls littéraux maximaux des clauses et, lors d'une paramodulation, à ne pas remplacer un terme par un autre plus grand ou égal.
Une modification de cette stratégie ordonnée est la stratégie dite positive. Elle précise en effet que toute déduction doit utiliser au moins une clause n'ayant que des littéraux positifs. Cette stratégie, proposée pour la résolution par J.A. Robinson dans [Rob65], est définie pour la paramodulation dans [Rus89].

De nombreux raffinements ont ensuite été définis pour la règle de paramodulation, comme la règle de *superposition* interdisant tout remplacement dans le plus petit membre d'une équation, ou les règles de *simplifications clausales* [Rus89], inspirées de l'algorithme de D.E. Knuth et P.B. Bendix [KB70], éliminant des clauses redondantes. En particulier, une définition générale de redondance d'une clause ou d'une inférence a été proposée par L. Bachmair et H. Ganzinger [BG94].

Insuffisance des techniques classiques

Les règles d'inférence mentionnées précédemment forment un système complet, mais restent inefficaces dans de nombreux cas. En particulier, la présence d'équations non orientables, c'est-à-dire dont les membres ne sont pas comparables, comme la commutativité $(f(x,y) \simeq f(y,x))$, gêne considérablement l'application de la règle de paramodulation. D'autres axiomes, comme l'associativité $(f(f(x,y),z) \simeq f(x,f(y,z)))$ combinée avec la commutativité, provoquent la divergence des déductions. En effet, si une paramodulation est effectuée dans le sous-terme $f(x,y)$ de cet axiome, la clause déduite pourra également être utilisée pour une paramodulation dans ce même sous-terme, et ainsi de suite.

Le traitement de tels axiomes nécessite donc un contrôle important et coûteux, afin d'éviter autant que possible ces nombreuses déductions qui s'avèrent être inutiles.

Déduction modulo une théorie équationnelle

La solution proposée par G. Plotkin [Plo72], pour éviter ces déductions redondantes, est d'utiliser un algorithme d'unification dans une théorie équationnelle présentée par un ensemble fini d'axiomes. J.R. Slagle [Sla74] a montré que la combinaison de règles de réécriture et de règles d'inférence utilisant l'unification, comme la résolution et la paramodulation, permet de construire des procédures complètes (pour réfuter des conjectures). Le résultat de J.R. Slagle portant sur des théories très limitées, D.S. Lankford et A. Ballantyne [LB77a] l'ont étendu aux théories *permutatives*, c'est-à-dire composées d'axiomes tels que tout opérateur apparaissant dans un membre d'un axiome apparaît aussi dans le second membre de cet axiome. L'exemple le plus important étant les théories associatives et commutatives [LB77b].

En 1981, G.E. Peterson et M.E. Stickel [PS81] ont proposé une procédure de complétion pour les théories présentées comme l'union d'un ensemble de règles de réécriture R et d'un ensemble d'équations non orientables E, qui engendre un système canonique de réductions. Cet algorithme utilise comme G. Plotkin les axiomes de E de manière implicite, mais il n'a été appliqué qu'aux théories associatives et commutatives.

La déduction modulo une théorie équationnelle E nécessite donc la définition de mécanismes simulant le rôle de ces axiomes, tels que des algorithmes d'unification, de filtrage et d'égalité modulo cette théorie E. Il faut également tenir compte des éventuelles paramodulations appliquées dans ces axiomes, engendrant des clauses appelées *extensions* des clauses initiales ; dans le cadre purement équationnel, la gestion des extensions est habituellement traitée

- soit en ajoutant une règle d'inférence correspondant à une paramodulation d'une équation dans un axiome de E, ce qui engendre les extensions des équations,

- soit en associant à chaque équation l'ensemble de ses extensions possibles, qui peuvent être utilisées par la règle de paramodulation.

Nous avons proposé une nouvelle technique pour traiter les extensions dans le cadre clausal, adaptée aux théories associatives et commutatives [RV91, RV95]. Elle consiste à définir deux nouvelles règles d'inférence, appelées *paramodulation contextuelle* et *paramodulation étendue*, simulant l'application d'une paramodulation avec une extension. Ainsi, nous n'avons plus besoin de créer explicitement les extensions des clauses.
Cette action de simuler les extensions a été appliquée pour la complétion équationnelle par S. Anantharaman, J. Hsiang et J. Mzali [AHM89] dans l'implantation du système `SbReve2`, puis par C. Marché [Mar93].

Dans ce document, nous généralisons cette technique à une théorie équationnelle E, et proposons une méthode que l'on pourrait qualifier de « compilatoire » pour gérer les extensions potentielles associées à la théorie E. Cette méthode consiste à extraire de la théorie E les différentes formes (appelées *contextes*) que pourront prendre les extensions. Un contexte est principalement représenté par un couple composé d'un terme et d'une position dans ce terme : ainsi, l'extension d'une équation $(l \simeq r)$ pour un contexte (e, p) est $(e[l]_p \simeq e[r]_p)$. Ce calcul de contextes incorpore un algorithme détectant les éventuelles redondances.

Les contextes calculés pour la théorie E, avant même de connaître le système de clauses à résoudre, sont utilisés dans les règles d'inférence de paramodulation contextuelle et de paramodulation étendue. En conséquence, nous n'engendrons pas explicitement les extensions des clauses, mais simulons leur rôle grâce aux règles précédentes.

Les règles d'inférence définies dans cette thèse décrivent les stratégies de paramodulation et de superposition basées sur les stratégies ordonnées et positives ; en particulier, nous apportons des raffinements tre‘s importants pour la strate'gie positive. A ces règles viennent s'ajouter des règles de simplification permettant d'éliminer des clauses redondantes. Ces systèmes sont réfutationnellement complets pour des théories équationnelles vérifiant la propriété de *régularité* :

> *Pour toute équation $(e_1 \simeq e_2)$ de la théorie E, l'ensemble des variables du terme e_1 doit être égal à l'ensemble des variables du terme e_2.*

Cette condition est imposée par les propriétés nécessaires à l'ordre comparant les termes. La preuve de complétude est basée sur la technique des arbres sémantiques transfinis de J. Hsiang et M. Rusinowitch [Rus89, HR91].

Nous appliquons et raffinons ces résultats à la classe des *théories associatives et commutatives*. Ces dernières sont très importantes car elles interviennent dans de nombreux domaines comme les calculs algébriques, la modélisation de processus concurrents ou la planification. Pour ces théories, nous ajoutons dans les règles d'inférence des conditions limitant leurs applications, comme par exemple interdire tout remplacement au niveau d'un opérateur associatif et commutatif f s'il se trouve juste sous un même opérateur f. D'autres raffinements très importants pour restreindre l'espace de recherche interviennent dans les règles de paramodulation contextuelle et étendue. En particulier, notre règle de paramodulation étendue peut être vue comme une généralisation de la règle de superposition de l'algorithme de B. Buchberger pour construire les bases de Gröbner [Buc65], ou une simulation du calcul des paires critiques étendues de C. Marché [Mar93].

Contraintes symboliques

La déduction dans une théorie équationnelle E s'appuie sur un algorithme d'unification dans cette théorie. Contrairement à la théorie vide où il n'existe qu'un unificateur principal, les algorithmes d'unification modulo E peuvent engendrer un très grand nombre d'unificateurs principaux, voire une infinité comme pour les théories associatives. Par exemple, E. Domenjoud [Dom92] a montré que l'unification des termes $x * x * x * x$ et $y_1 * y_2 * y_3 * y_4$, où l'opérateur $*$ est associatif et commutatif, engendrait 34 359 607 481 solutions principales.

Ainsi, malgré les nombreuses conditions ajoutées dans les règles d'inférence, l'espace de recherche reste énorme pour de nombreuses théories équationnelles.

Nous proposons donc une amélioration très importante pour la déduction dans des théories associatives et commutatives (AC), permettant de supprimer ces problèmes d'unification.

Ceux-ci sont conservés sous la forme de *contraintes d'unification AC* associées à chaque clause : pour une clause C et une conjonction c de contraintes d'unification $(t_1 =^?_{AC} t_2)$, $C\,[\![\, c \,]\!]$ représente l'ensemble des clauses $C\sigma$ telles que σ est solution des contraintes c. Ainsi, une seule clause est déduite lors d'une étape d'inférence, au lieu d'une clause par solution du problème d'unification. Ces travaux sont inspirés de la notion de résolution contrainte présentée par G. Huet [Hue72] pour la logique d'ordre supérieur, et étendent les résultats de C. Kirchner, H. Kirchner et M. Rusinowitch [KKR90] et R. Nieuwenhuis et A. Rubio [NR92a, NR92b], développés dans la théorie vide.

La règle de résolution pour les théories AC s'exprime alors par :

$$\frac{A_1 \vee D_1 \;\; [\![\, c_1 \,]\!] \qquad\qquad \neg A_2 \vee D_2 \;\; [\![\, c_2 \,]\!]}{D_1 \vee D_2 \;\; [\![\, c_1 \wedge c_2 \wedge (A_1 =^?_{AC} A_2) \,]\!]}$$

Ainsi, chaque clause déduite hérite des contraintes de ses clauses parentes, auxquelles s'ajoutent les problèmes d'unification AC introduits par la règle utilisée. De la même manière, la règle de paramodulation contrainte est définie par :

$$\frac{(l_1 \simeq r_1) \vee D_1 \;\; [\![\, c_1 \,]\!] \qquad\qquad L_2[l_2] \vee D_2 \;\; [\![\, c_2 \,]\!]}{L_2[r_1] \vee D_1 \vee D_2 \;\; [\![\, c_1 \wedge c_2 \wedge (l_1 =^?_{AC} l_2) \,]\!]}$$

Il est à noter que toutes les autres règles d'inférence (factorisation, réflexion, paramodulations contextuelle et étendue) peuvent être transformées de la même manière.

Les problèmes d'unification n'étant pas résolus, la stratégie contrainte permet également de simuler la *stratégie basique*, définie par :

> *Tout remplacement appliqué dans un terme introduit dans une clause par une déduction antérieure est inutile.*

Les solutions des contraintes n'étant pas calculées, aucune substitution n'est appliquée dans la clause déduite. La définition initiale de cette stratégie basique présentée par J.-M. Hullot [Hul80] pour la règle de surréduction dans la théorie vide, consiste à retenir les positions où une substitution a été appliquée, afin d'interdire toute future déduction au niveau de celles-ci. Le principe de cette stratégie a été incorporé dans des systèmes d'inférence basés sur les stratégies de paramodulation et de superposition dans la théorie vide par L. Bachmair, H. Ganzinger, C. Lynch et W. Snyder [BGLS92], avec cependant une différence significative : les substitutions ne sont plus appliquées dans les clauses déduites ; à chaque clause est associée une substitution, représentant le cumul des substitutions calculées pour déduire cette clause.

En appliquant la stratégie basique, une amélioration supplémentaire peut être ajoutée dans les règles de paramodulation : nous « bloquons » tout futur remplacement dans le terme r_1, car celui-ci est introduit dans la clause déduite par la règle d'inférence. Cette amélioration n'est pas compatible avec la stratégie de superposition. Cependant, les adaptations que nous avons définies pour ces stratégies de paramodulation et de superposition sont optimales (jusqu'à ce jour).

La stratégie contrainte présentée dans cette thèse met en œuvre d'autres types de contraintes symboliques. En fait, toutes les conditions qu'il faut habituellement vérifier pour appliquer une règle d'inférence sont codées sous forme de contraintes, les plus importantes étant les *contraintes* *d'ordre*. En effet, les conditions d'orientation d'une équation ou de maximalité d'un littéral dans une clause restent souvent non résolues, car l'ordre sur des termes avec variables n'est pas total. En conséquence, une étape d'inférence est appliquée si ces conditions sont satisfaisables, mais il est possible qu'une instance de la clause déduite falsifie l'une de ces conditions. Pour éviter ces problèmes d'incohérence dans une succession de déductions, nous ajoutons dans la partie contrainte de la clause déduite des contraintes de la forme $(t_1 \succ^? t_2)$.

Ainsi, une clause est déduite par une étape d'inférence s'il existe une solution de ses contraintes d'unification satisfaisant les autres contraintes symboliques.

La stratégie de déduction avec des contraintes symboliques est en quelque sorte un retour à l'esprit des travaux de J. Herbrand [Her30] où l'on tente de réduire le calcul des prédicats au calcul propositionnel. On peut considérer les formules contraintes comme des approximations des formules propositionnelles pouvant intervenir dans une preuve. En particulier, nous n'utilisons plus la règle de logique qui, à partir d'une formule $\forall x,\ p(x)$, déduit $p(t)$ pour un terme t. Ainsi, nous gardons la notion d'existence d'un ensemble fini d'instances closes insatisfaisables, sans chercher à les calculer.

Le système DATAC

Le logiciel DATAC, pour *Déduction Automatique dans des Théories Associatives et Commutatives*, est l'implantation des travaux réalisés au cours de cette thèse pour les théories associatives et commutatives. Il est écrit en CAML Light [LM93], langage fonctionnel de la famille ML, et est muni d'une interface développée en Tcl/Tk [Ous94]. Outre les stratégies de paramodulation et de superposition, le logiciel offre la possibilité d'utiliser la stratégie contrainte, la stratégie basique, ou bien la stratégie standard. Cette implantation nous a permis de comparer ces différentes stratégies.

Plan de la thèse

Le document est organisé de la façon suivante. Le premier chapitre est consacré à la présentation du cadre de travail, c'est-à-dire des notations et des concepts de base. Nous introduisons les ordres de simplification compatibles et décrivons quelques propriétés qui seront fondamentales pour la suite.

Le second chapitre soulève les problèmes rencontrés pour effectuer de la déduction modulo une théorie équationnelle E, puis présente des systèmes de règles d'inférence basés sur les stratégies de paramodulation et de superposition. Ces systèmes étant définis pour la stratégie ordonnée, nous décrivons ensuite leur adaptation à la stratégie positive. Enfin, nous proposons un ensemble de règles de simplification permettant d'éliminer des clauses redondantes. Nous terminons ce chapitre par une comparaison avec les travaux de G.D. Plotkin [Plo72], U. Wertz [Wer92] et E. Paul [Pau94].

Le troisième chapitre est consacré à l'étude des extensions de règles (sous forme de contextes) nécessaires à une théorie équationnelle E. Nous y définissons un algorithme de calcul des contextes combiné avec une détection de redondance. Nous proposons également un algorithme qui, appliqué juste avant d'effectuer une déduction avec un contexte, permet de déterminer si la clause déduite ne sera pas redondante par rapport à l'ensemble des clauses disponibles.

La preuve de complétude des systèmes d'inférence, décrits dans le deuxième chapitre pour une théorie équationnelle E, est détaillée dans le quatrième chapitre. Cette preuve est valable pour toute théorie E régulière, c'est-à-dire préservant les variables. La méthode de preuve est fondée sur la technique des arbres sémantiques transfinis. Une partie de ces preuves a été transférée dans l'Appendice A, afin de faciliter la compréhension de la technique utilisée.

Dans le cinquième chapitre, nous présentons le cas des théories associatives et commutatives, et nous y décrivons des améliorations très importantes dans l'application des règles d'inférence.

Le sixième chapitre présente la stratégie contrainte. Après avoir retracé l'historique de cette stratégie et précisé nos motivations, nous décrivons le langage de contraintes manipulé, puis définissons les règles d'inférence et de simplification. La preuve de complétude de cette stratégie s'inspire très naturellement de la preuve dans le cas non contraint. Nous présentons ensuite quelques règles pour simplifier les ensembles de contraintes engendrés.

Le septième chapitre est consacré à l'implantation dans le logiciel `DATAC` des travaux décrits dans les chapitres 4, 5 et 6, concernant les théories associatives et commutatives. Nous y détaillons les différentes stratégies proposées, accompagnées de quelques exemples de résolution, puis comparons ces stratégies.

Enfin, nous concluons sur des perspectives, des extensions et des applications potentielles de notre travail.

1
Cadre de travail

Nous présentons dans ce chapitre les principales définitions et notations utilisées dans ce document, adaptées pour la déduction modulo une théorie équationnelle E. Nous énonçons également les propriétés devant être vérifiées par l'ordre sur les termes, et décrivons les principaux ordres existants avec ces propriétés. Ce chapitre se termine par la construction d'ordres totaux sur les classes de congruence de termes et d'atomes engendrées par la théorie E, à partir d'un ordre total sur les termes clos.

1.1 Définitions et notations

Cette section est consacrée à la présentation des principales notations utilisées tout au long de cette thèse. Elles sont conformes aux notations standards définies dans [DJ90].

Termes et substitutions

Soient $\mathcal{F}$ un ensemble fini d'opérateurs de fonctions, et $\mathcal{X}$ un ensemble fini de variables. L'algèbre des termes composée à partir de $\mathcal{F}$ et $\mathcal{X}$ est notée $\mathcal{T}(\mathcal{F}, \mathcal{X})$. $\mathcal{T}(\mathcal{F})$ désigne l'ensemble des termes clos (l'Univers de Herbrand).

Soit $\mathcal{P}$ un ensemble fini d'opérateurs de prédicats, contenant le prédicat d'égalité $\simeq$. L'ensemble des atomes $\mathcal{A}(\mathcal{P}, \mathcal{F}, \mathcal{X})$ est $\{P(t_1, \ldots, t_n) \mid P \in \mathcal{P} \text{ et } t_i \in \mathcal{T}(\mathcal{F}, \mathcal{X})\}$. L'ensemble des atomes clos (la Base de Herbrand) est notée $\mathcal{A}(\mathcal{P}, \mathcal{F})$. Une équation est un atome dont le symbole de prédicat est $\simeq$. Dans ce document, nous supposerons que $\simeq$ est commutatif dans le sens où nous ne ferons pas la distinction entre les atomes $(s \simeq t)$ et $(t \simeq s)$. Un *littéral* est un atome (A) ou la négation d'un atome $(\neg A)$. Une *clause* C est un multi-ensemble de littéraux, notée sous la forme d'une disjonction : $C = L_1 \vee \ldots \vee L_n$. La clause ne contenant aucun littéral est appelée *clause vide*, et notée $\square$.

L'ensemble des variables d'un terme t est noté $\mathcal{V}ar(t)$. Une substitution est une fonction σ de $\mathcal{X}$ dans $\mathcal{T}(\mathcal{F}, \mathcal{X})$. $\mathcal{D}om(\sigma)$ désigne le sous-ensemble des variables de $\mathcal{X}$ telles que $\sigma(x) \neq x$, et $\mathcal{C}o\mathcal{D}om(\sigma)$ est l'ensemble $\{\sigma(x) \mid x \in \mathcal{D}om(\sigma)\}$. Une substitution σ est appliquée à un terme t si toutes les variables x de t sont remplacées par $\sigma(x)$; le résultat est noté $t\sigma$.

Pour exprimer plus efficacement les sous-termes et les substitutions, nous utilisons des *positions* (ou *occurrences*). Imaginons un terme représenté par un arbre : une position dans un terme indique un nœud de l'arbre ; elle est représentée par une suite d'entiers naturels $i_1 \cdot i_2 \cdot \ldots \cdot i_n$, où, partant de la racine du terme, chaque i_j est l'indice de la branche à emprunter pour passer du j-ième nœud au $j+1$-ième ; la suite vide, notée ϵ, représente la position vide (i.e. la racine). Soit p une position ; nous utiliserons $t|_p$ pour le sous-terme de t en p. Plus précisément, $t|_\epsilon = t$, et $f(t_1, \ldots, t_n)|_{i \cdot p} = t_i|_p$. Nous noterons $s[p \leftarrow t]$ ou $s[t]_p$ pour préciser que le sous-terme à la position p de s est t. Un sous-terme de t est dit *strict* s'il est distinct de t. L'opérateur au sommet d'un terme t est noté $\mathcal{H}ead(t)$, si t n'est pas une variable ; sinon, $\mathcal{H}ead(t)$ n'est pas défini. L'ensemble des positions non variables d'un terme t, c'est-à-dire des positions p de t telles que $t|_p \notin \mathcal{X}$, est noté $\mathcal{FP}os(t)$.

Théorie équationnelle

Soit E un ensemble d'équations. La congruence engendrée par cet ensemble E est appelée *E-égalité* et notée $=_E$. Une substitution σ est un *E-unificateur* de deux termes s et t si $s\sigma =_E t\sigma$; σ est un *E-unificateur principal* s'il n'existe pas d'autre E-unificateur ρ de s et t tel que, pour toute variable x de $\mathcal{D}om(\rho)$, $\sigma(x) =_E \tau(\rho(x))$ pour une substitution τ. Un problème d'unification entre deux termes s et t est noté $s =_E^? t$. L'ensemble des substitutions solution d'une conjonction c de problèmes d'unification est noté $\mathcal{S}ol(c)$. Une substitution σ est un *filtre* (modulo E) d'un terme s dans un terme t si $s\sigma =_E t$; l'ensemble des filtres de s dans t sera noté *E-match*(s, t).

La relation de sous-terme modulo E est notée $\trianglelefteq_E$; $s \trianglelefteq_E t$ s'il existe un terme t', E-égal à t, et une position p dans t' tels que $t'|_p =_E s$. Si la position p est stricte, la relation est notée $\triangleleft_E$.

La relation d'inclusion d'un ensemble de termes dans un autre ensemble de termes, utilisant la E-égalité pour comparer les termes, est notée $\subseteq_E$; $\Subset_E$ désigne son extension aux multi-ensembles de termes.

Une théorie E est dite *régulière* si, pour chacun de ses axiomes $(e_1 \simeq e_2)$, $\mathcal{V}ar(e_1) = \mathcal{V}ar(e_2)$.

Cohérence et complétude

Un ensemble de clauses S est *cohérent* s'il possède au moins un modèle. Un ensemble de clauses S est *E-incohérent* s'il n'a pas de modèle cohérent avec la théorie E.

Soit *INF* un ensemble de règles d'inférence et S un ensemble de clauses. *INF(S)* dénote l'ensemble obtenu en ajoutant à S toutes les clauses engendrées par application de règles de *INF* aux clauses de S. Soit $INF^0(S) = S$, $INF^{n+1}(S) = INF(INF^n(S))$, et $INF^*(S) = \bigcup_{n \geq 0} INF^n(S)$.

Un ensemble de règles d'inférence *INF* est *réfutationnellement complet pour une théorie E*, ou *E-complet* en abrégeant, si, pour tout ensemble de clauses S E-incohérent, $INF^*(S)$ contient la clause vide.

1.2 Ordres sur les termes

L'utilisation d'un *ordre*, relation irréflexive et transitive permettant de mesurer la complexité des termes, est indispensable en déduction automatique. Cela permet d'assurer la terminaison des simplifications et de restreindre l'application de règles d'inférence. Les ordres que nous utilisons doivent être des *ordres de simplification totaux*, c'est-à-dire avoir les « bonnes » propriétés énumérées ci-dessous.

Définition 1.1 (Ordre de simplification total) *Un ordre $>$ est un ordre de simplification total (TSO pour abréger) si*

1. *$>$ est bien fondé (ou nœthérien) : il n'existe pas de suite infinie de termes $s_1, s_2, \ldots$ telle que $s_i > s_{i+1}$ pour tout $i \geq 1$,*
2. *$>$ est monotone : étant donnés deux termes s et t tels que $s > t$, alors $w[s]_p > w[t]_p$ pour tout terme w et toute position p dans w,*
3. *$>$ a la propriété de sous-terme : si s est un sous-terme strict de t, alors $t > s$,*
4. *$>$ est stable par instanciation : pour toute substitution σ, et pour tous termes s et t tels que $s > t$, alors $s\sigma > t\sigma$,*
5. *$>$ est total sur l'ensemble des termes clos : pour tous termes clos s et t, on a soit $s > t$, soit $t > s$, soit $s = t$.*

Comme nous allons travailler dans une théorie équationnelle E, une autre propriété doit être respectée, la *E-compatibilité*.

Définition 1.2 (E-compatibilité) *Un ordre $>$ est E-compatible si, étant donnés deux termes s et t tels que $s > t$ et $s \neq_E t$, alors pour tous termes s' et t' respectivement E-égaux à s et t, $s' > t'$.*

La définition d'ordres de simplification totaux dans une théorie E est un problème très difficile. Le cas des théories associatives et commutatives (AC) a été traité ; E contient l'ensemble des axiomes

$$(A) \quad f(f(x,y),z) \simeq f(x, f(y,z)) \qquad\qquad (C) \quad f(x,y) \simeq f(y,x)$$

pour chaque opérateur f d'un sous-ensemble $\mathcal{F}_{AC}$ de $\mathcal{F}$. Dans ces théories, le problème est :

Comment tenir compte du fait que
tous les sous-termes d'un opérateur AC peuvent permuter ?

Trois types d'ordres ont été développés pour tenter de résoudre ce problème. Détaillons leurs approches.

1.2.1 Ordres récursifs sur les chemins

Rappelons la définition d'un ordre récursif sur les chemins, le RPO [Der82]. Pour cela, il est nécessaire d'ordonner les opérateurs par préférence et de s'assurer que deux opérateurs sont toujours comparables ; un tel ordre est appelé une *précédence totale sur les opérateurs* et noté $>_{\mathcal{F}}$. Une fonction τ associe un statut à chaque opérateur pour indiquer la manière de traiter ses arguments ; il peut être multi-ensemble, lexicographique de la gauche vers la droite (Lexico-GD) ou de la droite vers la gauche (Lexico-DG).
L'extension d'un ordre $>$ aux multi-ensembles est définie par :

Définition 1.3 (Extension multi-ensemble) $S = \{s_1, \ldots, s_m\} >^{mult} \{t_1, \ldots, t_n\} = T$ *si, définissant le multi-ensemble I comme l'intersection des multi-ensembles S et T, pour tout t_j de $T - I$, il existe un s_i dans $S - I$ tel que $s_i > t_j$.*

L'extension lexicographique d'un ordre $>$ est définie par :

Définition 1.4 (Extension lexicographique) *$(s_1, \ldots, s_m) >^{lex} (t_1, \ldots, t_m)$ s'il existe un indice $i \in [1, m]$ tel que : $\forall j \in [1, i[,\ s_j = t_j$ et $s_i > t_i$.*

Enonçons la définition de l'ordre RPO, faisant appel à ces extensions multi-ensembles et lexicographiques.

Définition 1.5 (Ordre RPO) *$s = f(s_1, \ldots, s_m) \ >_{RPO}\ g(t_1, \ldots, t_n) = t$ si*

- *soit $\exists i \in [1, m],\ s_i >_{RPO} t$,*
- *soit $f >_{\mathcal{F}} g$ et $\forall j \in [1, n],\ s >_{RPO} t_j$,*
- *soit $f = g$ (donc $\tau(f) = \tau(g)$) et*
  - *si $\tau(f) =$ Multi-ensemble, $\{s_1, \ldots, s_m\} >_{RPO}^{mult} \{t_1, \ldots, t_n\}$,*
  - *si $\tau(f) =$ Lexico-GD, $(s_1, \ldots, s_m) >_{RPO}^{lex} (t_1, \ldots, t_n)$,*
  - *si $\tau(f) =$ Lexico-DG, $(s_m, \ldots, s_1) >_{RPO}^{lex} (t_n, \ldots, t_1)$.*

Rappelons qu'un ordre défini sur les termes clos peut s'étendre sur les termes avec variables de la façon suivante :

Définition 1.6 (Extension aux termes variables) *Soient s et t deux termes de $\mathcal{T}(\mathcal{F}, \mathcal{X})$; $s > t$ si, pour toute substitution close σ définie sur $\mathcal{V}ar(s) \cup \mathcal{V}ar(t)$, $s\sigma > t\sigma$.*

Les premières adaptations aux théories AC ont été réalisées sur les ordres récursifs sur les chemins, et ont consisté à aplatir les termes au niveau des opérateurs AC et à donner un statut multi-ensemble à ceux-ci.

Définition 1.7 (Aplatissement) *La fonction aplat : $\mathcal{T}(\mathcal{F}) \to \mathcal{T}(\mathcal{F})$, aplatissant un terme clos t, est définie par :*

$$aplat(t) = \begin{cases} t & \text{si } t \text{ est une constante} \\ f(aplat(t_1), \ldots, aplat(t_n)) & \text{si } t = f(t_1, \ldots, t_n) \text{ et } f \notin \mathcal{F}_{AC} \\ f(t_{1,1}, \ldots, t_{1,k_1}, \ldots, t_{n,1}, \ldots, t_{n,k_n}) & \text{si } t = f(t_1, \ldots, t_n) \text{ et } f \in \mathcal{F}_{AC} \\ \quad \text{où, pour chaque } i \in [1, n], \text{ la séquence } t_{i,1}, \ldots, t_{i,k_i} \text{ est} & \\ \quad\quad \bullet \text{ soit réduite à } aplat(t_i) \text{ si } \mathcal{H}ead(t_i) \neq f,\ (k_i = 1) & \\ \quad\quad \bullet \text{ soit définie par } f(t_{i,1}, \ldots, t_{i,k_i}) = aplat(t_i). & \end{cases}$$

Un terme t est dit aplati si $aplat(t) = t$.

Nous noterons $\boldsymbol{\mathcal{H}terms(t, f)}$ le multi-ensemble des arguments de l'opérateur AC f au sommet de $aplat(t)$. Si $\mathcal{H}ead(t) \neq f$, alors $\mathcal{H}terms(t, f) = \{t\}$.

Le statut multi-ensemble associé aux opérateurs AC est indispensable, car ce sont en fait des *opérateurs variadiques* après aplatissement, c'est-à-dire n'ayant pas une arité fixe.

Ainsi, une première proposition d'ordre a été

$$s > t \quad \text{si} \quad aplat(s) >_{RPO} aplat(t)$$

Mais, l'ordre obtenu n'est pas un TSO, comme le montre l'exemple suivant (la propriété de monotonie n'est pas respectée).

Exemple 1.1 Les entiers naturels sont classiquement représentés par les règles $x + 0 \to x$ et $x + s(y) \to s(x + y)$, orientées grâce à la précédence $+ >_{\mathcal{F}} s >_{\mathcal{F}} 0$. $+$ est un symbole AC et a donc le statut multi-ensemble. Selon la précédence ainsi décrite, le terme $s(0) + s(0)$ est plus grand que le terme $s(s(0))$. Cependant, si nous incluons ces deux termes dans un même contexte $s(0) + [.]$ l'ordre est inversé, c'est-à-dire $s(0) + s(s(0)) >_{RPO} s(0) + s(0) + s(0)$. En effet, le multi-ensemble $\{s(0), s(s(0))\}$ est supérieur au multi-ensemble $\{s(0), s(0), s(0)\}$. ♦

Une nouvelle condition doit donc être ajoutée: les opérateurs AC doivent être minimaux pour la précédence. Mais, cela veut dire qu'un seul opérateur AC est autorisé, et que la représentation des entiers naturels citée dans l'Exemple 1.1 n'est plus orientable dans le sens usuel.

De nombreux ordres AC-compatibles ont été définis avec pour point commun de ne permettre l'utilisation que d'un seul opérateur AC. La principale question a lors été: *comment faire cohabiter plusieurs opérateurs AC ?*

L. Bachmair et D. Plaisted [BP85b] ont défini un ordre basé sur le RPO mais permettant plusieurs opérateurs AC, l'*Associative Path Ordering*. Pour cela, ils ont exploité des propriétés de distributivité entre ces opérateurs AC. L'ordre RPO n'est appelé qu'après avoir appliqué cette distributivité dans les termes à comparer.

Définition 1.8 (Ordre APO) *Etant donnés deux termes clos s et t,*

$$s >_{APO} t \quad si \quad aplat(distrib(s)) >_{RPO} aplat(distrib(t))$$

où la fonction $distrib : \mathcal{T}(\mathcal{F}) \to \mathcal{T}(\mathcal{F})$, appliquée à un terme t, est définie par

$$\begin{cases} t & si\ t\ est\ une\ constante \\ g(t'_1, t'_2) & si\ t = f(t_1, t_2),\ \exists i \in [1,2],\ t_i = g(t_{i1}, t_{i2}), \\ & \quad f, g \in \mathcal{F}_{AC},\ f\ est\ distributif\ dans\ g,\ et \\ & \quad \bullet\ soit\ i = 1,\ t'_1 = distrib(f(t_{11}, t_2))\ et\ t'_2 = distrib(f(t_{12}, t_2)) \\ & \quad \bullet\ soit\ i = 2,\ t'_1 = distrib(f(t_1, t_{21}))\ et\ t'_2 = distrib(f(t_1, t_{22})) \\ f(t'_1, \ldots, t'_n) & sinon\ (t = f(t_1, \ldots, t_n)\ et\ t'_i = distrib(t_i)) \end{cases}$$

La condition de minimalité des opérateurs AC pour la précédence est toujours présente, mais en plus l'opérateur AC distribué doit être plus grand que l'autre opérateur AC.
Cet ordre peut traiter plus de deux opérateurs AC, mais dans ce cas la précédence n'est plus totale, car au plus deux opérateurs AC peuvent être comparables avec cette approche.

Le principal défaut de cet ordre, outre la limitation du nombre d'opérateurs AC, est la minimalité des opérateurs AC pour la précédence. C. Delor et L. Puel [DP93] ont montré que l'ordre APO pouvait être considérablement amélioré:

1. il est possible de déclarer des constantes et des opérateurs unaires plus petits que des opérateurs AC,
2. il est possible d'avoir plus de deux opérateurs AC comparables.

Ces progrès sont possibles à condition d'appliquer des transformations sur les termes ; cela se traduit par la mise en forme normale des termes à comparer, grâce à un système de réécriture représentant les transformations à appliquer. Le lecteur intéressé par plus de détails pourra se reporter à [DP93].
De telles techniques avaient déjà été utilisées par I. Gnaedig et P. Lescanne [Gna86, GL86], sous la forme de systèmes de réécriture auxiliaires.

1.2.2 Interprétations polynômiales

Les interprétations polynômiales ont été introduites par D. Lankford [Lan79]. Elles consistent à associer un polynôme à chaque opérateur. Ainsi, un terme est interprété par le polynôme résultant de la composition des polynômes associés à chacun de ses opérateurs. A. Ben Cherifa et P. Lescanne [BC86, BCL87] ont décrit des techniques assez simples permettant de montrer qu'un polynôme P_1 est plus grand qu'un polynôme P_2.

Ils ont étendu ces calculs d'interprétations aux opérateurs AC en donnant le profil des polynômes à associer à de tels opérateurs.

Proposition 1.9 *L'interprétation de tout opérateur associatif-commutatif f (binaire) doit être un polynôme de la forme:*

$$[f](X,Y) \;=\; aXY + b(X+Y) + c \qquad \textit{avec} \qquad ac + b - b^2 = 0 \quad (a,b,c \in \mathbb{N})$$

Ces interprétations sont essentiellement utilisées pour prouver la terminaison d'un système de réécriture, c'est-à-dire que, pour chaque règle $l \to r$ de ce système, le terme l est supérieur au terme r. En voici un exemple :

Exemple 1.2 Soit le système suivant, décrivant un anneau :

$$\begin{array}{rcl} x+0 & \to & x \\ x+(-x) & \to & 0 \\ x*(y+z) & \to & (x*y)+(x*z) \\ (x+y)*z & \to & (x*z)+(y*z) \end{array}$$

La terminaison de ce système est prouvée en interprétant les opérateurs par

$$\begin{array}{rclcrcl} [+](X,Y) & = & X+Y+1 & \qquad & [-](X) & = & 2X \\ [*](X,Y) & = & XY & & [0] & = & 2 \end{array}$$

En effet, la première règle donne $X+3 > X$, la deuxième $3X+1 > 2$ (car X est supérieur ou égal à 2, interprétation de 0), la troisième $XY + XZ + X > XY + XZ + 1$ pour la même raison, et la quatrième $XZ + YZ + Z > XZ + YZ + 1$. ♦

Cependant, le principal reproche fait à ces techniques d'interprétation polynômiale est la nécessité de chercher des polynômes adéquats pour chaque exemple traité, ce qui est d'autant plus complexe lorsque le nombre d'opérateurs manipulés est élevé.

1.2.3 Interprétations AC

A ce jour, il n'existe que deux ordres ne limitant pas le nombre d'opérateurs AC. Il s'agit des ordres de P. Narendran et M. Rusinowitch [NR91], utilisant les interprétations polynômiales, et de A. Rubio et R. Nieuwenhuis [RN93, Rub94], basé sur le RPO.

I Ordre de P. Narendran et M. Rusinowitch

Cet ordre est un ordre de simplification AC-compatible et total sur les classes de congruence AC. Il est défini sur les termes clos par une interprétation dans les anneaux commutatifs libres. Ainsi, à chaque symbole de fonction f est associé une indéterminée X_f et l'interprétation I est calculée par :

- $I(f(t_1,\ldots,t_n)) = (X_f+1)((X_f^2+2X_f)I(t_1)\ldots I(t_n) + (X_f+1)(I(t_1)+\ldots+I(t_n))+1)$, pour tout opérateur f d'arité $n>0$
- $I(a) = X_a + 1$, pour toute constante a

Il faut noter que cette interprétation respecte la condition définie par A. Ben Cherifa sur la forme des polynômes interprétant un opérateur AC (Proposition 1.9), car les coefficients a, b et c sont respectivement $(X_f+1)(X_f^2+2X_f)$, $(X_f+1)(X_f+1)$ et (X_f+1), et $ac+b-b^2$ est bien égal à 0.

Utilisant un ordre $>_N$ pour comparer deux polynômes, basé sur une précédence totale entre les indéterminées X_f ($f \in \mathcal{F}$), l'ordre sur les termes clos est défini par :

Définition 1.10 $s = f(s_1,\ldots,s_m) \succ g(t_1,\ldots,t_n) = t$ *si*

- *soit* $I(s) >_N I(t)$,
- *soit* $I(s) = I(t)$ *et*
  - *si* $f \in \mathcal{F}_{AC}$, $\mathcal{H}terms(s,f) \succ^{mult} \mathcal{H}terms(t,f)$,
  - *si* $f \notin \mathcal{F}_{AC}$, $(s_1,\ldots,s_m) \succ^{lex} (t_1,\ldots,t_n)$.

Cet ordre est bien fondé, monotone et total sur les classes d'AC-congruence, ce qui implique la propriété de sous-terme. De plus, il est AC-compatible.

Le seul reproche qui puisse être fait à cet ordre est qu'il nécessite des calculs de polynômes pouvant être coûteux. Cependant, il est possible d'utiliser des polynômes plus simples pour interpréter des opérateurs non AC, comme l'a fait C. Marché dans sa thèse [Mar93].

II Ordre de A. Rubio et R. Nieuwenhuis

Contrairement à l'ordre précédent, cet ordre est défini de manière très classique puisqu'il est basé sur le RPO (Définition 1.5) et nécessite une précédence totale $>_{\mathcal{F}}$ sur les opérateurs. L'interprétation I d'un terme clos consiste à l'aplatir au niveau des opérateurs AC, puis à le transformer suivant les règles suivantes, où $\perp$ est la plus petite constante pour l'ordre de précédence :

- $f(\ldots, c, \ldots) \rightarrow f(\ldots, \perp, \ldots)$
  si $f \in \mathcal{F}_{AC}$, c est une constante ($\neq \perp$), et $f >_{\mathcal{F}} c$
- $f(\ldots, g(t_1, \ldots, t_n), \ldots) \rightarrow f(\ldots, t_j, \ldots)$
  si $f \in \mathcal{F}_{AC}$, $f >_{\mathcal{F}} g$, et t_j est maximal dans $\{t_1, \ldots, t_n\}$ pour l'ordre $>_{RPO}$

L'interprétation $I(s)$ d'un terme s consiste donc à lui trouver une forme normale en l'aplatissant et en appliquant les deux règles précédentes en suivant la stratégie en profondeur d'abord et toujours le plus à gauche possible. L'ordre entre deux termes clos est défini ainsi :

Définition 1.11 $s = f(s_1, \ldots, s_m) \succ g(t_1, \ldots, t_n) = t$ *si*

- *soit* $I(s) >_{RPO} I(t)$,
- *soit* $I(s) =_{AC} I(t)$ *et*
  - *si* $f \in \mathcal{F}_{AC}$, $\mathcal{H}terms(s, f) \succ^{mult} \mathcal{H}terms(t, f)$,
  - *si* $f \notin \mathcal{F}_{AC}$, $(s_1, \ldots, s_m) \succ^{lex} (t_1, \ldots, t_n)$.

Cet ordre a été étendu aux termes avec variables, en modifiant la seconde règle de réécriture dans la transformation : plusieurs termes t_j pouvant être susceptibles d'être maximaux dans $\{t_1, \ldots, t_n\}$, il faut créer un terme $f(\ldots, t_j, \ldots)$ pour chacun de ces t_j.

L'interprétation d'un terme donnant un ensemble de formes normales, la définition de l'ordre $\succ$ doit être modifiée :

Définition 1.12 $s = f(s_1, \ldots, s_m) \succ g(t_1, \ldots, t_n) = t$ *si*

- *soit,* $\forall t' \in I(t),\ \exists s' \in I(s),\ s' >_{RPO} t'$
- *soit,* $\forall t' \in I(t),\ \exists s' \in I(s),\ s' \geq_{RPO} t'$ *et*
  - *si* $f \in \mathcal{F}_{AC}$, $\mathcal{H}terms(s, f) \succ^{mult} \mathcal{H}terms(t, f)$,
  - *si* $f \notin \mathcal{F}_{AC}$, $(s_1, \ldots, s_m) \succ^{lex} (t_1, \ldots, t_n)$.

L'ordre obtenu est un ordre de simplification total (Définition 1.1).

Cependant, il possède un inconvénient majeur : étant donnés deux opérateurs AC $+$ et $*$, tels que $* >_{\mathcal{F}} +$; l'axiome de distributivité de $*$ par rapport à $+$ est orienté dans le sens inverse de ce qui est nécessaire : $x * (y + z) \prec (x * y) + (x * z)$. En effet, $I(x * (y + z)) = \{x * y, x * z\}$ et $I((x * y) + (x * z)) = \{(x * y) + (x * z)\}$, et le terme $(x * y) + (x * z)$ est supérieur aux termes $x * y$ et $x * z$.

A. Rubio a montré dans sa thèse [Rub94] que cet ordre peut être adapté aux théories associatives uniquement associatives.

1.3 Ordre total sur les classes de congruence

Les ordres sont utilisés pour limiter les applications des règles d'inférence. Tout au long de ce document, nous allons utiliser une stratégie dite « ordonnée », qui va consister à n'appliquer des déductions que si elles sont effectuées entre les littéraux maximaux des clauses ; de plus, lorsqu'une égalité sera utilisée pour effectuer un remplacement, nous vérifierons qu'il s'agit bien du remplacement du plus grand membre de l'égalité par le plus petit. Nous allons donc devoir comparer des termes ou des atomes, et la complétude des règles d'inférence que nous définirons sera basée sur l'utilisation d'un *ordre de simplification total*.

En fait, nous avons besoin d'un ordre total sur les classes de congruence engendrées par la théorie E. Voici comment nous le définissons :

Définition 1.13 (E-TSO sur les termes) *Soit $>$ un TSO E-compatible sur les termes. Un ordre total sur les classes de congruence engendrées par la théorie E (E-TSO pour abréger), noté $\succ$, est défini par : $s \succ t$ si $s > t$ et $s \neq_E t$.*

Il est aisé de vérifier que cet ordre $\succ$ est nœthérien, monotone, stable par instanciation, et possède les propriétés de sous-terme et de E-compatibilité.

Nous avons également besoin d'un ordre sur les atomes, qui est simplement défini par extension de l'ordre $>$ comme suit. Notons que cette extension préserve la propriété d'E-compatibilité sur les atomes.

Définition 1.14 (TSO sur les atomes) *Soit $>_{\mathcal{P}}$ une précédence totale sur les symboles de prédicat, "$\simeq$" étant le plus petit. Soit $>$ un TSO E-compatible sur les termes. Nous étendons cet ordre $>$ sur l'ensemble des atomes par :*

$P(s_1, \ldots, s_n) \; > \; Q(t_1, \ldots, t_m)$ *si*

- *soit* $P >_{\mathcal{P}} Q$,
- *soit* $P = Q$, *P n'est pas le prédicat d'égalité, et*
  - *soit* $(s_1, \ldots, s_n) \succ^{lex} (t_1, \ldots, t_n)$,
  - *soit* $P(s_1, \ldots, s_n) =_E Q(t_1, \ldots, t_n)$ *et* $(s_1, \ldots, s_n) >^{lex} (t_1, \ldots, t_n)$,
- *soit* $P = Q$, *P est le prédicat d'égalité, et, en supposant que* $s_1 \geq s_2$ *et* $t_1 \geq t_2$,
  - *soit* $\{s_1, s_2\} \succ^{mult} \{t_1, t_2\}$,
  - *soit* $\{s_1, s_2\} =_E \{t_1, t_2\}$ *et* $\{s_1, s_2\} >^{mult} \{t_1, t_2\}$.

Il est facile de vérifier que cet ordre sur les atomes est aussi un TSO. De même, nous l'étendons sur les classes de congruence :

Définition 1.15 (E-TSO sur les atomes) *Soit $>$ un TSO sur les atomes défini comme ci-dessus. L'ordre $\succ$ est étendu aux atomes par : $A \succ B$ si $A > B$ et $A \neq_E B$.*

Nous avons donc défini un ordre permettant de comparer deux atomes. Mais, dans les règles d'inférence que nous définirons plus tard, les comparaisons porteront sur des littéraux. En général, *comparer deux littéraux* revient à comparer les atomes correspondants. Mais, nous verrons que pour la stratégie de superposition, nous aurons besoin d'un ordre spécifique sur les littéraux (Définition 2.8).

2
Paramodulation modulo une théorie équationnelle

Nous présentons dans ce chapitre différents systèmes de règles d'inférence pour effectuer de la déduction dans une théorie équationnelle E. Ces systèmes sont basés sur les règles de paramodulation et de superposition, et font appel à une stratégie ordonnée obligeant à n'effectuer des inférences qu'entre les littéraux maximaux des clauses. Nous montrerons qu'un raffinement de cette stratégie ordonnée est également possible ; il s'agit de la stratégie positive.
Nous définissons ensuite des règles de simplification, dont le but est d'éliminer des clauses redondantes. Ces règles sont compatibles avec tous les systèmes d'inférence décrits dans ce chapitre.

Ces systèmes d'inférence sont réfutationnellement complets. La preuve, basée sur la technique des arbres sémantiques transfinis [HR91, Rus89], sera décrite dans le Chapitre 4.

Nous clorons ce chapitre par une présentation des travaux voisins effectués par G.D. Plotkin [Plo72], U. Wertz [Wer92] et E. Paul [Pau94].

Mais, commençons par étudier les différents problèmes soulevés pour effectuer des déductions modulo une théorie équationnelle. Notons immédiatement que la *théorie E considérée doit être régulière*. Cette condition n'apparaîtra pas dans les définitions des règles d'inférence, mais elle est indispensable pour la complétude de ces systèmes d'inférence comme nous le verrons dans le Chapitre 4.

2.1 Problématique

Pour travailler dans une théorie équationnelle E, nous avons besoin d'un ordre de simplification $\succ$, total sur les classes de congruence et E-compatible, défini sur les termes et les atomes, comme décrit dans le chapitre précédent (Définition 1.15). Nous avons également besoin d'un algorithme de E-unification.

A partir de cela, nous pouvons définir des règles d'inférence, représentées sous la forme de fractions avec les clauses initiales au-dessus de la barre et la clause déduite en-dessous. Ces règles, classiques en déduction automatique, sont la résolution

$$\frac{A_1 \vee D_1 \qquad \neg A_2 \vee D_2}{D_1\sigma \vee D_2\sigma} \quad \text{où } \sigma \text{ est un } E\text{-unificateur principal de } A_1 \text{ et } A_2,$$

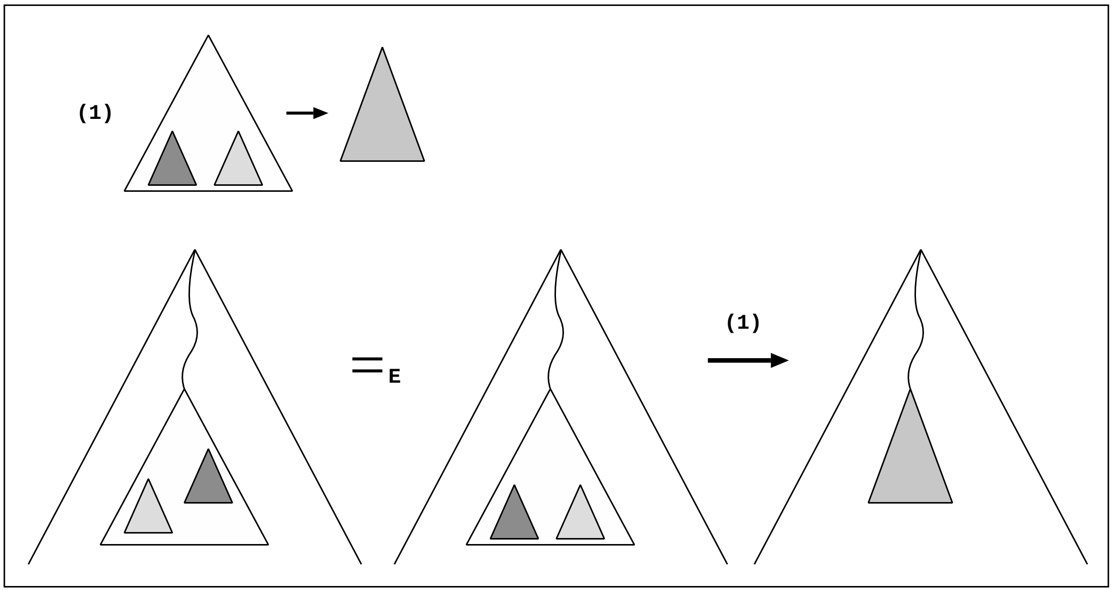


Figure 2.1: E-Paramodulation

et la paramodulation

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2 \vee D_2}{L_2[p_2 \leftarrow r_1]\sigma \vee D_1\sigma \vee D_2\sigma}$$ où σ est un E-unificateur principal de l_1 et du sous-terme à la position p_2 de L_2.

Le rôle de l'étape de E-unification dans cette dernière règle est schématisé dans la Figure 2.1, où deux termes (triangles) ont le même grisé s'ils sont E-unifiables.

Cependant, cette règle ne permet pas de traduire tous les remplacements possibles modulo les axiomes de la théorie E, comme le montre l'exemple suivant :

Exemple 2.1 Supposons que E représente la propriété d'associativité d'un opérateur f : $E = \{(f(f(x,y),z) \simeq f(x,f(y,z)))\}$. Le système de clauses suivant

$$\begin{cases} (f(a,b) \simeq c) & (1) \\ P(f(a,f(b,d))) & (2) \\ \neg P(f(c,d)) & (3) \end{cases}$$

est E-incohérent, mais les règles de résolution et de paramodulation dont nous disposons ne permettent pas de déduire une contradiction. En effet, pour pouvoir appliquer un remplacement grâce à la clause (1), il faut savoir que le sous-terme $f(a,f(b,d))$ de la clause (2) est E-égal à $f(f(a,b),d)$. Cette information ne peut être gérée par l'algorithme d'unification et la règle de paramodulation, car le remplacement n'est pas effectué à la même position que l'étape de E-égalité. ♦

La solution est d'appliquer le remplacement à la position de l'étape de E-égalité, en utilisant ce qui est appelé une *extension* de l'équation invoquée. Appliqué à l'exemple précédent, cela donne :

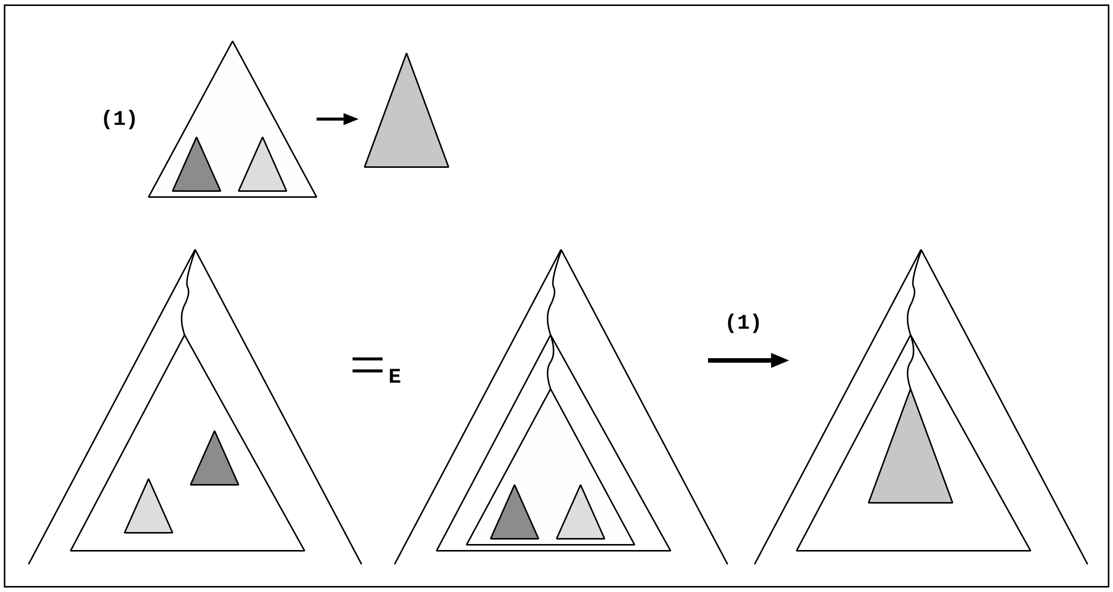


Figure 2.2: E-Paramodulation contextuelle

Exemple 2.2 (Suite de l'Exemple 2.1) Une extension de la clause (1) est créée en remplaçant le sous-terme $f(x, y)$ de l'axiome d'associativité par c, après l'avoir unifié avec $f(a, b)$ grâce à la substitution $\{x \mapsto a, y \mapsto b\}$.

$$\frac{(f(a,b) \simeq c) \qquad\qquad (f(\boldsymbol{f(x,y)}, z) \simeq f(x, f(y,z)))}{(f(\boldsymbol{c}, z) \simeq f(a, f(b, z)))}$$

Notons (1_{ext}) cette extension $(f(a, f(b, z)) \simeq f(c, z))$. Alors, une paramodulation de (1_{ext}) dans la clause (2) produit la clause $P(f(c, d))$ (4), la variable z ayant été instanciée par d. Une étape de résolution entre les clauses (3) et (4) engendre alors une contradiction, symbolisée par la clause vide $\Box$ (5). ♦

Nous rassemblons ces deux étapes, extension puis remplacement, dans une règle, appelée *paramodulation contextuelle* et schématisée par la Figure 2.2, qui permet donc de ne remplacer qu'un sous-terme (modulo des étapes de E-égalité) du terme à la position choisie. Le *contexte* est le terme étendant le membre gauche de l'équation utilisée ; dans l'exemple précédent, $f([\![\,]\!], d)$ est le contexte du remplacement (le symbole $[\![\,]\!]$ représente un sous-terme vide).

Ajouter cette nouvelle règle d'inférence ne suffit pas pour traiter complètement la théorie E. Il reste encore des déductions que nous n'arrivons pas à dériver comme le montre l'exemple suivant.

Exemple 2.3 L'opérateur f étant toujours associatif, le système suivant est E-incohérent.

$$\begin{cases} (f(a,b) \simeq c) & (1) \\ (f(b,d) \simeq e) & (2) \\ P(f(c,d)) & (3) \\ \neg P(f(a,e)) & (4) \end{cases}$$

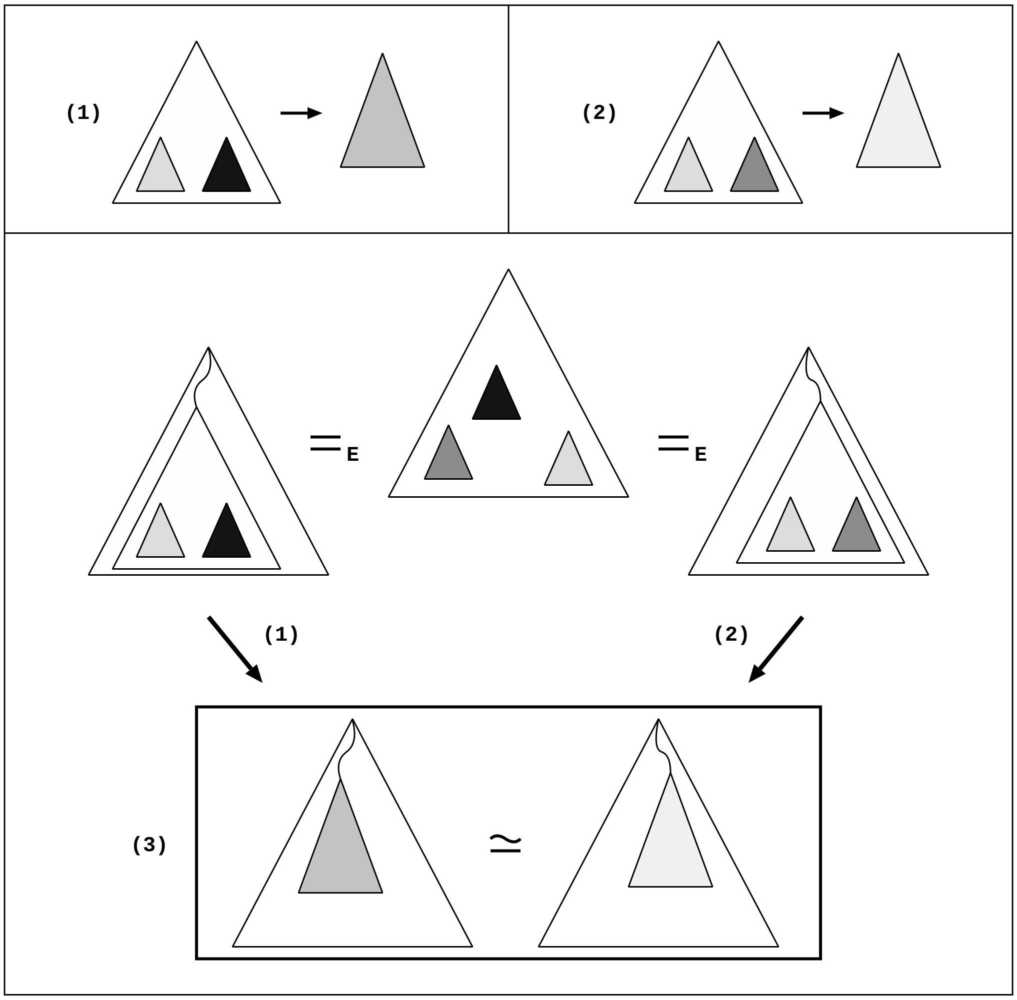


Figure 2.3: E-Paramodulation étendue

En effet, l'instance $(f(f(a,b),d) \simeq f(a,f(b,d)))$ de l'axiome d'associativité de f est réductible dans son membre gauche par la clause (1) et dans son membre droit par la clause (2), ce qui permet de déduire que l'équation $(f(c,d) \simeq f(a,e))$ est une conséquence logique du système précédent et de E. De cette nouvelle équation, une contradiction peut être dérivée entre les clauses (3) et (4).

Or, nous sommes dans l'impossibilité de montrer cette incohérence, car aucun remplacement ne peut être effectué avec nos règles de paramodulation et de paramodulation contextuelle. La raison en est simple : la déduction précédente a été faite entre une extension de la clause (1) et une extension de la clause (2), ce qui n'est pas prévu par nos règles. ♦

La solution est donc d'ajouter une nouvelle règle d'inférence appliquée entre deux clauses étendues, qui consiste à unifier les membres gauches des équations étendues pour déduire l'égalité des deux membres droits. Nous avons appelé cette règle *paramodulation étendue* et la Figure 2.3 schématise son rôle.

Voici ce que donne cette nouvelle règle lorsqu'elle est appliquée à l'exemple précédent :

Exemple 2.4 (Suite de l'Exemple 2.3) Les deux extensions sont créées ainsi :

$$\frac{(f(a,b)\simeq c)\ \ (1) \qquad\qquad (f(\boldsymbol{f(x,y)},z)\simeq f(x,f(y,z)))}{(f(\boldsymbol{c},z)\simeq f(a,f(b,z)))\ \ (1_{ext})}$$

$$\frac{(f(b,d)\simeq e)\ \ (2) \qquad\qquad (f(f(x,y),z)\simeq f(x,\boldsymbol{f(y,z)}))}{(f(f(x,b),d)\simeq f(x,\boldsymbol{e}))\ \ (2_{ext})}$$

Ensuite, un remplacement est effectué dans la clause (1_{ext}), en utilisant la clause (2_{ext}), grâce à la substitution $\{x \mapsto a, z \mapsto d\}$ qui unifie les termes $f(a, f(b, z))$ et $f(f(x, b), d)$. La clause engendrée, $(f(c,d)\simeq f(a,e))$ (5), va permettre de trouver une contradiction entre les clauses (3) et (4). ♦

Finalement, ces deux nouvelles règles de paramodulation simulent le rôle des extensions de clauses, engendrées habituellement par les axiomes de E. Et comme elles évitent de créer explicitement ces extensions, elles évitent également la mise en place du contrôle classiquement utilisé pour les gérer [PS81, Wer92, Pau94]. D'autre part, comme nous le verrons dans la suite de ce chapitre ainsi que dans le chapitre suivant, leur application peut être limitée grâce à des conditions assez simples.

2.2 Règles d'inférence

Nous définissons dans cette section les règles d'inférence pour les stratégies de paramodulation et de superposition. Elles font appel à un algorithme d'unification dans une théorie équationnelle E, et utilisent un ordre de simplification total sur les classes de congruence et E-compatible (E-TSO, Définition 1.15).

2.2.1 Stratégie de paramodulation

Ces règles d'inférence sont basées sur les règles classiques de résolution [Rob65] et de paramodulation [RW69], et manipulent les extensions engendrées par la théorie équationnelle E grâce aux règles de paramodulation contextuelle et étendue. La stratégie de déduction est dite *ordonnée* [Rus89] car, grâce à un ordre $\succ$, E-TSO, nous vérifions que chaque étape d'inférence est appliquée entre les littéraux maximaux des clauses concernées, et chaque terme remplacé doit l'être par un plus petit.
L'ordre $\succ$ n'étant pas total sur les termes et les atomes non clos, deux objets peuvent être incomparables ; donc, nous nous contentons de vérifier que l'objet devant être maximal n'est pas inférieur ou égal (noté $\not\preceq$) aux autres. De plus, la notation $L \not\preceq D$, où L est un littéral et D une clause, signifie qu'aucun atome de D n'est supérieur ou égal à l'atome correspondant à L.
Chaque règle fait appel à des clauses dont les ensembles de variables sont disjoints. Un exemple illustre leur application dans lequel la théorie E représente l'associativité d'un opérateur f : $E = \{(f(f(x,y),z)\simeq f(x,f(y,z)))\}$; pour ne pas surcharger ces exemples, nous supposerons dans chacun d'eux que les conditions d'ordre sont satisfaites pour un ordre et une précédence sur les opérateurs donnés.

La première règle est la factorisation, ayant pour but d'engendrer des clauses dont le littéral maximal ne possède qu'une seule occurrence.

Définition 2.1 (*E*-Factorisation)

$$\frac{L_1 \vee \ldots \vee L_n \vee D}{L_1\sigma \vee D\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{L_1 =^?_E L_2, \ldots, L_1 =^?_E L_n\}) \\ L_1\sigma \not\preceq D\sigma \end{cases}$$

Exemple 2.5 Une factorisation de la clause $P(f(a,y)) \vee P(f(f(x,b),x)) \vee Q(x,y)$ utilisant l'unificateur principal $\{x \mapsto a,\ y \mapsto f(b,a)\}$ produit la clause $P(f(a,f(b,a))) \vee Q(a,f(b,a))$. ♦

La règle suivante est la résolution dont nous avons vu une description sommaire au début de ce chapitre. Dans ce document, nous utiliserons toujours une règle de résolution binaire, c'est-à-dire ne traitant qu'un seul littéral par clause, contrairement à la règle de résolution de G.D. Plotkin (cf. Section 2.5.1).

Définition 2.2 (*E*-Résolution)

$$\frac{A_1 \vee D_1 \qquad \neg A_2 \vee D_2}{D_1\sigma \vee D_2\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{A_1 =^?_E A_2\}) \\ A_1\sigma \not\preceq D_1\sigma \vee D_2\sigma \end{cases}$$

Notons que toute résolution appliquée sur des littéraux équationnels est inutile, car elle équivaut à une paramodulation suivie d'une réflexion (cf. définitions suivantes).

Exemple 2.6 Si une résolution est effectuée entre la clause $P(f(a,y)) \vee Q(y)$ et la clause $\neg P(f(f(x,b),x)) \vee Q(x)$, avec la substitution $\sigma = \{x \mapsto a,\ y \mapsto f(b,a)\}$, la clause déduite est $Q(f(b,a)) \vee Q(a)$. ♦

L'utilisation du prédicat d'égalité a entraîné la création de la règle de paramodulation pour remplacer les axiomes décrivant les propriétés de ce prédicat. Mais, l'un de ces axiomes n'est pas traduit par cette règle ; il s'agit de la réflexivité $(x \simeq x)$. La règle suivante évite d'ajouter cet axiome à l'ensemble initial de clauses. Elle simule une étape de résolution avec $(x \simeq x)$.

Définition 2.3 (*E*-Réflexion)

$$\frac{\neg(l \simeq r) \vee D}{D\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{l =^?_E r\}) \\ (l \simeq r)\sigma \not\preceq D\sigma \end{cases}$$

Exemple 2.7 Une réflexion appliquée à la clause $\neg(f(a,y) \simeq f(f(a,b),x)) \vee (f(x,y) \simeq c)$ produit $(f(x,f(b,x)) \simeq c)$, en utilisant la substitution $\{y \mapsto f(b,x)\}$. ♦

Il est à noter que seules les règles de résolution et de réflexion peuvent engendrer la clause vide $(\square)$, car ce sont les seules règles qui suppriment des littéraux.

Définition 2.4 (*E*-Paramodulation)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2 \vee D_2}{L_2[p_2 \leftarrow r_1]\sigma \vee D_1\sigma \vee D_2\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{L_2|_{p_2} =^?_E l_1\}) \quad où \quad p_2 \in \mathcal{FP}os(L_2) \\ (l_1 \simeq r_1)\sigma \not\preceq D_1\sigma \\ L_2\sigma \not\preceq D_2\sigma \quad et \quad L_2\sigma \not\prec (l_1 \simeq r_1)\sigma \\ l_1\sigma \not\preceq r_1\sigma \end{cases}$$

Une paramodulation est donc appliquée à une position non variable du littéral L_2, à condition d'avoir bien orienté l'équation $(l_1 \simeq r_1)$. En plus de la maximalité des littéraux utilisés, une condition précise que l'équation servant au remplacement ne doit pas être plus grande que l'atome dans lequel le remplacement est effectué. La condition suivante peut être ajoutée :

> *Si le littéral L_2 est une équation négative $\neg(l_2 \simeq r_2)$, il faut vérifier que le remplacement a été appliqué dans le plus grand membre de cette équation, c'est-à-dire, en supposant que la position p_2 appartient à l_2 : $l_2\sigma \npreceq r_2\sigma$.*

Exemple 2.8 Une paramodulation de la clause $(f(a,y) \simeq c) \vee (g(y) \simeq d)$ à la position 1 du littéral $P(f(x, f(b,z)))$ de la clause $P(f(f(x,b),z)) \vee Q(x,z)$, utilisant la substitution $\sigma = \{x \mapsto a,\ y \mapsto f(b,z)\}$, produit la clause $P(c) \vee (g(f(b,z)) \simeq d) \vee Q(a,z)$. ♦

La règle d'inférence suivante est la paramodulation contextuelle, simulant une paramodulation avec une extension, grâce à la notion de contexte que nous étudierons dans le prochain chapitre. Nous avons vu dans la section précédente qu'une extension d'une équation $(l \simeq r)$ était engendrée grâce à une paramodulation dans un axiome $(e \simeq e')$ de E, à une position p ; l'extension est alors $(e' \simeq e[r]_p)\sigma$, pour un E-unificateur σ de l et $e|_p$. Mais, comme les termes e et e' sont E-égaux, cette extension est équivalente à $(e[l]_p \simeq e[r]_p)\sigma$. Le terme e et la position p indiquent donc le contexte de l'extension. Plus formellement, nous définissons un *contexte* par un triplet (e,p,c) où c est une conjonction de problèmes de E-unification apparus lors de la construction du terme e. $\mathcal{C}ont(l)$ désigne un ensemble de contextes $(e[l]_p, p, c)$ associés au terme l. Pour simplifier les notations dans les règles d'inférence suivantes, le terme $e[l]_p$ du contexte sera noté e.
Nous ne détaillons pas plus cette notion de contexte ici, et nous renvoyons le lecteur au Chapitre 3 entièrement consacré à ce sujet.

Définition 2.5 (E-Paramodulation contextuelle)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad\qquad L_2 \vee D_2}{L_2[p_2 \leftarrow e_1[r_1]_{p_1}]\sigma \vee D_1\sigma \vee D_2\sigma} \quad si \quad \begin{cases} (e_1,p_1,c_1) \in \mathcal{C}ont(l_1),\ \ p_2 \in \mathcal{FP}os(L_2) \\ \sigma \in \mathcal{S}ol(\{L_2|_{p_2} =^?_E e_1\} \cup c_1) \\ (l_1 \simeq r_1)\sigma \npreceq D_1\sigma \\ L_2\sigma \npreceq D_2\sigma \quad et \quad L_2\sigma \nprec (l_1 \simeq r_1)\sigma \\ l_1\sigma \npreceq r_1\sigma \\ \forall x \in \mathcal{V}ar(L_2|_{p_2}),\ l_1\sigma \ntrianglelefteq_E x\sigma \end{cases}$$

Les conditions d'application de cette règle sont sensiblement les mêmes que celles de la paramodulation. En plus, la substitution utilisée doit satisfaire les contraintes d'unification soulevées lors de l'élaboration du contexte utilisé. La dernière condition stipule que le terme à remplacer dans $L_2|_{p_2}$ ne doit pas avoir été introduit par la substitution, c'est-à-dire ne doit pas être un sous-terme d'un terme associé à une variable ; ce test est très important, car si la condition n'est pas vérifiée, l'inférence reviendrait à une paramodulation dans une variable, connue comme source d'inefficacité.
Comme pour la paramodulation, nous pouvons ajouter une condition précisant que toute paramodulation contextuelle doit s'appliquer dans le plus grand membre des équations négatives. Bien que l_1 soit un sous-terme strict de L_2, nous avons quand même précisé la condition

$L_2\sigma \not\prec (l_1 \simeq r_1)\sigma$, car la théorie E peut contenir des axiomes ne préservant pas la propriété de sous-terme.

Exemple 2.9 Une paramodulation contextuelle, appliquée de la clause $(f(y,b) \simeq c) \vee (g(y) \simeq d)$ dans le sous-terme $f(a, f(x,d))$ de la clause $P(f(a, f(x,d))) \vee Q(x)$, utilisant le contexte $(f(f(x_1,x_2),x_3), 1, \{f(x_1,x_2) =_E^? f(y,b)\})$ de $\mathcal{C}ont(f(y,b))$ et la substitution $\{x, x_2 \mapsto b,\ y, x_1 \mapsto a,\ x_3 \mapsto d\}$, engendre la clause $P(f(c,d)) \vee (g(a) \simeq d) \vee Q(b)$. ♦

La dernière règle d'inférence est la paramodulation étendue, simulant une paramodulation entre deux extensions.

Définition 2.6 (E-Paramodulation étendue)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad\qquad (l_2 \simeq r_2) \vee D_2}{(e_1[r_1]_{p_1} \simeq e_2[r_2]_{p_2})\sigma \vee D_1\sigma \vee D_2\sigma} \quad si \quad \begin{cases} (e_1, p_1, c_1) \in \mathcal{C}ont(l_1),\ (e_2, p_2, c_2) \in \mathcal{C}ont(l_2) \\ \sigma \in \mathcal{S}ol(\{e_1 =_E^? e_2\} \cup c_1 \cup c_2) \\ (l_1 \simeq r_1)\sigma \not\preceq D_1\sigma \quad et \quad (l_2 \simeq r_2)\sigma \not\preceq D_2\sigma \\ l_1\sigma \not\preceq r_1\sigma \quad et \quad l_2\sigma \not\preceq r_2\sigma \\ \forall x \in \mathcal{V}ar(l_2),\ l_1\sigma \not\trianglelefteq_E x\sigma \\ \forall x \in \mathcal{V}ar(l_1),\ l_2\sigma \not\trianglelefteq_E x\sigma \end{cases}$$

Cette règle s'applique en unifiant des contextes étendant le plus grand membre de l'équation maximale de chacune des deux clauses. Les deux dernières conditions vérifient que le plus grand membre d'une équation n'est pas introduit par l'instanciation d'une variable dans l'autre membre.

Exemple 2.10 Une paramodulation étendue entre les clauses $(f(x,b) \simeq c) \vee (g(x) \simeq x)$ et $(f(b, f(y,d)) \simeq y) \vee (h(y) \simeq b)$, utilisant les contextes $(f(f(x_1,x_2),x_3), 1, \{f(x_1,x_2) =_E^? f(x,b)\})$ de $\mathcal{C}ont(f(x,b))$ et $(f(y_1, f(y_2,y_3)), 2, \{f(y_2,y_3) =_E^? f(b, f(y,d))\})$ de $\mathcal{C}ont(f(b, f(y,d)))$, et la substitution $\{x_1, y_1 \mapsto x,\ x_2, y_2 \mapsto b,\ x_3, y_3 \mapsto f(y,d)\}$, permet de déduire la clause suivante : $(f(c, f(y,d)) \simeq f(x,y)) \vee (g(x) \simeq x) \vee (h(y) \simeq b)$. ♦

Les exemples que nous avons donnés pour chaque règle d'inférence ne considèrent à chaque déduction qu'une seule substitution solution, mais la plupart d'entre eux possèdent plusieurs solutions et, pour préserver la complétude, il est indispensable d'engendrer une clause pour chacune de celles-ci. La théorie considérée étant l'associativité, un problème d'unification peut avoir une infinité de solutions ; par exemple, $f(x,a) =_E^? f(a,x)$ a pour solutions les instanciations de x par a, $f(a,a)$, $f(a, f(a,a))$, ...
Pour de telles théories, il est donc nécessaire d'utiliser un semi-algorithme produisant un nombre fini de solutions, et d'intercaler la production avec d'autres étapes de déduction.

Soit *INF* l'ensemble des règles d'inférence définies dans cette section, pour la stratégie de E-paramodulation. Enonçons le théorème de E-complétude de *INF*.

Théorème 2.7 (Complétude de la E-paramodulation) *Si S est un ensemble de clauses E-incohérent, $INF^*(S)$ contient la clause vide.*

Ce théorème sera démontré dans la Section 4.3.

2.2.2 Stratégie de superposition

La stratégie de superposition est une variante très intéressante de la stratégie de paramodulation pour faire de la complétion. La principale différence réside dans le fait qu'une superposition n'est jamais appliquée dans le plus petit membre d'un littéral équationnel. La définition de cette stratégie se fait donc par l'ajout de règles traitant des remplacements dans les littéraux équationnels, et surtout par l'introduction d'un *nouvel ordre* $\succ_L$ *de comparaison des littéraux équationnels*, qui ne consiste plus à simplement comparer les atomes correspondants.

Définition 2.8 (Ordre sur les littéraux) *L'ordre $\succ_L$ entre deux littéraux est défini par :*

- $(l_1 \simeq r_1) \succ_L (l_2 \simeq r_2)$ *si* $(l_1 \simeq r_1) \succ (l_2 \simeq r_2)$,
- $(l_1 \simeq r_1) \succ_L \neg(l_2 \simeq r_2)$ *si* $(l_1 \simeq r_1) \succ (l_2 \simeq l_2)$ *et* $(l_1 \simeq r_1) \succ (r_2 \simeq r_2)$,
- $\neg(l_1 \simeq r_1) \succ_L (l_2 \simeq r_2)$ *si* $(l_1 \simeq r_1) \succ (l_2 \simeq \perp)$ *et* $(l_1 \simeq r_1) \succ (r_2 \simeq \perp)$, *où* $\perp$ *représente une constante minimale pour la précédence, n'appartenant pas à* $\mathcal{F}$,
- $\neg(l_1 \simeq r_1) \succ_L \neg(l_2 \simeq r_2)$ *si* $(l_1 \simeq r_1) \succ (l_2 \simeq r_2)$.

Toute comparaison entre d'autres littéraux revient à utiliser directement l'ordre $\succ$ sur les atomes correspondants.

Cet ordre doit remplacer $\succ$ dans les règles d'inférence définies pour la stratégie de paramodulation, à chaque fois que des littéraux sont comparés. Il apparaît dans les règles sous la forme $L \not\preceq_L D$ pour dire qu'aucun littéral de D n'est supérieur ou égal au littéral L.

Définissons maintenant les règles de superposition et de superposition contextuelle. Nous ne décrivons pas la règle de *superposition étendue* car la seule différence avec la paramodulation étendue (Définition 2.6) est l'utilisation de l'ordre sur les littéraux, modification immédiate dès que l'on utilise la stratégie de superposition.

Définition 2.9 (*E*-Superposition) *Cette règle se substitue à la E-Paramodulation (Définition 2.4) lorsque le littéral L_2, dans lequel le remplacement est effectué, est de la forme $(l_2 \simeq r_2)$ ou $\neg(l_2 \simeq r_2)$. Outre la manipulation d'un ordre sur les littéraux, les modifications suivantes sont à effectuer :*

- *ajouter* $l_2\sigma \not\preceq r_2\sigma$, *en supposant que le remplacement a lieu dans* l_2,
- *remplacer* $L_2\sigma \not\prec (l_1 \simeq r_1)\sigma$ *par* $L_2\sigma \not\preceq_L (l_1 \simeq r_1)\sigma$.

Définition 2.10 (*E*-Superposition contextuelle) *Cette règle se substitue à la E-paramodulation contextuelle (Définition 2.5) lorsque le littéral L_2, dans lequel le remplacement est effectué, est de la forme $(l_2 \simeq r_2)$ ou $\neg(l_2 \simeq r_2)$. La condition à ajouter est :*

- $l_2\sigma \not\preceq r_2\sigma$, *en supposant que le remplacement a lieu dans* l_2,

Une nouvelle règle d'inférence doit être ajoutée au système. Elle s'appelle *factorisation équationnelle* et a pour but de dériver des clauses dans lesquelles aucun littéral équationnel positif ne possède un membre gauche E-unifiable avec le membre gauche de l'équation positive maximale de cette clause.

Définition 2.11 (*E*-Factorisation équationnelle)

$$\frac{(l_1 \simeq r_1) \vee (l_2 \simeq r_2) \vee D}{(l_2 \simeq r_2)\sigma \vee \neg(r_1 \simeq r_2)\sigma \vee D\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{l_1 =^?_E l_2\}) \\ (l_1 \simeq r_1)\sigma \not\preceq_L (l_2 \simeq r_2)\sigma \vee D\sigma \\ l_1\sigma \not\preceq r_1\sigma \end{cases}$$

Ainsi cette règle remplace l'équation maximale par une diséquation (équation négative) formée avec les deux membres droits.

Soit *INF* l'ensemble des règles d'inférence définies dans cette section, pour la stratégie de E-superposition, c'est-à-dire les règles de E-factorisation, E-factorisation équationnelle, E-résolution, E-réflexion, E-superposition, E-superposition contextuelle et E-superposition étendue, auxquelles viennent s'ajouter les règles de E-paramodulation et E-paramodulation contextuelle applicables uniquement dans des littéraux non équationnels. Enonçons le théorème de E-complétude de *INF*.

Théorème 2.12 (Complétude de la *E*-superposition) *Si S est un ensemble de clauses E-incohérent, $INF^*(S)$ contient la clause vide.*

Ce théorème sera démontré dans la Section 4.4.

La stratégie de superposition que nous venons de définir utilise les règles de la stratégie de paramodulation. Cependant, il est possible de supprimer les règles de factorisation, paramodulation et paramodulation contextuelle si *le seul prédicat utilisé est le prédicat d'égalité*. Nous pouvons remarquer dans un premier temps que la règle de factorisation est inutile lorsqu'elle est appliquée sur des littéraux équationnels positifs, car des étapes de factorisation équationnelle puis de réflexion produisent le même résultat. Les factorisations entre diséquations peuvent être évitées en assouplissant les conditions de maximalité des diséquations dans les autres règles d'inférence : il suffit de remplacer les symboles $\not\preceq_L$ par $\not\prec_L$ afin de tolérer l'existence de diséquations E-égales à celle traitée.

Ces modifications apportent un gain car elles permettent d'économiser trois règles d'inférence, mais elles provoquent souvent le remplacement d'une déduction par plusieurs, comme nous venons de le préciser pour la factorisation de littéraux positifs, mais également pour la factorisation de littéraux négatifs, car une clause obtenue par paramodulation dans un littéral négatif d'une clause factorisée est alors obtenue après autant de paramodulations que de littéraux initialement factorisés. L'avantage est donc discutable.

2.3 Stratégie positive

La stratégie positive est un raffinement de la stratégie ordonnée utilisée dans la section précédente pour définir les règles d'inférence pour la E-paramodulation et la E-superposition. Cette stratégie s'appelle « positive » car une majorité d'inférences doit utiliser au moins une *clause positive*, c'est-à-dire une clause n'ayant que des littéraux positifs. Ainsi, dans les règles de paramodulation et de superposition définies auparavant, la clause contenant l'équation utilisée pour le remplacement doit être positive, et les remplacements doivent être appliqués en priorité dans les littéraux négatifs d'une clause. Définissons plus formellement cette stratégie.

Définition 2.13 (Stratégie positive) *Si une inférence utilise un littéral positif dans une clause, alors cette clause doit être positive, c'est-à-dire ne contenir aucun littéral négatif, et le littéral considéré doit être maximal dans la clause.*
Si une inférence utilise un littéral négatif dans une clause, alors ce littéral doit être maximal vis-à-vis des autres littéraux négatifs de cette clause.

Le but de cette stratégie est donc d'éviter l'augmentation du nombre de littéraux négatifs. En effet, la clause vide ne pouvant être engendrée que par suppression de littéraux, et comme les seules règles d'inférence concernées (résolution et réflexion) n'ôtent que des littéraux négatifs, la stratégie permet de savoir de quel ensemble de clauses très restreint la contradiction sera dérivée. Notons cependant qu'il existe une règle créant des littéraux négatifs : la factorisation équationnelle.
Cette stratégie offre également l'avantage d'être très explicite sur l'ordre d'application des règles d'inférence. Si nous voulons utiliser une équation pour effectuer une paramodulation, il faut d'abord éliminer les littéraux négatifs de la clause à laquelle elle appartient.

Enumérons les modifications à effectuer dans les règles d'inférence des sections précédentes :

- Factorisation : si les littéraux factorisés sont positifs, alors la clause les contenant doit être positive, et ces littéraux doivent être maximaux ; si les littéraux factorisés sont négatifs, alors ils doivent être maximaux par rapport aux autres littéraux négatifs de la clause,
- Factorisation équationnelle : appliquée uniquement sur une clause positive (les conditions de maximalité ne changent pas),
- Résolution : la clause contenant le littéral positif utilisé doit être positive et ce littéral doit être maximal ; le littéral négatif utilisé dans l'autre clause doit être le plus grand littéral négatif de celle-ci,
- Réflexion : la maximalité de l'équation négative n'est plus vérifiée que par rapport aux autres littéraux négatifs de la clause,
- Paramodulation, Paramodulation contextuelle, Superposition, Superposition contextuelle : la clause contenant l'équation utilisée pour le remplacement doit être positive, et cette équation doit être maximale dans la clause ; si le littéral L_2, dans lequel s'effectue le remplacement, est positif, alors la clause $L_2 \vee D_2$ doit être positive et L_2 doit être maximal dans celle-ci ; si le littéral L_2 est négatif, alors ce doit être le plus grand littéral négatif de la clause $L_2 \vee D_2$,
- Paramodulation étendue, Superposition étendue : elles ne sont plus appliquées qu'entre les équations maximales de deux clauses positives.

Soit INF un ensemble de règles d'inférence définies dans cette section, pour la stratégie de E-paramodulation ou de E-superposition. Enonçons le théorème de E-complétude de INF.

Théorème 2.14 (Complétude de la stratégie positive) *Si S est un ensemble de clauses E-incohérent, $INF^*(S)$ contient la clause vide.*

Ce théorème sera démontré dans la Section 4.5.

2.4 Règles de simplification

Cette section présente des mécanismes indispensables pour l'efficacité des démonstrateurs, puisqu'il s'agit de règles permettant d'éliminer des informations redondantes. Ces règles peuvent être décomposées en deux groupes : celles qui suppriment une clause, comme la subsomption, et celles qui remplacent une clause par une autre, comme les règles de simplification.

Une notion de redondance plus générale a été définie par L. Bachmair et H. Ganzinger [BG94]. Elle permet d'éliminer une clause si celle-ci est redondante par rapport à un ensemble de clauses. Nous étudions actuellement l'adaptation de notre technique de preuve de complétude à cette notion.

2.4.1 Subsomption

La règle de subsomption consiste à supprimer une clause si elle est redondante vis-à-vis d'une autre clause. Sa définition, comme toute définition de règle de simplification, fait appel à un algorithme de filtrage dans la théorie E.

Définition 2.15 (E-Subsomption) *Soient C_1 et C_2 deux clauses telles qu'il existe une substitution ρ vérifiant : $C_1\rho \subseteq_E C_2$ (les clauses sont considérées comme des multi-ensembles de littéraux). Alors, la clause C_1 subsume la clause C_2.*

Ainsi, la clause C_2 est subsumée par C_1 si chaque littéral de $C_1\rho$ apparaît dans C_2 avec le même nombre d'occurrences. Donc, la clause C_2 a au moins autant de littéraux que C_1.

L'application de la règle de subsomption consiste à supprimer les clauses subsumées.

Définition 2.16 (Application de la subsomption) *Soient C_1 et C_2 deux clauses. Si C_1 subsume C_2, alors la clause C_2 est supprimée.*

Considérer les clauses comme des multi-ensembles de littéraux est indispensable, comme le montre l'exemple suivant.

Exemple 2.11 Soit le système de clauses suivant, incohérent dans la théorie vide :

$$\left\{ \begin{array}{ll} (f(a,x) \simeq g(x)) \vee (f(x,a) \simeq g(x)) & (1) \\ (g(a) \simeq b) & (2) \\ \neg(f(a,a) \simeq b) & (3) \end{array} \right.$$

Supposons que, pour chaque équation, le membre gauche est plus grand que le membre droit. Une factorisation de la clause (1) engendre la clause $(f(a,a) \simeq g(a))$ (4). Cependant, si la définition de la subsomption teste l'inclusion des clauses comme une inclusion d'ensembles, la clause (1) subsume la clause (4) (en instanciant x par a). La clause (4) est donc supprimée.
Cependant, il n'est plus possible d'appliquer une seule inférence entre les clause (1), (2) et (3), car la condition de maximalité du littéral utilisé dans la clause (1) ne serait pas respectée. L'incohérence du système ne peut être démontrée. ♦

Nous montrerons (Théorème 2.21) que l'application de cette règle de subsomption est correcte, bien qu'elle ne fasse pas intervenir la notion de subsomption stricte, habituellement utilisée.

Une clause est strictement subsumée si elle est subsumée par une clause, mais ne subsume pas cette même clause. Par exemple, les clauses $P(f(x,y))$ et $P(z)$ se subsument mutuellement, si la théorie E est formée de l'axiome $(f(x,x) \simeq x)$. La subsomption stricte interdit donc de supprimer l'une des deux.

Ces problèmes de subsomption mutuelle sont résolus par l'application d'une stratégie très simple que nous détaillons dans la Section 2.4.4. Ne pas utiliser la subsomption stricte permet de supprimer les clauses qui sont E-égales, à un renommage de variable près, à une clause existant dans l'ensemble de clauses courant.

2.4.2 Simplification conditionnelle

Un deuxième type de règle de simplification est la simplification par remplacement, qui correspond en fait à une étape de réécriture dans une clause. Elle ne s'applique que si l'orientation de l'équation utilisée est certaine. Les règles que nous définissons sont dites « conditionnelles » car la clause permettant la simplification peut posséder d'autres littéraux que l'équation en question, à condition bien sûr que ces littéraux apparaissent aussi dans la clause simplifiée.
L'ordre utilisé pour comparer les littéraux est noté $\succ$, comme pour la stratégie de paramodulation ; mais, lorsque ces règles sont utilisées avec la stratégie de superposition, il faut le remplacer par $\succ_L$.

Définition 2.17 (E-Simplification)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2 \vee D_2}{L_2[p_2 \leftarrow r_1\rho] \vee D_2} \quad si \quad \begin{cases} \rho \in E\text{-}match(l_1, L_2|_{p_2}) \quad où \quad p_2 \in \mathcal{FP}os(L_2) \\ l_1\rho \succ r_1\rho \\ D_2 = D_2' \vee D_2'' \quad et \quad D_1\rho \subseteq_E D_2' \\ (l_1 \simeq r_1)\rho \;\prec\; L_2 \vee D_2'' \end{cases}$$

Cette règle s'applique donc si l_1 est filtré dans le sous-terme à la position p_2 de L_2. L'inclusion des littéraux de D_1 dans ceux de D_2' porte sur les ensembles et non sur les multi-ensembles. Cela signifie que la simplification est effectuée même si un littéral a un plus grand nombre d'occurrences dans $D_1\rho$ que dans D_2'. La dernière condition précise qu'un littéral de la clause $L_2 \vee D_2$, non utilisé dans D_2', doit être plus grand que l'équation $(l_1 \simeq r_1)$; ceci est imposé par notre technique de preuve de complétude.

La règle suivante est le pendant de la paramodulation (ou superposition) contextuelle. Son énoncé est plus compliqué que la précédente en raison du traitement de l'extension.

Définition 2.18 (E-Simplification contextuelle)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2 \vee D_2}{L_2[p_2 \leftarrow e_1[r_1]_{p_1}\sigma\rho] \vee D_2} \quad si \quad \begin{cases} (e_1, p_1, c_1) \in \mathcal{C}ont(l_1), \; p_2 \in \mathcal{FP}os(L_2) \\ \sigma \in \mathcal{S}ol(c_1) \quad et \quad \rho \in E\text{-}match(e_1\sigma, L_2|_{p_2}) \\ l_1\sigma\rho \succ r_1\sigma\rho \\ D_2 = D_2' \vee D_2'' \quad et \quad D_1\sigma\rho \subseteq_E D_2' \\ (l_1 \simeq r_1)\rho \;\prec\; L_2 \vee D_2'' \end{cases}$$

Ainsi, il faut d'abord trouver une solution au problème d'extension de l_1 pour ensuite filtrer le terme étendu dans le sous-terme $L_2|_{p_2}$. Comme pour la règle de E-simplification, nous vérifions qu'un littéral de $L_2 \vee D_2''$ est plus grand que $(l_1 \simeq r_1)$; ce test est trivial si la théorie E respecte la propriété de sous-terme, mais est indispensable sinon.

Définition 2.19 (Application des règles de simplification) *Soient C_1 et C_2 deux clauses. Si C_1 simplifie (contextuellement) C_2 en une clause C_2', alors la clause C_2 est remplacée par sa simplification C_2'.*

2.4.3 Autres règles de simplification

D'autres règles de simplification sont compatibles avec les stratégies de paramodulation et de superposition. Par exemple, il est possible de supprimer les **tautologies** ; il en existe deux sortes : les clauses contenant un littéral E-égal à une identité $(s \simeq s)$; et les clauses contenant des littéraux complémentaires, i.e. des clauses de la forme $L \vee \neg L' \vee D$, où L et L' sont E-égaux. Une autre règle importante est la **simplification clausale**, qui supprime toutes les instances d'un littéral L dans les clauses si $\neg L$ est une clause à part entière. On peut donc supprimer toute instance de $\neg(x \simeq x)$, ce qui est en quelque sorte une **réflexion triviale**. En fait, une simplification clausale revient à appliquer une résolution puis à supprimer l'une des clauses parentes par subsomption.

D'autres transformations sont possibles, comme les **simplifications par remplacement** qui consistent à remplacer une clause $\neg(s \simeq t) \vee D$ par $\neg(s \simeq t) \vee D'$, où D' correspond à D dans lequel tout sous-terme E-égal à s a été remplacé par t, ceci à condition que s soit strictement plus grand que t, pour assurer la diminution de la complexité de la clause.

2.4.4 Théorème de complétude

Soit INF l'un des ensembles de règles d'inférence définis dans les sections précédentes, c'est-à-dire pour la E-paramodulation ou la E-superposition, et pour la stratégie ordonnée ou positive. Soit INF_S le système composé de INF et des règles de simplification définies dans cette section.

Soit S_0 un ensemble de clauses. Une *dérivation* de S_0 est une suite d'ensembles de clauses $S_0, S_1, S_2, \ldots$, où chaque S_{i+1} est obtenu après une étape d'inférence ou de simplification de INF_S utilisant une ou plusieurs clauses de S_i.

Définition 2.20 *$INF(S)$ désignant l'ensemble des clauses pouvant être déduites d'un ensemble de clauses S en une étape par une règle de INF, une dérivation $S_0, S_1, S_2, \ldots$ est équitable si toute clause C appartenant à $\bigcap_{i \geq j} INF(S_i)$ (pour un indice j) est subsumée par une clause C' de $\bigcup_{i \geq 0} S_i$.*

Le problème de *subsomption mutuelle* de deux clauses est résolu en appliquant une stratégie de subsomption des clauses les plus récentes d'abord. Soit C_1 et C_2 deux clauses d'un ensemble S_i telles que C_1 subsume C_2, et C_2 subsume C_1. Soient j_1 et j_2 les indices minimaux tels que : $\forall k \in [j_1, i]$, $C_1 \in S_k$ et $\forall k \in [j_2, i]$, $C_2 \in S_k$. Si $j_1 \geq j_2$, la clause C_1 peut être supprimée ; sinon, la clause C_2 peut être supprimée.

Enonçons le théorème de E-complétude de INF en présence des règles de simplification.

Théorème 2.21 *Soit S un ensemble E-incohérent de clauses. Alors, toute dérivation équitable de S engendre la clause vide.*

Ce théorème sera démontré dans la Section 4.6.

2.5 Comparaison avec des travaux voisins

Cette section est consacrée à la comparaison de nos travaux avec ce qui a déjà été fait pour la déduction automatique modulo une théorie équationnelle. Nous allons plus particulièrement étudier les travaux de G.D. Plotkin [Plo72], U. Wertz [Wer92] et E. Paul [Pau94].

2.5.1 Travaux de G.D. Plotkin

Les règles d'inférence définies par G.D. Plotkin dans [Plo72] incluent la factorisation, car chacune d'entre elles n'agit pas sur un littéral d'une clause, mais sur un ensemble de littéraux E-unifiables. Nous dirons qu'une substitution σ est un E-unificateur d'ensembles de termes, d'atomes ou de littéraux $\{A_{1,1}, \ldots, A_{1,n_1}\}$, ..., $\{A_{m,1}, \ldots, A_{m,n_m}\}$, si

$$\forall i \in [1, m],\ \forall j \in [2, n_i],\ A_{i,1}\sigma =_E A_{i,j}\sigma$$

Définition 2.22 (Résolution)

$$\frac{A_1 \vee \ldots \vee A_n \vee D_1 \qquad \neg B_1 \vee \ldots \vee \neg B_m \vee D_2}{D_1\sigma \vee D_2\sigma}$$

si σ est un E-unificateur de l'ensemble d'atomes $\{A_1, \ldots, A_n, B_1, \ldots, B_m\}$

La règle suivante permet d'éliminer des équations négatives dont les deux membres sont E-unifiables. Dans les sections précédentes, nous avons appelé cette règle « réflexion ».

Définition 2.23 (Trivialisation)

$$\frac{\neg(l_1 \simeq r_1) \vee \ldots \vee \neg(l_n \simeq r_n) \vee D}{D\sigma}$$

si σ est un E-unificateur de l'ensemble de termes $\{l_1, r_1, \ldots, l_n, r_n\}$

La définition de la paramodulation fait appel au calcul d'une *paire spéciale*, notée $<x, t>$, où t est un terme possédant la variable x à une position p et tel que, pour toute position q dans t, indépendante de p ($p \neq q \cdot o$), $t|_q$ est une variable. Notons que toute variable de $\mathcal{V}ar(t)$ n'a qu'une seule occurrence dans t.

Définition 2.24 (Paramodulation)

$$\frac{(l_1 \simeq r_1) \vee \ldots \vee (l_n \simeq r_n) \vee D_1 \qquad L_1 \vee \ldots \vee L_m \vee D_2}{L'_1\sigma \vee D_1\sigma \vee D_2\sigma}$$

si, à chaque littéral L_i, correspond un atome $P(t_{i,1}, \ldots, t_{i,j}, \ldots, t_{i,p})$, et σ est un E-unificateur des ensembles $\{(l_1 \simeq r_1), \ldots, (l_n \simeq r_n)\}$, $\{L_1, \ldots, L_m\}$ et $\{t_{1,j}, t[x \leftarrow l_1]\}$, où $<x, t>$ est une paire spéciale.
Le littéral L'_1 correspond au littéral L_1 dans lequel le terme $t_{1,j}$ a été remplacé par $t[x \leftarrow r_1]$.

Ces règles d'inférence, bien que définies pour effectuer des déductions dans une théorie équationnelle E, ne font pas intervenir la notion d'extension ou toute autre notion semblable. Cela s'explique par les deux remarques suivantes sur la règle de paramodulation :

- Aucune condition d'orientation de l'équation n'est imposée ; il est ainsi possible de remplacer un terme par un plus grand, ce qui évite (indirectement) l'introduction d'une règle comme la paramodulation étendue.

- La construction de la paire spéciale est indépendante du sous-terme dans lequel le remplacement est effectué ; elle dissimule en fait la construction de contextes. En effet, comme les remplacements sont appliqués juste sous le symbole de prédicat, et non à une occurrence interne, le terme t de la paire spéciale peut être construit complètement différemment du terme $t_{1,j}$. La version de cette règle de paramodulation pour le cas clos est encore plus explicite à ce sujet : G.D. Plotkin précise que le terme $t_{1,j}$ est unifié avec un terme $w[l_1]$.

Notons également que les paramodulations dans des variables sont autorisées.

2.5.2 Travaux d'U. Wertz

U. Wertz définit dans sa thèse [Wer92] un ensemble de règles d'inférence basé sur la stratégie de superposition pour effectuer de la déduction automatique dans une théorie équationnelle régulière E.

Les règles de base utilisées sont la superposition, la réflexion et la factorisation équationnelle, définies pour la stratégie ordonnée. Elles sont complétées par deux approches $\mathcal{M}_E$ et $\mathcal{M}_{Ext}$ pour traiter les extensions.

Le système $\mathcal{M}_E$

Une nouvelle règle appelée *E-closure* est introduite ; elle consiste à effectuer une superposition d'une clause dans un axiome de la théorie E. L'inconvénient immédiat de cette règle est donc qu'elle permet d'engendrer des extensions de clauses, ainsi que des extensions de ces extensions, sans aucune restriction.

L'ensemble de ces règles d'inférence est prouvé réfutationnellement complet à condition que la théorie E soit *simple*, c'est-à-dire respecte la propriété suivante :

Aucun terme ne doit être E-égal à l'un de ses sous-termes stricts.

Le système $\mathcal{M}_{Ext}$

Ce second système ne contient pas de règle d'inférence supplémentaire. A chaque clause est associé l'ensemble de ses extensions et des extensions de ses extensions. Il est alors possible d'appliquer une superposition d'une clause étendue dans une autre clause, pouvant elle-même être étendue si la superposition s'effectue dans une équation positive et au sommet de son membre gauche.

Mettre à la disposition du système d'inférence toutes les extensions des clauses requiert la mise en place d'un contrôle sévère, afin de calculer l'ensemble des extensions pour chaque

clause, de modifier l'ensemble de ces extensions lorsqu'une clause est simplifiée, et de vérifier le bon usage de ces extensions dans les règles d'inférence.
D'autre part, lorsqu'une clause étendue est utilisée, la maximalité de l'équation utilisée est testée sur la clause étendue, et non sur la clause de départ.

Ce système est prouvé réfutationnellement complet à condition que la théorie E respecte la propriété suivante :

> *Si un terme s est E-égal à l'un de ses sous-termes stricts $s|_p$, alors pour tout terme t, $s[t]_p$ doit être E-égal à t.*

Nous montrerons dans notre preuve de complétude (preuve du Lemme A.2) le problème ayant provoqué l'ajout de cette restriction sur la théorie E, mais nous prouverons que cette condition est inutile. La seule condition à retenir sur la théorie E est qu'elle doit être régulière.
Notons que la propriété imposée par U. Wertz implique que la théorie E est régulière, mais la réciproque est fausse. Par exemple, la théorie $E = \{(f(x,e) \simeq x)\}$ est régulière mais ne satisfait pas la propriété précédente : $f(e,e) =_E e$, mais $\forall t \in \mathcal{T}(\mathcal{F}),\ f(e,t) =_E t$ est faux.

La preuve de complétude de ces deux systèmes est basée sur la technique de construction de modèles de L. Bachmair et H. Ganzinger [BG90]. U. Wertz utilise une notion de redondance de clauses plus générale que nous ; cette notion englobe nos règles de subsomption, simplifications.

2.5.3 Travaux d'E. Paul

Les travaux d'E. Paul sont assez proches de ceux décrits par U. Wertz, dans la mesure où il définit un système de règles d'inférence ayant à sa disposition l'ensemble des clauses étendues. Cependant, une règle d'inférence spéciale, appelée *paramodulation étendue*, est définie pour appliquer une paramodulation entre deux clauses étendues. E. Paul présente en plus un système de règles d'inférence basé la stratégie positive combinée avec la stratégie de superposition. Cette dernière n'est pas totalement appliquée, car certains remplacements dans le plus petit membre d'une équation sont permis par une règle spéciale, appelée « merging paramodulation », apparaissant dans les travaux de L. Bachmair et H. Ganzinger [BG90] et M. Haberstau [Hab90].

Comme pour le second système d'U. Wertz, ce système est prouvé réfutationnellement complet à condition que la théorie E respecte la propriété suivante :

> *Si un terme s est E-égal à l'un de ses sous-termes stricts $s|_p$, alors pour tout terme t, $s[t]_p$ doit être E-égal à t.*

La preuve de complétude est basée sur la technique des arbres sémantiques de J. Hsiang et M. Rusinowitch [Rus89, HR91]. La principale différence avec notre technique de preuve est la construction de l'arbre sémantique : E. Paul définit un arbre sémantique sur les classes d'équivalence pour la théorie E, alors que nous utilisons l'arbre sémantique correspondant à la théorie vide puis sélectionnons une branche consistante avec E.

3
Extensions de règles modulo une théorie équationnelle

La déduction dans une théorie équationnelle E en évitant l'usage explicite des axiomes de E oblige à simuler le rôle de ces axiomes, comme nous venons de le voir dans le chapitre précédent. Cela se traduit par l'utilisation d'un algorithme de E-unification, mais surtout par la manipulation de *clauses étendues*.
Cette technique fut proposée par G.E. Peterson et M.E. Stickel pour la définition d'un algorithme de complétion dans les théories équationnelles [PS81]. Elle fut également utilisée dans les travaux de H. Kirchner [Kir85] et J.-P. Jouannaud et H. Kirchner [JK86] sous le nom de paires critiques de cohérence.

Deux règles d'inférence pour la stratégie de E-paramodulation utilisent ces clauses étendues : la paramodulation contextuelle et la paramodulation étendue. Lors de leur définition, nous avons noté $\mathcal{C}ont(l)$ l'ensemble des extensions possibles d'une équation $(l \simeq r)$ par rapport à l ; chaque élément (e, p, c) de $\mathcal{C}ont(l)$, appelé *contexte*, étend l'équation $(l \simeq r)$ en $(e[l]_p \simeq e[r]_p)$.

Dans ce chapitre, nous allons montrer comment construire incrémentalement l'ensemble des contextes utiles pour une théorie E, en définissant un algorithme détectant la redondance d'un contexte par rapport à un ensemble de contextes déjà construits. Ce calcul ne dépend que de la théorie E, c'est-à-dire est effectué indépendamment de l'ensemble de clauses à compléter ou à résoudre.

Nous terminerons ce chapitre par une étude de la redondance d'un contexte pour une clause donnée, limitant ainsi les applications des règles de paramodulation contextuelle et étendue avec cette clause.

3.1 Les extensions vues par les contextes

Rappelons par un petit exemple pourquoi étendre des clauses est indispensable.

Exemple 3.1 Soit E la théorie représentant la propriété d'associativité d'un opérateur f. Soient $(f(a,b) \simeq c)$ (1) et $P(f(a, f(b,d)))$ (2) deux clauses. Aucun remplacement ne peut être effectué de (1) dans (2), cependant la clause (2) est E-égale à la clause $P(f(f(a,b),d))$ (2′) dans laquelle le sous-terme $f(a,b)$ peut être remplacé par c pour obtenir $P(f(c,d))$. Etendre la clause (1) consiste à trouver un terme $e[\cdot]_p$, appelé contexte, tel qu'une paramodulation appliquée de

$(e[f(a,b)]_p \simeq e[c]_p)$ dans (2) engendre $P(f(c,d))$ (où une clause qui lui est E-égale). Dans cet exemple, le contexte est $f(\cdot, x)$.

Le cas que nous venons de voir est simple car nous n'avons utilisé que des clauses closes. Mais, si la clause (2) est $P(f(u,v))$, où u et v sont des variables, la clause $P(c)$ peut être engendrée, mais il est également indispensable de pouvoir engendrer les clauses $P(f(c,x))$, $P(f(x,c))$ et $P(f(x, f(c,y)))$, grâce aux contextes respectifs $f(\cdot, x)$, $f(x, \cdot)$ et $f(x, f(\cdot, y))$, car elles seules permettent de trouver une contradiction si l'une des clauses suivantes est présente : $\neg P(f(c,d))$, $\neg P(f(d,c))$ ou $\neg P(f(d, f(c, d')))$. ♦

3.1.1 Construction des contextes

Un *contexte* est un triplet composé d'un terme, d'une position non variable dans ce terme, et de l'ensemble (i.e. la conjonction) des problèmes de E-unification apparus au cours de la construction du terme.

La Figure 3.1 décrit récursivement la construction des contextes par rapport aux axiomes de la théorie E. Un premier ensemble de contextes, noté $\mathcal{C}ont_0$, représente les contextes initiaux, c'est-à-dire désigne les positions internes non variables des membres des axiomes de E. Les ensembles $\mathcal{C}ont_{k+1}$ (où $k \geq 0$) représentent les contextes issus de l'empilement d'un élément de $\mathcal{C}ont_0$ à un contexte de $\mathcal{C}ont_k$, après renommage des variables. L'ensemble $\mathcal{C}ont(l)$ contient les triplets (e,p,c) des $\mathcal{C}ont_k$ ($k \geq 0$) pour lesquels le terme l est unifiable avec le sous-terme à la position p de e.
Ainsi, une extension d'une équation $(l \simeq r)$, où l est supposé plus grand que r, est une équation $(e[l]_p \simeq e[r]_p)$ telle que le contexte (e,p,c) appartient à $\mathcal{C}ont(l)$; les seules instances de cette extension pouvant être utilisées sont celles qui satisfont c.

$$
\begin{array}{rcl}
\mathcal{C}ont_0 & = & \{\ (e,p,\emptyset)\ \mid\ \exists (e \simeq e') \text{ ou } (e' \simeq e) \in E,\ p \in \mathcal{FP}os(e) \text{ et } p \neq \epsilon\ \} \\[1ex]
\mathcal{C}ont_{k+1} & = & \{\ (e_1[e_2]_{p_1},\ p_1{\cdot}p_2,\ c_2 \cup \{e_1|_{p_1} =^?_E e_2\})\ \mid\ (e_1,p_1,\emptyset) \in \mathcal{C}ont_0, \\
 & & \qquad (e_2,p_2,c_2) \in \mathcal{C}ont_k \text{ et } \mathcal{S}ol(c_2 \cup \{e_1|_{p_1} =^?_E e_2\}) \neq \emptyset\ \} \\[1ex]
\mathcal{C}ont(l) & = & \{\ (e,\ p,\ c \cup \{e|_p =^?_E l\})\ \mid\ (e,p,c) \in \bigcup_{k \geq 0} \mathcal{C}ont_k,\ \mathcal{S}ol(c \cup \{e|_p =^?_E l\}) \neq \emptyset\ \}
\end{array}
$$

Figure 3.1: Construction par récurrence des contextes

Cet algorithme s'arrête lorsqu'un ensemble de contextes $\mathcal{C}ont_n$ est vide. Mais, pour de très nombreuses théories, une infinité de contextes est engendrée par cet algorithme ; c'est pour cette raison que nous étudierons par la suite une notion de redondance d'un contexte par rapport à un ensemble de contextes déjà créés (notion qui devra être intégrée à cet algorithme).

Donnons un exemple de calcul de contextes.

Exemple 3.2 Soit E la théorie représentée par les propriétés d'associativité et de commutativité d'un opérateur f :

$$E \;=\; \{\ (f(f(x,y),z) \simeq f(x, f(y,z))),\ (f(x,y) \simeq f(y,x))\ \}$$

L'ensemble des contextes initiaux $\mathcal{C}ont_0$ est :

$$\{\quad (f(f(x,y),z), 1, \emptyset),\quad (f(x, f(y,z)), 2, \emptyset)\quad \}$$

Les contextes de $\mathcal{C}ont_1$ calculés à partir du premier contexte de $\mathcal{C}ont_0$ sont :

$$\begin{array}{ll} \{ & (f(f(f(x,y),z),z'),1.1,\{f(x',y') =^?_E f(f(x,y),z)\}), \\ & (f(x',f(f(x,y),z)),2.1,\{f(y',z') =^?_E f(f(x,y),z)\}) \quad \} \end{array}$$

Les contextes de $\mathcal{C}ont_1$ calculés à partir du second contexte de $\mathcal{C}ont_0$ sont :

$$\begin{array}{ll} \{ & (f(f(x,f(y,z)),z'),1.2,\{f(x',y') =^?_E f(x,f(y,z))\}), \\ & (f(x',f(x,f(y,z))),2.2,\{f(y',z') =^?_E f(x,f(y,z))\}) \quad \} \end{array}$$

En continuant d'appliquer l'algorithme de la Figure 3.1, nous construisons 8 contextes dans $\mathcal{C}ont_2$, 16 dans $\mathcal{C}ont_3$, et ainsi de suite. ♦

L'algorithme décrit dans la Figure 3.1, appliqué sur la théorie AC de l'exemple précédent engendre une infinité de contextes. Cependant, il est bien connu qu'une seule extension est nécessaire pour cette théorie. Cela montre bien que l'algorithme ne peut être utilisé sous sa forme actuelle, et motive ainsi les définitions d'algorithmes détectant la redondance de contextes dans les Sections 3.1.4 et 3.2.

3.1.2 Variante de la construction des contextes

Le calcul des contextes décrit dans la Figure 3.1 pourrait être syntaxiquement simplifié : il n'est pas indispensable de conserver les problèmes de E-unification rencontrés lors de la construction ; il suffit de tester la E-unifiabilité à chaque étape. Ainsi, pour une équation $(l \simeq r)$, tout contexte de $\mathcal{C}ont(l)$ est de la forme $(e,p,\emptyset)$, et l'extension correspondante est l'équation $(e[l]_p \simeq e[r]_p)$.

Cette simplification est correcte, car tout contexte possédant des contraintes d'unification est une instance du même contexte sans ces contraintes. Mais cela signifie qu'une instance d'un contexte peut être utilisée alors qu'elle ne respecte pas les contraintes d'unification accumulées lors de sa construction. De plus, la version non simplifiée a l'avantage de détecter les cas où aucune solution n'est utile, comme le montre l'exemple suivant :

Exemple 3.3 Soit E la théorie représentée par l'axiome $(f(x,e) \simeq x)$. $\mathcal{C}ont_0$ contient le contexte $(f(x,e),2,\emptyset)$ et $\mathcal{C}ont_1$ contient le contexte $(f(x,f(y,e)),2{\cdot}2,\{f(y,e) =^?_E e\})$, issu de l'application du contexte $(f(y,e),2,\emptyset)$ dans le sous-terme e (position 2) du contexte $(f(x,e),2,\emptyset)$. Or, la seule solution de la condition de E-unification est $\{y \mapsto e\}$, et le contexte est en fait $(f(x,f(e,e)),2{\cdot}2,\emptyset)$. L'algorithme de redondance que nous développerons dans la Section 3.2 permettra de montrer la redondance de ce contexte, car le sous-terme $f(e,e)$ est E-égal au sous-terme à la position $2{\cdot}2$, e.

Or, si nous ne tenons pas compte du problème de E-unification posé lors de la construction du contexte de $\mathcal{C}ont_1$, celui-ci devient $(f(x,f(y,e)),2{\cdot}2,\emptyset)$ et n'est pas redondant. ♦

Pour les raisons que nous venons d'énumérer, nous conserverons les contraintes de E-unification dans les contextes.

3.1.3 Couverture totale des contextes

Nous avons défini dans la Figure 3.1 un algorithme de construction de contextes à partir d'une théorie équationnelle E. L'utilisation de ces contextes dans les règles d'inférence a pour but d'éliminer toute manipulation explicite des équations de E, lors de la construction des extensions en particulier. Cependant, nous devons vérifier que ces contextes expriment complètement la théorie E ; c'est le but de cette section.

La preuve de cette propriété sera effectuée sur les termes clos, car nous l'utiliserons dans la preuve de complétude des systèmes d'inférence, preuve réalisée dans le cas clos. La propriété que nous voulons montrer est la suivante : si un terme clos t est réduit par une équation close $(l \simeq r)$ (ou l'une de ses extensions) à une position p (l étant plus grand que r, pour un ordre donné), alors, pour tout terme clos t' E-égal à t, t' et $t[r]_p$ peuvent être réduits en deux termes E-égaux par l'équation $(l \simeq r)$ et ses extensions calculées à partir des contextes.

Pour rester le plus général possible, nous n'utiliserons pas la notion citée ci-dessus de réduction par une équation bien précise. Nous dirons que le terme t, dont un sous-terme est E-égal à un contexte $e[l]_p$ de l (où p peut être la position vide ϵ, i.e. $e[l]_p = l$), est transformé (ou réduit) en $t[e[\cdot]_p]$; la transformation consiste donc à remplacer le sous-terme de t par le contexte dans lequel l est remplacé par une nouvelle constante, symbolisée par "$\cdot$". Nous montrerons alors que, pour tout terme t', E-égal à t, $t[e[\cdot]_p]$ et t' peuvent être transformés en deux termes E-égaux. Mais introduisons d'abord quelques notations et notions nouvelles.

Une étape de *E-égalité* entre deux termes clos t_1 et t_2, à une position p_1, utilisant un axiome $(e_1 \simeq e'_1)$ de E et une substitution σ_1, est décrite ainsi :

$$t_1 \xleftrightarrow[e_1 \simeq e'_1,\ \sigma_1]{p_1} t_2 \qquad \text{si} \ \ t_1|_{p_1} = e_1\sigma_1 \ \ \text{et} \ \ t_2 = t_1[e'_1\sigma_1]_{p_1}$$

Les deux termes t_1 et t_2 étant clos, la substitution σ_1 instancie toutes les variables de e_1 et e'_1 ; il ne s'agit donc pas d'un simple filtrage de e_1 dans $t_1|_{p_1}$, mais aussi de e'_1 dans $t_2|_{p_1}$, car la théorie E peut ne pas être régulière et certaines variables de e'_1 ne pas apparaître dans e_1. Lorsque les détails d'une étape de E-égalité ne sont pas importants, nous la notons simplement $\leftrightarrow_E$.

Un *ensemble de contextes $\mathcal{C}_l$ associé à un terme l* est un ensemble de couples (e_l, p_l), où e_l est un terme clos et p_l une position (éventuellement vide) dans e_l, tels que $e_l|_{p_l} =_E l$. Cet ensemble contient en fait les instances closes des contextes de $\mathcal{C}ont(l)$, ainsi que le couple (l, ϵ).

Un *algorithme de calcul de contextes* est un algorithme qui associe à tout terme clos l un ensemble de contextes $\mathcal{C}_l$.

Soit $\mathcal{C}_l$ un ensemble de contextes. Nous définissons la relation $\rightarrow_{\mathcal{C}_l,E}$ par :

$$t_1 \longrightarrow_{\mathcal{C}_l,E} t_2 \ \ \text{si} \ \ \exists (e_l, p_l) \in \mathcal{C}_l, \ \exists q \in \mathcal{FP}os(t_1), \ t_1|_q =_E e_l, \ t_2 = t_1[e_l[\cdot]_{p_l}]_q$$

Cette relation, appliquée à un couple (e_l, p_l) et une position q, est notée : $t_1 \longrightarrow^q_{(e_l,p_l),E} t_2$. Précisons que les clôtures transitive et réflexive transitive d'une relation $\rightarrow$ seront notées $\rightarrow^+$ et $\rightarrow^*$ respectivement.

Le but de cette section étant de montrer que les contextes expriment totalement la théorie E, rappelons la propriété que nous allons montrer :

> *Si un terme t est réductible en un terme t_1 par la relation $\rightarrow_{\mathcal{C}_l,E}$, alors pour tout terme t_2 E-égal à t, t_2 et t_1 sont réductibles par $\rightarrow_{\mathcal{C}_l,E}$ en deux termes E-égaux.*

Cette propriété est appelée *E-fermeture* de $\rightarrow_{\mathcal{C}_l,E}$. Pour prouver qu'elle est vérifiée, nous aurons besoin d'une propriété intermédiaire, appelée *E-fermeture locale*, qui diffère de la E-fermeture par l'hypothèse qu'il ne doit exister qu'une étape de E-égalité pour passer de t à t_2. Ces deux propriétés sont décrites dans la Figure 3.2.

$$
\begin{tikzcd}
t \arrow[r, leftrightarrow, "E"'] \arrow[d, "\mathcal{C}_l{,}E"'] & t_2 \arrow[dd, dotted, "\mathcal{C}_l{,}E"', "*"] \\
t_1 \arrow[d, dotted, "\mathcal{C}_l{,}E"', "*"] & \\
t'_1 \arrow[r, leftrightarrow, dotted, "*", "E"'] & t'_2
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
t \arrow[r, leftrightarrow, "*", "E"'] \arrow[d, "\mathcal{C}_l{,}E"'] & t_2 \arrow[dd, dotted, "\mathcal{C}_l{,}E"', "*"] \\
t_1 \arrow[d, dotted, "\mathcal{C}_l{,}E"', "*"] & \\
t'_1 \arrow[r, leftrightarrow, dotted, "*", "E"'] & t'_2
\end{tikzcd}
$$

Figure 3.2: E-fermeture locale et E-fermeture de $\rightarrow_{\mathcal{C}_l,E}$

Les définitions de ces propriétés sont assez proches des notions de E-cohérence et E-cohérence locale données par J.-P. Jouannaud et H. Kirchner dans [JK86]. Les principales différences sont :

1. Dans nos deux définitions, le terme t_2 peut être directement E-égal à t'_1, alors que les définitions de E-cohérence obligent à ce que t_2 soit réduit au moins une fois. Comme nous le verrons plus tard, cette différence vient du fait que nous ne nous restreignons pas à des théories équationnelles régulières, et que la propriété de E-compatibilité n'est pas requise pour l'ordre.

2. Dans la définition de la E-fermeture, nous ne considérons qu'une seule étape de réduction entre t et t_1, alors que la définition de E-cohérence précise qu'il peut y en avoir plusieurs.

Un ensemble de contextes $\mathcal{C}$ est dit *couvrant* si, pour tout terme clos l, la relation $\rightarrow_{\mathcal{C},E}$ est E-fermante.

Proposition 3.1 *L'algorithme de calcul de contextes décrit dans la Figure 3.1 construit un ensemble couvrant.*

La preuve de cette proposition est effectuée en deux étapes. La première consiste à montrer que, pour tout terme clos l, la relation $\rightarrow_{\mathcal{C}_l,E}$ est localement E-fermante (Lemme 3.2). La seconde étape consiste à montrer la E-fermeture de la relation $\rightarrow_{\mathcal{C}_l,E}$ (Lemme 3.3).

Lemme 3.2 *Pour tout terme clos l, la relation $\rightarrow_{\mathcal{C}_l,E}$ est localement E-fermante.*

Preuve : *Selon l'algorithme décrit dans la Figure 3.1, l'ensemble $\mathcal{C}_l$ associé au terme l est :*

$$\mathcal{C}_l = \{(l, \epsilon)\} \cup \{(e\tau, p) \mid (e, p, c) \in \mathcal{C}ont(l),\ \tau \in \mathcal{S}ol(c)\}$$

Soient t un terme clos, (e_l, p_l) un élément de $\mathcal{C}_l$, et $(e \simeq e')$ une équation de E tels que :

$$
\begin{tikzcd}
t \arrow[r, leftrightarrow, "p", "e \simeq e',\ \sigma"'] \arrow[d, "(e_l{,}p_l){,}E"', "q"] & t[e'\sigma]_p \\
t[e_l[\cdot]]_q &
\end{tikzcd}
$$

Tout au long de cette preuve, nous omettrons volontairement de préciser que la constante "$\cdot$" se trouve à la position p_l de e_l, afin de ne pas surcharger les notations.

1. *Si les positions p et q sont indépendantes : le terme à la position q dans $t[e'\sigma]_p$ peut être réduit par $\rightarrow_{(e_l,p_l),E}$, pour obtenir un terme E-égal à $t[e_l[\cdot]]_q$.*

$$\begin{tikzcd} & t[e'\sigma]_p \arrow[d, "(e_l,p_l),E"', "q"] \\ t[e_l[\cdot]]_q \arrow[r, leftrightarrow, "p", "e \simeq e',\ \sigma"'] & t[e'\sigma]_p[e_l[\cdot]]_q \end{tikzcd}$$

2. *Si $p = q{\cdot}o$ (où o peut être la position vide ϵ) : cela signifie que l'étape de E-égalité a été appliquée au niveau, ou en-dessous, du sous-terme à la position q de t. Donc, le sous-terme à la position q de $t[e'\sigma]_p$ est toujours E-égal à e_l, et la réduction peut donc y être appliquée.*

$$\begin{tikzcd} & t[e'\sigma]_p \arrow[dl, "q"', "(e_l,p_l),E"] \\ t[e_l[\cdot]]_q & \end{tikzcd}$$

3. *Si $q = p{\cdot}o$ (où $o \neq \epsilon$) : l'étape de E-égalité a été appliquée au-dessus de l'étape de réduction dans t. Il faut alors différencier deux cas, selon que la réduction a été appliquée à une position variable ou non de e.*

   (a) *Si $p \in \mathcal{FP}os(e)$, la réduction a été appliquée à une position non variable de e. Selon notre algorithme de construction de contextes, $e[e_l]$ fait partie des contextes de $\mathcal{C}ont(l)$, et donc le couple $(e[e_l], o{\cdot}p_l)$ appartient à $\mathcal{C}_l$; il peut alors être utilisé pour réduire $t[e'\sigma]_p$ en $t[e_l[\cdot]]_q$.*

$$\begin{tikzcd} & t[e'\sigma]_p \arrow[dl, "p"', "(e[e_l],o\cdot p_l),E"] \\ t[e_l[\cdot]]_q = t[e[e_l[\cdot]]_o]_p & \end{tikzcd}$$

   (b) *Si $p \notin \mathcal{FP}os(e)$, la réduction s'est produite dans un terme créé par la substitution σ. Soit x la variable de e concernée, instanciée par un terme s dont un sous-terme est E-égal à e_l. En réduisant toutes les autres occurrences de la variable x dans $e\sigma$, nous obtenons un terme $t[e\sigma']_p$, où la substitution σ' ne diffère de σ que par l'affection de $s[e_l[\cdot]]$ à la variable x. L'étape suivante consiste à réduire le terme $t[e'\sigma]$ en $t[e'\sigma']$; mais, si la théorie E n'est pas régulière, il est possible que la variable x n'ait aucune occurrence dans e'.*

      i. *Si $x \in \mathcal{V}ar(e')$, le terme $t[e'\sigma]$ peut effectivement être réduit en $t[e'\sigma']$, qui est E-égal à $t[e\sigma']$.*

$$\begin{tikzcd} & t[e'\sigma]_p \arrow[dd, "(e_l,p_l),E"', "+"] \\ t[e_l[\cdot]]_q \arrow[d, "(e_l,p_l),E"', "*"] & \\ t[e\sigma']_p \arrow[r, leftrightarrow, "p", "e \simeq e',\ \sigma'"'] & t[e'\sigma']_p \end{tikzcd}$$

ii. *Si $x \notin \mathcal{V}ar(e')$, aucune réduction n'a à être appliquée dans le terme $t[e'\sigma]$, et comme σ' ne diffère de σ que par la valeur associée à x, les termes $t[e'\sigma]$ et $t[e'\sigma']$ sont égaux, et donc E-égaux à $t[e\sigma']$.*

$$\begin{tikzcd}
 & t[e'\sigma]_p \arrow[dl, "p" ', "e \simeq e',\ \sigma'"] \\
t[e_l[\cdot]]_q \arrow[d, "(e_l,p_l),E" ', "*"] & \\
t[e\sigma']_p \arrow[uur, shift left, leftrightarrow] &
\end{tikzcd}$$

Dans chaque cas, nous avons montré que les termes $t[e_l[\cdot]]_q$ et $t[e'\sigma]_p$ pouvaient être réduits par $\rightarrow_{\mathcal{C}_l,E}$ en deux termes E-égaux. La relation $\rightarrow_{\mathcal{C}_l,E}$ est donc localement E-fermante, et ceci pour un terme l quelconque. □

Lemme 3.3 *Pour tout terme clos l, la relation $\rightarrow_{\mathcal{C}_l,E}$ est E-fermante.*

Preuve : *La relation $\rightarrow_{\mathcal{C}_l,E}$ est E-fermante si, étant donné un terme t réductible par $\rightarrow_{(e_l,p_l),E}$ (où $(e_l,p_l) \in \mathcal{C}_l$) en un terme $t[e_l[\cdot]]$, pour tout terme t' E-égal à t, t' et $t[e_l[\cdot]]$ sont réductibles par $\rightarrow_{\mathcal{C}_l,E}$ en deux termes E-égaux. Décrivons la situation de départ en décomposant la E-égalité de t et t' en un nombre fini d'étapes de E-égalité.*

$$\begin{tikzcd}
t \arrow[r, leftrightarrow, "p_1", "e_1 \simeq e'_1,\ \sigma_1" '] \arrow[d, "(e_l,p_l),E" ', "q"] & t_1 \arrow[r, leftrightarrow, "p_2", "e_2 \simeq e'_2,\ \sigma_2" '] & t_2 \arrow[r, leftrightarrow] & \cdots \arrow[r, leftrightarrow] & t_{n-1} \arrow[r, leftrightarrow, "p_n", "e_n \simeq e'_n,\ \sigma_n" '] & t_n = t' \\
t[e_l[\cdot]]_q & & & & &
\end{tikzcd}$$

La preuve de la propriété de E-fermeture se fait en appliquant récursivement la propriété de E-fermeture locale en partant du terme t, à chaque fois que l'on va rencontrer un terme duquel partent une étape de E-égalité et une réduction par $\rightarrow_{\mathcal{C}_l,E}$.

Décrivons la situation après avoir appliqué la propriété de E-fermeture locale sur le terme t, selon les cas soulevés dans la preuve du Lemme 3.2 :

Cas 1. *Le terme t_1 est réductible à la position q par $\rightarrow_{(e_l,p_l),E}$ en un terme E-égal à $t[e_l[\cdot]]_q$. La E-fermeture locale doit alors être appliquée pour t_1, entre cette réduction et la E-égalité avec t_2.*

Cas 2. *Le terme t_1 est réductible à la position q par $\rightarrow_{(e_l,p_l),E}$ en $t[e_l[\cdot]]_q$. La E-fermeture locale doit alors être appliquée pour t_1, entre cette réduction et la E-égalité avec t_2.*

Cas 3a. *Le terme t_1 est réductible à la position q par $\rightarrow_{(e_1[e_l],o\cdot p_l),E}$ en $t[e_1[e_l[\cdot]]_o]_{p_1}$. La E-fermeture locale doit alors être appliquée pour t_1, entre cette réduction et la E-égalité avec t_2.*

Cas 3(b)i. *Le terme $t[e_l[\cdot]]_q$ est réductible par $\rightarrow^*_{(e_l,p_l),E}$ en un terme $t[e_1\sigma'_1]_{p_1}$. Le terme t_1 est réductible en au moins une étape par $\rightarrow_{(e_l,p_l),E}$ en un terme $t[e'_1\sigma'_1]_{p_1}$; nous noterons $t'_1[e_l[\cdot]]$ le terme issu de la première réduction par $\rightarrow_{(e_l,p_l),E}$, car la position* *de la réduction n'est pas connue. La propriété de E-fermeture locale doit alors être appliquée sur t_1, entre la première réduction par $\rightarrow_{(e_l,p_l),E}$ et la E-égalité avec t_2.*

$$\begin{tikzcd}
t \arrow[r, leftrightarrow, "p_1", "e_1 \simeq e'_1,\ \sigma_1"'] \arrow[d, "(e_l,p_l),E"', "q"] & t_1 \arrow[r, leftrightarrow, "p_2", "e_2 \simeq e'_2,\ \sigma_2"'] \arrow[d, "(e_l,p_l),E"'] & t_2 \arrow[r, leftrightarrow] & \cdots \\
t[e_l[\cdot]]_q \arrow[d, "(e_l,p_l),E"', "*"] & t'_1[e_l[\cdot]] \arrow[d, "(e_l,p_l),E"', "*"] & & \\
t[e_1\sigma'_1]_{p_1} \arrow[r, leftrightarrow, "p_1", "e_1 \simeq e'_1,\ \sigma'_1"'] & t[e'_1\sigma'_1]_{p_1} & &
\end{tikzcd}$$

Etudions les différentes possibilités pour cette E-égalité, car certains cas sont relativement complexes.

Cas 1. *Si t_2 est réductible en un terme t'_2 E-égal à $t'_1[e_l[\cdot]]$, les étapes de E-fermetures locales suivantes s'appliquent sur t_2, entre la réduction en t'_2 et la E-égalité avec t_3, ainsi que sur $t[e_l[\cdot]]_q$, entre les éventuelles réductions par $\rightarrow_{(e_l,p_l),E}$ et la E-égalité avec t'_2.*

Cas 2. et 3a. *Si t_2 est réductible en $t'_1[e_l[\cdot]]$, l'étape de E-fermeture locale suivante s'applique sur t_2, entre la réduction en $t'_1[e_l[\cdot]]$ et la E-égalité avec t_3.*

Cas 3(b)i. *La situation est décrite dans le schéma ci-dessous.*

$$\begin{tikzcd}
& t[e'_1\sigma_1]_{p_1} = t_1 = t_1[e_2\sigma_2]_{p_2} \arrow[rr, leftrightarrow, "p_2", "e_2 \simeq e'_2,\ \sigma_2"'] \arrow[d, "(e_l,p_l),E"'] & & t_2 \arrow[dd, "(e_l,p_l),E"', "+"] \\
& t'_1[e_l[\cdot]] \arrow[dl, "*", "(e_l,p_l),E"'] \arrow[dr, "*", "(e_l,p_l),E"'] & & \\
t[e'_1\sigma'_1]_{p_1} & & t_1[e_2\sigma'_2]_{p_2} \arrow[r, leftrightarrow, "p_2", "e_2 \simeq e'_2,\ \sigma'_2"'] & t_1[e'_2\sigma'_2]_{p_2}
\end{tikzcd}$$

Le problème qui paraît se poser vient de la réduction possible du terme $t'_1[e_l[\cdot]]$ de deux manières différentes par $\rightarrow_{(e_l,p_l),E}$. Il semble qu'une propriété de confluence soit nécessaire.

Cependant, ces réductions s'appliquent dans des variables de e'_1 et de e_2, au niveau de sous-termes E-égaux à e_l. Plus précisément, ces sous-termes sont égaux (syntaxiquement, i.e. pas E-égaux) au sous-terme dans lequel la réduction a été appliquée pour transformer t_1 en $t'_1[e_l[\cdot]]$.

Donc, les termes $t[e'_1\sigma'_1]_{p_1}$ et $t_1[e_2\sigma'_2]_{p_2}$ peuvent être réduits par $\rightarrow_{(e_l,p_l),E}$ en un même terme t''_1.

La propriété de E-fermeture locale doit alors être appliquée sur les termes t_2, $t[e'_1\sigma'_1]_{p_1}$ et $t_1[e_2\sigma'_2]_{p_2}$.

Cas 3(b)ii. *Si t_2 est E-égal à $t'_1[e_l[\cdot]]$, l'étape de E-fermeture locale suivante s'applique sur $t'_1[e_l[\cdot]]$, entre les éventuelles réductions par $\rightarrow_{(e_l,p_l),E}$ et la E-égalité avec t_2. Si le terme $t'_1[e_l[\cdot]]$ n'est pas réduit par $\rightarrow_{(e_l,p_l),E}$, alors t_2 est E-égal à $t[e'_1\sigma'_1]_{p_1}$.*

L'application du cas 3(b)i de E-fermeture locale sur le terme t est donc compatible avec tous les cas de E-fermeture locale applicables sur le terme t_1.

Cas 3(b)ii. *Le terme $t[e_l[\cdot]]_q$ est réductible par $\rightarrow_{(e_l,p_l),E}$ en un terme E-égal à t_1, et donc E-égal à t'. La propriété de E-fermeture est donc satisfaite.*

Nous avons ainsi montré comment appliquer récursivement la propriété de E-fermeture locale pour obtenir finalement que les termes $t[e_l[\cdot]]$ et t' se réduisent par $\rightarrow_{(e_l,p_l),E}$ en deux termes E-égaux. □

Ceci clôt la preuve de couverture de la théorie E par l'ensemble des contextes construits dans la Figure 3.1.

Voici un exemple d'application récursive de la propriété de E-fermeture locale entre deux termes E-égaux. Notons que la théorie équationnelle utilisée n'est pas régulière, et que l'ensemble des axiomes la représentant n'est pas canonique sur les termes clos.

Exemple 3.4 Soit E la théorie des groupes représentée par les axiomes suivants :

$$\begin{array}{rcll} (x*y)*z & \simeq & x*(y*z) & \quad (1) \\ x*0 & \simeq & x & \quad (2) \\ x*i(x) & \simeq & 0 & \quad (3) \end{array}$$

Considérons le terme clos $0 * i(i(a))$, où a est une constante, et considérons que le sous-terme $i(a)$ est réductible. Le problème que l'on se pose est de faire la correspondance entre ce terme réductible et le terme a, qui est E-égal à $0 * i(i(a))$. Détaillons la preuve de E-fermeture de l'ensemble de contextes $\mathcal{C}_{i(a)}$.

$$\begin{tikzcd}
0 * i(i(a)) \arrow[r, leftrightarrow, "1", "(3)"'] \arrow[d, "(i(a),\epsilon),E"', "2\cdot 1"] & (a*i(a))*i(i(a)) \arrow[r, leftrightarrow, "\epsilon", "(1)"'] \arrow[d, dotted, "(i(a),\epsilon),E"', "2\cdot 1"] & a*(i(a)*i(i(a))) \arrow[r, leftrightarrow, "2", "(3)"'] \arrow[d, dotted, "(i(a),\epsilon),E"', "2\cdot 2\cdot 1"] & a*0 \arrow[r, leftrightarrow, "\epsilon", "(2)"'] & a \\
0*i(\cdot) \arrow[r, dotted, leftrightarrow, "1", "(3)"'] \arrow[dr, dashed, "1", "(a*i(a),2),E"'] & (a*i(a))*i(\cdot) \arrow[r, dotted, leftrightarrow, "\epsilon", "(1)"'] & a*(i(a)*i(\cdot)) \arrow[d, dotted, "(i(a),\epsilon),E"', "2\cdot 1"] & & \\
 & (a*\cdot)*i(\cdot) \arrow[r, dashed, leftrightarrow, "\epsilon", "(1)"'] & a*(\cdot*i(\cdot)) \arrow[uur, dashed, "2", "(3)"'] & &
\end{tikzcd}$$

Cette preuve de E-fermeture entre les termes $0 * i(i(a))$ et a nous donne en plus la marche à suivre pour résoudre des systèmes E-incohérents, comme par exemple

$$\{\ (i(a) \simeq b),\ P(0 * i(b)),\ \neg P(a)\ \}$$

où nous retrouvons une équation traduisant la réductibilité de $i(a)$, une clause contenant $0 * i(b)$ et une dernière contenant a.
Il faut donc appliquer une paramodulation contextuelle de $(i(a) \simeq b)$ dans $P(0*i(b))$, à la position $1\cdot 1$, avec le contexte $(x*i(x), 2, \emptyset)$ et la substitution $\{x \mapsto a\}$; la clause déduite est $P((a*b)*i(b))$. Une résolution entre cette dernière clause et $\neg P(a)$ engendre alors la clause vide. ♦

3.1.4 Notion de redondance

La définition du calcul des contextes donnée dans la Figure 3.1 est relativement primaire. Elle ne peut être mise en œuvre efficacement sans introduire la notion de contextes inutiles. Cette notion est très liée au test de convergence des paires critiques de cohérence de H. Kirchner [Kir85] dans les techniques de complétion, mais notre méthode offre l'avantage de les détecter à l'avance, c'est-à-dire avec pour seule donnée la théorie E.

Considérons l'exemple où E contient les propriétés d'associativité et de commutativité d'un opérateur f.

Exemple 3.5 (Associativité et commutativité) Soit E la théorie représentée par l'ensemble d'axiomes $\{(f(f(x,y),z) \simeq f(x,f(y,z))),\ (f(x,y) \simeq f(y,x))\}$, l'ensemble initial des contextes est :

$$\mathcal{C}ont_0 \ = \ \{\ (f(f(x,y),z),1,\emptyset),\ (f(x,f(y,z)),2,\emptyset)\ \}$$

Nous obtenons donc déjà deux solutions, alors qu'il est bien connu qu'une équation $(l \simeq r)$ ne s'étend que d'une seule manière dans les théories AC, $(f(l,z) \simeq f(r,z))$, c'est-à-dire en ajoutant une nouvelle variable z liée à l et r par l'opérateur AC f.
Les deux solutions trouvées nous disent que cette nouvelle variable peut être ajoutée soit à droite (première solution), soit à gauche (seconde solution) du terme l : $f(l,z)$ et $f(x,l)$. Or, ces deux termes sont AC-égaux à un renommage de variables près. L'une de ces deux solutions est donc inutile, et l'ensemble initial des contextes $\mathcal{C}ont_0$ est réduit à $\{\ (f(f(x,y),z),1,\emptyset)\ \}$. ♦

Nous venons donc de montrer que certains contextes sont redondants, car E-égaux à un autre contexte, modulo un renommage des variables. Cependant, d'autres contextes sont inutiles, comme le montre l'exemple suivant, toujours avec les théories AC.

Exemple 3.6 (Suite de l'Exemple 3.5) Nous avons montré que $\mathcal{C}ont_0$ était réduit à un seul contexte, $(f(f(x,y),z),1,\emptyset)$. Calculons $\mathcal{C}ont_1$: considérant les contextes $(f(f(x_1,y_1),z_1),1,\emptyset)$ et $(f(f(x_2,y_2),z_2),1,\emptyset)$, renommages de l'élément de $\mathcal{C}ont_0$, le contexte suivant appartient à $\mathcal{C}ont_1$:

$$(\ f(f(f(x_2,y_2),z_2),z_1),\ 1{\cdot}1,\ \{f(x_1,y_1) =^?_E f(f(x_2,y_2),z_2)\}\ )$$

Or, le terme $f(f(f(x_2,y_2),z_2),z_1)$ n'est pas AC-égal à $f(f(x,y),z)$ à un renommage de variables près. Cependant, nous pouvons remarquer qu'il est AC-égal au terme $f(f(x_2,y_2),f(z_2,z_1))$, dans lequel le sous-terme $f(x_2,y_2)$ (où le remplacement doit avoir lieu) est resté inchangé et s'est juste déplacé de la position $1{\cdot}1$ à la position 1. Ce terme est moins général que $f(f(x,y),z)$, car il en est une instance par la substitution $\{x \mapsto x_2, y \mapsto y_2, z \mapsto f(z_2,z_1)\}$.
Toute extension utilisant le contexte de $\mathcal{C}ont_1$ est donc une instance d'une extension utilisant le contexte de $\mathcal{C}ont_0$. Le contexte calculé pour $\mathcal{C}ont_1$ est ainsi redondant, et il n'existe qu'un seul contexte, c'est-à-dire qu'une seule manière d'étendre une équation, pour les théories associatives et commutatives. ♦

Ainsi, les contextes redondants ne sont pas simplement définis par un renommage des variables, mais par un filtrage. Donnons une définition de la redondance d'un contexte (sans contraintes d'unification) lorsqu'elle est détectée à une position p du terme le composant.

Définition 3.4 *Un contexte $(e_1,p_1,\emptyset)$ est* **redondant à une position p** *pour un ensemble de contextes $\mathcal{C}$ si $p \in \mathcal{FP}os(e_1)$ et $p_1 = p{\cdot}q$ (où q est une position non vide), et s'il existe un contexte (e_2,p_2,c_2) dans $\mathcal{C}$ et une solution σ de c_2 tels que :*

Un contexte $(e_1, p_1, \emptyset)$ est E-redondant pour un ensemble de contextes $\mathcal{C}$ si,

1. soit $(e_1, p_1, \emptyset)$ est redondant au sommet pour $\mathcal{C}$,
2. soit, pour tout terme e'_1, E-égal à une instance close $e_1\sigma$ de e_1,
   (a) il existe une position p'_1 dans $\mathcal{FP}os(e'_1)$ telle que $(e'_1, p'_1, \emptyset)$ est redondant au sommet par $(e_1, p_1, \emptyset)$, ou bien
   (b) la représentation e'_1 du contexte $(e_1, p_1, \emptyset)$ est E-redondante à une position p' pour $\mathcal{C}$.

Un contexte (e_1, p_1, c_1) est E-redondant pour un ensemble de contextes $\mathcal{C}$ si, pour toute solution σ_1 de c_1, $(e_1\sigma_1, p_1, \emptyset)$ est E-redondant pour $\mathcal{C}$.

Figure 3.3: E-redondance d'un contexte (version simple)

- $e_2[\cdot]_{p_2}\sigma \ =_E \ (e_1|_p)[\cdot]_q$, *afin de garantir l'équivalence des termes* $e_2\sigma$ *et* $e_1|_p$,
- $(e_2|_{p_2})\sigma \ =_E \ e_1|_{p_1}$, *pour s'assurer que les sous-termes où ont lieu les remplacements sont bien équivalents.*

$(e_1, p_1, \emptyset)$ est dit redondant en p pour $\mathcal{C}$, par $(e_2\sigma, p_2, \emptyset)$. Le contexte $(e_1, p_1, \emptyset)$ est **redondant au sommet** *si $p = \epsilon$. Notons que la substitution σ, en plus d'être solution de c_2, est un filtrage de e_2 dans e_1.*

La définition suivante introduit une notion de redondance très différente de la précédente, puisqu'elle s'applique à un terme clos e'_1 et pour un contexte $(e_1, p_1, \emptyset)$ donné. Ce terme clos est appelé une représentation du contexte, car il représente un terme dans lequel le contexte pourrait être appliqué. Dire que e'_1 est E-redondant pour le contexte $(e_1, p_1, \emptyset)$ signifie que ce dernier n'a pas besoin d'être appliqué dans un terme égal à e'_1.

Définition 3.5 *Soit $(e_1, p_1, \emptyset)$ un contexte. Soit e'_1 un terme clos et σ une substitution close tels que e'_1 est E-égal à $e_1\sigma$. La représentation e'_1 du contexte $(e_1, p_1, \emptyset)$ est dite* **E-redondante à une position p** *pour un ensemble de contextes $\mathcal{C}$ s'il existe un terme e_2 E-égal à $e'_1|_p$ et une position p_2 dans $\mathcal{FP}os(e_2)$ tels que :*

- *$(e_2, p_2, \emptyset)$ est redondant au sommet pour $\mathcal{C}$, par un contexte $(e_3, p_3, \emptyset)$,*
- *$(e_1\sigma, p_1, \emptyset)$ est redondant au sommet par $(e'_1[e_3]_p, p{\cdot}p_3, \emptyset)$.*

Il faut noter que la position p peut être la position vide.

La Figure 3.3 décrit un algorithme permettant de reconnaître quelques contextes redondants pour la théorie E. Cet algorithme est assez simple, mais il n'est efficace que sur des théories respectant la propriété de sous-terme, c'est-à-dire telles qu'un terme n'est jamais E-égal à l'un de ses sous-termes. Nous détaillerons dans la prochaine section (Section 3.2) un algorithme plus complexe permettant d'identifier un grand nombre de contextes redondants supplémentaires, particulièrement dans ces théories ne respectant pas la propriété de sous-terme.

Lemme 3.6 *L'ensemble de contextes calculé par l'algorithme de la Figure 3.1, muni de la détection des contextes E-redondants décrite dans la Figure 3.3, est couvrant.*

Preuve : *Il n'est pas besoin de reprendre en détail la preuve de E-fermeture des ensembles de contextes décrite dans la section précédente (Section 3.1.3), car la E-redondance d'un contexte exprimée dans la Figure 3.3 peut être interprétée comme suit.*

1. *S'il existe un contexte $(e_2, p_2, \emptyset)$ dans $\mathcal{C}$ tel que tout remplacement avec le contexte $(e_1, p_1, \emptyset)$ est une instance d'un remplacement avec ce contexte $(e_2, p_2, \emptyset)$, alors utiliser ce contexte $(e_2, p_2, \emptyset)$ au lieu de $(e_1, p_1, \emptyset)$ produit exactement le même résultat.*
2. *Portons notre attention sur les termes dans lesquels le contexte $(e_1, p_1, \emptyset)$ pourrait être appliqué. La première remarque que nous pouvons faire est que ce contexte n'a pas besoin d'être utilisé avec les termes qui sont des instances de e_1, car le remplacement peut être effectué directement à la position p_1. Nous pouvons généraliser cette remarque en déclarant $(e_1, p_1, \emptyset)$ inutile si tous les termes dans lesquels il pourrait être appliqué peuvent être traités sans contexte ou par un autre contexte de $\mathcal{C}$. Mais, effectuer ce test pour tous les termes E-égaux à e_1 ne suffit pas, car un terme E-égal à une instance de e_1 peut ne pas être une instance d'un terme E-égal à e_1. Ainsi, le contexte $(e_1, p_1, \emptyset)$ est E-redondant si, pour tout terme e'_1 E-égal à une instance close $e_1\sigma$ de e_1,*
   (a) *soit un terme E-égal à $e_1|_{p_1}$ apparaît à une position p'_1 de e'_1, i.e. le remplacement peut être directement effectué à cette position, mais il faut vérifier en plus que le résultat est identique à celui obtenu en appliquant le contexte $(e_1, p_1, \emptyset)$,*
   (b) *soit un contexte de $\mathcal{C}$ peut être appliqué à une position p' de e'_1 pour obtenir le même résultat qu'en appliquant le contexte $(e_1, p_1, \emptyset)$ au sommet de e'_1.*

Dans la pratique, vérifier le second point ne consiste pas à étudier toutes les instances closes de e_1, mais plus simplement à énumérer les différentes formes que peuvent prendre ces instances. Nous pouvons d'ailleurs noter que, si le contexte $(e_1, p_1, \emptyset)$ est redondant à une position $p \neq \epsilon$ par un contexte $(e_2, p_2, \emptyset)$, alors toutes les représentations $e'_1 =_E e_1\sigma$ telles que

$$\exists p' \in \mathcal{FP}os(e'_1),\ e'_1|_{p'} =_E (e_1|_p)\sigma \quad \text{et} \quad e'_1[\cdot]_{p'} =_E e_1[\cdot]_p\sigma$$

sont E-redondantes à la position p' par le contexte $(e_2\sigma, p_2, \emptyset)$. □

Le calcul des contextes défini dans la Figure 3.1 doit être appliqué en tenant compte de cette définition de redondance. Un algorithme simple serait, pour chaque contexte nouvellement construit, de vérifier qu'il n'est pas E-redondant pour l'ensemble $\mathcal{C}$ des contextes déjà construits, puis d'ôter de $\mathcal{C}$ les contextes devenus E-redondants par l'ajout de ce nouveau contexte. Il serait même intéressant, lors d'une inférence avec contexte, de vérifier que l'instance utilisée du contexte, et même la représentation de son terme dans la clause où le remplacement doit avoir lieu, n'est pas redondante.

Détaillons quelques exemples de calculs de contextes utiles.

Exemple 3.7 (Associativité) Soit $E = \{(f(f(x, y), z) \simeq f(x, f(y, z)))\}$. L'ensemble initial des contextes est :

$$\mathcal{C}ont_0 \;=\; \{\ (f(f(x, y), z), 1, \emptyset),\ (f(x, f(y, z)), 2, \emptyset)\ \}$$

Aucun des ces contextes n'est E-redondant pour l'autre. Le calcul des contextes au niveau suivant donne :

$$\mathcal{C}ont_1 = \left\{ \begin{array}{l} (\ f(f(f(x_2,y_2),z_2),z_1),\ 1{\cdot}1,\ \{f(x_1,y_1) \stackrel{?}{=}_E f(f(x_2,y_2),z_2)\}\ ) \\ (\ f(x_1,f(x_2,f(y_2,z_2))),\ 2{\cdot}2,\ \{f(y_1,z_1) \stackrel{?}{=}_E f(x_2,f(y_2,z_2))\}\ ) \\ (\ f(f(x_2,f(y_2,z_2)),z_1),\ 1{\cdot}2,\ \{f(x_1,y_1) \stackrel{?}{=}_E f(x_2,f(y_2,z_2))\}\ ) \\ (\ f(x_1,f(f(x_2,y_2),z_2)),\ 2{\cdot}1,\ \{f(y_1,z_1) \stackrel{?}{=}_E f(f(x_2,y_2),z_2)\}\ ) \end{array} \right\}$$

Les deux premiers sont redondants au sommet pour $\mathcal{C}ont_0$. Les deux derniers sont équivalents, à un renommage de variables près.
Il faut noter que les problèmes de E-unification peuvent être supprimés, car le membre gauche est un terme faisant intervenir des variables qui n'apparaissent plus dans le contexte et n'apportent aucune contrainte sur les variables du terme du triplet.
Le seul contexte utile de l'ensemble $\mathcal{C}ont_1$ est donc $(f(f(x_2,f(y_2,z_2)),z_1),\ 1{\cdot}2,\ \emptyset)$, car, bien qu'il soit redondant à la position 1 pour le second contexte de $\mathcal{C}ont_0$, la représentation de son terme sous la forme $f(f(x_2,y_2),f(z_2,z_1))$ n'est pas redondante. De plus, tout contexte de $\mathcal{C}ont_2$ est redondant au sommet pour ce dernier, car il exprime que l'extension se fait en ajoutant une variable de chaque coté du terme à étendre, alors que les contextes de $\mathcal{C}ont_2$ expriment qu'il peut y avoir plusieurs variables de chaque coté.
Donc, l'ensemble des contextes possibles pour l'axiome d'associativité est :

$$\bigcup_{k\geq 0} \mathcal{C}ont_k = \{\ (f(f(x,y),z),1,\emptyset),\ (f(x,f(y,z)),2,\emptyset),\ (f(f(w,f(x,y)),z),1{\cdot}2,\emptyset)\ \}$$

ce qui signifie qu'une équation s'étend en ajoutant une nouvelle variable soit à droite, soit à gauche, soit de chaque coté. ♦

Les deux cas que nous venons de voir (AC et A) sont assez particuliers, car leur nombre de contextes utiles est fini. Cependant, il ne faut pas perdre de vue que de nombreuses théories en engendrent un nombre infini. En voici un exemple.

Exemple 3.8 (Distributivité à droite) Soit $E = \{(f(x,g(y,z)) \simeq g(f(x,y),f(x,z)))\}$. L'ensemble de contextes $\mathcal{C}ont_0$ est composé de trois éléments, tous indispensables :

$$(f(x_1,g(x_2,x_3)),\ 2,\ \emptyset) \qquad (g(f(x_1,x_2),f(x_1,x_3)),\ 1,\ \emptyset) \qquad (g(f(x_1,x_2),f(x_1,x_3)),\ 2,\ \emptyset)$$

En utilisant la notation (i,j) pour désigner le j^e élément de $\mathcal{C}ont_i$, et $j1(i2,j2)$ pour indiquer la construction d'un contexte à partir du $j1^e$ élément de $\mathcal{C}ont_0$ et de $(i2,j2)$, l'ensemble $\mathcal{C}ont_1$ est composé des contextes utiles suivants (aux contraintes simplifiées) :

$$\begin{array}{ll} 1(0,1) & (\ f(x_1,f(x_2,g(x_3,x_4))),\ 2{\cdot}2,\ \emptyset\ ) \\ 2(0,2) & (\ g(g(f(x_1,x_2),f(x_1,x_3)),f(x_1,x_4)),\ 1{\cdot}1,\ \emptyset\ ) \\ 3(0,2) & (\ g(f(x_1,x_2),g(f(x_1,x_3),f(x_1,x_4))),\ 2{\cdot}1,\ \emptyset\ ) \\ 2(0,3) & (\ g(g(f(x_1,x_2),f(x_1,x_3)),f(x_1,x_4)),\ 1{\cdot}2,\ \emptyset\ ) \\ 3(0,3) & (\ g(f(x_1,x_2),g(f(x_1,x_3),f(x_1,x_4))),\ 2{\cdot}2,\ \emptyset\ ) \end{array}$$

Les quatres autres contextes calculables sont E-redondants. Par exemple, le contexte construit par $2(0,1)$,

$$(\ g(f(x_1,g(x_2,x_3)),f(x_4,x_6)),\ 1{\cdot}2,\ \{f(x_4,x_5) \stackrel{?}{=}_E f(x_1,g(x_2,x_3))\}\ ),$$

se simplifie en $(g(f(x_1, g(x_2, x_3)), f(x_1, x_6)),\ 1{\cdot}2,\ \emptyset)$, car l'unique solution de la contrainte de E-unification est $\{x_4 \mapsto x_1,\ x_5 \mapsto g(x_2, x_3)\}$. Le terme de ce contexte est redondant à la position 1 pour le contexte $(0, 1)$. Ses représentations sont :

- $g(g(f(x_1, x_2), f(x_1, x_3)), f(x_1, x_6))$, terme E-redondant à la position 1 pour $(0, 1)$,
- $f(x_1, g(g(x_2, x_3), x_6))$, terme redondant au sommet pour son contexte $2(0, 1)$.

Nous avons donc déjà 8 contextes utiles, et en prêtant attention à leur construction, il est assez facile de s'apercevoir que l'on peut en engendrer une infinité. Par exemple, le contexte $(1, 1)$ a été créé par un plongement du contexte $(0, 1)$ dans lui-même. Et, si on applique à nouveau $(1, 1)$ dans $(0, 1)$ pour obtenir $(2, 1)$, et ainsi de suite, une chaîne infinie de contextes va être engendrée. La forme de ses éléments est la suivante :

$$(\ f(x_1, f(x_2, \ldots f(x_n, g(x_{n+1}, x_{n+2})))),\ 2^n,\ \emptyset\ )$$

Ces contextes sont indispensables, car ils permettent de retrouver le sous-terme $g(x_{n+1}, x_{n+2})$, où le remplacement doit avoir lieu, à partir de la représentation

$$g(f(x_1, f(x_2, \ldots f(x_n, x_{n+1}))), f(x_1, f(x_2, \ldots f(x_n, x_{n+2}))))$$

♦

3.2 Détection de contextes redondants

L'algorithme donné dans la section précédente pour détecter la redondance de contextes est assez simple et efficace pour de nombreuses théories équationnelles, mais il s'avère insuffisant pour les théories simples, c'est-à-dire telles qu'un terme est E-égal à l'un de ses sous-termes. Etudions quelques exemples pour bien cerner les différents problèmes.

Exemple 3.9 Soit E la théorie $\{f(g(x)) \simeq g(x)\}$. La construction des contextes donne :

$$\begin{array}{rcl} \mathcal{C}ont_0 & = & \{\ (f(g(x)), 1, \emptyset)\ \} \\ \mathcal{C}ont_1 & = & \{\ (f(f(g(x))), 1{\cdot}1, \emptyset)\ \} \\ \vdots & & \\ \mathcal{C}ont_n & = & \{\ (f^{n+1}(g(x)), 1^{n+1}, \emptyset)\ \} \\ \vdots & & \end{array}$$

Or, aucun de ces contextes n'est E-redondant selon l'algorithme décrit dans la Figure 3.3. Cependant, en examinant la théorie, on s'aperçoit que seul le contexte de $\mathcal{C}ont_0$ est utile. Voyons cela sur le système E-incohérent suivant :

$$\left\{ \begin{array}{ll} (g(a) \simeq b) & (1) \\ P(b) & (2) \\ \neg P(f(f(b))) & (3) \end{array} \right.$$

En posant que $g(a)$ est plus grand que b, aucune déduction ne peut être effectuée entre la clause (1) et les clauses (2) et (3). Par contre, une paramodulation contextuelle de (1) dans (1), avec le contexte $(f(g(x)), 1, \emptyset)$ de $\mathcal{C}ont_0$ permet de déduire la clause $(f(b) \simeq b)$ (4). Cette dernière montre facilement la contradiction entre les clauses (2) et (3).

Nous pouvons remarquer alors que toute inférence utilisant un contexte pour la clause (1) produit une clause dans laquelle le contexte est réduit par (4) en b. Ainsi, toute paramodulation étendue, utilisant donc deux contextes de (1), produit une clause qui se simplifie en une tautologie $(b \simeq b)$. En ce qui concerne les paramodulations contextuelles possibles à une position p d'un terme, elles peuvent être remplacées par une paramodulation directement à la position p, car le sous-terme à cette position est E-unifiable avec $g(a)$ (grâce à la forme de la théorie E), et la clause déduite est identique à celle obtenue par une paramodulation contextuelle, suivie de la simplification de la clause déduite par (4).

Donc, le contexte de $\mathcal{C}ont_0$ sert à engendrer des clauses comme (4), mais tous les autres contextes sont inutiles. ♦

Exemple 3.10 Soit E la théorie $\{f(f(g(x))) \simeq f(g(x))\}$. La construction des contextes donne :

$$\begin{array}{lcl} \mathcal{C}ont_0 & = & \{\ (f(f(g(x))), 1, \emptyset),\ (f(f(g(x))), 1{\cdot}1, \emptyset),\ (f(g(x)), 1, \emptyset)\ \} \\ \mathcal{C}ont_1 & = & \{\ (f(f(f(g(x)))), 1{\cdot}1, \emptyset),\ (f(f(f(g(x)))), 1{\cdot}1{\cdot}1, \emptyset)\ \} \\ \vdots & & \end{array}$$

Aucun contexte de $\mathcal{C}ont_0$ n'est E-redondant. Cependant, les contextes de $\mathcal{C}ont_1$ le sont, car on retrouve le même problème que dans l'exemple précédent, grâce aux déductions faites par une paramodulation contextuelle et une paramodulation étendue.

En effet, si l'on dispose d'une équation $(l \simeq r)$ où l est E-unifiable avec $f(g(x))$, une paramodulation contextuelle de $(l \simeq r)$ dans $(l \simeq r)$, avec le contexte $(f(f(g(x))), 1, \emptyset)$, produit l'équation $(f(r) \simeq r)$.

Si l'on dispose d'une équation $(g \simeq d)$ où d est E-unifiable avec $g(x)$, une paramodulation étendue entre $(g \simeq d)$ et $(g \simeq d)$, avec les contextes $(f(f(g(x))), 1{\cdot}1, \emptyset)$ et $(f(g(x)), 1, \emptyset)$, produit l'équation $(f(f(d)) \simeq f(d))$.
Grâce à ces deux équations, tous les contextes autres que ceux de $\mathcal{C}ont_0$ sont inutiles. ♦

Ainsi, il est possible de détecter la redondance d'un contexte en tenant compte des clauses qui seront déduites par des étapes d'inférence. Nous présentons dans la Figure 3.4 un algorithme plus complet que celui de la Figure 3.3, où nous détectons les cas semblables à ceux décrits dans les exemples précédents, c'est-à-dire lorsqu'un contexte produirait systématiquement une clause simplifiable en une seconde clause, qui peut être obtenue directement sans contexte, ou avec un autre contexte.

Malheureusement, nous n'avons pas encore réussi à prouver que cet algorithme produit un ensemble couvrant. Nous énonçons donc ce résultat comme une conjecture.

Conjecture 3.7 *L'algorithme de construction des contextes de la Figure 3.1, muni de la détection des contextes E-redondants décrite dans la Figure 3.4, est couvrant.*

Expliquons cependant en quelques mots nos motivations pour utiliser le cas 2 de la Figure 3.4.

> *Si un sous-terme à une position p ($p \neq \epsilon$) de e_1 est E-redondant pour $\mathcal{C}$, par un contexte $(e_2, p_2, \emptyset)$, et*

Un contexte $(e_1, p_1, \emptyset)$ est E-redondant pour un ensemble de contextes $\mathcal{C}$ si,

1. soit $(e_1, p_1, \emptyset)$ est redondant au sommet pour $\mathcal{C}$,
2. soit $(e_1, p_1, \emptyset)$ est redondant à une position $p \neq \epsilon$ pour un contexte $(e_2, p_2, \emptyset)$ de $\mathcal{C}$, et
   (a) soit $e_1|_p =_E e_1|_{p_1}$ et le contexte $(e_1[e_1|_{p_1}]_p, p, \emptyset)$ est E-redondant pour $\mathcal{C}$,
   (b) soit $(e_1, p_1, \emptyset)$ est redondant à une position q, plus interne que p ($q = p{\cdot}o$ où $o \neq \epsilon$), pour un contexte $(e_3, p_3, \emptyset)$ de $\mathcal{C}$, et
      - les termes $e_1|_p$ et $e_1|_q$ sont E-égaux, et
      - le contexte $(e_1[e_1|_q]_p, p{\cdot}q', \emptyset)$ est E-redondant pour $\mathcal{C}$, où q' désigne la position non vide telle que $p_1 = q{\cdot}q'$.
3. soit, pour tout terme e'_1, E-égal à une instance close $e_1\sigma$ de e_1,
   (a) il existe une position p'_1 dans $\mathcal{FP}os(e'_1)$ telle que $(e'_1, p'_1, \emptyset)$ est redondant au sommet par $(e_1, p_1, \emptyset)$, ou bien
   (b) la représentation e'_1 du contexte $(e_1, p_1, \emptyset)$ est E-redondante à une position p' pour $\mathcal{C}$.

Un contexte (e_1, p_1, c_1) est E-redondant pour un ensemble de contextes $\mathcal{C}$ si, pour toute solution σ_1 de c_1, $(e_1\sigma_1, p_1, \emptyset)$ est E-redondant pour $\mathcal{C}$.

Figure 3.4: E-redondance d'un contexte (version complète)

(a) *si ce sous-terme est E-égal au terme $e_1|_{p_1}$ où le remplacement doit être effectué, alors, pour toute équation $(l \simeq r)$ où l est E-unifiable avec $e_1|_{p_1}$, une paramodulation contextuelle de $(l \simeq r)$ dans $(l \simeq r)$ avec le contexte $(e_2, p_2, \emptyset)$ engendrera l'équation $(e_2[r]_{p_2} \simeq r)$.*
Ainsi, si le contexte $(e_1, p_1, \emptyset)$ est utilisé avec $(l \simeq r)$, le terme $e_1[r]_{p_1}$, apparaissant dans la clause déduite, est simplifiable par $(e_2[r]_{p_2} \simeq r)$ en $e_1[r]_p$, ce qui revient à utiliser directement le contexte $(e_1, p, \emptyset)$. Nous vérifions que le problème peut effectivement être ramené à ce dernier contexte par un test de E-redondance de $(e_1[e_1|_{p_1}]_p, p, \emptyset)$ pour $\mathcal{C}$.

(b) *si un autre sous-terme de e_1, à une position q plus interne que p, est E-redondant pour $\mathcal{C}$, par un contexte $(e_3, p_3, \emptyset)$, et les sous-termes aux positions p et q de e_1 sont E-égaux, alors cela signifie que, pour toute équation $(l \simeq r)$ où l est E-unifiable avec $e_1|_{p_1}$, une paramodulation étendue entre $(l \simeq r)$ et $(l \simeq r)$ avec les contextes $(e_2, p_2, \emptyset)$ et $(e_3, p_3, \emptyset)$ de $\mathcal{C}$ permet de déduire l'équation $(e_2[r]_{p_2} \simeq e_3[r]_{p_3})$.*
Donc si le contexte $(e_1, p_1, \emptyset)$ est utilisé avec $(l \simeq r)$, le terme $e_1[r]_{p_1}$, apparaissant dans la clause déduite, est simplifiable par $(e_2[r]_{p_2} \simeq e_3[r]_{p_3})$ en $e_1[e_3[r]_{p_3}]_p$. Cela revient à utiliser directement le contexte $(e_1[e_1|_q]_p, p, \emptyset)$. L'existence de ce dernier est testée par sa E-redondance pour $\mathcal{C}$.

Donnons maintenant quelques exemples de constructions de contextes pour des théories équationnelles courantes.

Exemple 3.11 Soit E la théorie des monoïdes.

$$E = \{\ (f(f(x,y),z) \simeq f(x,f(y,z))),\ (f(x,0) \simeq x),\ (f(0,x) \simeq x)\ \}$$

La construction des contextes donne :

$$\begin{array}{lcl} \mathcal{C}ont_0 & = & \{\ (f(f(x,y),z),1,\emptyset),\ (f(x,f(y,z)),2,\emptyset),\ (f(x,0),2,\emptyset),\ (f(0,x),1,\emptyset)\ \} \\ \mathcal{C}ont_1 & = & \{\ (f(w,f(f(x,y),z)),2{\cdot}1,\emptyset)\ \} \\ \mathcal{C}ont_2 & = & \emptyset \end{array}$$

Seuls ces 5 contextes sont utiles lorsque l'on travaille modulo un monoïde. ♦

Exemple 3.12 Soit E la théorie des groupes, représentée par le système suivant :

$$E = \{\ (f(f(x,y),z) \simeq f(x,f(y,z))),\ (f(x,0) \simeq x),\ (f(x,i(x)) \simeq 0)\ \}$$

La construction des contextes donne :

$$\begin{array}{lcl} \mathcal{C}ont_0 & = & \{\ (f(f(x,y),z),1,\emptyset),\ (f(x,f(y,z)),2,\emptyset),\ (f(x,0),2,\emptyset),\ (f(x,i(x)),2,\emptyset)\ \} \\ \mathcal{C}ont_1 & = & \{\ (f(w,f(f(x,y),z)),2{\cdot}1,\emptyset),\ (f(x,f(y,i(y))),2{\cdot}2,\emptyset),\ (f(f(x,i(x)),y),1{\cdot}2,\emptyset)\ \} \\ \mathcal{C}ont_2 & = & \{\ (f(x,f(f(y,i(y)),z)),2{\cdot}1{\cdot}2,\emptyset)\ \} \\ \mathcal{C}ont_3 & = & \emptyset \end{array}$$

Il existe donc 8 contextes utiles pour la théorie des groupes. ♦

Exemple 3.13 Soit E la théorie des groupes abéliens.

$$E = \{\ (f(f(x,y),z) \simeq f(x,f(y,z))),\ (f(x,y) \simeq f(y,x)),\ (f(x,0) \simeq x),\ (f(x,i(x)) \simeq 0)\ \}$$

La construction des contextes donne :

$$\begin{array}{lcl} \mathcal{C}ont_0 & = & \{\ (f(f(x,y),z),1,\emptyset),\ (f(x,0),2,\emptyset),\ (f(x,i(x)),2,\emptyset)\ \} \\ \mathcal{C}ont_1 & = & \{\ (f(f(x,i(x)),y),1{\cdot}2,\emptyset)\ \} \\ \mathcal{C}ont_2 & = & \emptyset \end{array}$$

Il n'existe ainsi que 4 contextes utiles pour la théorie des groupes abéliens. ♦

3.3 Elimination d'inférences utilisant des contextes

Nous avons défini dans les sections précédentes le calcul des extensions engendrées par la théorie E, sous forme de contextes et en tenant compte d'une notion de redondance. Nous avons également montré dans la Section 2.2 comment simuler leur rôle grâce à l'utilisation de ces contextes dans les règles de paramodulation contextuelle et étendue. Mais, au vu d'un ensemble de clauses, il est possible d'associer à chaque clause un ensemble de *contextes inutiles*, ce qui rend inutile les règles de paramodulation contextuelle et étendue avec ces contextes.

Cette notion d'extensions inutiles a été définie pour les théories associatives et commutatives dans la procédure de G.E. Peterson et M.E. Stickel [PS81] par L. Bachmair et N. Dershowitz dans [BD89], et elle correspond au test de confluence des paires critiques de cohérence [Kir85]. Notons qu'elle correspond à la définition de déductions redondantes [BG94], car toute paramodulation contextuelle ou étendue avec l'une de ces extensions déclarées inutiles engendre une clause redondante.

Avant de définir cette notion d'extensions inutiles, rappelons que l'extension d'une règle $(l_1 \to r_1)$ est notée $(f(l_1, z) \to f(r_1, z))$, où z est une variable n'apparaissant ni dans l_1 ni dans r_1, car le seul contexte utile issu des propriétés AC de l'opérateur f est $(f(f(x, y), z), 1, \emptyset)$ (provenant de l'axiome d'associativité $(f(f(x, y), z) \simeq f(x, f(y, z)))$).

Définition 3.8 (Extension inutile dans les théories AC**)** *Soit R un système de réécriture. Soit $(l_1 \to r_1)$ une règle de R. L'extension $(f(l_1, z) \to f(r_1, z))$ est inutile s'il existe une règle $(l_2 \to r_2)$ dans R et une substitution σ_2 telles que $f(l_1, z) =_{AC} l_2\sigma_2$ et le terme $f(r_1, z)$ se réécrit par R (et modulo AC) en un terme AC-égal à $r_2\sigma_2$.*

Nous étendons cela pour définir dans quel cas un contexte est inutile pour une équation donnée, dans une théorie équationnelle E.

Définition 3.9 (Contexte inutile) *Soit S un ensemble d'équations et $(l_1 \simeq r_1)$ l'une d'entre elles, telle que $l_1 \succ r_1$. Un contexte (e_1, p_1, c_1) de $\mathcal{C}ont(l_1)$ est inutile si, pour toute solution σ_1 de c_1, il existe une équation $(l_2 \simeq r_2)$ dans S et une substitution σ_2 telles que $e_1[l_1]_{p_1}\sigma_1 =_E l_2\sigma_2$ et le terme $e_1[r_1]_{p_1}\sigma_1$ est simplifiable (au sens des règles de simplification définies dans la Section 2.4.2) en un terme E-égal à $r_2\sigma_2$.*

Cette définition est donnée pour un ensemble d'équations S, mais elle peut être facilement étendue à un ensemble de clauses, à condition de vérifier en plus que, pour chaque littéral L_2 appartenant à la même clause que $(l_2 \simeq r_2)$, $L_2\sigma_2$ est également un littéral de l'instance par σ_1 de la clause contenant $(l_1 \simeq r_1)$.

Preuve: (Correction de la définition des contextes inutiles)

L'inutilité de ces contextes est montrée très simplement : soit $C_1 = (l_1 \simeq r_1) \vee D_1$ une clause utilisée dans une inférence avec la solution σ_1 d'un contexte (e_1, p_1, c_1) de $\mathcal{C}ont(l_1)$, jugé inutile par la définition précédente (grâce à une clause $C_2 = (l_2 \simeq r_2) \vee D_2$). Détaillons chaque inférence possible :

- *Paramodulation contextuelle de C_1 dans une clause $C_3 = L_3 \vee D_3$.*

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad\qquad L_3 \vee D_3}{L_3[p_3 \leftarrow e_1[r_1]_{p_1}]\sigma \vee D_1\sigma \vee D_3\sigma}$$

Notons d'abord que la substitution σ est une instance de la substitution σ_1, car σ utilise σ_1 comme solution de c_1 : $\forall x \in \mathcal{D}om(\sigma)$, $x\sigma =_E x\sigma_1\rho$, pour une substitution ρ.

Comme le contexte est inutile, la définition nous dit que $e_1[r_1]_{p_1}\sigma_1$ est simplifiable en $r_2\sigma_2$, et $D_2\sigma_2$ est inclus dans $D_1\sigma_1$. Donc, la clause déduite est simplifiable de la même manière en une clause C E-égale à $L_3\sigma[r_2\sigma_2\rho]_{p_3} \vee D_1\sigma \vee D_3\sigma$.

Or, la définition nous dit également que le terme $l_2\sigma_2$ est E-égal au terme $e_1[l_1]_{p_1}\sigma_1$;

donc, $L_3|_{p_3}\sigma =_E e_1[l_1]_{p_1}\sigma =_E l_2\sigma_2\rho$, *et une paramodulation de la clause* C_2 *dans* C_3, *à la position* p_3 *de* L_3, *est possible :*

$$\frac{(l_2 \simeq r_2) \vee D_2 \qquad\qquad L_3 \vee D_3}{L_3[p_3 \leftarrow r_2]\sigma' \vee D_2\sigma' \vee D_3\sigma'}$$

La clause ainsi obtenue est plus générale que (i.e. subsume) la clause C, *d'où l'intérêt d'éviter la première inférence.*

- *Paramodulation étendue entre* C_1 *et une clause* $C_3 = (l_3 \simeq r_3) \vee D_3$.

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad\qquad (l_3 \simeq r_3) \vee D_3}{(e_1[r_1]_{p_1} \simeq e_3[r_3]_{p_3})\sigma \vee D_1\sigma \vee D_3\sigma}$$

Comme dans le cas précédent, la clause déduite est simplifiable en une clause C E-*égale à* $(r_2\sigma_2\rho \simeq e_3[r_3]_{p_3}\sigma) \vee D_1\sigma \vee D_3\sigma$. *La définition d'un contexte inutile précise que* $l_2\sigma_2 =_E e_1[l_1]_{p_1}\sigma_1$ *; donc, le terme* $e_3[r_3]_{p_3}\sigma$, E-*égal à* $e_1[l_1]_{p_1}\sigma$, *est aussi* E-*égal au terme* $l_2\sigma_2\rho$. *La clause* C *est donc subsumée par* C_2.

- *Paramodulation étendue entre* C_1 *et une clause* $C_3 = (l_3 \simeq r_3) \vee D_3$, *utilisant la solution* σ_3 *d'un contexte* (e_3, p_3, c_3) *de* $\mathcal{C}ont(l_3)$, *également jugé inutile grâce à une clause* $C_4 = (l_4 \simeq r_4) \vee D_4$.

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad\qquad (l_3 \simeq r_3) \vee D_3}{(e_1[r_1]_{p_1} \simeq e_3[r_3]_{p_3})\sigma \vee D_1\sigma \vee D_3\sigma}$$

Pour les même raisons que les cas précédents, la clause déduite est simplifiable en une clause C E-*égale à* $(r_2\sigma_2\rho \simeq r_4\sigma_4\rho') \vee D_1\sigma \vee D_3\sigma$. *De plus, notant* ρ' *la substitution telle que* $\forall x \in \mathcal{D}om(\sigma)$, $x\sigma =_E x\sigma_3\rho'$, *les termes* $l_2\sigma_2\rho$ *et* $l_4\sigma_4\rho'$ *sont* E-*égaux, car* $l_2\sigma_2 =_E e_1[l_1]_{p_1}\sigma_1$ *et* $l_4\sigma_4 =_E e_3[l_3]_{p_3}\sigma_3$. *Une paramodulation appliquée entre les clauses* C_2 *et* C_4 *produit la clause*

$$\frac{(l_2 \simeq r_2) \vee D_2 \qquad\qquad (l_4 \simeq r_4) \vee D_4}{(r_2 \simeq r_4)\sigma' \vee D_2\sigma' \vee D_4\sigma'}$$

qui est plus générale que C.

Ainsi, nous avons montré qu'une clause engendrée par une inférence utilisant un contexte dit inutile était redondante. □

Donc, à chaque fois que nous voudrons appliquer une paramodulation contextuelle ou étendue avec une clause C, il faudra vérifier que le contexte utilisé est bien utile pour l'équation à étendre de C. Ce test est très important pour limiter les déductions, comme cela a déjà été montré pour les théories associatives et commutatives dont nous donnons un exemple ci-dessous.

Exemple 3.14 Pour un opérateur AC f, les équations suivantes n'ont pas besoin d'être utilisées dans les règles de paramodulation contextuelle et étendue, car chaque équation étudiée à part rend son extension redondante.

$$\begin{array}{rcl} f(x,0) & \simeq & 0 \\ f(x,1) & \simeq & x \\ f(x,o(y)) & \simeq & o(f(x,y)) \\ f(x,g(y,z)) & \simeq & g(f(x,y),f(x,z)) \end{array}$$

Alors que les extensions des équations suivantes sont indispensables, si elles sont traitées indépendamment des équations précédentes :

$$\begin{array}{lclcl} f(x,i(x)) & \simeq & 1 & \text{s'étend en} & f(f(x,i(x)),z) \simeq f(1,z) \\ f(x,s(y)) & \simeq & g(x,f(x,y)) & \text{s'étend en} & f(f(x,s(y)),z) \simeq f(g(x,f(x,y)),z) \end{array}$$

Cependant, si ces six équations sont rassemblées en un même ensemble de clauses, l'extension de la dernière devient redondante : le membre gauche est AC-égal à $f(f(x,z),s(y))$, instance de $f(x,s(y))$ en remplaçant x par $f(x,z)$; le membre droit, $f(g(x,f(x,y)),z)$, est simplifiable par l'équation $(f(x,g(y,z))\simeq g(f(x,y),f(x,z)))$ en $g(f(x,z),f(f(x,y),z))$, terme AC-égal à $g(f(x,z),f(f(x,z),y))$, instance de $g(x,f(x,y))$ toujours en remplaçant x par $f(x,z)$. Toutes les conditions sont donc réunies pour affirmer que l'extension de la sixième équation est inutile.

Dans la pratique, l'extension de la cinquième équation n'est pas utilisée sous la forme donnée ci-dessus. En effet, il est plus avantageux de ne considérer les extensions qu'après avoir simplifié au maximum leur membre droit, car cela évite de répéter ces simplifications après chaque utilisation de l'extension. Donc, l'extension simplifiée de l'équation $(f(x,i(x))\simeq 1)$ est $(f(f(x,i(x)),z)\simeq z)$. ♦

La définition des contextes redondants donnée dans la section précédente est en fait très proche de cette notion de contexte utile pour une clause. Plus particulièrement, les points 2a et 2b de la Figure 3.4 déclarent certains contextes E-redondants en anticipant la déduction de clauses par paramodulation contextuelle et étendue. Avec la définition de contexte inutile pour une clause, ces contextes auraient également été déclarés inutiles dès que ces clauses auraient été effectivement déduites.
Reprenons l'Exemple 3.9 pour montrer cela.

Exemple 3.15 Soit E la théorie $\{f(g(x))\simeq g(x)\}$.

- Avec notre définition de contextes E-redondants, seul le contexte $(f(g(x)),1,\emptyset)$ est utile. En effet, pour toute équation $(l\simeq r)$ telle que l est E-unifiable avec $g(x)$, une paramodulation contextuelle de $(l\simeq r)$ dans elle-même avec le contexte précédent engendre l'équation $(f(r)\simeq r)$.
- Avec la définition d'un contexte inutile pour l'équation $(l\simeq r)$, dès que $(f(r)\simeq r)$ est engendrée, tout contexte $(f^i(g(x)),1^i,\emptyset)$ utilisé avec $(l\simeq r)$ devient inutile :

$$f^i(l) =_E l \text{ et le terme } f^i(r) \text{ est simplifié par } (f(r)\simeq r) \text{ en } r$$

Notons que l'unique contexte non E-redondant devient inutile pour l'équation $(l\simeq r)$ dès que $(f(r)\simeq r)$ est engendrée. ♦

Notre définition de E-redondance des contextes peut paraître faire double emploi dans ces cas particuliers, mais elle apporte des avantages essentiels :

- Les contextes sont déclarés E-redondants avant même de connaître les clauses du système, et donc avant que les étapes de paramodulation contextuelle et étendue ne soient appliquées pour engendrer les clauses telle que $(f(r)\simeq r)$ dans l'exemple précédent.

- Cette détection préventive de contextes E-redondants permet de limiter le nombre de contextes à notre disposition, et même, comme dans l'exemple 3.9, de ne conserver qu'un nombre fini de contextes au lieu d'un nombre infini.

4
Complétude de la paramodulation équationnelle

Le problème consistant à prouver la complétude de stratégies de preuve de théorèmes avec égalité est resté essentiel depuis les premiers pas de la déduction automatique. L'une des questions posées était de savoir si le système d'inférence composé de la résolution et de la paramodulation était complet sans les axiomes de réflexivité fonctionnelle et sans paramoduler dans les variables. Une première preuve indirecte de l'inutilité des axiomes de réflexivité fonctionnelle a été donnée par D. Brand [Bra75]. La première preuve directe a été décrite par G.E. Peterson [Pet83] à l'aide des arbres sémantiques. Cependant, cette preuve nécessite l'utilisation d'un ordre de simplification isomorphique à ω sur les atomes clos. J. Hsiang et M. Rusinowitch [Rus89, HR91] ont développé une nouvelle méthode basée sur les arbres sémantiques transfinis pour éviter cette condition et permettre ainsi une plus grande classe d'ordres. Leur méthode a été appliquée à la procédure de complétion de Knuth-Bendix [HR87] et à la complétion conditionnelle [KR87].

D'autres techniques ont été proposées par J. Pais et G.E. Peterson [PP91] et L. Bachmair et H. Ganzinger [BG90]. La première est basée sur le *forcing* et utilise également l'induction transfinie pour construire le modèle de Herbrand d'un ensemble de clauses. La seconde est basée sur la définition de systèmes de réécriture canoniques pour représenter les interprétations équationnelles.

Nous allons utiliser dans ce chapitre la technique des arbres sémantiques transfinis pour prouver la complétude réfutationnelle des règles d'inférence définies dans le Chapitre 2. Dans un premier temps, nous avons étendu cette technique aux théories associatives et commutatives [RV91, RV95]. Nous présentons dans ce chapitre son extension aux théories équationnelles régulières. Cette technique des arbres sémantiques transfinis a également été étendue aux théories AC, puis à une théorie équationnelles E, par E. Paul [Pau92, Pau94]. Cependant, les méthodes obtenues sont assez différentes puisque E. Paul construit un arbre sémantique modulo la théorie E, puis considère la branche droite de cet arbre, alors que nous utilisons l'arbre sémantique de la théorie vide, puis sélectionnons une branche consistante avec la théorie E.

L'adaptation proposée de la technique des arbres sémantiques à la stratégie de superposition permet de montrer la complétude de cette stratégie dans sa totalité. Les adaptations précédemment proposées autorisent soit quelques déductions avec des littéraux non maximaux dans leur clause [Rus91], soit quelques remplacements dans le plus petit membre d'une équation [Hab90].

La preuve de complétude réfutationnelle présentée dans ce chapitre n'est valable que si la théorie équationnelle utilisée est régulière. Ce résultat étend les travaux d'U. Wertz [Wer92] et d'E. Paul [Pau94], qui posent la condition suivante :

> *Si un terme s est E-égal à l'un de ses sous-termes stricts $s|_p$, alors pour tout terme t, $s[t]_p$ doit être E-égal à t.*

Comme nous l'avons précisé dans la Section 2.5.2, cette condition implique la régularité de E, mais la réciproque est fausse.

4.1 Technique des arbres sémantiques

Commençons par introduire les principales notions de cette technique, mais nous recommandons la lecture de [Rus89, HR91] pour plus de détails.

Soit $>$ un ordre de simplification total E-compatible défini sur $\mathcal{A}(\mathcal{P},\mathcal{F},\mathcal{X}) \cup \mathcal{T}(\mathcal{F},\mathcal{X})$ (Définition 1.14), et soit $\succ$ l'ordre correspondant total sur les classes de congruence engendrées par la théorie E considérée (Définition 1.15).
La *base de Herbrand* $\mathcal{A}(\mathcal{P},\mathcal{F})$, c'est-à-dire l'ensemble des atomes clos, peut être ordonnée en une séquence croissante $\{A_i\}_{i<\lambda}$ par l'ordre $>$; i est appelé l'*ordinal* de l'atome A_i et λ l'*ordinalité* de la séquence. Un ordinal α est un *ordinal limite* s'il n'admet pas de prédécesseur : l'atome A_α est le plus petit atome supérieur à une séquence croissante infinie d'atomes. Le successeur d'un ordinal α sera noté $\alpha + 1$, et son prédécesseur α^- (s'il existe).
Etant donné un atome A_α, W_α dénote le *segment initial* formé de l'ensemble des atomes plus petits que A_α : $W_\alpha = \{A_i \mid i < \alpha\}$.

Etant donné un ordinal α, une *interprétation partielle* sur W_α est une fonction I (parfois notée I_α) de W_α dans $\{V, F\}$ définie par :

- Si $(s \simeq s) \in W_\alpha$, alors $I(s \simeq s) = V$,
- Si $(s \simeq t)$, $B[s]$, $B[t] \in W_\alpha$ et $I(s \simeq t) = V$, alors $I(B[s]) = I(B[t])$.

Une *interprétation* est une interprétation partielle définie sur W_λ, i.e. toute la base de Herbrand.

Cette définition d'interprétation s'étend naturellement aux clauses par : soient I une interprétation (partielle) sur W_α, A un élément de W_α, et C la clause close $L_1 \vee \ldots \vee L_n$ dont les atomes appartiennent à W_α. Alors,

- $I(\neg A) \;=\; \neg I(A)$,
- $I(C) \;=\; I(L_1) \vee \ldots \vee I(L_n)$.

I satisfait une clause close C si $I(C) = V$. Sinon, *I falsifie C*.

En ce qui concerne les clauses avec variables, une interprétation *I satisfait une clause C* (ou bien *C est valide dans I*) si, pour toute instance close C' de C, $I(C') = V$. De même,

> *Une clause C est cohérente* s'il existe une interprétation pour laquelle elle est valide. Sinon, *C est incohérente*.
> *Un ensemble de clauses S est cohérent* s'il existe une interprétation satisfaisant toutes les clauses de S. Sinon, *S est incohérent*.

Lorsque l'on travaille dans une théorie équationnelle E, on dit qu'un ensemble de clauses S est *E-incohérent* si $S \cup E$ est incohérent.

Soient u et v deux atomes clos et I une interprétation sur W_α. u est *I-réductible* en v par l'équation $(s \simeq t)$, noté $u \longrightarrow_I^{s \simeq t} v$, si $(s \simeq t)$ est un atome de W_α tel que :

$$u = u[s]\ ,\ s > t\ ,\ u > (s \simeq t)\ ,\ I(s \simeq t) = V \quad et \quad v = u[t]$$

Un atome qui n'est pas I-réductible est dit *I-irréductible*. Au cours des preuves à venir, il nous arrivera souvent d'omettre de préciser le nom de l'interprétation pour ne pas surcharger l'écriture ; nous parlerons donc de *réductibilité* et d'*irréductibilité* d'atomes.

Le théorème suivant énonce que, pour tester la I-réductibilité, il suffit d'utiliser des équations qui, elles, sont I-irréductibles.

Théorème 4.1 (Théorème de la Réduction) *Un atome clos est I-réductible si et seulement s'il est I-réductible par une équation I-irréductible.*

Cet autre théorème montre que les interprétations peuvent être construites par récurrence, comme dans [Pet83].

Théorème 4.2 *Soit $I : W_{\alpha+1} \to \{V, F\}$ telle que I est une interprétation sur W_α. Soit J la restriction de I à W_α. I est une interprétation sur $W_{\alpha+1}$ si et seulement si :*

1. *A_α est J-réductible en un atome B et $I(A_\alpha) = I(B)$, ou bien*
2. *A_α est J-irréductible, de la forme $(t \simeq t)$, et $I(A_\alpha) = V$, ou bien*
3. *A_α est J-irréductible mais pas de la forme $(t \simeq t)$.*

Dans le théorème précédent, I est une *extension* (ou *successeur*) de J. L'ensemble formé par toutes les interprétations partielles est appelé *arbre sémantique transfini*. Un *nœud* est un élément de cet arbre. Un *chemin* (ou *branche*) est une séquence de nœuds $\{I_i\}_{i<\alpha}$ telle que α est un ordinal $(\leq \lambda)$ et le domaine de I_i est W_i.

Par convention, lorsqu'une interprétation I_α admet deux extensions, son *successeur gauche* (ou *nœud gauche*) satisfait l'atome A_α et son *successeur droit* (ou *nœud droit*) falsifie A_α.

Notre arbre sémantique étant construit dans la théorie vide, une interprétation I peut ne pas être un modèle de la théorie E, c'est-à-dire I peut interpréter différemment deux atomes E-égaux. Introduisons donc la notion d'inconsistance d'une interprétation pour la théorie E.

Définition 4.3 (E-(in)consistance) *Une interprétation I définie sur W_α est E-consistante si, pour tout atome A dans W_α, et pour tout atome B E-égal à A,*

- *si $B \in W_\alpha$, alors $I(A) = I(B)$,*
- *si $B \notin W_\alpha$ et $B \longrightarrow_I^{l \simeq r} B'$ (avec $l \succ r$), alors $I(A) = I(B')$.*

Sinon, I est dite E-inconsistante.

Cette définition ne prend pas en compte les atomes B hors de W_α qui sont I-réductibles en un B' par une équation $(l \simeq r)$ où $l =_E r$, car cela revient à faire le test sur l'atome B'. De même, si l'atome B n'appartient pas à W_α et est I-irréductible, l'interprétation I est considérée comme E-consistante, car elle n'influe pas sur la valeur de B.

Définition 4.4 (Arbre sémantique cohérent) *Soit $\mathcal{T}$ un arbre sémantique transfini. L'arbre sémantique cohérent d'un ensemble de clauses S, noté $MCT(S)$, est le sous-arbre maximal de $\mathcal{T}$ tel que, pour chaque nœud I dans $MCT(S)$,*

- *soit I est E-consistante et, pour toute clause C', E-égale à une instance close d'une clause C de S et dont les atomes appartiennent au domaine de I, $I(C') = V$,*
- *soit I est E-inconsistante, mais satisfait toute équation $(u \simeq u')$ de son domaine telle que $u =_E u'$.*

Un *nœud d'échec* I est soit un nœud E-consistant falsifiant une clause C', E-égale à une instance close d'une clause C de S, soit un nœud E-inconsistant falsifiant une équation $(u \simeq u')$ où $u =_E u'$; nous dirons que la clause C' (resp. $(u \simeq u')$) *étiquette* le nœud d'échec I.
Si J est le dernier nœud d'un chemin de $MCT(S)$, alors toute extension de J est un nœud d'échec. Un *chemin maximal* dans $MCT(S)$ est un chemin dont les extensions ne sont pas dans $MCT(S)$; ces extensions sont donc des nœuds d'échec.

Le lemme suivant établit que si une interprétation I_α, définie sur W_α, est un nœud d'échec, alors α ne peut pas être un ordinal limite. Sa preuve est identique à celle présentée dans [HR91] pour la théorie vide.

Lemme 4.5 (Lemme de Clôture) *Soit S un ensemble de clauses. Alors, tout chemin maximal de $MCT(S)$ admet un dernier élément (dans $MCT(S)$).*

Le second lemme ci-dessous énonce une propriété très importante de $MCT(S)$: aucune interprétation E-inconsistante n'est un modèle de S.

Lemme 4.6 *Si une interprétation I de $MCT(S)$ est E-inconsistante, alors tout chemin passant par I est maximal.*

Il s'agit en fait d'une conséquence immédiate du second point de la construction de $MCT(S)$ (Définition 4.4), car dans un chemin, si deux atomes E-égaux sont interprétés différemment, cela signifie qu'une équation $(u \simeq u')$, où $u =_E u'$, a été (ou sera) falsifiée par un nœud de ce chemin. Ainsi, ce nœud est un nœud d'échec et le chemin est maximal, i.e. n'est pas défini sur toute la base de Herbrand.

Une conséquence de ces deux lemmes est :

Corollaire 4.7 *Soit S un ensemble de clauses ; S est E-incohérent si et seulement si tout chemin maximal de $MCT(S)$ s'étend en un nœud d'échec.*

Il est inutile d'ajouter les axiomes de la théorie E à S, car nous avons vu que les seuls chemins pouvant être définis sur toute la base de Herbrand sont E-consistants.

Voici le théorème de complétude de règles d'inférence pour les arbres sémantiques dans une théorie équationnelle E :

Théorème 4.8 (Théorème Fondamental) *Un ensemble de règles d'inférence INF est E-complet si et seulement si $MCT(INF^*(S))$ ne contient que l'interprétation vide à chaque fois que S est E-incohérent.*

4.2 Lemme de relèvement

Le but du lemme de relèvement est de montrer que toute inférence appliquée au niveau clos peut être relevée au niveau général, c'est-à-dire qu'une inférence appliquée entre des instances closes de clauses peut également être appliquée entre les clauses générales correspondantes.

Le principal problème est le relèvement de la règle de paramodulation, illustré par l'exemple suivant.

Exemple 4.1 Soient $P(x,x,a)$ et $(a \simeq b)$ deux clauses $(a \succ b)$; considérant l'instance close $P(a,a,a)$ de $P(x,x,a)$, une paramodulation dans le troisième argument produit $P(a,a,b)$, et la même paramodulation dans la clause générale produit $P(x,x,b)$, qui admet bien $P(a,a,b)$ pour instance.
Cependant, si la paramodulation est appliquée dans le premier argument, produisant $P(b,a,a)$, il est impossible d'engendrer une clause admettant $P(b,a,a)$ pour instance à partir de la clause générale. ♦

La solution est de ne considérer que les instances closes calculées à partir d'une substitution remplaçant les variables par des termes irréductibles, ou plus précisément irréductibles modulo la théorie E. Si l'on reprend l'exemple précédent, la seule instance à considérer est $P(b,b,a)$, car a est réductible. Alors, toute clause déduite par paramodulation dans une instance close de $P(x,x,a)$ est une instance d'une clause déduite par paramodulation dans $P(x,x,a)$.

Enonçons la définition d'irréductibilité d'une substitution :

Définition 4.9 *Soit I une interprétation (partielle) et σ et θ deux substitutions closes. σ est dite I-réductible en θ, dénoté $\sigma \to_I \theta$, si σ est identique à θ sauf pour une variable x, et $\sigma(x) =_E t \longrightarrow_I^{l \simeq r} \theta(x)$, où $l \succ r$ c'est-à-dire l n'est pas E-égal à r. Si σ ne peut être réduite, elle est dite I-irréductible.*

Mais, notre arbre sémantique étant construit dans la théorie vide, l'interprétation I de la définition précédente peut ne pas être un modèle de la théorie E, et une clause $C\sigma$ peut ne pas avoir la même interprétation que la clause $C\theta$. Nous ne pourrons donc appliquer cette définition qu'aux interprétations E-consistantes.

Le théorème suivant énonce que l'on peut ne considérer que les clauses dont la partie substitution est irréductible pour une interprétation E-consistante.

Théorème 4.10 (Théorème de la Substitution Irréductible) *Soient I une interprétation (partielle) E-consistante et $C\sigma$ une clause close dont les atomes sont tous dans le domaine de I. Alors, il existe une substitution close I-irréductible θ telle que $I(C\sigma) = I(C\theta)$.*

La preuve de ce théorème, basée sur la nœthérianité de la relation $\to_I$, est détaillée pour la théorie vide dans [HR91]. Elle reste valide dans notre cas grâce à la propriété de E-consistance de I. Ce théorème permet donc de relever la paramodulation du cas clos au cas général en ne tenant compte, dans les clauses closes, que des positions qui existent déjà dans les clauses générales correspondantes.

Plus généralement, nous considérerons à partir de maintenant qu'une clause C', E-égale à une instance close $C\sigma$ d'une clause C de S, étiquette un nœud d'échec I si la substitution σ est

I-irréductible. Une telle substitution existe toujours car, par définition, un nœud d'échec étiqueté par une clause (et non une équation triviale pour E) est une interprétation E-consistante.

Revenons au relèvement proprement dit des règles d'inférence ; il se fait en deux étapes : d'une part montrer que les conditions pour appliquer une règle sont toujours valables au cas général, et d'autre part que la clause déduite admet bien pour instance la clause déduite dans le cas clos. Nous allons donc utiliser la propriété suivante, issue de la stabilité par instanciation de l'ordre $\succ$.

Proposition 4.11 *Soient A et B deux objets (termes ou atomes). Si $A\sigma \succ B\sigma$ pour une substitution σ, alors $A \not\preceq B$.*

Une conséquence de cette proposition est :

Corollaire 4.12 *Soit σ un E-unificateur des objets $A_1, \ldots, A_m$, et B_1 et B_2 deux objets tels que : $B_1\sigma \succ B_2\sigma$. Il existe un E-unificateur principal τ de $A_1, \ldots, A_m$ tel que : $B_1\tau \not\preceq B_2\tau$.*

Ce corollaire permet de relever les conditions d'ordre des règles d'inférence. La substitution τ est définie par : $\mathcal{D}om(\tau) \subseteq \mathcal{D}om(\sigma)$, $\forall x \in \mathcal{D}om(\sigma)$, $x\sigma =_E x\tau\rho$, pour une substitution close ρ.

Lemme 4.13 (Lemme de Relèvement) *Soient $C_1, \ldots, C_n$ des clauses et σ une substitution close définie sur $\mathcal{V}ar(C_1) \cup \ldots \cup \mathcal{V}ar(C_n)$. Si une règle d'inférence R appliquée entre $C_1\sigma, \ldots, C_n\sigma$ produit une clause C, la même règle R peut être appliquée entre les clauses $C_1, \ldots, C_n$. Elle engendre une clause D qui admet C comme instance close.*

Preuve : *L'inférence appliquée sur les clauses $C_1\sigma, \ldots, C_n\sigma$ utilise une substitution σ' unifiant des objets $A_1\sigma, \ldots, A_m\sigma$ modulo E, et produit une clause C de la forme $C'\sigma\sigma'$. Bien que σ soit close, σ' peut ne pas être l'identité, car certaines règles d'inférence introduisent de nouvelles variables (paramodulations contextuelle et étendue). Soit τ un E-unificateur principal de $A_1, \ldots, A_m$ tel que : $\forall x \in \mathcal{D}om(\sigma') \cup \mathcal{D}om(\sigma)$, $x\sigma\sigma' =_E x\tau\rho$, pour une substitution close ρ.*
Les conditions d'ordre apparaissant dans les règles d'inférence portent toujours sur des objets des clauses initiales. Comme elles sont valides pour l'inférence R sur les clauses closes et comme la substitution σ est irréductible, le Corollaire 4.12 nous permet d'affirmer qu'elles ne sont pas contredites lorsque la substitution τ leur est appliquée.

Si R est une paramodulation ou une superposition (normale ou contextuelle), la position p, où l'inférence est appliquée, existe aussi dans la clause générale par irréductibilité de σ.

La dernière condition devant être vérifiée par τ concerne la règle de paramodulation contextuelle (le cas de la superposition est identique), et consiste à vérifier que le membre gauche de l'équation n'a pas été introduit dans une variable du sous-terme où le remplacement a eu lieu.

> *Soient $C_1 = (l_1 \simeq r_1) \vee C_1'$ et $C_2 = L_2 \vee C_2'$ deux clauses. L'étape de paramodulation contextuelle est appliquée de C_1 dans C_2 à la position p_2 de L_2. Nous avons à montrer la propriété suivante : $\forall x \in \mathcal{V}ar(L_2|_{p_2})$, $l_1\tau \not\trianglelefteq_E x\tau$. Si une telle variable existe, $l_1\tau\rho$ est également un sous-terme modulo E de $x\tau\rho$. Or, le terme $x\tau\rho$ est égal au terme $x\sigma$, par définition de τ, et il existe donc un terme t, E-égal à $x\sigma$, qui est réductible par $(l_1\sigma \simeq r_1\sigma)$. Cela contredit l'hypothèse d'irréductibilité de la substitution σ.*

Ainsi, toutes les conditions sont remplies, et la règle R peut être appliquée entre les clauses $C_1, \ldots, C_n$, pour produire la clause $D\tau$. La clause C déduite au cas clos étant égale à $C'\sigma\sigma'$ et comme $\sigma\sigma' =_E \tau\rho$, la clause $D\tau\rho$ est E-égale à $C'\sigma\sigma'$, et donc à C. Nous avons ainsi montré que la clause C est une instance close de la clause $D\tau$, modulo E. □

Ce Lemme de Relèvement est valable pour les stratégies de paramodulation et de superposition, ainsi que pour la stratégie positive.

4.3 Preuve de complétude

Dans cette section, nous allons prouver la complétude réfutationnelle des règles d'inférence pour la stratégie de paramodulation décrite dans le Chapitre 2 (Section 2.2.1). Les principales différences entre cette preuve et la preuve dans la théorie vide sont la construction de la branche droite dans l'arbre sémantique, et les nombreux cas supplémentaires introduits par la théorie E concernant les nœuds d'échec. Il nous faut montrer que chacun de ces cas est résolu par nos règles d'inférence.

Soit INF l'ensemble des règles d'inférence décrites dans la section 2.2.1, c'est-à-dire les règles de E-factorisation, E-résolution, E-réflexion, E-paramodulation, E-paramodulation contextuelle et E-paramodulation étendue. Cette section est entièrement consacrée à la preuve du Théorème de Complétude de la E-paramodulation (Théorème 2.7). Au cours de ces preuves, nous ne manipulerons que des clauses closes, car nous avons montré dans la section précédente comment passer au cas général en utilisant le Lemme de Relèvement et le Théorème des Substitutions Irréductibles. Traçons d'abord les grandes lignes de la preuve :

Schéma de Preuve : *La preuve se faisant par contradiction, nous supposons que $INF^*(S)$ ne contient pas la clause vide ; alors, $MCT(INF^*(S))$ n'est pas vide. Nous définirons (Définition 4.15) par récurrence une suite de nœuds dans $MCT(INF^*(S))$, appelée branche droite, et prouverons (Proposition 4.18) que tous ses nœuds sont E-consistants, i.e. compatibles avec les axiomes de la théorie E. Dans ce but, nous construirons cette branche droite pour qu'elle évite ce que nous appellerons des nœuds de quasi-échec (Définition 4.14).*

Le dernier nœud Q_γ de cette branche droite s'étend en un ou plusieurs nœuds d'échec (ou de quasi-échec). Nous montrerons dans la Proposition 4.19 qu'une étape d'inférence, utilisant les clauses étiquetant ces nœuds, peut être appliquée et engendrer une clause falsifiée par Q_γ. Nous aurons alors une contradiction avec le fait que Q_γ appartient à $MCT(INF^*(S))$, ce qui signifiera que l'arbre sémantique cohérent est vide, et donc que la clause vide appartient à $INF^*(S)$. □

4.3.1 Construction de la branche droite

Pour prouver le Théorème de Complétude, nous raisonnons par contradiction. Donc, supposons que S est un ensemble E-incohérent de clauses, mais que $INF^*(S)$ ne contient pas la clause vide ; dans ce cas, $MCT(INF^*(S))$ n'est pas vide. Définissons les ensembles suivants :

$$\begin{array}{rcl} \mathcal{GS} & = & \{\ C \mid \exists C' \in INF^*(S),\ C =_E C'\sigma,\ \text{pour une instance close } C'\sigma \text{ de } C'\ \} \\ \mathcal{E} & = & \{\ (u \simeq u') \mid u, u' \in \mathcal{T}(\mathcal{F}),\ u =_E u'\ \} \end{array}$$

$\mathcal{GS}$ est l'ensemble de toutes les instances closes des clauses de $INF^*(S)$, et $\mathcal{E}$ est l'ensemble de toutes les équations closes triviales pour la théorie E. Ces deux ensembles contiennent en fait toutes les clauses pouvant étiqueter un nœud d'échec au bout d'un chemin de $MCT(INF^*(S))$, selon la définition d'un nœud d'échec (page 72). Par définition d'un arbre sémantique cohérent, $MCT(\mathcal{GS})$ est équivalent à $MCT(INF^*(S))$; donc, $MCT(\mathcal{GS})$ est également non vide.

Dans les preuves qui vont suivre, une équation $(u \simeq v)$ vérifiera implicitement $u \geq v$. D'autre part, nous pouvons toujours supposer qu'il existe un unique littéral maximal dans une clause close, car la règle de factorisation permet d'éliminer les autres occurrences de ce littéral. Nous n'insisterons pas plus sur ce point car il est identique au cas standard [HR91], c'est-à-dire dans la théorie vide.

La technique utilisée dans la théorie vide consiste à construire une suite d'interprétations partielles, par récurrence transfinie, en suivant la branche droite de $MCT(\mathcal{GS})$. Ensuite, il est montré que cette branche est vide, ce qui contredit le fait que $MCT(\mathcal{GS})$ n'est pas vide. Cependant, en ce qui nous concerne, cette branche droite peut être E-inconsistante et donc ne pas représenter la théorie dans laquelle nous travaillons. Nous allons ainsi définir et utiliser le chemin E-consistant le plus à droite, c.-à-d. le chemin le plus à droite dont chaque nœud est E-consistant.

Pour détecter les E-inconsistances le plus tôt possible, nous avons à définir une extension de la notion de nœud d'échec, les *nœuds de quasi-échec*. Mais, comme ces inconsistances peuvent être bien dissimulées, cette notion est assez compliquée. Elle est illustrée par la Figure 4.1.

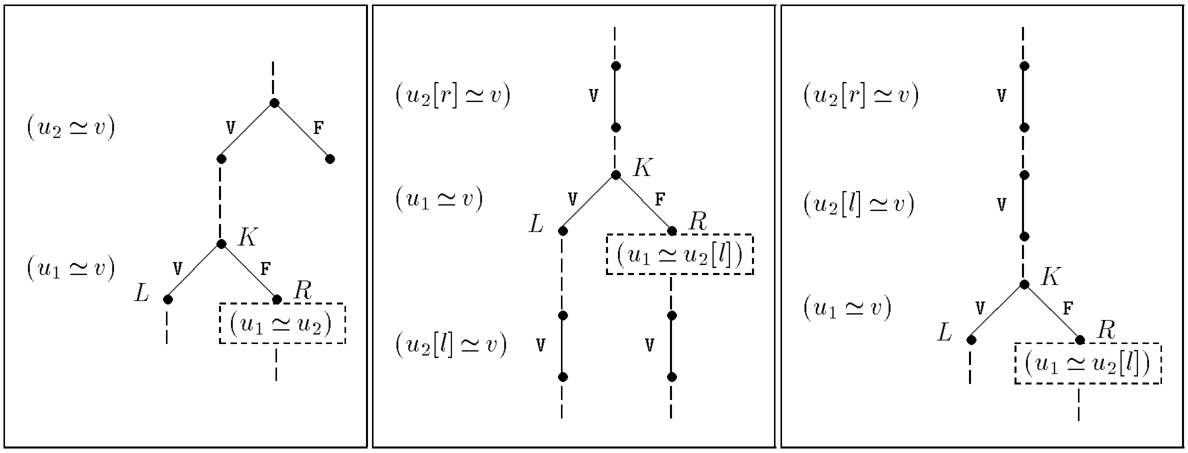


Figure 4.1: Nœuds de quasi-échec (R) où $u_1 =_E u_2$

Définition 4.14 *Soit K un nœud de $MCT(\mathcal{GS})$, défini sur W_α, tel que A_α est un atome équationnel irréductible $(u_1 \simeq v)$ où $u_1 \succ v$; K admet deux extensions L et R : $L(A_\alpha) = V$ et $R(A_\alpha) = F$. Alors, R est un nœud de quasi-échec s'il existe un terme u_2, E-égal à u_1, tel que :*

- *soit l'atome $(u_2 \simeq v)$ appartient à W_α, est irréductible et $K(u_2 \simeq v) = V$,*
- *soit l'atome $(u_2 \simeq v)$ est réductible par une équation $(l \simeq r)$ où $l \succ r$, et K satisfait l'atome $(u_2[r] \simeq v)$.*

L'étiquette d'un tel nœud de quasi-échec est $(u_1 \simeq u_2)$.

Notons qu'un nœud de quasi-échec est E-inconsistant. Dans le premier cas, nous choisirons u_2 minimal, ce qui garantit que le nœud droit au niveau de l'atome $(u_2 \simeq v)$ est soit un nœud d'échec, soit un nœud de quasi-échec satisfaisant le deuxième cas de la définition précédente. Dans le second cas de cette définition, nous pouvons supposer que $(l \simeq r)$ est un atome irréductible, grâce au Théorème de la Réduction.

Maintenant, définissons la *branche droite* de $MCT(\mathcal{GS})$, c'est-à-dire le chemin le plus à droite qui ne contient pas de nœuds de quasi-échec. Ensuite, nous prouverons qu'il est E-consistant.

Définition 4.15 *La branche droite de $MCT(\mathcal{GS})$ est l'interprétation partielle Q_γ, définie sur W_γ, et construite comme suit : Q_0 est l'interprétation vide ; supposant que Q_i est définie pour tout $i < \alpha$, Q_α est défini (si possible) par :*

- *Si α est un ordinal limite, comme dans le cas classique [HR91], on définit simplement Q_α par $\bigcup_{i<\alpha} Q_i$. Alors, par le Lemme de Clôture, Q_α appartient à $MCT(\mathcal{GS})$.*
- *Si α n'est pas un ordinal limite, alors α admet un prédécesseur α^-. Plusieurs cas peuvent se produire :*
  - *Si Q_{α^-} a exactement un successeur J,*
    - *Si J appartient à $MCT(\mathcal{GS})$, alors Q_α est l'interprétation J,*
    - *Si J n'appartient pas à $MCT(\mathcal{GS})$, alors le dernier nœud Q_γ de la branche droite est Q_{α^-},*
  - *Si Q_{α^-} a exactement deux successeurs L et R, $L(A_{\alpha^-}) = V$ et $R(A_{\alpha^-}) = F$, alors*
    - *Si R est un nœud d'échec ou de quasi-échec,*
      - *Si L est un nœud d'échec, alors γ est égal à α^- ,*
      - *Si L n'est pas un nœud d'échec, alors Q_α est l'interprétation L ,*
    - *Si R n'est ni un nœud d'échec, ni de quasi-échec, alors Q_α est R.*

La proposition suivante est une conséquence du Théorème de la Réduction ; elle énonce que tout atome réductible est réductible (modulo E) par une équation dont le nœud droit est un nœud d'échec.

Proposition 4.16 *Soit Q_α la restriction à W_α de la branche droite Q_γ. Si l'atome A_α est réductible par une équation $(l \simeq r)$, avec $l \succ r$, alors il existe un atome $A_{\alpha'}$, E-égal à A_α, qui est réductible par une équation $(g \simeq d)$ telle que $g \succ d$ et le nœud droit de $(g \simeq d)$ est un nœud d'échec étiqueté par une clause de $\mathcal{GS}$. De plus, $Q_\alpha(A_\alpha[r]) = Q_\alpha(A_{\alpha'}[d])$.*

Preuve : *Supposons que cette proposition soit vraie pour tous les atomes plus petits que (mais non E-égaux à) A_α.*
Le Théorème de la Réduction nous permet de choisir $(l \simeq r)$ irréductible ; soit R_l son nœud droit : $R_l(l \simeq r) = F$.

- *Si R_l est un nœud d'échec, alors il ne peut pas être étiqueté par une équation de $\mathcal{E}$ puisque $l \succ r$ implique que $l \neq_E r$. Donc, $(l \simeq r)$ est l'équation recherchée.*
- *Si R_l est un nœud de quasi-échec, étiqueté par $(l \simeq l')$ où $(l' \simeq r)$ est irréductible (et plus petit que $(l \simeq r)$), il faut alors reprendre tout le raisonnement avec l'atome $(l' \simeq r)$ et son nœud droit.*

- *Sinon, R_l est un nœud de quasi-échec étiqueté par $(l \simeq l'[a])$, tel que l'atome $(l'[a] \simeq r)$ est réductible par une équation $(a \simeq b)$ en $(l'[b] \simeq r)$ $(a \succ b)$; de plus, $R_l(l'[b] \simeq r) = V$. L'atome $A_\alpha[l'[a]]$, E-égal à $A_\alpha[l]$, est ainsi réductible par $(a \simeq b)$ en $A_\alpha[l'[b]]$, et $Q_\alpha(A_\alpha[l'[b]]) = Q_\alpha(A_\alpha[r])$.*
  Par hypothèse de récurrence, nous avons supposé que cette proposition était vraie pour tout atome plus petit que A_α. Donc, il existe une équation $(a' \simeq b')$ $(a' \succ b')$, dont le nœud droit est un nœud d'échec, réduisant un atome $(l'' \simeq r)$ E-égal à $(l[a] \simeq r)$ en conservant la valeur de l'interprétation de ces atomes. Cela nous permet de dire que l'équation $(a' \simeq b')$ est l'atome recherché, puisqu'il réduit l'atome $A_\alpha[l'']$, E-égal à A_α, en un atome ayant la même interprétation que $A_\alpha[r]$.

Dans les trois cas nous avons trouvé des atomes $A_{\alpha'}$ et $(g \simeq d)$ vérifiant les conditions énoncées. □

Une autre proposition importante est citée puis prouvée ci-dessous. Elle montre qu'une règle de E-paramodulation (contextuelle) peut être appliquée quand deux clauses étiquettent des nœuds d'échec et que le littéral maximal de l'une est réductible par le littéral maximal de l'autre.

Proposition 4.17 *Soient $C_1 = (l_1 \simeq r_1) \vee D_1$ et $C_2 = L_2 \vee D_2$ deux clauses de $\mathcal{GS}$ étiquetant des nœuds d'échec le long de la branche droite de $MCT(\mathcal{GS})$, et telles qu'il existe un littéral L, E-égal à L_2, réductible par $(l_1 \simeq r_1)$ $(l_1 \succ r_1)$. Alors, le dernier nœud de la branche droite Q_γ falsifie une clause de $\mathcal{GS}$.*

Preuve : *Les atomes $(l_1 \simeq r_1)$ et L_2 étant maximaux dans leur clause respective, Q_γ falsifie D_1 et D_2. D'autre part, L étant réductible à une position q par $(l_1 \simeq r_1)$ en $L[r_1]_q$ et par définition d'un nœud d'échec (propriété de E-consistance respectée), les atomes correspondant aux littéraux L et $L[r_1]_q$ ont même valeur de vérité (celle de l'atome de L_2).*
Soient $C'_1 = (l'_1 \simeq r'_1) \vee D'_1$ et $C'_2 = L'_2 \vee D'_2$ les clauses de $INF^*(S)$ admettant respectivement C_1 et C_2 comme instances closes pour une substitution σ. Comme L est réductible par $(l_1 \simeq r_1)$ et la théorie E est régulière, nous avons vu dans le Chapitre 3 sur la gestion des extensions que le littéral $L'_2\sigma$ est réductible soit par $(l_1 \simeq r_1)$, soit par une extension de $(l_1 \simeq r_1)$ (Lemme 3.3: propriété de E-fermeture). Il existe donc une position o dans $L'_2\sigma$ telle que : $o \in \mathcal{FP}os(L'_2\sigma)$ et soit $L'_2\sigma|_o =_E l'_1\sigma$ et $L'_2\sigma[\cdot]_o =_E L[\cdot]_q$, soit il existe un contexte $(e_1, p_1, c_1) \in \mathcal{C}ont(l_1)$ et une substitution close σ' solution de c_1 tels que $L'_2\sigma|_o =_E e_1\sigma'[l'_1\sigma]_{p_1}$.

- *Si $L'_2\sigma|_o$ est E-égal à $l'_1\sigma$, alors une* ***E*-paramodulation** *de $C'_1\sigma$ dans $C'_2\sigma$, à la position o, permet d'engendrer la clause $L'_2\sigma[r'_1\sigma]_o \vee D'_1\sigma \vee D'_2\sigma$. La clause déduite est E-égale à $L[r_1]_q \vee D_1 \vee D_2$.*
- *Si $L'_2\sigma|_o$ est E-égal à un terme $e_1\sigma'[l'_1\sigma]_{p_1}$, alors une* ***E*-paramodulation contextuelle** *de $C'_1\sigma$ dans $C'_2\sigma$, à la position o et utilisant le contexte (e_1, p_1, c_1), permet d'engendrer la clause $L'_2\sigma[e_1\sigma'[r'_1\sigma]_{p_1}]_o \vee D'_1\sigma \vee D'_2\sigma$. Par E-consistance du nœud d'échec étiqueté par la clause C_2, le littéral $L'_2\sigma[e_1\sigma'[r'_1\sigma]_{p_1}]_o$ a la même interprétation que L.*

Ainsi, dans les deux cas, la clause déduite appartient à $\mathcal{GS}$, mais est falsifiée par Q_γ. Un nœud d'échec devrait donc couper la branche droite. Nous en concluons que Q_γ ne peut pas appartenir à $MCT(\mathcal{GS})$. □

Il faut noter que, dans la preuve précédente, la position o a été trouvée dans $L_2'\sigma$, mais en fait il s'agit d'une position de L_2', car le Lemme de Relèvement nous dit qu'il ne faut considérer que les substitutions irréductibles. Dans cette preuve, nous n'avons pas mentionné que l'équation $(l_1 \simeq r_1)$ ne doit pas être plus grande que l'atome correspondant au littéral L_2, mais nous verrons dans les preuves à venir que cette condition est toujours vérifiée.

4.3.2 Consistance de la branche droite

Proposition 4.18 *Si $MCT(\mathcal{GS})$ n'est pas vide, sa branche droite Q_γ n'est pas vide et est E-consistante.*

La preuve de cette Proposition, donnée ci-dessous, fait appel à plusieurs lemmes qui seront énoncés et démontrés dans l'Appendice A.1.

Preuve: *Premièrement, Q_γ n'est pas vide car, par construction de l'arbre sémantique, A_0 est un atome $(a \simeq a)$ et $Q_1(A_0) = V$; donc, Q_1 ne peut être ni un nœud d'échec ni un nœud de quasi-échec.*

Enfin, nous avons à prouver que Q_γ est E-consistant. Pour cela, nous raisonnons par contradiction : si Q_γ est E-inconsistant, il existe un ordinal minimal α tel que Q_α est E-inconsistant. Nécessairement, **α n'est pas un ordinal limite** *car, comme dans le cas classique [HR91], si α est un ordinal limite, Q_α est défini par $\bigcup_{i<\alpha} Q_i$, et donc l'un des Q_i est E-inconsistant, ce qui est impossible puisque nous avons choisi α minimal.*

Comme α n'est pas un ordinal limite, il admet un prédécesseur α^-. Notons K l'interprétation partielle Q_{α^-} et B l'atome A_{α^-}. Par minimalité de α, K est E-consistant, et

$$\exists A_\beta =_E B \ \text{ tel que } \begin{array}{l} \bullet \ \text{soit } A_\beta \in W_\alpha \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta) \\ \bullet \ \text{soit } A_\beta \notin W_\alpha,\ A_\beta \longrightarrow_K^{l \simeq r} A_\beta[r] \text{ où } l \succ r, \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta[r]) \end{array}$$

La preuve qu'une telle interprétation Q_α n'existe pas se décompose ainsi : nous prouvons dans un premier temps (Lemme A.1) que B est une équation $(u_2 \simeq v)$ et A_β une équation $(u_1 \simeq v)$, où $u_1 =_E u_2$, $u_2 \succ v$ et v est irréductible ; alors, en supposant que A_β est le plus petit atome E-égal à B permettant de détecter la E-inconsistance de Q_α, nous détaillons les différents cas possibles :

- *Si K a exactement un successeur, Q_α (Section A.1, Paragraphe I) : nous montrons d'abord que B est réductible, puis que A_β est irréductible et enfin que $K(A_\beta) = F$. Cela signifie qu'un nœud de quasi-échec devrait couper la branche droite au niveau du nœud droit de l'atome A_β.*
- *Si K a exactement deux successeurs L et R, au moins l'un d'entre eux n'est ni un nœud d'échec ni un nœud de quasi-échec (Q_α), $L(B) = V$ et $R(B) = F$.*

- *Si Q_α est L (Section A.1, Paragraphe II) : l'atome A_β, falsifié par K, est réductible par une équation $(l \simeq r)$ dont le nœud droit est un nœud d'échec ; R est un nœud de quasi-échec, c'est-à-dire qu'il existe un atome E-égal à B qui est réductible par une équation $(g \simeq d)$ dont le nœud droit est un nœud d'échec. A partir de cela, nous en déduisons que Q_α ne peut pas appartenir à $MCT(\mathcal{GS})$, car elle falsifie une clause de $\mathcal{GS}$ engendrée par une paramodulation étendue entre les clauses étiquetant les nœuds droits de $(g \simeq d)$ et $(l \simeq r)$.*
- *Si Q_α est R (Section A.1, Paragraphe III) : l'atome A_β est valide pour K et, qu'il soit réductible ou pas, Q_α devrait être un nœud de quasi-échec.*

Donc, dans tous les cas, nous trouvons une contradiction avec le fait que Q_α appartient à la branche droite de $MCT(\mathcal{GS})$. L'unique conclusion est que la branche droite Q_γ est E-consistante. □

4.3.3 Génération de la clause vide

Comme nous venons de montrer que, si la clause vide n'a pas été engendrée, la branche droite Q_γ de $MCT(\mathcal{GS})$ n'est pas vide et est E-consistante, il nous reste à montrer que Q_γ falsifie une clause de $\mathcal{GS}$, et donc que $MCT(\mathcal{GS})$ est vide. Cela clôturera la preuve du Théorème de Complétude, car l'unique raison pour laquelle l'arbre sémantique cohérent est vide est que la clause vide appartient à $\mathcal{GS}$, et donc à $INF^*(S)$.

Voici donc la dernière proposition qui montre que la branche droite est vide, et donc que la clause vide a été engendrée par notre système de règles d'inférence.

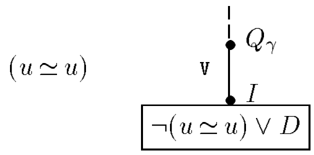
Proposition 4.19 *La branche droite est vide, et donc la clause vide appartient à $INF^*(S)$.*

Preuve : *Le dernier nœud Q_γ de la branche droite est défini sur W_γ, c'est-à-dire pour tous les atomes A_i tels que $i < \gamma$; il appartient à $MCT(\mathcal{GS})$ et chacune de ses extensions est un nœud d'échec ou de quasi-échec. Comme dans la théorie vide, nous avons trois cas principaux à étudier :*

1. *Soit Q_γ a une seule extension I et A_γ est irréductible,*
2. *Soit Q_γ a deux extensions L et R,*
3. *Soit Q_γ a une seule extension I et A_γ est réductible.*

Détaillons chacun de ces cas.

Cas 1 : *Q_γ a une extension I et A_γ est irréductible.*



Par construction de l'arbre sémantique, I satisfait l'atome A_γ, qui est en fait une équation de la forme $(u \simeq u)$, où u est un terme clos irréductible. I étant un nœud d'échec, il falsifie une clause C de $\mathcal{GS}$.

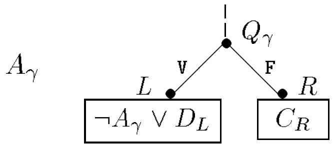
Q_γ ne falsifiant pas C et Q_γ et I ne différant que par la valeur associée à A_γ, nous avons

$$C \;=\; \neg(u \simeq u) \vee D \quad \text{et} \quad Q_\gamma(D) = F,$$

pour une clause D dont les atomes sont plus petits que $(u \simeq u)$.
La clause C appartient à $\mathcal{GS}$ et il existe une clause C' dans $INF^*(S)$ et une substitution close σ telles que $C'\sigma =_E C$.
Par une étape de E-réflexion dans la clause $C'\sigma$, nous déduisons une clause E-égale à D. Donc, D appartient à $\mathcal{GS}$, et comme elle est falsifiée par Q_γ, cette dernière interprétation ne peut appartenir à $MCT(\mathcal{GS})$, et donc à la branche droite.

Ce cas ne peut donc pas se produire.

Cas 2 : *Q_γ a deux extensions L et R (donc A_γ est irréductible).*



Le nœud Q_γ a deux extensions qui sont des nœuds d'échec ou de quasi-échec. Comme l'extension gauche L satisfait l'atome A_γ, L est un nœud d'échec et falsifie une clause C_L de $\mathcal{GS}$. Cette clause est de la forme $\neg A_\gamma \vee D_L$, où chaque atome de D_L est plus petit que A_γ. Q_γ et L ne différant que par la valeur associée à A_γ, Q_γ falsifie également D_L.
Notons C'_L la clause $\neg A'_\gamma \vee D'_L$ de $INF^*(S)$ qui admet C_L comme instance close (modulo E), pour une substitution σ.

Quant à R, il s'agit soit d'un nœud d'échec, soit d'un nœud de quasi-échec. Etudions ces deux cas.

Cas 2.1 : *Si R est un nœud d'échec.*
La clause C_R étiquetant R appartient soit à $\mathcal{GS}$, soit à $\mathcal{E}$.

Cas 2.1.1 : *Si $C_R \in \mathcal{GS}$.*
C_R est de la forme $A_\gamma \vee D_R$ et il existe une clause C'_R de $INF^*(S)$ telle que $C'_R\sigma =_E C_R$. Comme pour D_L, Q_γ falsifie D_R.
Donc, une étape de E-résolution peut être appliquée entre les clauses $C'_L\sigma$ et $C'_R\sigma$, et déduit une clause E-égale à $D_L \vee D_R$; cette clause appartient ainsi à $\mathcal{GS}$, et, comme elle est falsifiée par Q_γ, l'interprétation Q_γ ne peut pas être dans $MCT(\mathcal{GS})$.

Cas 2.1.2 : *Si $C_R \in \mathcal{E}$.*
Cela signifie que A_γ est une équation $(u \simeq v)$ où u et v sont E-égaux. Une E-réflexion dans la clause $C'_L\sigma$ permet de déduire la clause $D'_L\sigma$, E-égale à D_L. D_L est donc une clause de $\mathcal{GS}$, et nous en concluons que Q_γ ne peut être un nœud de la branche droite puisqu'elle falsifie D_L.

Cas 2.2 : *Si R est un nœud de quasi-échec.*
Par définition d'un nœud de quasi-échec, A_γ est un atome $(u_2 \simeq v)$, où $u_2 \succ v$, et R est étiqueté par une équation $(u_2 \simeq u_1)$ où u_2 et u_1 sont E-égaux. Différencions les deux possibilités pour ce nœud de quasi-échec :

- *si* $(u_1 \simeq v)$ *est irréductible, cet atome est validé par* Q_γ. *Or,* $\neg(u_1 \simeq v) \vee D_L$ *étant une clause* E*-égale à* C_L*, elle appartient aussi à* $\mathcal{GS}$ *et un nœud d'échec étiqueté par cette clause devrait couper la branche droite au niveau de* $(u_1 \simeq v)$.

- *si* $(u_1 \simeq v)$ *est réductible par une équation* $(g \simeq d)$ *(où* $g \succ d$*) en* $(u_1[d] \simeq v)$ *et* $Q_\gamma(u_1[d] \simeq v) = V$*, la Proposition 4.16 nous permet d'affirmer qu'il existe une équation* $(l \simeq r)$*, dont le nœud droit est un nœud d'échec, qui réduit un atome* E*-égal à* $(u_1[g] \simeq v)$*, c'est-à-dire à* A_γ. *Soit* C_1 *la clause étiquetant le nœud droit de cet atome* $(l \simeq r)$*. Par la Proposition 4.17, une inférence (*E*-paramodulation ou* E*-paramodulation contextuelle) de* C_1 *dans* C_L *engendre une clause falsifiée par* Q_γ.

Dans les deux cas, nous avons montré que Q_γ *falsifiait une clause de* $\mathcal{GS}$*. Cette interprétation ne peut donc pas appartenir à la branche droite.*

Le cas 2 est également impossible.

Cas 3 : Q_γ *a une extension* I *et* A_γ *est réductible.*

Soit $(g \simeq d)$ *une équation telle que* $g > d$, $Q_\gamma(g \simeq d) = V$ *et il existe une position* p *dans* A_γ *telle que*

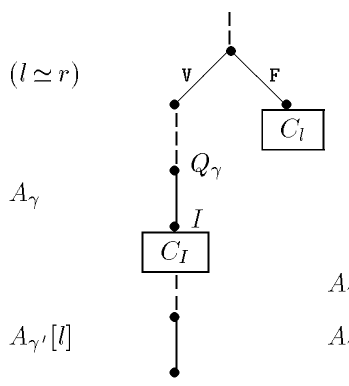
- $A_\gamma|_p = g$ *si* A_γ *n'est pas un littéral équationnel, ou bien* $I(A_\gamma) = F$,
- $s|_p = g$ *si* A_γ *est une équation* $(s \simeq t)$*, où* $s > t$ *et* $I(A_\gamma) = V$.

Dans le second cas, l'existence de $(g \simeq d)$ *n'est pas immédiate. Nous pouvons la prouver ainsi : supposons que* s *est irréductible ; alors* t *doit être réductible en un terme* t' *; mais, il est aisé de vérifier que* $Q_\gamma(s \simeq t') = V$ *et* $(s \simeq t')$ *est plus petit que* $(s \simeq t)$ *; donc* $(s \simeq t)$ *est réductible en* s *par* $(s \simeq t')$*, ce qui contredit l'hypothèse de départ.*

Notons que l'extension I *de* Q_γ *est un nœud d'échec, car* A_γ *est réductible, étiqueté par une clause* C_I *de* $\mathcal{GS}$*, car autrement* Q_γ *serait* E*-inconsistant :* A_γ *serait un atome* $(u \simeq v)$ *où* $u =_E v$, $Q_\gamma(u \simeq v) = Q_\gamma(u[d] \simeq v) = F$*, alors que* Q_γ *validerait* $(u \simeq u[d])$*,* E*-égal à* $(u[d] \simeq v)$*, puisque* $(u \simeq u[d])$ *est réductible en* $(u[d] \simeq u[d])$.

D'autre part, g *ne peut être* E*-égal à* d*, car une clause* E*-égale à celle étiquetant* I *devrait alors couper la branche droite au niveau de l'atome* $A_\gamma[d]$.

Donc, l'atome A_γ *est réductible par* $(g \simeq d)$ *et* $g \succ d$*. Par la Proposition 4.16, il existe un atome* $A_{\gamma'}$ E*-égal à* A_γ *et réductible par une équation irréductible* $(l \simeq r)$ *(où* $l \succ r$*) dont le nœud droit est un nœud d'échec étiqueté par une clause* C_l *de* $\mathcal{GS}$*. Une étape de* E*-paramodulation, ou de* E*-paramodulation contextuelle, peut alors être appliquée de* C_l *dans* C_I *et engendrer une clause de* $\mathcal{GS}$ *falsifiée par* Q_γ *(Proposition 4.17).*



$$A_{\gamma'}[l] \longrightarrow^{l \simeq r}_{Q_\gamma} A_{\gamma'}[r]$$

$$A_{\gamma'}[l] =_E A_\gamma \quad \text{et} \quad Q_\gamma(A_{\gamma'}[r]) = I(A_\gamma)$$

Ainsi, dans ce dernier cas nous avons également trouvé une contradiction avec le fait que Q_γ appartient à la branche droite de $MCT(\mathcal{GS})$.

Nous venons de détailler tous les cas possibles. Comme ils se concluent tous de la même manière (Q_γ ne peut pas appartenir à la branche droite), nous en déduisons que cette branche droite est vide, et donc que le sous-arbre $MCT(\mathcal{GS})$ l'est également. Or, le seul moyen de ne laisser que l'interprétation vide dans $MCT(\mathcal{GS})$ est que la clause vide appartienne à $INF^*(S)$. Ceci conclut la preuve du Théorème de Complétude. □

4.4 Stratégie de superposition

Soit INF l'ensemble des règles d'inférence décrites dans la section 2.2.2, c'est-à-dire les règles de E-factorisation, E-factorisation équationnelle, E-résolution, E-réflexion, E-superposition, E-superposition contextuelle et E-superposition étendue, auxquelles viennent s'ajouter les règles de E-paramodulation et E-paramodulation contextuelle applicables uniquement dans des littéraux non équationnels.

Cette section est consacrée à la preuve de E-complétude de INF (Théorème 2.12), c'est-à-dire à montrer que, pour tout ensemble E-incohérent de clauses S, $INF^*(S)$ contient la clause vide. Nous effectuerons la preuve dans le cas clos, grâce au Lemme de Relèvement et au Théorème des Substitutions Irréductibles.

Comme pour la stratégie de paramodulation, la preuve du Théorème de Complétude de la E-superposition se fait par contradiction. Supposons que S soit un ensemble E-incohérent de clauses, mais que $INF^*(S)$ ne contienne pas la clause vide. Alors, l'arbre sémantique cohérent $MCT(INF^*(S))$ n'est pas vide. Pour représenter l'ensemble des clauses pouvant étiqueter un nœud d'échec dans cet arbre, nous définissons les ensembles suivants :

$$\begin{array}{rcl} \mathcal{GS} & = & \{\ C \mid \exists C' \in INF^*(S),\ C =_E C'\sigma,\ \text{pour une instance close } C'\sigma \text{ de } C'\ \} \\ \mathcal{E} & = & \{\ (u \simeq u') \mid u, u' \in \mathcal{T}(\mathcal{F}),\ u =_E u'\ \} \end{array}$$

$\mathcal{GS}$ est l'ensemble de toutes les instances closes des clauses de $INF^*(S)$, et $\mathcal{E}$ l'ensemble de toutes les équations closes triviales pour la théorie E. Par définition d'un arbre sémantique cohérent, $MCT(\mathcal{GS})$ est équivalent à $MCT(INF^*(S))$; donc, $MCT(\mathcal{GS})$ est également non vide.

Nous détaillons dans cette section les différentes adaptations nécessaires pour exprimer la stratégie de superposition lors de la construction de la branche droite de $MCT(\mathcal{GS})$, puis montrons que cette branche est E-consistante, et enfin qu'elle ne peut être que vide, ce qui implique que le sous-arbre $MCT(\mathcal{GS})$ est vide et donc que la clause vide appartient à $INF^*(S)$.

4.4.1 Adaptation de la technique des arbres sémantiques

La stratégie de superposition est différente de la stratégie de paramodulation par le fait qu'elle interdit tout remplacement dans le plus petit membre (membre droit) d'une équation. Cependant, lors de la mise en place d'un nœud d'échec dans l'arbre sémantique, il est possible que le littéral maximal de la clause étiquetant ce nœud soit une équation réductible dans son membre droit. Nous avons déjà vu pour la stratégie de paramodulation que, si ce littéral est négatif, la réductibilité du membre droit implique la réductibilité du membre gauche. Mais si ce littéral est positif, il nous faut absolument éviter d'appliquer une superposition dans cette clause, car elle ne pourrait être appliquée que dans le membre droit. Dans ce but, nous introduisons une nouvelle notion de nœud d'échec, appelée *nœud d'échec distant* et illustrée par la Figure 4.2, qui consiste à dévier la branche droite pour qu'elle n'atteigne pas de tels nœuds d'échec.
N'oublions pas également que la stratégie de superposition met en œuvre un ordre sur les littéraux favorisant les littéraux équationnels négatifs (Définition 2.8). Ainsi, la maximalité d'un littéral dans une clause devra être vérifiée car l'ordre n'est plus tout-à-fait celui des atomes dans l'arbre sémantique.

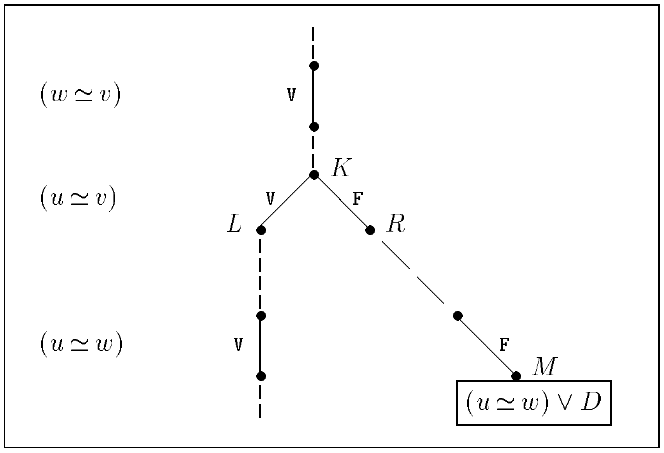


Figure 4.2: Nœud d'échec distant (R)

Définition 4.20 *Soit K un nœud de $MCT(\mathcal{GS})$, défini sur W_α, tel que A_α est un atome équationnel $(u \simeq v)$, avec $u \succ v$; K admet deux extensions L et R : $L(A_\alpha) = V$, $R(A_\alpha) = F$ et $R \in MCT(\mathcal{GS})$. Alors, R est un nœud d'échec distant au niveau d'un atome $(u \simeq w)$ si cette interprétation R est la racine d'un sous-arbre de $MCT(\mathcal{GS})$ dont la branche droite se termine par un nœud d'échec M, étiqueté par une clause $(u \simeq w) \vee D$ et tel que : $w \succ v$, $K(w \simeq v) = V$, $\forall v' < w$, $M(u \simeq v') = F$.*

Pour simplifier, nous dirons qu'un tel nœud R est *étiqueté par la clause* $(u \simeq w) \vee D$. Notons que les nœuds K, R et M sont E-consistants puisque M est E-consistant par définition d'un nœud d'échec.

Maintenant, définissons la *branche droite* de $MCT(\mathcal{GS})$, c'est-à-dire le chemin le plus à droite qui ne contient pas de nœuds de quasi-échec ou d'échec distant. Ensuite, nous prouverons qu'elle est E-consistante.

Définition 4.21 *La branche droite de $MCT(\mathcal{GS})$ est l'interprétation partielle Q_γ, définie sur W_γ, et construite comme suit : Q_0 est l'interprétation vide ; supposant que Q_i est définie pour tout $i < \alpha$, Q_α est défini (si possible) par :*

- *Si α est un ordinal limite, comme dans le cas classique [HR91], on définit simplement Q_α par $\bigcup_{i<\alpha} Q_i$. Alors, par le Lemme de Clôture, Q_α appartient à $MCT(\mathcal{GS})$.*
- *Si α n'est pas un ordinal limite, alors α admet un prédécesseur α^-. Plusieurs cas peuvent se produire :*
  - *Si Q_{α^-} a exactement un successeur J,*
    - *Si J appartient à $MCT(\mathcal{GS})$, alors Q_α est l'interprétation J,*
    - *Si J n'appartient pas à $MCT(\mathcal{GS})$, alors le dernier nœud Q_γ de la branche droite est Q_{α^-},*
  - *Si Q_{α^-} a exactement deux successeurs L et R, $L(A_{\alpha^-}) = V$ et $R(A_{\alpha^-}) = F$, alors*
    - *Si R est un nœud de quasi-échec, d'échec, ou d'échec distant,*
      - *Si L est un nœud d'échec, alors γ est égal à α^-,*
      - *Si L n'est pas un nœud d'échec, alors Q_α est l'interprétation L,*
    - *Si R n'est ni un nœud de quasi-échec, ni un nœud d'échec, ni un nœud d'échec distant, alors Q_α est R.*

Soit R un nœud d'échec distant comme décrit dans la Définition 4.20, étiqueté par une clause C. Détaillons quelques propriétés de cette clause C :

Propriétés des nœuds d'échec distants :

1. C ne peut contenir de littéral $\neg(u' \simeq w')$ où $u' =_E u$. En effet, cela signifie que l'atome $(u' \simeq w')$ est valide pour M, mais comme $(u \simeq w')$ doit être falsifié par M, l'interprétation M serait E-inconsistante. Cette propriété garantit donc que le littéral $(u \simeq w)$ est bien maximal dans la clause C pour l'ordre $\succ_L$.

2. La clause C ne contient pas de littéral positif $(u' \simeq w')$ où $u' =_E u$ et $K(w \simeq w') = V$. En effet, une étape de factorisation équationnelle sur la clause $C = (u \simeq w) \vee (u' \simeq w') \vee D$ produit la clause $(u' \simeq w') \vee \neg(w \simeq w') \vee D$ plus petite que C, qui devrait donc étiqueter un nœud d'échec au-dessus de M.

3. Si la clause C contient des littéraux positifs $(u' \simeq w')$, où $u' =_E u$, qui ne satisfont pas la propriété précédente, alors ces littéraux peuvent être remplacés dans C par les littéraux $(u \simeq w')$, car $(u \simeq w')$ est falsifié par définition.

La propriété suivante présente un cas dans lequel la maximalité d'un littéral dans une clause étiquetant un nœud d'échec est garantie.

Propriété des nœuds d'échec :

1. Soit Q_α une interprétation E-consistante, restriction de la branche droite à W_α, telle que A_α soit irréductible. Si une clause $L \vee D$ étiquette une extension de Q_α (L est A_α ou $\neg A_\alpha$), alors L est maximal dans $L \vee D$ pour l'ordre $\succ_L$.
   Cette propriété est évidente si A_α n'est pas un atome équationnel ou bien si $L = \neg A_\alpha$. Par contre, si $L = A_\alpha = (u \simeq w)$ $(u \succ w)$, D ne doit pas contenir de littéral $\neg(u' \simeq v)$, où $u' =_E u$. Cette condition est facile à vérifier, car un tel atome $(u' \simeq v)$ serait valide pour Q_α de même que $(u \simeq v)$ par E-consistance, et $(u \simeq w)$ devrait être réductible par $(u \simeq v)$. Or, A_α est irréductible.

La proposition suivante correspond à la Proposition 4.16 pour la stratégie de paramodulation. Elle exprime que si un atome A_α est réductible, il existe une équation dont le nœud droit est étiqueté par une clause de $\mathcal{GS}$ et telle que son équation maximale réduit un atome E-égal à A_α. Ces clauses de $\mathcal{GS}$ pourront servir pour les trois règles de superposition.

Proposition 4.22 *Soit Q_α la restriction à W_α de la branche droite Q_γ. Si l'atome A_α est réductible par une équation $(l \simeq r)$ où $l \succ r$, alors il existe un atome $A_{\alpha'}$, E-égal à A_α, qui est réductible par une équation $(g \simeq d)$ où $g \succ d$ et telle que le nœud droit de $(g \simeq d)$ est un nœud d'échec ou un nœud d'échec distant étiqueté par une clause C_g de $\mathcal{GS}$. De plus, $Q_\alpha(A_\alpha[r]) = Q_\alpha(A_{\alpha'}[d])$ et, si l'interprétation au niveau de l'atome $(g \simeq d)$ est E-consistante, la clause C_g ne contient pas de littéral $\neg(g' \simeq d')$ où g' est E-égal à g.*

Preuve : *La preuve de l'existence des atomes $A_{\alpha'}$ et $(g \simeq d)$ est faite exactement comme dans la preuve de la Proposition 4.16, avec en plus la possibilité pour le nœud droit de l'atome $(g \simeq d)$ d'être un nœud d'échec distant.*

La proposition demande également qu'il n'existe pas de littéral $\neg(g' \simeq d')$, où g' est E-égal à g, dans la clause étiquetant le nœud droit de $(g \simeq d)$. Cette propriété est importante, car elle permet de garantir que l'atome au niveau duquel se trouve le nœud d'échec (sous le nœud droit de $(g \simeq d)$) est bien maximal dans la clause C_g ; cet atome est $(g \simeq d)$ si le nœud droit de $(g \simeq d)$ est un nœud d'échec, ou bien $(g \simeq w)$ s'il s'agit d'un nœud d'échec distant.
Par définition, cette propriété est vraie si le nœud droit de $(g \simeq d)$ est un nœud d'échec distant.
Donc, supposons que le nœud droit de $(g \simeq d)$, noté R_g, est un nœud d'échec et qu'il existe un atome $(g' \simeq d')$ minimal, tel que $g' =_E g$ et le littéral $\neg(g' \simeq d')$ appartient à C_g. Notons L_g le nœud gauche de $(g \simeq d)$, restriction de la branche droite aux atomes inférieurs à $(g \simeq d)$, ce dernier atome compris : $L_g(g' \simeq d') = R_g(g' \simeq d') = V$.

- *si $L_g(d \simeq d') = V$, l'atome $(g' \simeq d)$ étant réductible en $(g' \simeq d')$, il est valide pour Q_α et R_g devrait donc être un nœud de quasi-échec.*
- *si $L_g(d \simeq d') = F$, l'atome $(g' \simeq d)$ est falsifié par Q_α, mais les interprétations L_g et Q_α sont alors E-inconsistantes.*

Ainsi, si l'on suppose que L_g est E-consistante, il n'existe pas de tel atome $(g' \simeq d')$. Le littéral $(g \simeq d)$ est donc maximal dans C_g pour l'ordre $\succ_L$. □

La prochaine proposition est l'équivalent de la Proposition 4.17 pour la stratégie de paramodulation. Elle précise que si une superposition ou une superposition contextuelle est appliquée entre deux clauses étiquetant des nœuds d'échec ou d'échec distants le long de la branche droite, alors la clause déduite est falsifiée par Q_γ, ce qui remet en cause l'appartenance de Q_γ à la branche droite.

Proposition 4.23 *Soit $C_1 = (l_1 \simeq r_1) \vee D_1$ et $C_2 = L_2 \vee D_2$ deux clauses de $\mathcal{GS}$ étiquetant des nœuds d'échec ou d'échec distant le long de la branche droite de $MCT(\mathcal{GS})$, et telles qu'il existe un littéral L, E-égal à L_2, réductible par $(l_1 \simeq r_1)$ où $l_1 \succ r_1$. Alors, si les conditions d'ordre pour appliquer une superposition (contextuelle) de C_1 dans C_2 sont vérifiées, Q_γ falsifie une clause de $\mathcal{GS}$.*

La preuve de cette proposition est similaire à celle de la stratégie de paramodulation. Les conditions d'ordre introduites par la stratégie de superposition ne posent pas de problème car elle sont supposées correctes. La raison en est simple : à chaque utilisation de cette proposition, nous montrerons la correction de ces conditions de maximalité de littéraux et de termes. Mais, nous pouvons déjà noter que si la clause C_1 étiquette un nœud d'échec distant ou bien un nœud d'échec, successeur droit d'un nœud de la branche droite, l'atome $(l_1 \simeq r_1)$ est maximal dans la clause C_1 pour l'ordre $\succ_L$ (propriétés des nœuds d'échec et d'échec distants citées précédemment).

4.4.2 Consistance de la branche droite

La proposition suivante énonce que la branche droite construite est E-consistante. Sa preuve différant quelque peu de celle de la Proposition 4.18, nous allons détailler à nouveau les principaux cas.

Proposition 4.24 *Si $MCT(\mathcal{GS})$ n'est pas vide, sa branche droite Q_γ n'est pas vide et est E-consistante.*

Les lemmes intermédiaires utilisés dans la preuve de cette proposition sont démontrés dans l'Appendice A.2.

Preuve : *La branche droite n'est pas vide car, par construction de l'arbre sémantique, A_0 est un atome $(a \simeq a)$ et $Q_1(A_0) = V$; donc, Q_1 ne peut être ni un nœud d'échec, ni un nœud d'échec distant, ni un nœud de quasi-échec.*

Pour montrer que l'interprétation Q_γ est E-consistante, nous raisonnons par contradiction : si Q_γ est E-inconsistant, il existe un ordinal minimal α tel que Q_α est E-inconsistant. Cet ordinal ne peut être un ordinal limite car, comme dans le cas de la théorie vide [HR91], si α est un ordinal limite, Q_α est défini par $\bigcup_{i<\alpha} Q_i$, et donc l'un des Q_i doit être E-inconsistant, ce qui est impossible puisque nous avons choisi α minimal.

α admet donc un prédécesseur α^-. Soit K l'interprétation partielle Q_{α^-}, et notons B l'atome A_{α^-}. Par minimalité de α, K est E-consistante et

$$\exists A_\beta =_E B \text{ tel que } \begin{array}{l} \bullet \text{ soit } A_\beta \in W_\alpha \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta) \\ \bullet \text{ soit } A_\beta \notin W_\alpha,\ A_\beta \longrightarrow_K^{l \simeq r} A_\beta[r] \text{ où } l \succ r, \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta[r]) \end{array}$$

Il a été prouvé dans le Lemme A.1 que B est une équation $(u_2 \simeq v)$ et A_β une équation $(u_1 \simeq v)$, où $u_1 =_E u_2$, $u_2 \succ v$ et v est irréductible. Dès lors, supposons que A_β est le plus *petit atome E-égal à B permettant de détecter la E-inconsistance de Q_α. Plusieurs cas sont à envisager, mais nous montrerons dans les lemmes qui vont suivre qu'ils sont tous impossibles :*

- *Si K a exactement un successeur, Q_α (Section A.2, Paragraphe I) : B est réductible, A_β est irréductible et $K(A_\beta) = F$. Cela signifie qu'un nœud de quasi-échec devrait couper la branche droite au niveau du nœud droit de l'atome A_β.*
- *Si K a exactement deux successeurs L et R, au moins l'un d'entre eux (Q_α) n'est ni un nœud d'échec, ni un nœud d'échec distant, ni un nœud de quasi-échec ; $L(B) = V$ and $R(B) = F$.*
  - *Si Q_α est L (Section A.2, Paragraphe II) : l'atome A_β, falsifié par K, est réductible par une équation $(l \simeq r)$ dont le nœud droit est un nœud d'échec ; R est un nœud de quasi-échec, c'est-à-dire qu'il existe un atome E-égal à B qui est réductible par une équation $(g \simeq d)$ dont le nœud droit est un nœud d'échec. A partir de cela, nous en déduisons que Q_α ne peut pas appartenir à $MCT(\mathcal{GS})$, car elle falsifie une clause de $\mathcal{GS}$ engendrée par une paramodulation étendue entre les clauses étiquetant les nœuds droits de $(g \simeq d)$ et $(l \simeq r)$.*
  - *Si Q_α est R (Section A.2, Paragraphe III) : l'atome A_β est valide pour K, mais Q_α devrait être un nœud de quasi-échec.*

Donc, dans tous les cas, nous trouvons une contradiction avec le fait que Q_α appartient à la branche droite de $MCT(\mathcal{GS})$. L'unique conclusion est que la branche droite Q_γ est E-consistante. □

4.4.3 Génération de la clause vide

La branche droite étant E-consistante, il nous reste à montrer qu'en fait elle est vide. Pour cela, nous étudions dans la preuve de la proposition suivante chacun des cas possibles pour le dernier nœud Q_γ de cette branche, et nous montrons qu'ils sont tous impossibles.

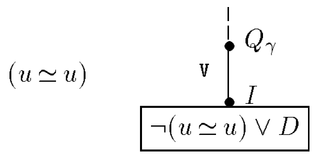
Proposition 4.25 *La branche droite est vide, et donc la clause vide appartient à $INF^*(S)$.*

Preuve : *Le dernier nœud Q_γ de la branche droite est donc défini sur W_γ, c'est-à-dire sur tous les atomes A_i pour $i < \gamma$; il appartient à $MCT(\mathcal{GS})$ et chacune de ses extensions est un nœud d'échec, d'échec distant ou de quasi-échec. Comme pour la stratégie de paramodulation, nous avons trois cas principaux à étudier :*

1. *Soit Q_γ a une seule extension I et A_γ est irréductible,*
2. *Soit Q_γ a deux extensions L et R,*
3. *Soit Q_γ a une seule extension I et A_γ est réductible.*

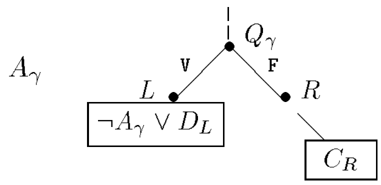
Détaillons ces différents cas :

Cas 1 : *Q_γ a une extension I et A_γ est irréductible.*



A_γ est une équation $(u \simeq u)$, où u est un terme clos irréductible. La clause C de $\mathcal{GS}$ étiquetant le nœud d'échec I est de la forme $\neg(u \simeq u) \vee D$. Ce cas soulève une contradiction, car Q_γ falsifie la clause D de $\mathcal{GS}$, produite par une étape de E-réflexion sur C.

Cas 2 : *Q_γ a deux extensions L et R (donc A_γ est irréductible).*



Le nœud Q_γ a deux extensions qui sont des nœuds d'échec, d'échec distants ou de quasi-échec. Comme l'extension gauche L satisfait l'atome A_γ, L est un nœud d'échec et falsifie une clause C_L de $\mathcal{GS}$. Cette clause est de la forme $\neg A_\gamma \vee D_L$, où chaque atome de D_L est plus petit que A_γ. Q_γ et L ne différant que par la valeur associée à A_γ, Q_γ falsifie également D_L.

Notons C'_L la clause $\neg A'_\gamma \vee D'_L$ de $INF^*(S)$ qui admet C_L comme instance close (modulo E), pour une substitution σ.

Quant à R, il s'agit soit d'un nœud d'échec, soit d'un nœud d'échec distant, soit d'un nœud de quasi-échec. Etudions ces trois cas.

Cas 2.1 : *Si R est un nœud d'échec.*

La clause C_R étiquetant R appartient alors soit à $\mathcal{GS}$, soit à $\mathcal{E}$.

Cas 2.1.1 : *Si $C_R \in \mathcal{GS}$.*

C_R est de la forme $A_\gamma \vee D_R$ et il existe une clause C'_R de $INF^*(S)$ telle que $C'_R\sigma =_E C_R$. Comme pour D_L, Q_γ falsifie D_R.

Donc, un étape de E-résolution peut être appliquée entre les clauses $C'_L\sigma$ et $C'_R\sigma$, et déduit une clause E-égale à $D_L \vee D_R$; cette clause appartient ainsi à $\mathcal{GS}$, et, comme elle est falsifiée par Q_γ, l'interprétation Q_γ ne peut pas être dans $MCT(\mathcal{GS})$.

Cas 2.1.2 : *Si $C_R \in \mathcal{E}$.*

Cela signifie que A_γ est une équation $(u \simeq v)$ où u et v sont E-égaux. Alors, une E-réflexion dans la clause $C'_L\sigma$ permet de déduire la clause $D'_L\sigma$, E-égale à D_L. D'où, D_L appartient à $\mathcal{GS}$, et nous en concluons que Q_γ ne peut être un nœud de la branche droite puisqu'elle falsifie D_L.

Cas 2.2 : *Si R est un nœud d'échec distant.*

Dans ce cas, A_γ est un atome $(u \simeq v)$ et le nœud d'échec M_R sous R se trouve au niveau d'un atome $(u \simeq w)$ et est étiqueté par une clause $C_R = (u \simeq w) \vee D_R$; de plus, $Q_\alpha(w \simeq v) = V$. Il existe donc une clause C'_R dans $INF^*(S)$ telle que $C'_R\sigma =_E C_R$. Si nous appliquons une E-superposition de la clause $C'_R\sigma$ dans la *clause $C'_L\sigma$, E-égale à $\neg(u \simeq v) \vee D_L$, la clause déduite est E-égale à $\neg(w \simeq v) \vee D_L \vee D_R$. Cette dernière clause appartient donc à $\mathcal{GS}$ et est falsifiée par M_R, ce qui contredit la construction de la branche droite car il devrait y avoir un nœud d'échec au-dessus de M_R.*

Cas 2.3 : *Si R est un nœud de quasi-échec.*

Par définition d'un nœud de quasi-échec, A_γ est un atome $(u_2 \simeq v)$, où $u_2 \succ v$, et R est étiqueté par une équation $(u_2 \simeq u_1)$ où u_2 et u_1 sont E-égaux. Différencions les deux possibilités pour ce nœud de quasi-échec :

- *si $(u_1 \simeq v)$ est irréductible, cet atome est validé par Q_γ. Or, $\neg(u_1 \simeq v) \vee D_L$ étant une clause E-égale à C_L, elle appartient aussi à $\mathcal{GS}$ et un nœud d'échec étiqueté par cette clause devrait couper la branche droite au niveau de $(u_1 \simeq v)$.*
- *si $(u_1 \simeq v)$ est réductible par une équation $(g \simeq d)$ (où $g \succ d$) en $(u_1[d] \simeq v)$ et $Q_\gamma(u_1[d] \simeq v) = V$, la Proposition 4.22 nous permet d'affirmer qu'il existe une équation $(l \simeq r)$, dont le nœud droit est un nœud d'échec ou d'échec distant, qui réduit un atome E-égal à $(u_1[g] \simeq v)$, c'est-à-dire à A_γ. Soit C_1 la clause étiquetant le nœud droit de cet atome $(l \simeq r)$. Par la Proposition 4.23, une inférence (E-superposition ou E-superposition contextuelle) de C_1 dans C_L engendre une clause falsifiée par Q_γ.*
  Les conditions de maximalité des littéraux pour effectuer cette déduction sont satisfaites car le remplacement s'effectue dans un littéral négatif.

Dans les deux cas, nous avons montré que Q_γ falsifiait une clause de $\mathcal{GS}$. Cette interprétation ne peut donc pas appartenir à la branche droite.

Le cas 2 est donc impossible.

Cas 3 : *Q_γ a une extension I et A_γ est réductible.*

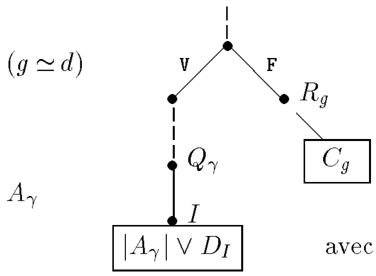
L'interprétation I ne peut être ni un nœud de quasi-échec ni un nœud d'échec distant car A_γ est réductible. Donc, I est un nœud d'échec étiqueté par une clause C_I. Cette clause ne peut pas appartenir à $\mathcal{E}$ car dans ce cas, A_γ serait une équation $(s \simeq s')$ où s et s' sont E-égaux ; alors, $(s \simeq s')$ serait réductible par une équation $(g \simeq d)$ en $(s[d] \simeq s')$; or ce dernier atome, falsifié par Q_γ, est E-égal à $(s[d] \simeq s)$, atome valide dans Q_γ car réductible en $(s[d] \simeq s[d])$; cela contredit la E-consistance de Q_γ.
Ainsi, C_I est une clause de $\mathcal{GS}$. Notons d'autre part que l'atome A_γ ne peut être réduit en un atome A'_γ E-égal, car une clause C'_I de $\mathcal{GS}$, E-égale à C_I, devrait étiqueter un nœud d'échec au niveau de cet atome A'_γ.

Donc, par la Proposition 4.22, il existe deux atomes $A_{\gamma'}$ et $(g \simeq d)$ tels que $A_{\gamma'}$ est E-égal à A_γ, réductible par $(g \simeq d)$, atome irréductible où $g \succ d$, et $I(A_\gamma) = I(A_{\gamma'}[d])$. Soit R_g le nœud droit de l'atome $(g \simeq d)$. Il s'agit soit d'un nœud d'échec, soit d'un nœud d'échec distant. Notons $C_g = (g \simeq w) \vee D_g$ la clause l'étiquetant (w est en fait le terme d si R_g est un nœud d'échec).

Détaillons maintenant les différents cas possible :

Cas 3.1 : *Si A_γ n'est pas équationnel.*

La clause C_I est de la forme $|A_\gamma| \vee D_I$, où $|A_\gamma|$ dénote le littéral A_γ si l'atome A_γ est falsifié par I, et le littéral $\neg A_\gamma$ sinon.

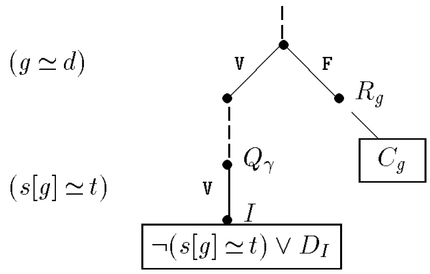


avec : $|A_\gamma| =$ si $I(A_\gamma) = F$ alors A_γ, sinon $\neg A_\gamma$

Le littéral $|A_\gamma|$ est bien maximal dans la clause C_I, au sens de la stratégie de superposition. Ainsi, la Proposition 4.23 nous dit qu'une E-superposition ou une E-superposition contextuelle de C_g dans C_I produit une clause falsifiée par Q_γ. Donc, cela contredit la construction effectuée de la branche droite.

Cas 3.2 : *Si $I(A_\gamma) = V$ et $A_\gamma = (s \simeq t)$.*

La clause C_I est de la forme $\neg(s \simeq t) \vee D_I$, et le littéral $\neg(s \simeq t)$ est bien maximal au sens de la stratégie de superposition. De plus, la réduction s'applique dans le terme s, si t est réductible en un terme t', l'atome $(s \simeq t')$ est valide pour Q_γ et réduit donc $(s \simeq t)$ dans s.



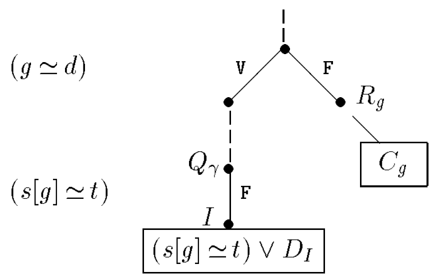
- *Si tout atome de D_g est plus petit que l'atome $(s \simeq t)$ alors, comme dans le cas précédent, la Proposition 4.23 nous permet d'appliquer une E-superposition ou une E-superposition contextuelle de C_g dans C_I pour déduire une clause falsifiée par Q_γ, ce qui contredit l'appartenance de Q_γ à la branche droite.*
- *S'il existe un atome plus grand que $(s \simeq t)$ dans D_g, cela signifie que R_g est un nœud d'échec distant, $s = g$, et cet atome supérieur à $(g \simeq t)$ est de la forme $(g \simeq w')$, falsifié par le nœud d'échec M_g au bout de la branche droite de R_g : $C_g = (g \simeq w) \vee (g \simeq w') \vee D'_g$.*

  Il faut donc montrer que l'on peut dériver une clause falsifiée par M_g, coupant la branche entre R_g et M_g.

  Une E-superposition de C_g dans C_I produit la clause $\neg(w \simeq t) \vee D_I \vee (g \simeq w') \vee D'_g$. Pour que cette clause soit falsifiée par M_g, il faut que tout littéral de D_I le soit. Mais, il peut exister un atome $(g \simeq u)$ tel que $D_I = \neg(g \simeq u) \vee D'_I$, $Q_\gamma(g \simeq u) = V$ et $M_g(g \simeq u) = F$. Il faut alors appliquer une E-superposition de C_g dans chacun de ces littéraux de D_I pour obtenir enfin la clause de $\mathcal{GS}$ recherchée, i.e. falsifiée par M_g : $\neg(w \simeq t) \vee \neg(w \simeq u) \vee D'_I \vee (g \simeq w') \vee D'_g$.

Cas 3.3 : *Si $I(A_\gamma) = F$, $A_\gamma = (s \simeq t)$ et s est réductible.*

La clause C_I est de la forme $(s \simeq t) \vee D_I$. Jusqu'à présent, lorsqu'un atome comme A_γ était réductible, nous avons utilisé une étape de E-superposition (contextuelle) pour trouver une contradiction. Mais dans le cas considéré, le problème *de la maximalité de l'atome A_γ dans C_I se pose. En effet, s'il existe un littéral $\neg(s' \simeq u)$ dans D_I tel que $s' =_E s$, nous ne pouvons pas appliquer de E-superposition dans A_γ.*



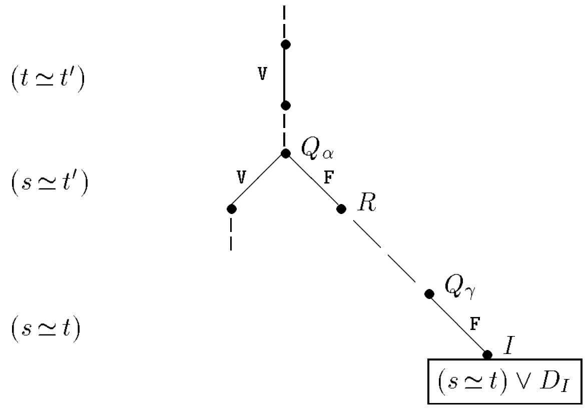
Supposons donc que les clauses C_I et C_g sont les plus petites clauses de $\mathcal{GS}$ pouvant étiqueter les nœuds I et R_g respectivement, pour l'ordre $\succ^{mult}$.

- *Supposons qu'il existe au moins un littéral de la forme $\neg(s \simeq u)$ dans D_I ; notons $D_I = \neg(s \simeq u) \vee D'_I$. Le littéral $(s \simeq t)$ n'est donc pas maximal dans C_I. Mais, si nous appliquons une E-superposition (contextuelle) de C_g dans chacun de ces littéraux, la clause C déduite, appartenant à $\mathcal{GS}$, est : $(s \simeq t) \vee \neg(s[d] \simeq u) \vee D'_I \vee D_g$.*
  Si $(s \simeq t) > (g \simeq w)$, la clause C est falsifiée par I et plus petite que C_I ; cela contredit le choix de C_I minimale.
  Si $(s \simeq t) \leq (g \simeq w)$, R_g est un nœud d'échec distant ; notons M_g le nœud d'échec au bout de la branche droite sous R_g ; la clause C est falsifiée par M_g et plus petite que C_g ; cela contredit également le choix de C_g minimale.

- *S'il n'existe pas de littéral $\neg(s' \simeq u)$ dans D_I tel que $s' =_E s$, le littéral $(s \simeq t)$ est maximal dans C_I. Une E-superposition (contextuelle) de C_g dans C_I produit la clause $C = (s[w] \simeq t) \vee D_I \vee D_g$ de $\mathcal{GS}$.*
  Si $(s \simeq t) > (g \simeq w)$, la clause C est falsifiée par Q_γ, ce qui contredit son appartenance à la branche droite.
  Si $(s \simeq t) \leq (g \simeq w)$, R_g est un nœud d'échec distant ; notons M_g le nœud d'échec au bout de la branche droite sous R_g ; la clause C est falsifiée par M_g et plus petite que C_g ; cela contredit le choix de C_g minimale.

Cas 3.4 : *Si $I(A_\gamma) = F$, $A_\gamma = (s \simeq t)$ et s est irréductible.*
Le nœud d'échec I est étiqueté par une clause $C_I = (s \simeq t) \vee D_I$. Il n'existe aucun littéral négatif $\neg(s' \simeq u)$ dans D_I tel que s' est E-égal à s, par E-consistance de Q_γ.
Soit t' le plus petit terme inférieur à t et tel que $Q_\gamma(t \simeq t') = V$ et $(s \simeq t')$ est irréductible. L'atome $(s \simeq t)$ se réduisant par $(t \simeq t')$ en $(s \simeq t')$, ce dernier atome $(s \simeq t')$ est falsifié par Q_γ. Soit α l'indice de $(s \simeq t')$; Q_α est la restriction de Q_γ à W_α. Soit R l'extension droite de Q_α : $R(s \simeq t') = F$.

La situation est décrite par la figure suivante :



Le nœud R ressemble énormément à un nœud d'échec distant. Pour conclure qu'il s'agit en effet d'un nœud d'échec distant, il nous suffit de vérifier la propriété suivante : $\forall u < t,\ I(s \simeq u) = F$. Or, aucun terme u ne peut contredire cela, car sinon $(s \simeq t)$ serait réductible par ce $(s \simeq u)$, ce qui est interdit par l'hypothèse d'irréductibilité de s.

Ainsi, chaque cas mène à une contradiction.

Comme tous les cas ont montré une incohérence avec la construction de la branche droite, nous en déduisons que cette branche droite est vide, et donc que le sous-arbre $MCT(\mathcal{GS})$ l'est également. Or, le seul moyen de ne laisser que l'interprétation vide dans $MCT(\mathcal{GS})$ est que la clause vide appartienne à $INF^*(S)$. Ceci conclut la preuve de complétude. □

4.5 Stratégie positive

Soit INF un ensemble de règles d'inférence défini pour la stratégie positive dans la Section 2.3. Rappelons que le but de cette stratégie est de ne faire une inférence que si elle utilise au moins une clause positive, c'est-à-dire une clause ne contenant que des littéraux positifs.

Détaillons la preuve du Théorème de Complétude de la stratégie positive (Théorème 2.14), qui consiste en fait à montrer quelques propriétés pour ensuite reprendre les preuves des théorèmes de complétude de la E-paramodulation (Théorème 2.7 prouvé dans la Section 4.3) et de la E-superposition (Théorème 2.12 prouvé dans la Section 4.4) pour la stratégie ordonnée.

Preuve : *La preuve de ce théorème se fait par contradiction. Supposons que S est un ensemble E-incohérent de clauses, mais que $INF^*(S)$ ne contient pas la clause vide. Alors, l'arbre sémantique cohérent $MCT(INF^*(S))$ n'est pas vide. Comme pour la stratégie ordonnée, définissons les ensembles suivants :*

$$\begin{aligned} \mathcal{GS} &= \{\ C \mid \exists C' \in INF^*(S),\ C =_E C'\sigma,\ \text{pour une instance close } C'\sigma \text{ de } C'\ \} \\ \mathcal{E} &= \{\ (u \simeq u') \mid u, u' \in \mathcal{T}(\mathcal{F}),\ u =_E u'\ \} \end{aligned}$$

$\mathcal{GS}$ contient toutes les instances closes de clauses de $INF^*(S)$, et $\mathcal{E}$ contient toutes les équations triviales pour la théorie E. L'arbre sémantique cohérent $MCT(\mathcal{GS})$ est équivalent à $MCT(INF^*(S))$; il est donc non vide.

Nous construisons la branche droite Q_γ de $MCT(\mathcal{GS})$ comme pour la stratégie ordonnée, c'est-à-dire selon la Définition 4.15 pour la stratégie de paramodulation, ou bien la Définition 4.21 pour la stratégie de superposition.

Rappelons que si un nœud Q_α de cette branche droite admet une extension droite R n'appartenant pas à la branche droite, alors l'interprétation R est soit un nœud de quasi-échec, soit un nœud d'échec, soit un nœud d'échec distant. Supposons que les nœuds R ainsi définis et étiquetés par une clause de $\mathcal{GS}$, le sont par une clause positive.

Grâce à cette hypothèse de positivité des clauses étiquetant les nœuds droits, les preuves de complétude de la stratégie ordonnée pour la paramodulation (Théorème 2.7) et la superposition (Théorème 2.12) peuvent être reprises mot pour mot afin de dériver une contradiction au fait que $MCT(\mathcal{GS})$ n'est pas vide. D'autre part, toutes les conditions de maximalité des littéraux utilisés, apparaissant dans les règles d'inférence, découlent très simplement de l'hypothèse précédente. Rappelons ces conditions :

- *Si une inférence fait appel à une clause positive, alors elle doit utiliser le littéral maximal de cette clause,*
- *Si une inférence utilise un littéral négatif d'une clause, alors ce littéral doit être maximal vis-à-vis des autres littéraux négatifs de cette clause.*

Outre ces conditions de maximalité, l'obligation d'effectuer des remplacements uniquement depuis une clause positive, et d'appliquer des paramodulations étendues uniquement entre clauses positives, vient du fait que ces déductions utilisent toujours des clauses étiquetant des nœuds droits. □

Il nous faut donc montrer le lemme suivant :

Lemme 4.26 *Les nœuds d'échec et d'échec distants, extensions droites de nœuds de la branche droite, sont étiquetés par des clauses positives de $\mathcal{GS}$.*

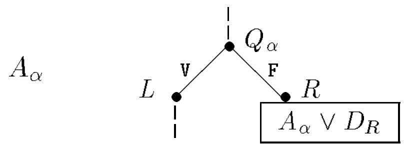
Preuve : *La preuve de ce lemme est réalisée par récurrence sur ces nœuds d'échec et d'échec distants R.*

Etape initiale : *Soit Q_α la plus petite restriction de la branche droite Q_γ qui admet deux extensions L et R, telles que $L(A_\alpha) = V$, $R(A_\alpha) = F$ et R est un nœud d'échec ou d'échec distant, étiqueté par une clause C_R de $\mathcal{GS}$. Supposons que C_R est la plus petite clause de $\mathcal{GS}$ falsifiée par R pour l'ordre $>^{mult}$, extension aux multi-ensembles de littéraux de $>$.*

Notons dans un premier temps que l'interprétation R ne peut pas être un nœud d'échec distant, car l'atome A_α serait une équation $(u \simeq v)$ et la clause C_R serait de la forme $(u \simeq w) \vee D_R$; l'atome $(w \simeq v)$, valide pour Q_α, réduisant $(u \simeq w)$ en $(u \simeq v)$, la Proposition 4.22 affirme qu'il existe alors un atome irréductible, dont le nœud droit est étiqueté par une clause de $\mathcal{GS}$, réduisant un atome E-égal à $(u \simeq w)$. Or, ceci est impossible car nous avons supposé que R était la plus petite interprétation étiquetée par une clause de $\mathcal{GS}$.

La clause C_R est donc de la forme $A_\alpha \vee D_R$, où tout atome de D_R est plus petit que A_α. Nous pouvons également supposer qu'aucun atome de D_R n'est E-égal à A_α, car

une étape de factorisation permet d'éliminer ces occurrences multiples de A_α.
Soit $C'_R = A'_\alpha \vee D'_R$ *la clause de* $INF^*(S)$ *telle que* $C_R =_E C'_R\sigma$ *pour une substitution close* σ.



Nous allons montrer dans un premier temps que tout atome de D_R *est irréductible, puis que tout littéral de* D_R *est positif.*

- *S'il existe un littéral réductible* L_R *de* D_R $(D_R = L_R \vee D_{2R})$*, alors par le Théorème de la Réduction, il est réductible par une équation irréductible* $(l \simeq r)$.
  l *ne peut être* E*-égal à* r*, car la clause* $A_\alpha \vee L_R[r] \vee D_{2R}$ *appartiendrait également à* $\mathcal{GS}$*, et comme elle est falsifiée par* R *et plus petite que* C_R*, cela contredit la minimalité de* C_R.
  Selon les Propositions 4.16 et 4.22, il existe un atome E*-égal à* L_R *qui est réductible par une équation irréductible, dont le nœud droit est étiqueté par une clause de* $\mathcal{GS}$*. Une telle équation ne peut exister, car* R *est le plus petit nœud falsifiant une clause de* $\mathcal{GS}$ *le long de la branche droite.*
  Donc, le littéral L_R *ne peut être réductible, et la clause* C_R *est ainsi irréductible.*
- *S'il existe un littéral* $\neg A_\beta$ *dans* D_R $(D_R = \neg A_\beta \vee D_{2R})$*, l'atome* A_β *est donc valide pour* Q_α *et nous venons de montrer qu'il est irréductible. Ainsi, la restriction* Q_β *de* Q_α *à* W_β *admet soit une seule extension et* A_β *est une équation de la forme* $(u \simeq u)$*, soit deux extensions et le nœud droit de* A_β *est un nœud d'échec étiqueté par une équation de* $\mathcal{E}$*, ce qui signifie que* A_β *est une équation* $(u \simeq v)$ *où* $u =_E v$*, ou bien le nœud droit de* A_β *est un nœud de quasi-échec.*
  - *Si* A_β *est une équation* $(u \simeq v)$ *où* $u =_E v$*, et éventuellement* $u = v$*, alors une étape de* E*-réflexion sur la clause* $C'_R\sigma = A'_\alpha\sigma \vee \neg A'_\beta\sigma \vee D'_{2R}\sigma$ *produit la clause* $A'_\alpha\sigma \vee D'_{2R}\sigma$*. La clause* $A_\alpha \vee D_{2R}$ *appartient donc à* $\mathcal{GS}$*, est falsifiée par* R *et est plus petite que* C_R*. Cela contredit le choix de* C_R *comme la clause minimale de* $\mathcal{GS}$ *falsifiée par* R.
  - *Le nœud droit de* A_β *ne peut être un nœud de quasi-échec, car dans ce cas les Propositions 4.16 et 4.22 nous disent qu'il existe une équation irréductible dont le nœud droit est étiqueté par une clause de* $\mathcal{GS}$*, qui réduit un atome* E*-égal à* A_β*. Or,* R *est le plus petit nœud étiqueté par une clause de* $\mathcal{GS}$.

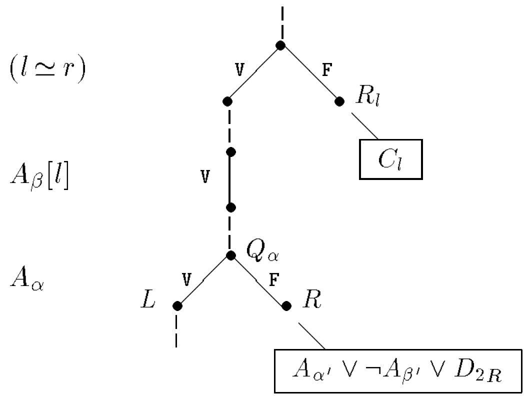
Ainsi, R *est étiqueté par une clause positive de* $\mathcal{GS}$.

Etape de récurrence : *Soit* Q_α *un nœud de la branche droite admettant deux extensions* L *et* R *telles que* $L(A_\alpha) = V$, $R(A_\alpha) = F$ *et* R *est un nœud d'échec ou d'échec distant, étiqueté par une clause* C_R *de* $\mathcal{GS}$*. Supposons que* C_R *est la plus petite clause de* $\mathcal{GS}$ *falsifiée par* R *pour l'ordre* $>^{mult}$*, extension aux multi-ensembles de littéraux de* $>$.
Supposons également par hypothèse de récurrence *que pour tout nœud de la branche droite, plus petit que* Q_α *et admettant une extension droite étiquetée par une clause de* $\mathcal{GS}$*, cette clause est positive.*

La clause C_R est de la forme $A_{\alpha'} \vee D_R$, où $A_{\alpha'}$ représente soit l'atome A_α si R est un nœud d'échec, soit l'atome sous R au niveau duquel se trouve le nœud d'échec si R est un nœud d'échec distant.

Nous allons montrer que tout littéral de D_R est positif en deux étapes : d'abord, s'il existe un littéral négatif dans D_R, il est irréductible ; ensuite, il existe une clause dans $\mathcal{GS}$ ne contenant pas de littéral négatif.

- *S'il existe un littéral négatif réductible $\neg A_{\beta'}$ dans D_R, il ne peut être réduit par une équation $(l \simeq r)$ où les termes l et r sont E-égaux, par minimalité de C_R. Donc, grâce aux Propositions 4.16 et 4.22, il existe une équation irréductible $(l \simeq r)$ réduisant un atome A_β, E-égal à $A_{\beta'}$, dont le nœud droit R_l est étiqueté par une clause C_l de $\mathcal{GS}$. Par hypothèse de récurrence, C_l est une clause positive. Notons $A_{\alpha'} \vee \neg A_{\beta'} \vee D_{2R}$ la clause C_R, et supposons que le littéral $\neg A_{\beta'}$ a été choisi maximal parmi les littéraux négatifs de la clause C_R.*



Soient $C'_l = (l' \simeq r') \vee D'_l$ et $C'_R = A'_{\alpha'} \vee \neg A'_{\beta'} \vee D'_{2R}$ deux clauses de $INF^*(S)$ admettant respectivement C_l et C_R pour instance close pour une substitution σ : $C'_l\sigma =_E C_l$ et $C'_R\sigma =_E C_R$. Etudions les différents cas possibles :

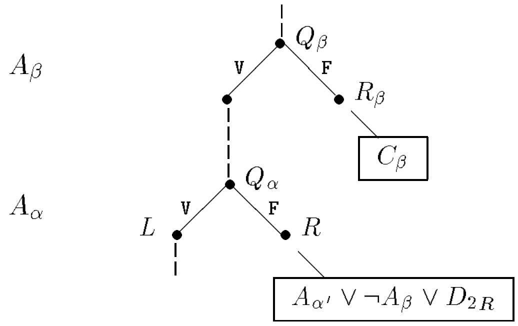
  - *Si A_β est une équation $(u \simeq v)$ où $u =_E v$, alors une E-réflexion appliquée sur la clause $C'_R\sigma$ produit une clause E-égale à $A_{\alpha'} \vee D_{2R}$; cette dernière clause appartient donc à $\mathcal{GS}$ et est plus petite que C_R, ce qui contredit la minimalité de C_R.*
  - *Si A_β n'est pas une équation triviale, une E-paramodulation ou une E-paramodulation contextuelle peut être appliquée de $C'_l\sigma$ dans $C'_R\sigma$ pour produire une clause E-égale à $A_{\alpha'} \vee \neg A_\beta[r'\sigma] \vee D_{2R} \vee D_l$, appartenant à $\mathcal{GS}$ et falsifiée par R. Cette clause est plus petite que C_R pour l'ordre $>^{mult}$, car l'atome $A_{\beta'}$ a été remplacé par un ensemble d'atomes qui lui sont tous strictement inférieurs. Cela contredit le choix de C_R comme la plus petite clause falsifiée par R.*

    Pour la stratégie de superposition, si A_β est un atome équationnel, il faut vérifier en plus que le remplacement ne s'effectue par dans le plus petit membre de cette équation. L'atome A_β étant valide pour Q_α, il est toujours possible de choisir $(l \simeq r)$ réduisant le plus grand membre de son équation.

Les deux cas que nous venons d'étudier sont donc impossibles car la clause C_R a été choisie minimale dans le sous-ensemble de $\mathcal{GS}$ des clauses falsifiées par R.

Ainsi, s'il existe un littéral négatif dans C_R, celui-ci est irréductible.

- *Supposons maintenant qu'il existe un littéral négatif irréductible $\neg A_\beta$ dans D_R : $D_R = \neg A_\beta \vee D_{2R}$. Soit $C'_R = A'_{\alpha'} \vee \neg A'_\beta \vee D'_{2R}$ la clause de $INF^*(S)$ telle que $C'_R\sigma =_E C_R$ pour une substitution close σ. Supposons que le littéral $\neg A_\beta$ a été choisi maximal parmi les littéraux négatifs de la clause C_R.*



Soit Q_β la restriction de Q_α à W_β. L'atome A_β étant irréductible, soit Q_β n'admet qu'une seule extension et alors A_β est une équation $(u \simeq u)$, soit Q_β admet deux extensions ; dans ce second cas, son extension droite R_β est soit un nœud d'échec ou d'échec distant étiqueté par une clause $C_\beta = A_\beta \vee D_\beta$, soit un nœud de quasi-échec. Par hypothèse de récurrence, nous pouvons supposer que la clause C_β est positive. Etudions ces différents cas :

Cas 1 : *Si A_β est une équation $(u \simeq v)$ où $u =_E v$.*
Alors, une étape de E-réflexion peut être appliquée dans la clause $C'_R\sigma$ pour engendrer une clause E-égale à $A_{\alpha'} \vee D_{2R}$; cette dernière clause appartient donc à $\mathcal{GS}$. Mais, comme elle est falsifiée par R et plus petite que C_R, ce cas ne peut se produire puisque C_R a été choisie minimale.

Cas 2 : *Si R_β est un nœud d'échec.*
Soit $C'_\beta = A'_\beta \vee D'_\beta$ une clause de $INF^*(S)$ telle que $C'_\beta\sigma =_E C_\beta$. La clause $C'_\beta\sigma$ étant positive par hypothèse de récurrence, une E-résolution peut être appliquée entre les clauses $C'_\beta\sigma$ et $C'_R\sigma$ pour engendrer la clause $A'_{\alpha'}\sigma \vee D'_{2R}\sigma \vee D'_\beta\sigma$, E-égale à $A_{\alpha'} \vee D_{2R} \vee D_\beta$ qui appartient donc à $\mathcal{GS}$. Cette dernière clause étant plus petite que C_R et falsifiée par R, cela contredit le choix de C_R minimale.

Cas 3 : *Si R_β est un nœud d'échec distant.*
L'atome A_β est une équation $(u \simeq v)$, et la clause C_β étiquette un nœud d'échec sous R_β au niveau d'un atome $(u \simeq w)$ tel que $Q_\beta(w \simeq v) = V$.
Soit $C'_\beta = (u' \simeq w') \vee D'_\beta$ une clause de $INF^*(S)$ telle que $C'_\beta\sigma =_E C_\beta$. Une E-paramodulation (ou E-superposition) appliquée de $C'_\beta\sigma$ dans $C'_R\sigma$ produit une clause E-égale à $A_{\alpha'} \vee \neg(w \simeq v) \vee D_{2R} \vee D_\beta$. Cette clause de $\mathcal{GS}$ est falsifiée par R et plus petite que C_R.

Cas 4 : *Si R_β est un nœud de quasi-échec.*
Par définition d'un nœud de quasi-échec, cela signifie qu'il existe un atome E-égal à A_β qui est réductible par une équation $(l \simeq r)$ où $l \succ r$. Mais, nous retrouvons le cas déjà résolu d'un littéral négatif réductible de D_R. Cette situation est donc également impossible.

Comme dans tous les cas nous avons trouvé une contradiction, nous en concluons que tout nœud d'échec ou d'échec distant R, extension droite d'un nœud de la branche droite, est étiqueté par une clause positive de $\mathcal{GS}$. Ceci conclut la preuve du lemme. □

4.6 Règles de simplification

Soit INF un ensemble de règles d'inférence défini dans l'une des Sections 2.2.1, 2.2.2 et 2.3, c'est-à-dire pour la E-paramodulation ou la E-superposition, et pour la stratégie ordonnée ou positive. Soit INF_S le système composé de INF et des règles de simplification définies dans la Section 2.4.

Nous allons montrer dans cette section que les règles de simplification ajoutées aux règles d'inférence préservent la complétude réfutationnelle (Théorème 2.21).

La méthode de preuve de complétude utilisée dans les sections 4.3 et 4.4 ne peut pas être appliquée ici, car, si deux clauses permettent d'en déduire une troisième, rien n'assure que ces deux clauses apparaîtront un jour dans le même S_i ; donc, l'inférence peut ne jamais être considérée. Nous allons montrer qu'heureusement cela ne gêne en rien la déduction de la clause vide.

A partir de maintenant, nous supposons que S_0 est un ensemble de clauses E-incohérent, et que $S_0, S_1, S_2, \ldots$ est une dérivation équitable. S^* dénote $\bigcup_{i \geq 0} S_i$.

Comme pour prouver la complétude des stratégies de paramodulation et de superposition sans règles de simplification, nous définissons les ensembles de clauses suivants :

$$\begin{array}{rcl} \mathcal{GS} & = & \{\ C \mid \exists C' \in S^*,\ C =_E C'\sigma, \text{ pour une instance close } C'\sigma \text{ de } C'\ \} \\ \mathcal{E} & = & \{\ (u \simeq u') \mid u, u' \in \mathcal{T}(\mathcal{F}),\ u =_E u'\ \} \end{array}$$

La construction de la branche droite Q_γ de l'arbre sémantique cohérent $MCT(\mathcal{GS})$ ne diffère pas : Définition 4.15 pour la stratégie de paramodulation, Définition 4.21 pour la stratégie de superposition.

Définition 4.27 *Une clause C est persistante dans une dérivation $S_0, S_1, S_2, \ldots$ s'il existe un indice k tel que C appartient à $\bigcap_{i \geq k} S_i$.*

La preuve de complétude en présence de règles de simplification se fait habituellement en montrant que tout nœud d'échec le long de la branche droite de $MCT(\mathcal{GS})$ est étiqueté par une clause persistante de $\mathcal{GS}$. Ceci n'est possible que s'il n'existe pas de suite infinie décroissante de clauses pour l'ordre de subsomption. Cette propriété, montrée par D. Loveland [Lov78] pour l'ordre de subsomption stricte dans la théorie vide, n'est pas valable pour l'ordre de subsomption. Et même pour l'ordre de subsomption stricte, cette propriété ne peut être étendue qu'aux théories équationnelles E dont les classes de congruence sont finies.

Lemme 4.28 *Soit E une théorie équationnelle ayant des classes de congruence finies. Il n'existe pas de séquence infinie $C_0, C_1, C_2, \ldots$ telle que la clause C_{i+1} E-subsume strictement la clause C_i, pour tout i.*

La preuve de ce lemme est très simple. Elle se fait par récurrence sur la taille des clauses. Donnons maintenant un exemple de théorie équationnelle ne respectant par le lemme précédent.

Exemple 4.2 Soit E la théorie composée de l'axiome $(f(x,x) \simeq x)$. Les classes de congruence définies par E sont infinies. Par exemple, pour une constante a,

$$a =_E f(a,a) =_E f(f(a,a),a) =_E f(a,f(a,a)) =_E f(f(f(a,a),a),a) =_E \dots$$

Notons $\gtrdot_E$ l'ordre de E-subsomption stricte : $A \gtrdot_E B$ si B subsume strictement A. Alors, la séquence suivante est infinie

$$P(a) \gtrdot_E P(f(x_1,a)) \gtrdot_E P(f(x_2,f(x_1,a))) \gtrdot_E P(f(x_3,f(x_2,f(x_1,a)))) \gtrdot_E \dots$$

car, si on instancie la variable x_1 de l'un de ces atomes par a, l'atome obtenu est E-égal à l'atome précédent dans la séquence, à un renommage de variables près. ♦

Notons $\gtrdot_E$ l'ordre de E-subsomption. Cet ordre n'est pas nœthérien, et nous ne pouvons donc pas assurer que si une clause C_1 de $\mathcal{GS}$ est subsumée par une clause C_2, alors il existe une clause C_3 de $\mathcal{GS}$ subsumant C_1 mais non subsumée. Nous allons orienter notre raisonnement sur la persistance des nœuds d'échec au lieu des clauses. Mais auparavant, définissons quelques notations :

$GR(S)$: ensemble des clauses E-égales à une instance close d'une clause de S, i.e. $\{ C \mid \exists C' \in S \ , \ C =_E C'\sigma \text{ , pour une substitution close } \sigma \}$

et pour un nœud d'échec K :

Σ_K : ensemble des clauses de S^* falsifiées par K, i.e. $\{ C' \in S^* \mid \exists C \in GR(\{C'\}) \ , \ K(C) = F \}$

TG_K : ensemble des clauses minimales de $GR(\Sigma_K)$ pour l'ordre $\succ$ (ou $\succ_L$)

Π_K : sous-ensemble des clauses de Σ_K ayant une instance (modulo E) dans TG_K

Notons que : $TG_K \subseteq GR(\Pi_K)$.

Définition 4.29 *Soit K un nœud d'échec de $MCT(\mathcal{GS})$ étiqueté par une clause de $\mathcal{GS}$. K est un nœud d'échec persistant s'il existe un indice k tel que : $\forall i \geq k,\ \exists C \in GR(S_i),\ K(C) = F$.*

Dans la proposition suivante, nous montrons que tout nœud d'échec K, étiqueté par une clause de Π_K le long de la branche droite Q_γ de $MCT(\mathcal{GS})$, est un nœud d'échec persistant. Dans la preuve, nous ne parlerons pas explicitement des nœuds d'échec distants, car ils servent à annoncer un nœud d'échec plus bas dans l'arbre, et c'est ce nœud-là que nous considérerons en parlant d'un nœud d'échec le long de la branche droite.

Proposition 4.30 *Soit K un nœud d'échec, extension d'un nœud Q_α de la branche droite, ou plus précisément extension droite de Q_α ou extension de Q_γ. Si K est étiqueté par une clause de $\mathcal{GS}$, alors K est un nœud d'échec persistant.*

Preuve : *Soit Q_α un nœud de la branche droite tel que*

- *soit il admet deux extensions L et R, R est un nœud d'échec ou d'échec distant étiqueté par une clause de $\mathcal{GS}$ $(R(A_\alpha) = F)$ et $K = R$,*

- *soit il s'agit du dernier nœud (Q_γ) de la branche droite et K est une extension de Q_α, nœud d'échec étiqueté par une clause de $\mathcal{GS}$.*

Par hypothèse de récurrence, supposons que tous les nœuds d'échec le long de la branche droite sont des nœuds d'échec persistants, ceci jusqu'à K exclus. Pour prouver que K est également un nœud d'échec persistant, nous allons montrer que la clause $C_K \in TG_K$ l'étiquetant ne peut être simplifiée, et que si elle est subsumée, il existe toujours une autre clause de $\mathcal{GS}$ falsifiée par K. Soit C'_K une clause de Π_K telle que $C_K =_E C'_K\sigma$, pour une substitution close σ.

Si C'_K est simplifiée : *Il existe un indice j pour lequel C_K appartient à S_j mais pas à S_{j+1}. Soit $(l'_1 \simeq r'_1) \vee D'_1$ la clause de S_j simplifiant C'_K ; par définition des règles de simplification et de simplification contextuelle, la réduction est appliquée à une position non-variable p d'un littéral L'_K, à l'aide d'une substitution ρ ; la clause C'_K peut être décomposée en $L'_K \vee D'_{1K} \vee D'_{2K}$, et $D'_1\rho$ est inclus dans D'_{1K} modulo E, i.e. $D'_1\rho \subseteq_E D'_{1K}$. La clause C_K peut être décomposée de la même manière en $L_K \vee D_{1K} \vee D_{2K}$, où $L_K =_E L'_K\sigma$, $D_{1K} =_E D'_{1K}\sigma$ et $D'_1\rho\sigma \subseteq_E D_{1K} =_E D'_{1K}\sigma$.*

D'autre part, il existe un atome A' dans $L'_K \vee D'_{2K}$ qui est plus grand que $(l'_1\rho \simeq r'_1\rho)$. Par stabilité de l'ordre sur les atomes, $A'\sigma$ est plus grand que $(l'_1\rho\sigma \simeq r'_1\rho\sigma)$. La clause $(l'_1\rho\sigma \simeq r'_1\rho\sigma) \vee D_{1K}$ est ainsi plus petite que C_K. Or, cette clause C_K appartenant à TG_K, K ne peut pas falsifier une clause plus petite, et comme K falsifie déjà D_{1K}, K doit satisfaire l'atome $(l'_1\rho\sigma \simeq r'_1\rho\sigma)$.

Un autre point important est qu'il existe un littéral L''_K, E-égal à L_K, réductible par $(l'_1\rho\sigma \simeq r'_1\rho\sigma)$, et tel que $L''_K[r'_1\rho\sigma]$ est falsifié par K. Ceci découle de la réductibilité (contextuelle) de L'_K et de la E-consistance de K (par définition d'un nœud d'échec).

Une instance close de la clause déduite par l'étape de simplification (contextuelle) de $(l'_1 \simeq r'_1) \vee D'_1$ dans C'_K est $L''_K[r'_1\rho\sigma] \vee D_{1K} \vee D_{2K}$. Cette clause appartient à $GR(S_{j+1})$ et donc à $GR(S^*)$; elle est falsifiée par K et plus petite que C_K pour l'ordre $\succ$, car $l'_1\rho\sigma \succ r'_1\rho\sigma$. Ceci contredit l'appartenance de C_K à TG_K.

Si C'_K est subsumée : *Il existe un indice j pour lequel C'_K appartient à S_j mais pas à S_{j+1}. Soit D'_1 la clause de S_j subsumant C'_K. Ainsi, il existe une substitution ρ telle que $D'_1\rho \subseteq_E C'_K$. La clause $D'_1\rho\sigma$ n'est ni plus petite que $C'_K\sigma$ pour l'ordre $\succ$, car C'_K appartient à Π_K, ni plus grande que $C'_K\sigma$ par définition de la subsomption. Les clauses $D'_1\rho\sigma$ et $C'_K\sigma$ sont donc E-égales.*

La clause D'_1 appartient également à Π_K, car $K(D'_1\rho\sigma) = F$. Donc, K falsifie quand même une clause de $GR(S_{j+1})$.

En conclusion, la clause C'_K de Π_K ne peut pas être simplifiée et, si elle est subsumée à un indice j, il existe une clause dans chaque S_i $(i > j)$ falsifiée par K. Donc, le nœud d'échec K est persistant. □

Dans la preuve de cette proposition, nous n'avons pas fait que montrer que K était un nœud d'échec persistant, mais nous avons montré que la clause C_K l'étiquetant est persistante, dans le sens où elle étiquettera toujours K même si elle ne sera pas toujours considérée comme l'instance d'une même clause.

Les preuves de compatibilité des autres règles de simplification mentionnées dans la Section 2.4.3 sont très faciles :

Elimination des tautologies : une clause contenant une équation E-égale à $(t \simeq t)$, ou contenant un littéral L et un littéral E-égal à $\neg L$ ne peut étiqueter un nœud d'échec par définition.

Simplifications clausales : si une clause $\neg L \vee D$ étiquette un nœud d'échec, et s'il existe une clause réduite à L dans $\mathcal{GS}$, alors la clause D étiquette soit ce même nœud d'échec, soit un nœud d'échec plus haut dans la branche.

Réflexion triviale : une clause $\neg(t \simeq t') \vee D$, où t et t' sont E-égaux, ne peut pas étiqueter un nœud d'échec par définition.

Simplification par remplacement : si une clause $\neg(l \simeq r) \vee D$ étiquette un nœud d'échec I et $l \succ r$, cela signifie que l'atome $(l \simeq r)$ est valide pour I. Donc, si D contient un sous-terme E-égal à l (notons-le $D[l]$), D étant falsifié par I, $D[r]$ est également falsifié par I. La clause $\neg(l \simeq r) \vee D[r]$ étiquette donc soit I, soit une restriction de I.

Proposition 4.31 *La branche droite Q_γ de l'arbre sémantique cohérent $MCT(\mathcal{GS})$ est E-consistante.*

Preuve : *Les seuls points qui peuvent poser un problème dans les preuves pour les stratégies de paramodulation (Proposition 4.18) et de superposition (Proposition 4.24) sont les étapes d'inférence entre des nœuds d'échec ou d'échec distants, extensions droites de nœuds de la branche droite.*

Or, nous venons de montrer dans la proposition précédente que tout nœud d'échec étiqueté par une clause de $\mathcal{GS}$ le long de la branche droite est persistant, sans jamais avoir à supposer que cette branche est E-consistante. Nous avons même montré qu'une clause étiquetant un tel nœud d'échec est persistante. Donc, toutes les inférences réalisées dans la preuve de la E-consistance de Q_γ restent applicables. $\square$

Nous n'allons pas reprendre les preuves des Propositions 4.19 et 4.25 pour montrer que la branche droite de $MCT(\mathcal{GS})$ est vide, et donc que la clause vide a été engendrée dans la dérivation $S_0, S_1, S_2, \ldots$, car toutes les inférences effectuées dans ces preuves utilisent donc des clauses étiquetant des nœuds persistants. Ces inférences seront toujours applicables, et le nœud d'échec qu'elles provoquent dans la branche droite est persistant.

5
Déduction dans les théories associatives et commutatives

Les théories associatives et commutatives représentent un cas bien particulier de théories équationnelles ; elles ont été très largement étudiées en raison de leurs fréquentes applications. D'autre part, les récentes définitions d'ordres de simplification totaux sur les classes de AC-congruence ont accentué l'intérêt de ces théories (cf. Section 1.2).

Le remplacement de ces axiomes d'associativité et de commutativité par des mécanismes, tels qu'un algorithme d'unification et des règles d'inférence, a été initialement proposé par G.E. Peterson et M.E. Stickel [PS81, Sti84] pour la réécriture, grâce à une extension de la procédure de complétion de Knuth-Bendix. Nous présentons dans ce chapitre l'adaptation des stratégies de paramodulation et de superposition du Chapitre 2 aux théories AC, travaux parus dans [RV91, Vig94b, RV95].

Un système de règles d'inférence pour les théories AC a été définis, pour la stratégie de paramodulation, par E. Paul [Pau92] ; sa preuve de complétude est basée sur les arbres sémantiques transfinis de M. Rusinowitch [Rus89] et J. Hsiang et M. Rusinowitch [HR91], mais diffère de la nôtre par la construction de l'arbre sémantique pour les classes de congruence AC. Les systèmes définis pour la stratégie de superposition par U. Wertz [Wer92], L. Bachmair et H. Ganzinger [BG93] sont démontrés réfutationnellement complets grâce à la technique de construction de modèles de L. Bachmair et H. Ganzinger [BG90] ; cependant, les extensions de clauses y sont explicitement créées, ce qui nécessite la mise au point d'un contrôle spécial pour les gérer. Notons que très récemment, L. Bachmair et H. Ganzinger ont modifié leur technique de preuve pour utiliser une autre approche très intéressante, fondée sur la construction d'un modèle équationnel à partir de modèles non-équationnels (ayant une relation transitive) [BG93].

Le principal avantage des théories AC est qu'elles admettent des « extensions » très simples. Nous avons vu dans le Chapitre 3 (Exemple 3.5) qu'il n'existe en effet qu'un seul contexte utile par opérateur AC f :

$$\bigcup_{k\geq 0} \mathcal{C}ont_k = \{\ (f(f(x,y),z),\ 1,\ \emptyset) \mid f \in \mathcal{F}_{AC}\ \}$$

Ainsi, le seul moyen d'étendre une équation est d'unifier son membre gauche avec le sous-terme $f(x,y)$ de ce contexte, et l'extension d'une équation $(l \simeq r)$ est donc $(f(l,z) \simeq f(r,z))$. Il est inutile de calculer les extensions des extensions ; ce sont des instances de l'extension précédente : $(f(f(l,z),z') \simeq f(f(r,z),z'))$ est l'instance de $(f(l,z) \simeq f(r,z))$, en substituant $f(z,z')$ à z.

Nous montrons dans ce chapitre les différentes améliorations qu'il est possible d'apporter aux règles d'inférence définies dans le Chapitre 2. Ces transformations sont compatibles avec les stratégies de paramodulation et de superposition, ainsi que la stratégie positive et les règles de simplification.

5.1 Notations particulières

Les théories associatives et commutatives étant des cas très particuliers de théories équationnelles, nous commençons par définir un ensemble de notations adaptées à ces théories.

Supposons que les opérateurs d'un sous-ensemble $\mathcal{F}_{AC}$ de $\mathcal{F}$ sont associatifs et commutatifs, ce qui signifie que, pour tout opérateur f de $\mathcal{F}_{AC}$, les axiomes $(f(f(x,y),z) \simeq f(x,f(y,z)))$ et $(f(x,y) \simeq f(y,x))$ appartiennent à la théorie à considérer. La congruence sur $\mathcal{T}(\mathcal{F},\mathcal{X})$ engendrée par des axiomes, appelée AC-égalité, est notée $=_{AC}$. Mentionnons tout de suite que nous n'allons pas utiliser la représentation aplatie des termes (cf. Définition 1.7) afin de rester homogène avec notre technique de preuve qui, manipulant un ordre total sur les termes clos, a besoin de différencier tous termes syntaxiquement différents, même s'ils sont AC-égaux.

Nous notons $\cap_{AC}$ (resp. $\subseteq_{AC}$) l'intersection (resp. l'inclusion) d'ensembles, comparant les objets grâce à l'AC-égalité. Par exemple, $\{a*b,b,c\} \cap_{AC} \{b*a,c,d\} = \{a*b,c\}$, $\{a*b,c\} \subseteq_{AC} \{b*a,b,c\}$, où $*$ est AC. L'extension de $\subseteq_{AC}$ aux multi-ensembles est notée $\Subset_{AC}$. Ainsi, $\{a*b,c,c\} \Subset_{AC} \{b*a,c,b,c\}$, mais $\{a*b,c,c\} \not\Subset_{AC} \{b*a,b,c\}$.

$\mathcal{MP}os(t)$ désigne l'ensemble des positions non variables p du terme t telles que, si l'opérateur f en p est AC, alors il n'est pas sous le même opérateur f dans t.
Etant donné un terme t et un opérateur AC f, $\mathcal{H}terms(t,f)$ est le multi-ensemble des termes sous f au sommet de $aplat(t)$ (Définition 1.7). Par exemple, $\mathcal{H}terms(a*((a*x)*g(b)),*)$ est $\{a,a,x,g(b)\}$.

5.2 Règles d'inférence

Les définitions dans des théories associatives et commutatives des règles d'inférence de factorisation, de réflexion et de résolution correspondent exactement à celles définies dans une théorie équationnelle générale (Section 2.2.1). Seules les règles de paramodulation contiennent quelques améliorations supplémentaires.

La première de ces améliorations est la restriction des remplacements aux positions maximales des opérateurs AC, notées $\mathcal{MP}os(t)$ pour un terme t. Ceci revient à travailler uniquement avec des termes aplatis au niveau des opérateurs AC (cf. Définition 1.7). La raison de cette condition est très simple :

> Soit f un opérateur AC. Si un remplacement peut être appliqué au niveau du sous-terme $f(t_1,t_2)$ d'un terme $f(f(t_1,t_2),t_3)$, un remplacement avec contexte peut être effectué au sommet du terme et donner exactement le même résultat ; le sous-terme t_3 se trouve alors placé dans le contexte.

Il est vrai que cette condition oblige à faire appel à de plus gros problèmes d'unification AC, mais ces inférences auraient de toute façon été effectuées. Il s'agit donc d'une amélioration effective, proposée par M. Lai [Lai89] et E. Domenjoud [Dom91].

Donnons la définition de la règle de paramodulation.

Définition 5.1 (AC-Paramodulation)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2 \vee D_2}{L_2[p_2 \leftarrow r_1]\sigma \vee D_1\sigma \vee D_2\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{L_2|_{p_2} =^?_{AC} l_1\}) \quad où\ p_2 \in \mathcal{MP}os(L_2) \\ (l_1 \simeq r_1)\sigma \not\preceq D_1\sigma \\ L_2\sigma \not\preceq D_2\sigma \quad et \quad L_2\sigma \not\prec (l_1 \simeq r_1)\sigma \\ l_1\sigma \not\preceq r_1\sigma \end{cases}$$

Pour définir la paramodulation contextuelle, il n'est pas besoin de choisir un contexte pour le membre gauche l_1, car, comme seul le contexte $(f(f(x,y),z),1,\emptyset)$ est utile pour un opérateur $AC\ f$, il peut être directement utilisé.

Définition 5.2 (AC-Paramodulation contextuelle)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad L_2 \vee D_2}{L_2[p_2 \leftarrow f(r_1,z)]\sigma \vee D_1\sigma \vee D_2\sigma} \quad si \quad \begin{cases} \sigma \in \mathcal{S}ol(\{L_2|_{p_2} =^?_{AC} f(l_1,z)\}) \\ \qquad où\ \ f \in \mathcal{F}_{AC},\ p_2 \in \mathcal{MP}os(L_2) \\ \qquad et\ \ z\ est\ une\ nouvelle\ variable \\ (l_1 \simeq r_1)\sigma \not\preceq D_1\sigma \quad et \quad L_2\sigma \not\preceq D_2\sigma \\ l_1\sigma \not\preceq r_1\sigma \\ \forall w \in \mathcal{H}terms(L_2|_{p_2}, f), \\ \qquad \mathcal{H}terms(l_1\sigma, f) \not\subseteq_{AC} \mathcal{H}terms(w\sigma, f) \end{cases}$$

La condition $L_2\sigma \not\prec (l_1 \simeq r_1)\sigma$, apparaissant dans le cas d'une théorie générale E, a été supprimée car, les théories AC respectant la propriété de sous-terme et le terme $l_1\sigma$ étant un sous-terme strict de $L_2\sigma$, elle est triviale.

Outre les conditions classiques d'orientation de l'équation et de maximalité des littéraux, cette règle fait appel à une condition sur les $\mathcal{H}terms$ du sous-terme à la position p_2 du littéral L_2. Le multi-ensemble $\mathcal{H}terms(L_2|_{p_2}, f)$ contient

- des termes restreints à une variable ; préciser que les $\mathcal{H}terms$ de $l_1\sigma$ ne doivent pas être inclus dans les $\mathcal{H}terms$ de l'instanciation par σ de ces variables signifie que le terme $l_1\sigma$ ne doit pas avoir été créé de toute pièce par l'algorithme d'unification AC dans l'une de ces variables ;

- des termes dont l'opérateur au sommet n'est pas f ; la condition sur les $\mathcal{H}terms$ signifie alors que le terme $l_1\sigma$ ne doit pas être AC-égal à l'instance par σ de l'un de ces termes ; ceci revient à interdire le cas où l'opérateur au sommet de $l_1\sigma$ n'est pas f.

La dernière règle de paramodulation, la paramodulation étendue, applique aussi directement l'unique contexte associé à un opérateur AC. Elle met en œuvre des conditions sur les $\mathcal{H}terms$ qui apportent un contrôle très sévère sur son application.

Définition 5.3 (*AC*-Paramodulation étendue)

$$\frac{(l_1 \simeq r_1) \vee D_1 \qquad\qquad (l_2 \simeq r_2) \vee D_2}{(f(r_1,z_1) \simeq f(r_2,z_2))\sigma \vee D_1\sigma \vee D_2\sigma}$$

$$si \begin{cases} \sigma \in \mathcal{S}ol(\{f(l_1,z_1) =^?_{AC} f(l_2,z_2)\}) \\ \qquad\qquad \textit{où} \;\; f \in \mathcal{F}_{AC},\; z_1 \textit{ et } z_2 \textit{ sont de nouvelles variables} \\ (l_1 \simeq r_1)\sigma \not\preceq D_1\sigma \quad et \quad (l_2 \simeq r_2)\sigma \not\preceq D_2\sigma \\ l_1\sigma \not\preceq r_1\sigma \quad et \quad l_2\sigma \not\preceq r_2\sigma \\ \mathcal{H}terms(l_1\sigma, f) \;\cap_{AC}\; \mathcal{H}terms(l_2\sigma, f) \;\neq\; \emptyset \\ \mathcal{H}terms(z_1\sigma, f) \;\cap_{AC}\; \mathcal{H}terms(z_2\sigma, f) \;=\; \emptyset \end{cases}$$

La première condition sur les $\mathcal{H}terms$ précise que les termes $l_1\sigma$ et $l_2\sigma$ doivent avoir au moins un sous-terme en commun (modulo AC), et la seconde condition précise qu'ils doivent même en avoir un maximum, c'est-à-dire qu'aucun sous-terme ne doit apparaître à la fois dans $z_1\sigma$ et dans $z_2\sigma$.

Cette règle de paramodulation étendue peut être vue comme une généralisation de la règle de superposition de l'algorithme de B. Buchberger pour construire les bases de Gröbner [Buc65]. Les conditions sur les $\mathcal{H}terms$ ont une signification très proche des critères de paires critiques assurant un recouvrement des membres gauches de règles [Buc79]. Les relations entre les techniques de construction des bases de Gröbner et les techniques de réécriture ont d'ailleurs été étudiées par F. Winkler [Win84] et R. Bündgen [Bün91], mais surtout par C. Marché [Mar93].

Les deux exemples suivants mettent en évidence le rôle de ces conditions sur les $\mathcal{H}terms$. Ils montrent que, lorsque ces conditions ne sont pas respectées, la clause produite est redondante.

Exemple 5.1 Si $l_1\sigma$ et $l_2\sigma$ n'ont pas de $\mathcal{H}terms$ communs :

$$\frac{(f(a,b) \simeq c) \qquad\qquad (f(d,e) \simeq g)}{(f(c, f(d,e)) \simeq f(g, f(a,b)))} \quad \text{où} \quad \sigma = \{z_1 \mapsto f(d,e), z_2 \mapsto f(a,b)\}$$

Intuitivement, la clause déduite peut être à nouveau réduite par les deux clauses initiales en une tautologie $(f(c,g) \simeq f(g,c))$. D'autre part, toute inférence utilisant la clause déduite peut être remplacée par des inférences utilisant ces deux clauses initiales. ♦

Exemple 5.2 Si $z_1\sigma$ et $l_2\sigma$ ont un $\mathcal{H}terms$ commun :

$$\frac{(f(a, f(b,g)) \simeq c) \qquad\qquad (f(f(g,d),a) \simeq e)}{(f(c, f(g,d)) \simeq f(e, f(b,g)))} \quad \text{où} \quad \sigma = \{z_1 \mapsto f(g,d), z_2 \mapsto f(b,g)\}$$

Une autre paramodulation étendue peut être appliquée entre ces deux clauses en respectant les conditions sur les $\mathcal{H}terms$. La clause déduite est $(f(c,d) \simeq f(e,b))$. Or, la première clause déduite $(f(c, f(g,d)) \simeq f(e, f(b,g)))$ est en fait une extension de cette seconde déduction ; elle est donc inutile. ♦

Pour montrer l'efficacité de ces deux conditions sur les $\mathcal{H}terms$, donnons quelques statistiques :

- Une paramodulation étendue appliquée entre deux équations, dont les membres gauches sont $f(x_1, x_2)$ et $f(y_1, y_2)$, fait appel au problème d'unification AC

$$f(f(x_1, x_2), z_1) =^?_{AC} f(f(y_1, y_2), z_2)$$

qui admet 265 solutions minimales. Les conditions sur les $\mathcal{H}terms$ permettent de n'en retenir que 103.

- Pour une paramodulation étendue appliquée entre les membres gauches $f(x_1, f(x_2, x_3))$ et $f(y_1, f(y_2, y_3))$, le problème d'unification AC

$$f(f(x_1, f(x_2, x_3)), z_1) =^?_{AC} f(f(y_1, f(y_2, y_3)), z_2)$$

admet 41503 solutions minimales. Seules 18456 satisfont les conditions sur les $\mathcal{H}terms$.

Ces deux exemples montrent des cas extrêmes, car le nombre de solutions reste énorme. Cependant, ils donnent une idée du gain apporté, car ces conditions (et plus particulièrement la seconde) peuvent être très facilement intégrées dans un algorithme d'unification AC.

Les systèmes d'inférence INF ainsi formés pour les stratégies de paramodulation et de superposition (et en présence des règles de simplifications) sont réfutationnellement complets.

Théorème 5.4 (Théorème de complétude) *Soit S un ensemble AC-incohérent de clauses. Alors, toute dérivation équitable de S engendre la clause vide.*

Preuve : *La preuve de ce théorème est un cas particulier de la preuve des théorèmes cités dans le Chapitre 2, pour la déduction modulo une théorie équationnelle régulière. Les conditions supplémentaires sur les $\mathcal{H}terms$ s'intègrent très naturellement dans ces preuves de la manière suivante :*

- *Paramodulation contextuelle : la condition sur les $\mathcal{H}terms$ est une conséquence de l'irréductibilité de la substitution close appliquée aux clauses.*
- *Paramodulation étendue : les membres gauches doivent posséder au moins un $\mathcal{H}terms$ en commun, car sinon la clause déduite se réduit en une tautologie (cf. Exemple 5.1), ce qui contredit la consistance du morceau de branche droite déjà construit (cf. preuve du Lemme A.2). Ces membres gauches doivent partager en plus un maximum de $\mathcal{H}terms$, car sinon une autre paramodulation étendue engendre une clause plus petite (cf. Exemple 5.2), également falsifiée par la branche droite.*

Le relèvement au cas général de cette preuve est effectué grâce aux lemmes décrits dans la section suivante. □

5.3 Relèvement des conditions sur les $\mathcal{H}terms$

Nous avons montré dans la Section 4.2 comment relever les règles d'inférence du cas clos au cas général, en n'utilisant que des substitutions irréductibles. Les preuves décrites sont valables pour les théories associatives et commutatives, mais nous devons nous assurer que les conditions ajoutées sur les $\mathcal{H}terms$ dans les règles de paramodulation contextuelle et paramodulation étendue peuvent également être relevées.

Lemme 5.5 (Relèvement de la AC-paramodulation contextuelle) *Soient deux clauses $C_1 = (l_1 \simeq r_1) \vee D_1$ et $C_2 = L_2 \vee D_2$ et σ une substitution close. Si une AC-paramodulation contextuelle de $C_1\sigma$ dans $C_2\sigma$, à une position $p_2 \in \mathcal{MP}os(L_2\sigma)$, produit une clause C, alors une paramodulation contextuelle peut être appliquée de C_1 dans C_2 à cette même position p_2, produisant une clause D admettant C pour instance close.*

Preuve: *Grâce au Lemme de Relèvement 4.13 et à l'hypothèse d'irréductibilité de la substitution σ, nous savons que la position p_2 est une position non variable de L_2, i.e. $p_2 \in \mathcal{MP}os(L_2)$. La paramodulation contextuelle de $C_1\sigma$ dans $C_2\sigma$ utilise une substitution σ' telle que: $L_2\sigma|_{p_2} =_{AC} f(l_1\sigma, z\sigma')$, où f est un opérateur AC.*
Il existe une substitution τ, AC-unificateur principal de $f(l_1, z)$ et $L_2|_{p_2}$, et une substitution close ρ telles que: $\forall x \in \mathcal{D}om(\sigma)$, $x\sigma =_{AC} x\tau\rho$, et $z\sigma' =_{AC} z\tau\rho$. Les conditions d'ordre sont relevées comme dans le Lemme 4.13.

La dernière condition à relever est celle sur les $\mathcal{H}terms$ du sous-terme à la position p_2 de L_2. Nous savons que:

$$\forall w \in \mathcal{H}terms(L_2\sigma|_{p_2}, f),\ \mathcal{H}terms(l_1\sigma, f) \not\subseteq_{AC} \mathcal{H}terms(w, f)$$

c'est-à-dire: $\forall w \in \mathcal{H}terms(L_2\sigma|_{p_2}, f)$, $l_1\sigma \neq_{AC} w$; cela assure que l'opérateur au sommet de $l_1\sigma$ est f. Il nous faut montrer que:

$$\forall w \in \mathcal{H}terms(L_2|_{p_2}, f),\ \mathcal{H}terms(l_1\tau, f) \not\subseteq_{AC} \mathcal{H}terms(w\tau, f)$$

Supposons qu'un tel terme w ne satisfasse pas cette dernière condition ; w étant issu de $\mathcal{H}terms(L_2|_{p_2}, f)$, il ne possède pas f en son sommet ; il s'agit donc soit d'une variable, soit d'un terme avec au sommet un opérateur g ; ce dernier cas peut être immédiatement éliminé, car il signifierait que g est l'opérateur au sommet de $l_1\tau$, ce qui est impossible ($l_1\tau\rho =_{AC} l_1\sigma$).
w ne peut donc être qu'une variable. Cependant, le terme $l_1\tau$ étant inclus dans $w\tau$, son instance $l_1\tau\rho$ est incluse dans $w\tau\rho$; or, les termes $w\tau\rho$ et $w\sigma$ étant AC-égaux et w étant une variable, $w\sigma$ est réductible (modulo AC) par $(l_1\sigma \simeq r_1\sigma)$, ce qui contredit l'hypothèse d'irréductibilité de σ.

La condition sur les $\mathcal{H}terms$ étant ainsi montrée pour la substitution τ, nous pouvons en déduire de la manière suivante que le terme $l_1\tau$ n'est pas réduit à une variable: si $l_1\tau = x \in \mathcal{X}$, comme $f(x, z\tau) =_{AC} (L_2|_{p_2})\tau$, cela implique qu'il existe un terme w de $\mathcal{H}terms(L_2|_{p_2}, f)$, tel que $w\tau$ est soit x, soit $f(x, w')$ pour un terme w', ce qui contredit la propriété précédente sur les $\mathcal{H}terms$.

Toutes les conditions sont réunies pour appliquer la suite du Lemme de Relèvement 4.13, assurant que la clause déduite par une paramodulation contextuelle de C_1 dans C_2 admet C pour instance close (modulo AC). □

Montrons également le relèvement des conditions sur les $\mathcal{H}terms$ dans la règle de paramodulation étendue dans les théories AC.

Lemme 5.6 (Relèvement de la AC-paramodulation étendue) *Soient deux clauses $C_1 = (l_1 \simeq r_1) \vee D_1$ et $C_2 = (l_2 \simeq r_2) \vee D_2$, et σ une substitution close. Si une AC-paramodulation étendue entre $C_1\sigma$ et $C_2\sigma$ produit une clause C, alors une paramodulation étendue peut être appliquée entre C_1 et C_2, produisant une clause D admettant C pour instance close.*

Preuve : *La paramodulation étendue entre $C_1\sigma$ et $C_2\sigma$ utilise une substitution σ' telle que : $f(l_1\sigma, z_1\sigma') =_{AC} f(l_2\sigma, z_2\sigma')$, où f est un opérateur AC.*
Comme dans la preuve du lemme précédent, le Lemme de Relèvement 4.13 et à l'hypothèse d'irréductibilité de la substitution σ nous garantissent l'existence d'une substitution τ, AC-unificateur principal de $f(l_1, z_1)$ et $f(l_2, z_2)$, et une substitution close ρ telles que : $\forall x \in \mathcal{D}om(\sigma)$, $x\sigma =_{AC} x\tau\rho$, $z_1\sigma' =_{AC} z_1\tau\rho$, et $z_2\sigma' =_{AC} z_2\tau\rho$. Les conditions d'ordre sont relevées comme dans le Lemme 4.13.

Montrons le relèvement des conditions sur les $\mathcal{H}terms$. Nous savons que :

$$\begin{cases} \mathcal{H}terms(l_1\sigma, f) \cap_{AC} \mathcal{H}terms(l_2\sigma, f) \neq \emptyset & (1) \\ \mathcal{H}terms(z_1\sigma', f) \cap_{AC} \mathcal{H}terms(z_2\sigma', f) = \emptyset & (2) \end{cases}$$

Montrons dans un premier temps que $\mathcal{H}terms(z_1\tau, f) \cap_{AC} \mathcal{H}terms(z_2\tau, f) = \emptyset$. Supposons qu'il existe deux éléments AC-égaux a et b de $\mathcal{H}terms(z_1\tau, f)$ et $\mathcal{H}terms(z_2\tau, f)$ respectivement. Donc les termes $a\rho$ et $b\rho$ sont AC-égaux, de même que les multi-ensembles $\mathcal{H}terms(a\rho, f)$ et $\mathcal{H}terms(b\rho, f)$. D'où, $\mathcal{H}terms(z_1\tau\rho, f) \cap_{AC} \mathcal{H}terms(z_2\tau\rho, f) \neq \emptyset$. Ceci contredit la propriété (2), car $\tau\rho =_{AC} \sigma\sigma'$.
Ainsi, $\mathcal{H}terms(z_1\tau, f) \cap_{AC} \mathcal{H}terms(z_2\tau, f) = \emptyset$.

Montrons maintenant que les termes $l_1\tau$ et $l_2\tau$ partagent au moins un sous-terme. Par définition de τ, on a : $f(l_1\tau, z_1\tau) =_{AC} f(l_2\tau, z_2\tau)$. Donc,

$$\mathcal{H}terms(l_1\tau, f) \cup_{|AC} \mathcal{H}terms(z_1\tau, f) =_{AC} \mathcal{H}terms(l_2\tau, f) \cup_{|AC} \mathcal{H}terms(z_2\tau, f)$$

Nous avons prouvé que $\mathcal{H}terms(z_1\tau, f)$ et $\mathcal{H}terms(z_1\tau, f)$ n'ont aucun élément en commun. Donc,

$$\begin{cases} \mathcal{H}terms(z_1\tau, f) \subseteq_{AC} \mathcal{H}terms(l_2\tau, f) \\ \mathcal{H}terms(z_2\tau, f) \subseteq_{AC} \mathcal{H}terms(l_1\tau, f) \end{cases}$$

Si nous supposons que $\mathcal{H}terms(z_1\tau, f) \subset_{AC} \mathcal{H}terms(l_2\tau, f)$, tous les sous-termes de $l_2\tau$ qui ne sont pas dans $z_1\tau$, sont dans $l_1\tau$. Donc, $\mathcal{H}terms(l_1\tau, f)$ et $\mathcal{H}terms(l_2\tau, f)$ ont des éléments en commun.

Le second cas est $\mathcal{H}terms(z_1\tau, f) =_{AC} \mathcal{H}terms(l_2\tau, f)$; cela signifie que $l_2\tau =_{AC} z_1\tau$ et $l_1\tau =_{AC} z_2\tau$, et en conséquence : $l_2\sigma =_{AC} z_1\sigma$ et $l_1\sigma =_{AC} z_2\sigma$. La propriété (2) dit que $z_1\sigma$ et $z_2\sigma$ n'ont pas de $\mathcal{H}terms$ communs, donc $l_1\sigma$ et $l_2\sigma$ non plus ; cela contredit la propriété (1). Donc, ce cas est aussi impossible.

Nous pouvons facilement vérifier que ni $l_1\tau$ ni $l_2\tau$ ne sont des variables ; en effet, si nous supposons que $l_1\tau$ est une variable x, comme $l_2\tau$ et $l_1\tau$ partagent des $\mathcal{H}terms$, la variable x appartient à $\mathcal{H}terms(l_2\tau, f)$; d'autre part, cela signifie que $z_2\tau$ est inclus dans $z_1\tau$, ce qui est en contradiction avec le fait que ces deux termes n'ont aucun $\mathcal{H}terms$ en commun. Donc, $\mathcal{H}ead(l_1\tau) = \mathcal{H}ead(l_2\tau) = f$.

Les conditions sur les $\mathcal{H}terms$ étant valides pour la substitution τ, une paramodulation étendue peut effectivement être appliquée entre les clauses C_1 et C_2, avec τ pour AC-unificateur principal, et la clause déduite admet la clause C comme instance close (Lemme 4.13), modulo AC. □

5.4 Exemple de résolution

Nous détaillons dans cette section un exemple d'application des règles d'inférence définies dans ce chapitre, qui consiste à prouver la propriété $(x_1^* \cdot x_1^* \simeq x_1^*)$ dans les Algèbres de Kleene, décrites par l'axiomatisation de D. Kozen [Koz91] suivante :

$(K, +, 0)$ est un monoïde commutatif idempotent : $AC(+)$

(1) $(x_1 + 0 \simeq x_1)$
(2) $(x_1 + x_1 \simeq x_1)$

$(K, \cdot, 1)$ est un monoïde :

(3) $(x_1 \cdot (x_2 \cdot x_3) \simeq (x_1 \cdot x_2) \cdot x_3)$
(4) $(1 \cdot x_1 \simeq x_1)$
(5) $(x_1 \cdot 1 \simeq x_1)$

$(K, +, \cdot, 0, 1)$ est un semi-anneau idempotent :

(6) $(x_1 \cdot (x_2 + x_3) \simeq (x_1 \cdot x_2) + (x_1 \cdot x_3))$
(7) $((x_1 + x_2) \cdot x_3 \simeq (x_1 \cdot x_3) + (x_2 \cdot x_3))$
(8) $(0 \cdot x_1 \simeq 0)$
(9) $(x_1 \cdot 0 \simeq 0)$

Définition de l'opérateur *étoile* de Kleene :

(10) $(1 + (x_1 \cdot x_1^*) \simeq x_1^*)$
(11) $(1 + (x_1^* \cdot x_1) \simeq x_1^*)$
(12) $\neg((x_1 \cdot x_2) + x_2 \simeq x_2) \vee ((x_1^* \cdot x_2) + x_2 \simeq x_2)$
(13) $\neg((x_1 \cdot x_2) + x_1 \simeq x_1) \vee ((x_1 \cdot x_2^*) + x_1 \simeq x_1)$

Pour montrer la propriété citée précédemment, nous ajoutons au système une clause représentant la négation de cette propriété (où A est une nouvelle constante) :

(C) $\neg(A^* \cdot A^* \simeq A^*)$

L'ordre utilisé pour comparer les termes est un ordre récursif sur les chemins, basé sur l'APO de L. Bachmair et D. Plaisted [BP85b]. La précédence sur les opérateurs est : $\cdot >_{\mathcal{F}} + >_{\mathcal{F}} {}^* >_{\mathcal{F}} 1 >_{\mathcal{F}} 0$.

La séquence de déductions suivante montre comment une contradiction peut être dérivée du système formé des 13 clauses représentant la théorie des algèbres de Kleene et de la clause (C). Notons que nous avons utilisé la stratégie de superposition.

`Superposition étendue entre 2 et 10`

(14) $(1 + (x_1 \cdot x_1^*) \simeq x_1^* + 1)$

`Simplification de 10 dans 14`

(14) $(x_1^* + 1 \simeq x_1^*)$

`Superposition étendue entre` 14 `et 10`

(15) $(x_1^* + (x_2 \cdot x_2^*) \simeq x_2^* + x_1^*)$

`Superposition de` 14 `dans 7`

(16) $((x_2^* \cdot x_1) + (1 \cdot x_1) \simeq x_2^* \cdot x_1)$

`Simplification de 4 dans 16`

(16) $((x_2^* \cdot x_1) + x_1 \simeq x_2^* \cdot x_1)$

`Simplification de 16 dans 12`

(12) $\neg((x_1 \cdot x_2) + x_2 \simeq x_2) \;\vee\; (x_1^* \cdot x_2 \simeq x_2)$

`Superposition de 12 dans` C

(17) $\neg(A^* \simeq A^*) \;\vee\; \neg((A \cdot A^*) + A^* \simeq A^*)$

`Réflexion triviale dans 17`

(17) $\neg((A \cdot A^*) + A^* \simeq A^*)$

`Simplification de 15 dans 17`

(17) $\neg(A^* + A^* \simeq A^*)$

`Simplification clausale dans 17 grâce à 2`

(17) $\square$

6
AC-Paramodulation contrainte

Nous avons défini depuis le début de ce document des ensembles de règles d'inférence pour effectuer des déductions modulo une théorie équationnelle E. L'application de ces règles est restreinte grâce à des conditions d'ordre, et dans le chapitre précédent, consacré aux cas particuliers des théories associatives et commutatives (AC), nous avons décrit des mécanismes supplémentaires permettant de limiter l'espace de recherche.

Cependant, malgré ces conditions, l'appel d'un algorithme d'unification AC pouvant engendrer de très nombreuses solutions entame fortement l'efficacité du processus d'inférence. Aussi, nous présentons dans ce chapitre une amélioration très importante pour limiter le nombre de clauses engendrées, et donc le nombre de déductions possibles. Il s'agit de l'association de contraintes symboliques à chaque clause, permettant entre autres de retarder, voire d'éviter, la résolution des problèmes d'unification AC.

Un premier type de contrainte symbolique simule la *stratégie basique*, qui consiste à interdire tout remplacement dans un sous-terme introduit dans une clause par une substitution lors d'une déduction antérieure. Par exemple, considérant l'étape de paramodulation suivante, avec la substitution $\{x \mapsto c,\ y \mapsto a\}$,

$$\frac{(f(a,x) \simeq b) \qquad P(f(y,c)) \vee Q(y,b)}{P(b) \vee Q(\boxed{a}, b)}$$

le sous-terme $f(y,c)$ est remplacé par b, en supposant que $f(a,c) \not\preceq b$ (orientation de l'équation) et $P(f(a,c)) \not\preceq Q(a,b)$ (maximalité du littéral utilisé). Le sous-terme a ayant été introduit dans la clause déduite par l'instanciation de la variable y, tout futur remplacement dans a est interdit. L'idée de cette restriction est simple : si un remplacement doit être appliqué dans a, alors il doit avoir lieu dans la clause $(f(a,x) \simeq b)$, avant d'effectuer l'étape de paramodulation précédente.
Il est possible de compléter cette définition en interdisant tout futur remplacement dans le terme introduit dans la clause déduite par le remplacement (b dans l'exemple précédent). Ceci ne s'applique qu'à la stratégie de paramodulation, car elle permet des remplacements dans les membres non maximaux des équations.

Nous traduisons cette stratégie basique par des *contraintes d'unification* AC, dans la lignée des travaux de C. Kirchner, H. Kirchner et M. Rusinowitch [KKR90] et de R. Nieuwenhuis et A. Rubio [NR92a, NR92b]. Ainsi, à chaque clause C est associée une conjonction de contraintes, notée $[\![c]\!]$, formée des contraintes d'unification AC posées lors de la déduction de C, ainsi que des contraintes héritées de ses clauses parentes. L'exemple précédent devient :

$$\frac{(f(a,x) \simeq b) \ \llbracket c_1 \rrbracket \qquad P(f(y,c)) \vee Q(y,b) \ \llbracket c_2 \rrbracket}{P(b) \vee Q(y,b) \ \llbracket c_1 \wedge c_2 \wedge c_3 \rrbracket}$$

où c_3 contient la contrainte d'unification AC $(f(a,x) =^?_{AC} f(y,c))$.

Une clause contrainte ainsi déduite représente autant de clauses qu'il y a de solutions minimales à ses contraintes d'unification AC. Cela représente un gain considérable en espace, pour n'avoir engendré qu'une seule clause, mais le gain est aussi très important en calcul, car rechercher toutes les solutions minimales d'un problème d'unification AC est doublement exponentiel, alors que tester l'existence d'une solution n'est qu'exponentiel.

A ces contraintes d'unification, nous ajoutons des *contraintes d'ordre*, représentant les conditions des règles d'inférence. Ainsi, dans l'exemple précédent, c_3 contient également les contraintes $(f(a,x) \succ^? b)$ et $(P(f(y,c)) \succ^? Q(y,b))$. Ces contraintes permettent de transmettre d'une inférence à la suivante les hypothèses sur la maximalité des termes.

Dans la première partie de ce chapitre, nous retraçons la genèse de la stratégie basique et des contraintes symboliques. Après avoir introduit quelques notations, nous présentons des ensembles de règles d'inférence basés sur les stratégies de paramodulation et de superposition, puis détaillons différentes règles de simplification. La preuve de complétude réfutationnelle de ces systèmes est dérivée très naturellement de la preuve dans le cas non contraint. Dans la dernière section, nous discutons des avantages et des inconvénients d'appliquer la stratégie basique au lieu de la stratégie contrainte, c'est-à-dire de résoudre les problèmes d'unification AC, mais de ne pas appliquer dans les clauses les substitutions solutions, comme cela a été réalisé pour la théorie vide par L. Bachmair, H. Ganzinger, C. Lynch et W. Snyder [BGLS92].

6.1 Historique et motivations

Rappelons les travaux ayant conduit à la stratégie basique, et à l'usage des contraintes symboliques. Nous montrerons ensuite les avantages essentiels apportés par les contraintes dans les mécanismes de déduction modulo des théories associatives et commutatives.

6.1.1 Stratégie basique et contraintes symboliques

L'idée d'interdire tout remplacement dans un terme créé par l'instanciation d'une variable a été proposée en premier par J.R. Slagle [Sla74], avec la définition d'*inférences bloquées*, caractérisées par l'utilisation de substitutions associant un terme irréductible (au sens de la réécriture) à chaque variable. Cette propriété était également sous-jacente dans la *méthode de modification* de D. Brand [Bra75].

Ce concept a été étendu aux théories permutatives (incluant les théories AC) par A.M. Ballantyne et D.S. Lankford [BL79]. Son adaptation à la technique de surréduction a été réalisée par J.-M. Hullot dans sa thèse [Hul80], dans des dérivations dites *basiques*, grâce à la définition de l'ensemble des positions dans lesquelles les étapes de surréduction sont autorisées ; la règle d'inférence issue de cela fut appelée *surréduction basique*.

Dans une optique différente mais très proche du principe de la stratégie basique, D. Kapur, D.R. Musser et P. Narendran [KMN85] ont défini la règle de *superposition primale*, tenant compte du fait que de nombreuses superpositions sont inutiles dans la procédure de D.E. Knuth et P.B. Bendix [KB70].

La stratégie basique a été adaptée à la paramodulation par W. Snyder et C. Lynch [SL91], puis a été complètement définie, pour des systèmes de règles d'inférence basés sur la superposition et la paramodulation, par L. Bachmair, H. Ganzinger, C. Lynch et W. Snyder dans [BGLS92] ; à chaque clause est associée une substitution, représentant le cumul des substitutions calculées pour déduire cette clause. Conserver cette substitution hors de la clause évite le calcul les positions dans lesquelles les remplacements sont autorisés.

Cependant, la plupart de ces travaux sur la stratégie basique ont été motivés initialement par l'idée de bloquer tout terme créé par un algorithme d'unification, ce qui fut partiellement réalisé dans le cadre de la logique d'ordre supérieur par G. Huet dans sa thèse [Hue72], où une règle de résolution, basée sur la résolution de J.A. Robinson [Rob65], fut décrite de telle manière que les problèmes d'unification n'aient pas à être résolus, ces derniers étant conservés comme contraintes des clauses déduites. Ce principe fut appliqué par C. Kirchner et H. Kirchner [KK89] pour définir une règle de superposition et un algorithme de complétion pour le raisonnement équationnel contraint. La règle de surréduction basique de J.-M. Hullot [Hul80] fut présentée sous une forme contrainte par W. Nutt, P. Réty et G. Smolka dans [NRS90].
R. Caferra et N. Zabel [CZ92] ont défini des règles d'inférence de résolution et de factorisation, pour la déduction dans la logique du premier ordre sans égalité ; le système d'inférence proposé permet de construire des modèles, en combinant des conjonctions et des disjonctions de contraintes d'unification, de disunification. Cette technique ne peut être simulée par les mécanismes classiques de résolution de contraintes comme ceux décrits dans [JL87, Bür91].

Les contraintes d'unification ont été utilisées par R. Nieuwenhuis et A. Rubio [NR92a] pour définir un ensemble de règles d'inférence basé sur la stratégie de superposition, dans lequel aucun unificateur n'a besoin d'être calculé.

Un second type de contraintes fut utilisé pour la complétion sans échec par G.E. Peterson [Pet90] et U. Martin et T. Nipkow [MN90] ; il s'agit des contraintes d'ordre. Elles permettent d'orienter des axiomes tels que la commutativité : $x * y \rightarrow y * x$ `si` $x > y$. L'utilisation de telles contraintes est également motivée par les résultats sur la décidabilité des contraintes d'ordre de H. Comon [Com90] (l'ordre lexicographique sur les chemins (LPO) est décidable) et J.-P. Jouannaud et M. Okada [JO91] (l'ordre lexicographique et multi-ensemble sur les chemins (RPO) est décidable).

La combinaison de contraintes d'unification et de contraintes d'ordre fut proposée par C. Kirchner, H. Kirchner et M. Rusinowitch [KKR90] lors de la définition de règles d'inférence pour les stratégies de superposition et de paramodulation ; pour être complet, ces systèmes nécessitaient soit de résoudre quelques problèmes d'unification, soit d'appliquer des remplacements dans des termes appartenant aux contraintes. R. Nieuwenhuis et A. Rubio [NR92b] ont défini des règles d'inférence basées sur la stratégie de superposition qui ne nécessitaient aucun calcul d'unificateur, et dans lesquelles les conditions d'ordre (orientation d'équations, maximalité d'un littéral dans une clause) étaient conservées.

Nous présentons dans ce chapitre l'extension de ce dernier résultat aux théories associatives et commutatives. Le système d'inférence ainsi défini est réfutationnellement complet, sans jamais nécessiter le calcul d'unificateurs AC. Ce résultat a été présenté dans [Vig94a].
Nous devons cependant préciser que le premier système de règles d'inférence ne nécessitant aucun calcul d'unificateurs AC est dû à R. Nieuwenhuis et A. Rubio [NR94, Rub94], car dans une première version [Vig93], nous devions calculer des unificateurs dans des cas très restreints.

6.1.2 Avantages des contraintes symboliques

Les contraintes symboliques que nous allons manipuler dans la suite de ce chapitre sont donc de deux types : contraintes d'unification AC et contraintes d'ordre.

La déduction dans des théories associatives et commutatives fait appel à un algorithme d'unification [Sti81, BHK+88]. Cet algorithme est décidable et finitaire (i.e. il existe un nombre fini de solutions à un problème d'unification AC). Cependant, le calcul de l'ensemble des solutions minimales étant doublement exponentiel [KN92], il est nécessaire de les éviter. Ainsi, les contraintes d'unification nécessitent seulement de tester l'existence d'une solution, ce qui est seulement exponentiel. Outre ce calcul simplifié, il n'est plus nécessaire d'engendrer une clause pour chaque solution minimale, ce qui a pour effet de diminuer aussi le nombre de déductions potentielles.

Exemple 6.1 Soit $*$ un opérateur AC. Considérons une étape de paramodulation de l'équation $(x * x * x * x \simeq x)$ dans la clause $P(y_1 * y_2 * y_3 * y_4) \vee Q(y_1, y_2, y_3, y_4)$, au niveau du sous-terme $y_1 * y_2 * y_3 * y_4$; une clause doit être engendrée pour chaque AC-unificateur principal, ce qui fait 34 359 607 481 nouvelles clauses (toutes indispensables), comme l'a montré E. Domenjoud [Dom92].
Cette même étape de paramodulation avec une contrainte d'unification, n'engendre qu'une seule clause : $P(x) \vee Q(y_1, y_2, y_3, y_4)\ [\![\, x * x * x * x =^?_{AC} y_1 * y_2 * y_3 * y_4 \,]\!]$. ♦

L'avantage de ce type de contraintes était déjà souligné dans [KK89, KKR90] et [Bür91].

En ce qui concerne les contraintes d'ordre, leur but est de conserver une trace des choix effectués au cours des déductions, que ce soit pour orienter une équation ou pour choisir un littéral maximal dans une clause. Ainsi, tout nouveau choix reste cohérent avec les précédents. L'exemple suivant montre le problème soulevé en cas d'absence de ces contraintes.

Exemple 6.2 Etant donné le système de clauses suivant,

$$\begin{cases} (f(b) \simeq y) & (1) \\ (g(f(a)) \simeq a) & (2) \\ P(g(f(b))) & (3) \\ \neg P(a) & (4) \end{cases}$$

montrons qu'il est incohérent, en utilisant l'ordre LPO avec la précédence $g >_{\mathcal{F}} f >_{\mathcal{F}} a >_{\mathcal{F}} b$.

$$\begin{array}{rcc} \text{Paramodulation de (1) dans (3) :} & P(g(y)) & (5) \\ \text{Paramodulation de (2) dans (5) :} & P(a) & (6) \\ \text{Résolution entre (4) et (6) :} & \square & (7) \end{array}$$

Une contradiction a bien été dérivée, mais étudions plus en détail les deux étapes de paramodulation. Dans la première, le sous-terme $f(b)$ a été remplacé par y, bien que ces deux termes soient incomparables par l'ordre LPO, car la condition testée est $f(b) \not\preceq y$; cependant, le but étant de remplacer un terme par un plus petit, la condition à retenir est $f(b) \succ^? y$.
Dans la seconde étape de paramodulation, la variable y a été instanciée par $f(a)$. Or, cette instanciation falsifie la contrainte précédente ($f(b) \prec f(a)$).
Cela signifie que la dérivation précédente est incorrecte (et donc inutile), mais cela ne remet pas en cause l'incohérence du système initial, car il existe toujours une dérivation correcte de la clause vide :

Paramodulation de (1) dans (2) :	$(g(f(b)) \simeq a)$	(5)	$\{y \mapsto f(a)\},\ f(a) \succ f(b)$
Paramodulation de (5) dans (3) :	$P(a)$	(6)	
Résolution entre (4) et (6) :	$\square$	(7)	

♦

6.2 Notations et langage de contraintes

Après avoir présenté quelques notations venant s'ajouter à celles définies dans le chapitre précédent (Section 5.1), nous décrirons celles concernant les contraintes symboliques.

La relation $\trianglelefteq_{AC}$ représente la propriété de sous-terme dans les théories AC, c'est-à-dire $t_1 \trianglelefteq_{AC} t_2$ s'il existe un terme t'_2 AC-égal à t_2 et une position p dans t'_2 tels que $t'_2|_p$ est égal à t_1. Par exemple, $a * b \trianglelefteq_{AC} b * (a * c)$, où $*$ est AC. Cette relation est étendue aux ensembles de termes : $\{s_1, \ldots, s_m\} \trianglelefteq_{AC} \{t_1, \ldots, t_n\}$ si $\forall i \in [1, m],\ \exists j \in [1, n],\ s_i \trianglelefteq_{AC} t_j$.

L'ensemble des substitutions σ permettant de filtrer un terme t_1 dans un terme t_2 modulo AC ($t_1\sigma =_{AC} t_2$) est noté $AC\text{-}match(t_1, t_2)$.

Nos règles d'inférence manipulent des *clauses contraintes*, notées $C\,[\![\, c \,]\!]$ comme dans [NR92a], où C est une clause et c est une conjonction de contraintes. Les contraintes de base sont des formules atomiques du type $(s =^?_{AC} t)$ (contrainte d'unification AC) ou $(s \succ^? t)$ (contrainte d'ordre) pour deux objets s et t. Les solutions de contraintes sont des substitutions sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ notées SU. Une conjonction de contraintes est dite *satisfaisable* si elle admet au moins une solution. Pour une contrainte c, $\mathcal{S}ol(c)$ désigne l'ensemble des solutions définies par :

- $\mathcal{S}ol(t_1 =^?_{AC} t_2) \ = \ \{\ \sigma \in SU \mid t_1\sigma =_{AC} t_2\sigma\ \}$
- $\mathcal{S}ol(t_1 \succ^? t_2) \ = \ \{\ \sigma \in SU \mid t_1\sigma \succ t_2\sigma\ \}$

Des contraintes d'ordre dérivées de la précédente seront utilisées par la suite pour exprimer, soit qu'un terme est plus supérieur ou égal à un autre, soit qu'un littéral est supérieur à tous les littéraux d'une clause.

- $\mathcal{S}ol(t_1 \succeq^? t_2) \ = \ \{\ \sigma \in SU \mid t_1\sigma \succeq t_2\sigma\ \}$
- $\mathcal{S}ol(L \succ^? (L_1 \vee \ldots \vee L_n)) \ = \ \{\ \sigma \in SU \mid \forall i \in [1, n],\ L\sigma \succ L_i\sigma\ \}$

D'autres symboles sont ajoutés dans le langage des contraintes, comme $\mathbb{T}$, $\mathbb{F}$ et $\wedge$. Ils sont interprétés par :

- $\mathcal{S}ol(\mathbb{T}) = \{ \sigma \in SU \}$
- $\mathcal{S}ol(\mathbb{F}) = \emptyset$
- $\mathcal{S}ol(c_1 \wedge c_2) = \mathcal{S}ol(c_1) \cap \mathcal{S}ol(c_2)$

Les contraintes définies ci-dessus peuvent faire appel à un algorithme de résolution. Cependant, notre but étant de retenir dans la partie contrainte toutes les conditions d'application des règles d'inférence, nous devons définir des contraintes d'un type très différent, dont le test de satisfaisabilité sera : il existe une solution aux contraintes d'unification et d'ordre ne contredisant pas ces contraintes spécifiques. La première de ces contraintes a pour but de tester que l'opérateur au sommet d'un terme t est bien un opérateur donné f ; elle n'est pas résolue si t est une variable, et fait donc appel à la fonction $\mathcal{H}ead$.

- $\mathcal{H}ead(t) =^? f$

Les autres contraintes font appel à la fonction $\mathcal{H}terms$, pour tester l'intersection d'ensembles ($\cap^?$) ou la non inclusion de multi-ensembles ($\not\subseteq^?$), en utilisant l'AC-égalité $=_{AC}$ pour comparer les termes.

- $\mathcal{H}terms(t_1, f) \cap^? \mathcal{H}terms(t_2, f) =^? \emptyset$
- $\mathcal{H}terms(t_1, f) \cap^? \mathcal{H}terms(t_2, f) \neq^? \emptyset$
- $\mathcal{H}terms(t_1, f) \not\subseteq^? \mathcal{H}terms(t_2, f)$

Etant donnée une substitution σ, par abus de notation nous écrirons $C \llbracket \sigma \rrbracket$ la clause contrainte $C \llbracket \bigwedge_{x \in \mathcal{D}om(\sigma)} (x =^?_{AC} x\sigma) \rrbracket$.

6.3 Règles d'inférence

Dans les règles définies dans cette section, nous ne précisons pas que l'ensemble des contraintes des clauses déduites doit être satisfaisable, car il existe deux points de vue à ce sujet :

- Vérifier à chaque déduction que les contraintes sont satisfaisables, ce qui est très coûteux mais évite de nombreuses déductions inutiles.
- Vérifier la satisfaisabilité des contraintes uniquement lorsqu'une contradiction a été engendrée, ce qui provoque la déduction de nombreuses clauses inutiles.

Ces deux stratégies ne sont pas incompatibles et pourraient très bien être combinées. Le choix est laissé à l'utilisateur.

6.3.1 Stratégie de paramodulation

Nous définissons ci-dessous six règles d'inférence qui sont en fait les adaptations à la stratégie contrainte et aux théories AC (Section 5.2) des règles définies dans la Section 2.2.1. Les exemples illustrant leur application utilisent l'opérateur AC $*$ et la précédence $P >_{\mathcal{P}} Q >_{\mathcal{P}} \simeq$ sur les opérateurs de prédicats, et $* >_{\mathcal{F}} g >_{\mathcal{F}} a >_{\mathcal{F}} b >_{\mathcal{F}} d >_{\mathcal{F}} e$ sur les opérateurs de fonctions. Les contraintes d'ordre et d'unification AC seront systématiquement simplifiées au moyen de règles que nous définirons dans la Section 6.6.

Définition 6.1 (AC-Factorisation contrainte)

$$\frac{L_1 \vee \ldots \vee L_n \vee D \;[\![c_1]\!]}{L_1 \vee D \;[\![c_1 \wedge c_2]\!]} \quad \textit{où } c_2 \textit{ est } \; (L_1 =^?_{AC} L_2) \wedge \ldots \wedge (L_1 =^?_{AC} L_n) \wedge (L_1 \succ^? D)$$

Exemple 6.3 Si nous appliquons cette règle de factorisation sur la clause $P(x_1 * y) \vee P(b * x) \vee Q(x, y) \;[\![x_1 =^?_{AC} a]\!]$, nous obtenons $P(x_1 * y) \vee Q(x, y) \;[\![c]\!]$, où c est simplifié en $(x_1 =^?_{AC} a) \wedge (x_1 * y =^?_{AC} b * x)$, car la contrainte d'ordre $(P(x_1 * y) \succ^? Q(x, y))$ est triviale et la contrainte d'unification AC $(P(x_1 * y) =^?_{AC} P(b * x))$ peut être décomposée en $(x_1 * y =^?_{AC} b * x)$. ♦

Les deux règles d'inférence suivantes sont les seules pouvant engendrer la *clause vide* (dénotée $\Box$), i.e. une clause symbolisant une contradiction, sans littéraux et avec un ensemble de contraintes satisfaisable.

Définition 6.2 (AC-Résolution contrainte)

$$\frac{A_1 \vee D_1 \;[\![c_1]\!] \qquad \neg A_2 \vee D_2 \;[\![c_2]\!]}{D_1 \vee D_2 \;[\![c_1 \wedge c_2 \wedge c_3]\!]} \quad \textit{où } c_3 \textit{ est } \; (A_1 =^?_{AC} A_2) \wedge (A_1 \succ^? D_1 \vee D_2)$$

Exemple 6.4 Une étape de résolution entre les clauses $P(x_1 * y) \vee Q(y) \;[\![x_1 =^?_{AC} a]\!]$ et $\neg P(x_2 * x) \vee Q(x) \;[\![x_2 =^?_{AC} b]\!]$ permet de déduire la clause $Q(y) \vee Q(x) \;[\![c]\!]$, où c est simplifié en $(x_1 =^?_{AC} a) \wedge (x_2 =^?_{AC} b) \wedge (x_1 * y =^?_{AC} x_2 * x)$. ♦

Définition 6.3 (AC-Réflexion contrainte)

$$\frac{\neg(l \simeq r) \vee D \;[\![c_1]\!]}{D \;[\![c_1 \wedge c_2]\!]} \quad \textit{où } c_2 \textit{ est } \; (l =^?_{AC} r) \wedge ((l \simeq r) \succ^? D)$$

Exemple 6.5 Une réflexion sur la clause $\neg(x_1 * y \simeq b * x) \vee (x * y \simeq d) \;[\![x_1 =^?_{AC} a]\!]$ permet de déduire la clause $(x * y \simeq d) \;[\![c]\!]$, où c est $(a * y =^?_{AC} b * x) \wedge ((a * y \simeq b * x) \succ^? (x * y \simeq d))$. ♦

La règle suivante est l'adaptation de la paramodulation à la stratégie contrainte. En plus de l'ajout des contraintes d'ordre et d'unification AC, le membre droit r_1 de l'équation servant au remplacement n'est pas inséré dans la clause déduite. Il est remplacé par une nouvelle variable x_{r_1} et la contrainte $(x_{r_1} =^?_{AC} r_1)$ est ajoutée.

Définition 6.4 (AC-Paramodulation contrainte)

$$\frac{(l_1 \simeq r_1) \vee D_1 \;[\![c_1]\!] \qquad L_2 \vee D_2 \;[\![c_2]\!]}{L_2[p_2 \leftarrow x_{r_1}] \vee D_1 \vee D_2 \;[\![c_1 \wedge c_2 \wedge c_3]\!]}$$

$$\begin{array}{l} \textit{où } x_{r_1} \textit{ est une nouvelle variable, } p_2 \in \mathcal{MP}os(L_2) \\ \textit{et } c_3 \textit{ est } \; (L_2|_{p_2} =^?_{AC} l_1) \wedge (x_{r_1} =^?_{AC} r_1) \\ \qquad \wedge (l_1 \succ^? r_1) \wedge ((l_1 \simeq r_1) \succ^? D_1) \wedge (L_2 \succ^? D_2) \wedge (L_2 \succeq^? (l_1 \simeq r_1)) \end{array}$$

Comme pour la règle de paramodulation sans contraintes, si le littéral L_2 est une équation négative $\neg(l_2 \simeq r_2)$, et en supposant que la position p_2 appartient à l_2, la contrainte $(l_2 \succ^? r_2)$ peut être ajoutée dans c_3.

Exemple 6.6 Une paramodulation de $(x_1 * y \simeq d) \vee (g(y) \simeq e)\ [\![x_1 =^?_{AC} a]\!]$ dans $P(x_2 * z) \vee Q(z)\ [\![x_2 =^?_{AC} b]\!]$, dans le sous-terme $x_2 * z$, produit la clause $P(x_{r_1}) \vee (g(y) \simeq e) \vee Q(z)\ [\![c]\!]$ où l'ensemble de contraintes c est simplifié en $(a * y =^?_{AC} b * z) \wedge (x_{r_1} =^?_{AC} d)$. ♦

Dans la règle suivante, un nouveau type de variable est introduit ; il s'agit de *variables d'extension*, indicées par un opérateur AC, pour représenter les sous-termes non utilisés (le contexte) lors d'une étape de paramodulation contextuelle. Le but de ces variables spéciales sera décrit dans la prochaine section.

Définition 6.5 (AC-Paramodulation contextuelle contrainte)

$$\frac{(l_1 \simeq r_1) \vee D_1\ [\![c_1]\!] \qquad L_2 \vee D_2\ [\![c_2]\!]}{L_2[p_2 \leftarrow f(x_{r_1}, z_f)] \vee D_1 \vee D_2\ [\![c_1 \wedge c_2 \wedge c_3 \wedge c_3']\!]}$$

où x_{r_1} est une nouvelle variable, $p_2 \in \mathcal{MP}os(L_2)$,
z_f est une nouvelle variable d'extension associée à l'opérateur AC f ,
et c_3 est $(L_2|_{p_2} =^?_{AC} f(l_1, z_f)) \wedge (x_{r_1} =^?_{AC} r_1) \wedge (l_1 \succ^? r_1) \wedge ((l_1 \simeq r_1) \succ^? D_1) \wedge (L_2 \succ^? D_2)$
et c_3' est la conjonction des contraintes $\mathcal{H}terms(l_1, f) \not\subseteq^? \mathcal{H}terms(t, f)$, pour chaque terme t de $\mathcal{H}terms(L_2|_{p_2}, f)$ qui n'est pas une variable d'extension pour f,
à laquelle on ajoute la condition $(\mathcal{H}ead(l_1) =^? f)$.

La condition de non inclusion des $\mathcal{H}terms$ pourrait être remplacée par la conjonction des deux problèmes de AC-disunification suivants :

$$(l_1 \neq^? t) \ \wedge\ (f(l_1, x) \neq^? t)$$

pour une nouvelle variable x. En effet, $\mathcal{H}terms(l_1, f) \not\subseteq^? \mathcal{H}terms(t, f)$ signifie que l_1 ne doit pas être AC-égal à t, et que t ne doit pas être AC-égal à un terme $f(l_1, s)$.

Exemple 6.7 Si nous appliquons une paramodulation contextuelle de la clause $(x_1 * b \simeq d) \vee (g(x_1) \simeq e)\ [\![x_1 =^?_{AC} a]\!]$ dans la clause $P(x_2 * (u * e)) \vee Q(u)\ [\![x_2 =^?_{AC} b]\!]$, dans le sous-terme $x_2 * (u * e)$, nous déduisons la clause $P(x_{r_1} * z_*) \vee (g(x_1) \simeq e) \vee Q(u)\ [\![c]\!]$. L'ensemble de contraintes c est simplifié en $(x_1 =^?_{AC} a) \wedge (a * z_* =^?_{AC} u * e) \wedge (x_{r_1} =^?_{AC} d) \wedge (\{a, b\} \not\subseteq^? \mathcal{H}terms(u, *))$. ♦

Définition 6.6 (AC-Paramodulation étendue contrainte)

$$\frac{(l_1 \simeq r_1) \vee D_1\ [\![c_1]\!] \qquad (l_2 \simeq r_2) \vee D_2\ [\![c_2]\!]}{(f(x_{r_1}, z_1) \simeq f(x_{r_2}, z_2)) \vee D_1 \vee D_2\ [\![c_1 \wedge c_2 \wedge c_3]\!]}$$

où x_{r_1}, x_{r_2}, z_1 et z_2 sont de nouvelles variables, $f \in \mathcal{F}_{AC}$
et c_3 est $(f(l_1, z_1) =^?_{AC} f(l_2, z_2)) \wedge (x_{r_1} =^?_{AC} r_1) \wedge (x_{r_2} =^?_{AC} r_2)$
$\wedge (l_1 \succ^? r_1) \wedge (l_2 \succ^? r_2) \wedge ((l_1 \simeq r_1) \succ^? D_1) \wedge ((l_2 \simeq r_2) \succ^? D_2)$
$\wedge (\mathcal{H}terms(l_1, f) \cap^? \mathcal{H}terms(l_2, f) \neq^? \emptyset)$
$\wedge (\mathcal{H}terms(z_1, f) \cap^? \mathcal{H}terms(z_2, f) =^? \emptyset)$

Dans la clause déduite, les termes r_1 et r_2 sont remplacés par des variables comme dans les deux règles précédentes. Par contre, l'utilisation de variables d'extension est inutile dans cette règle, comme nous le montrerons dans la preuve de complétude du système que nous venons de définir.

Exemple 6.8 Une paramodulation étendue entre les clauses $(x_1*b \simeq d)\vee(g(x_1)\simeq d)$ $[\![x_1 =^?_{AC} a]\!]$ et $(a * x_2 \simeq e) \vee (g(x_2)\simeq b)$ $[\![x_2 =^?_{AC} d]\!]$ permet de déduire la clause $(x_2 * x_2 \simeq x_{r_2} * z_2) \vee (g(x_1)\simeq d)\vee(g(x_2)\simeq b)$ $[\![c]\!]$, et la forme simplifiée de c est $(x_1 =^?_{AC} a)\wedge(x_2 =^?_{AC} d)\wedge(x_{r_2} =^?_{AC} e)\wedge (z_2 =^?_{AC} b)$. ♦

La section suivante est consacrée à l'explication détaillée des contraintes présentes dans ces règles d'inférences. Ensuite, nous montrerons que cette stratégie contrainte s'applique également à la stratégie de superposition.

6.3.2 A propos des conditions dans les règles d'inférence

Les règles d'inférence décrites dans la section précédente incluent les conditions énoncées dans le Chapitre 5, c'est-à-dire :

- lorsqu'une règle de paramodulation est appliquée au niveau d'un opérateur AC, il s'agit toujours d'une position maximale de cet opérateur dans le littéral considéré ($\mathcal{MP}os$),
- dans la règle de paramodulation étendue, les conditions sur les $\mathcal{H}terms$ indiquent que nous devons considérer que les termes l_1 et l_2 partagent un maximum de sous-termes : ils en possèdent au moins un, et les contextes z_1 et z_2 ne doivent pas avoir de sous-terme commun.

La principale difficulté pour intégrer les contraintes d'unification AC dans les règles porte sur le traitement du contexte lors d'une paramodulation contextuelle. Cette règle a pour but de ne remplacer qu'une partie du sous-terme dans lequel l'inférence est appliquée. C'est ce qu'exprime la contrainte d'unification : $L_2|_{p_2} =^?_{AC} f(l_1, z_f)$. Le sous-terme de L_2 à la position p_2 est divisé en deux parties : l'une doit être unifiable avec l_1, la seconde contient le contexte représenté par la variable z_f.
Pour la stratégie de paramodulation, il est bien connu que tout remplacement dans une variable est source d'inefficacité. Il est donc naturel de vérifier que, dans une étape de paramodulation contextuelle, le terme l_1 ne soit pas trouvé dans une variable de $L_2|_{p_2}$; c'est le rôle des conditions sur les $\mathcal{H}terms$. Elles précisent que les seules solutions valables sont celles où toute variable x de $\mathcal{H}terms(L_2|_{p_2}, f)$ n'est pas instanciée par un terme contenant l_1. En fait, cette condition sur les $\mathcal{H}terms$ est étendue à chaque terme de $\mathcal{H}terms(L_2|_{p_2}, f)$ pour garantir que le symbole de tête de l_1 est bien f.

Cependant, avec ces conditions sur les $\mathcal{H}terms$ et comme nous ne calculons pas les unificateurs, les termes constituant le contexte ne peuvent être isolés et ne peuvent donc pas être replacés dans la clause déduite. Cela provoque la perte de la propriété de complétude de notre système de règles d'inférence, comme le montre l'exemple suivant :

Exemple 6.9 Etant données trois clauses incohérentes (modulo les propriétés d'associativité et de commutativité de l'opérateur $*$)

$$(a * b \simeq d)\ [\![\mathbb{T}]\!]\ (1) \qquad P((a * a) * (b * b))\ [\![\mathbb{T}]\!]\ (2) \qquad \neg P(d * d)\ [\![\mathbb{T}]\!]\ (3)$$

une paramodulation contextuelle de (1) dans (2) dans le sous-terme $(a * a) * (b * b)$ produit la clause $P(x * z) \llbracket (x =^?_{AC} d) \wedge (z =^?_{AC} a * b) \rrbracket$ (4). Mais, une paramodulation contextuelle de (1) dans (4) au niveau du sous-terme $x * z$ ne peut pas être appliquée car le membre gauche $a * b$ de l'équation utilisée pour le remplacement serait trouvé dans la variable z. Ainsi, il est impossible de dériver une contradiction.
Cependant, si nous permettons cette paramodulation contextuelle de (1) dans (4), la clause déduite est $P(y * z') \llbracket (z =^?_{AC} a * b) \wedge (z' =^?_{AC} d) \wedge (y =^?_{AC} d) \rrbracket$ (5). Alors, une résolution entre (3) et (5) engendre la clause vide $\Box \llbracket (z =^?_{AC} a * b) \wedge (z' =^?_{AC} d) \rrbracket$. ♦

A partir de cet exemple, une solution pour retrouver l'indispensable propriété de complétude est de permettre des paramodulations indirectes dans des variables. Nous restreignons ces applications à des variables spéciales, représentant le contexte dans les paramodulations contextuelles. Elles sont appelées *variables d'extension* et indicées par l'opérateur AC grâce auquel elles ont été créées. Dans l'exemple précédent, les variables z et z' devraient être notées z_* et z'_*. La différenciation entre ces variables et les autres termes et variables est faite dans les contraintes de non-inclusion du membre gauche l_1 dans les $\mathcal{H}terms$ de $L_2|_{p_2}$: ces contraintes sont limitées aux termes qui ne sont pas des variables d'extension liées à l'opérateur AC considéré dans l'inférence.

Dans la règle d'AC-paramodulation contextuelle sans contraintes (Définition 5.2), vérifier que l_1 n'était pas inclus dans l'instance d'une variable permettait également de s'assurer que l'opérateur au sommet de l'instance de l_1 était bien l'opérateur AC f. Ne plus appliquer cette condition sur les $\mathcal{H}terms$ de $L_2|_{p_2}$ nous oblige à ajouter la contrainte $(\mathcal{H}ead(l_1) =^? f)$.

En résumé, *les variables d'extension sont créées par la règle de paramodulation contextuelle et utilisées uniquement par cette même règle.* Dans toutes les autres règles, ces variables doivent être considérées comme des variables normales, et la règle suivante doit être appliquée dès que possible.

Définition 6.7 (Elimination des variables d'extension) *Si une variable d'extension z_f de l'ensemble de contraintes c d'une clause $C \llbracket c \rrbracket$ n'apparaît pas dans C, alors l'application de la règle d'élimination des variables d'extension consiste à remplacer z_f par une nouvelle variable z.*

Cette définition est motivée par les propriétés suivantes :

1. Si une variable d'extension z_f apparaît seulement dans les contraintes d'une clause, elle ne peut être utilisée pour une étape de paramodulation contextuelle et a donc perdu son intérêt.

2. Toute variable d'extension z_f ne peut avoir qu'une occurrence dans une clause : elle est créée par une étape de paramodulation contextuelle et ne peut être dupliquée car cela nécessiterait l'instanciation d'une variable, ce qui est impossible (aucun unificateur n'est calculé).

3. Pour la même raison que précédemment, une variable d'extension ne peut apparaître que sous l'opérateur auquel elle est liée.

Il est à noter que cette notion de variable d'extension n'apparaît pas dans la règle de paramodulation étendue, car nous montrerons dans la preuve de complétude que tout futur remplacement dans ces contextes est inutile.

Dans les règles de *paramodulation* et de *paramodulation contextuelle*, un raffinement peut être ajouté : si le littéral L_2 est une équation négative $\neg(l_2 \simeq r_2)$ et en supposant que la position p se trouve dans le terme l_2, la contrainte d'ordre $(l_2 \succ^? r_2)$ peut être ajoutée pour spécifier que le remplacement doit s'effectuer dans le plus grand membre de l'équation.

L'ensemble de règles d'inférence ainsi défini limite fortement l'espace de recherche, grâce aux contraintes d'ordre, mais surtout grâce aux contraintes d'unification AC qui permettent de remplacer les calculs d'unificateurs principaux par un simple test de satisfaisabilité.

6.3.3 Stratégie de superposition

L'ensemble de règles d'inférence donné précédemment peut également être adapté à la stratégie de superposition. Les principales différences entre les stratégies de superposition et de paramodulation, déjà étudiées dans la Section 2.2.2, sont :

- Toute superposition dans le plus petit membre d'une équation est interdite,
- L'ordre $\succ_L$ comparant des littéraux équationnels est défini pour favoriser les équations négatives,
- Une règle supplémentaire est indispensable pour conserver la complétude ; il s'agit de la factorisation équationnelle.

Ainsi, les règles de paramodulation et de paramodulation contextuelle ne traitent plus que les remplacements dans des littéraux non-équationnels. Les remplacements dans des équations sont traités par les règles de superposition et de superposition contextuelle. Comme pour la stratégie non contrainte, la règle de superposition étendue est identique à la paramodulation étendue.

Cependant, une différence importante entre les stratégies de paramodulation et de superposition lorsque les contraintes d'unification sont introduites est que le membre droit d'une équation utilisée pour un remplacement doit être placé dans la clause déduite, et non remplacé par une nouvelle variable comme c'était le cas pour la stratégie de paramodulation. Ceci est dû à l'interdiction de réduire ce membre droit par les règles de superposition. Donc, dans les deux règles de paramodulation et les trois règles de superposition le membre droit de l'équation n'est plus abstrait dans la clause déduite.

Les contraintes représentant la maximalité d'un littéral équationnel $(l \simeq r)$ ou $\neg(l \simeq r)$ (où l est supposé plus grand que r) dans une clause sont interprétées ainsi :

- $(l \simeq r) \succ_L^? D$ se traduit par $(l \simeq r) \succ^? L$ pour tout littéral positif L de D et, $l \succ^? s$ et $l \succ^? t$ pour tout littéral négatif $\neg(s \simeq t)$ de D.
- $\neg(l \simeq r) \succ_L^? D$ se traduit par $(l \simeq r) \succ^? L$ pour tout littéral négatif L de D et, $(l \simeq l) \succ^? L$ pour tout littéral positif L de D.

La règle de factorisation équationnelle contrainte, qui a pour but d'engendrer des clauses ne possédant pas plusieurs équations positives de membres gauches AC-unifiables, est définie par :

Définition 6.8 (AC-Factorisation équationnelle contrainte)

$$\frac{(l_1 \simeq r_1) \vee (l_2 \simeq r_2) \vee D \ \llbracket c_1 \rrbracket}{(l_2 \simeq r_2) \vee \neg(r_1 \simeq r_2) \vee D \ \llbracket c_1 \wedge c_2 \rrbracket}$$

où c_2 *est* $(l_1 =^?_{AC} l_2) \wedge (l_1 \succ^? r_1) \wedge ((l_1 \simeq r_1) \succ^?_L (l_2 \simeq r_2) \vee D)$

R. Nieuwenhuis et A. Rubio ont défini un système de règles d'inférence similaire pour la stratégie de superposition [NR94, Rub94]. Nos deux systèmes ne calculent pas les unificateurs AC et permettent d'effectuer indirectement des remplacements dans une variable, dans la règle de superposition contextuelle. Mais, ils n'ont pas restreint cela à ce que nous appelons des variables d'extension. Ainsi, ils permettent des étapes de déduction inutiles comme nous le montrons dans l'exemple suivant.

Exemple 6.10 Etant donné le système

$$((a * a) * x \simeq a * x) \ \llbracket \mathbb{T} \rrbracket \ (1) \qquad (b * c \simeq c) \ \llbracket \mathbb{T} \rrbracket \ (2) \qquad P((a * b) * (a * c)) \ \llbracket \mathbb{T} \rrbracket \ (3)$$

la clause $P(a * x) \llbracket x =^?_{AC} b * c \rrbracket$ (4) est déduite par une superposition de (1) dans (3). Alors, R. Nieuwenhuis et A. Rubio permettent d'appliquer une superposition contextuelle de (2) dans (4), ce qui engendre la clause $P(c * y) \llbracket y =^?_{AC} a \rrbracket$ (5). En oubliant que nous n'appliquons les remplacements qu'aux positions maximales des opérateurs AC, cette dernière étape revient à une superposition de (2) dans la variable x, ce qui est bien connu pour être inutile.
Avec nos contraintes, nous interdisons cela car une clause similaire à la clause (5) peut être dérivée par une superposition contextuelle de (2) dans (3), $P(c * z_*) \llbracket z_* =^?_{AC} a * a \rrbracket$ (4′), suivie d'une superposition de (1) dans (4′), $P(a * x) \llbracket x =^?_{AC} c \rrbracket$ (5′).
Les clauses (5) et (5′) paraissent quelque peu différentes, mais en propageant les sous-termes contraints dans les deux séquences de déduction, les termes a et c sont tous les deux bloqués et la clause déduite dans les deux cas devrait être $P(y * x) \llbracket (x =^?_{AC} c) \wedge (y =^?_{AC} a) \rrbracket$ (5″). ♦

De plus, R. Nieuwenhuis et A. Rubio n'ont aucune restriction sur les $\mathcal{H}terms$ dans leurs règles de superposition contextuelle et étendue.

Au vu des définitions des stratégies de paramodulation et de superposition, nous pouvons remarquer que *la différence entre ces deux stratégies est assez faible* : la stratégie de paramodulation permet des remplacements dans le membre droit d'une équation, mais bloque ce membre droit lorsque l'équation sert pour effectuer une paramodulation ; la stratégie de superposition interdit tout remplacement dans le membre droit d'une équation, mais permet des remplacements dans ce terme, à l'intérieur d'une clause déduite par superposition depuis cette équation. Nous détaillerons un peu plus cette comparaison dans la Section 7.3.2.

6.3.4 Stratégie positive

Nous avons vu dans la Section 2.3 une stratégie appelée *positive*. Son principe est le suivant :

- Si une inférence utilise un littéral positif dans une clause, alors cette clause doit être positive, c'est-à-dire ne contenir aucun littéral négatif, est le littéral considéré doit être maximal dans la clause.

- Si une inférence utilise un littéral négatif dans une clause, alors ce littéral doit être maximal vis-à-vis des autres littéraux négatifs de cette clause.

Les modifications apportées aux systèmes d'inférence basés sur les stratégies de paramodulation et de superposition sont entièrement compatibles avec l'utilisation de contraintes symboliques. Ainsi, nous ne détaillons pas les règles d'inférence obtenues, mais insistons sur le fait que cette stratégie est très importante.

6.4 Règles de simplification

Dans cette section, nous définissons des règles de simplification compatibles avec la stratégie contrainte décrite dans ce chapitre. Comme l'ont montré C. Lynch et W. Snyder [LS93], la notion de redondance en présence de contraintes d'unification est très complexe. Les conditions d'application des règles présentées ci-dessous sont assez fortes, car une simplification ne peut être appliquée sur une clause contrainte que si toutes ses instances peuvent être simplifiées dans les mêmes conditions ; nous donnons donc d'autres versions de ces règles qui ne s'appliquent que sur une instance d'une clause. Ces dernières sont essentielles pour l'efficacité d'un démonstrateur de théorèmes.

6.4.1 Subsomption contrainte

Le but de la subsomption est d'éliminer une clause redondante. Sa définition classique est *une clause C_1 subsume une clause C_2 s'il existe une substitution ρ telle que $C_1\rho \Subset C_2$*. Comme nous l'avons vu dans la Section 2.4.1, les clauses sont considérées comme des multi-ensembles de littéraux, ce qui signifie que C_1 n'a pas plus de littéraux que C_2. L'extension de cette définition au cas contraint se fait ainsi :

Définition 6.9 (AC-Subsomption contrainte) *Une clause contrainte $C_1 [\![c_1]\!]$ subsume une clause contrainte $C_2 [\![c_2]\!]$ s'il existe une solution σ_1 de c_1 telle que : pour toute solution σ_2 de c_2, il existe une substitution ρ vérifiant que $C_1\sigma_1\rho$ est un sous-ensemble de $C_2\sigma_2$, i.e.*

$$\exists \sigma_1 \in \mathcal{S}ol(c_1),\ \forall \sigma_2 \in \mathcal{S}ol(c_2),\ \exists \rho,\ C_1\sigma_1\rho \Subset_{AC} C_2\sigma_2$$

La stratégie contrainte provoque la perte de la complétude, comme cela a été montré par L. Bachmair, H. Ganzinger, C. Lynch et W. Snyder dans [BGLS92]. Cela peut se produire lorsqu'un terme $x\sigma_1\rho$ associé à une variable x du domaine de σ_1 n'est pas inclus (au sens de la relation de sous-terme) dans un terme du co-domaine de σ_2, noté $\mathcal{C}o\mathcal{D}om(\sigma_2)$. Nous devons donc vérifier que cette condition est bien satisfaite.

Définition 6.10 (Application de la subsomption) *Si une clause contrainte $C_1 [\![c_1]\!]$ subsume une clause contrainte $C_2 [\![c_2]\!]$ et si les substitutions σ_1, σ_2 et ρ définies dans la Définition 6.9 satisfont*

$$\mathcal{C}o\mathcal{D}om(\sigma_1\rho)\ \trianglelefteq_{AC}\ \mathcal{C}o\mathcal{D}om(\sigma_2)$$

alors la clause $C_2 [\![c_2]\!]$ est supprimée.

Notons que $\mathcal{C}o\mathcal{D}om(\sigma_1\rho)$ désigne l'ensemble des termes du co-domaine de σ_1 instanciés par ρ. Cette définition doit être quelque peu modifiée pour tenir compte des variables d'extension introduites par les règles de paramodulation et de superposition contextuelles contraintes. Cette modification intervient dans le calcul du co-domaine des σ_i : si une variable d'extension z_f appartient au domaine d'un σ_i, et apparaît sous l'opérateur AC f dans C_i, l'ensemble $\mathcal{H}terms(z_f\sigma_i, f)$ doit remplacer $z_f\sigma_i$ dans $\mathcal{C}o\mathcal{D}om(\sigma_i)$.
A partir de maintenant, à chaque fois que nous parlerons du co-domaine d'une substitution, nous ferons référence à la définition précédente.

Comme pour le cas non contraint, l'application de cette règle de subsomption doit être accompagnée d'une stratégie de suppression des clauses les plus récemment engendrées en priorité.

6.4.2 Simplification contrainte

Une étape de simplification est en fait une étape de réécriture : une clause $L_2 \vee D_2$ est simplifiée par une équation $(l_1 \simeq r_1)$ à la position p_2 du littéral L_2 s'il existe une substitution ρ telle que : $L_2|_{p_2} = l_1\rho$ et $l_1\rho \succ r_1\rho$. Cette définition est étendue au cas contraint et pour les théories associatives et commutatives par :

Définition 6.11 (AC-Simplification contrainte) *Une clause $(l_1 \simeq r_1) \vee D_1[\![c_1]\!]$ simplifie une clause $L_2 \vee D_2[\![c_2]\!]$ à une position $p_2 \in \mathcal{MP}os(L_2)$ si :*

$$\exists \sigma_1 \in \mathcal{S}ol(c_1),\ \exists \rho_1,\ \forall \sigma_2 \in \mathcal{S}ol(c_2),\ \exists \rho'_1 \in AC\text{-}match(l_1\sigma_1\rho_1, L_2|_{p_2}\sigma_2),\quad \textit{tels que}$$

$$\begin{cases} \mathcal{D}om(\sigma_1) \cap \mathcal{D}om(\rho_1) = \emptyset \\ D_2\sigma_2 =_{AC} D'_2\sigma_2 \vee D''_2\sigma_2 \quad \textit{et} \quad D_1\sigma_1\rho_1\rho'_1 \subseteq_{AC} D'_2\sigma_2 \\ l_1\sigma_1\rho_1\rho'_1 \succ r_1\sigma_1\rho_1\rho'_1 \\ (l_1 \simeq r_1)\sigma_1\rho_1\rho'_1 \prec L_2\sigma_2 \vee D''_2\sigma_2 \end{cases}$$

Nous venons donc de définir une simplification conditionnelle, car la clause permettant la simplification peut contenir d'autres littéraux que l'équation $(l_1 \simeq r_1)$, à condition que ceux-ci soient également présents dans la clause simplifiée. L'énoncé de cette règle de simplification est assez complexe, et plus particulièrement le choix des substitutions, car il nous faut différencier la partie du filtre qui instanciera la clause déduite (ρ_1) de celle qui apparaîtra dans la partie contrainte $(\sigma_1\rho'_1)$, comme nous le verrons un peu plus loin.

Il est à noter que nous avons utilisé la notion d'inclusion entre ensembles et non multi-ensembles. Cela signifie qu'un littéral peut apparaître plusieurs fois dans $D_1\sigma_1\rho$ et n'apparaître qu'une fois dans $D'_2\sigma_2$. De plus, un littéral de C_2, n'apparaissant pas dans C_1, doit être plus grand que l'équation $(l_1 \simeq r_1)$.

Définition 6.12 (AC-Simplification contextuelle contrainte) *Une simplification contextuelle de $(l_1 \simeq r_1) \vee D_1[\![c_1]\!]$ dans $L_2 \vee D_2[\![c_2]\!]$ à une position $p_2 \in \mathcal{MP}os(L_2)$ est possible si :*

$$\exists \sigma_1 \in \mathcal{S}ol(c_1),\ \exists \rho_1,\ \forall \sigma_2 \in \mathcal{S}ol(c_2),\ \exists \rho'_1 \in AC\text{-}match(f(l_1\sigma_1, z)\rho_1, L_2|_{p_2}\sigma_2),\quad \textit{tels que}$$

$$\begin{cases} z \textit{ est une nouvelle variable et } \mathcal{H}ead(l_1\sigma_1\rho_1\rho'_1) = f \in \mathcal{F}_{AC} \\ \mathcal{D}om(\sigma_1) \cap \mathcal{D}om(\rho_1) = \emptyset \\ D_1\sigma_1\rho_1\rho'_1 \subseteq_{AC} D_2\sigma_2 \\ l_1\sigma_1\rho_1\rho'_1 \succ r_1\sigma_1\rho_1\rho'_1 \end{cases}$$

Cette règle est la contrepartie de la paramodulation (ou superposition) contextuelle. Mais ici, nous ne pouvons pas utiliser la notion de variable d'extension pour exprimer le contexte, car ce dernier (représenté par $z\rho_1$) doit être replacé dans la clause déduite.
Contrairement à la règle de simplification contrainte, il est inutile de vérifier qu'un atome de $L_2\sigma_2 \vee D_2\sigma_2$ est plus grand que $(l_1 \simeq r_1)\sigma_1\rho$, car $L_2\sigma_2$ satisfait trivialement cette condition.

L'exemple suivant montre la perte de complétude si la notion de variable d'extension était utilisée dans la règle de simplification contextuelle.

Exemple 6.11 Etant donné le système AC-incohérent suivant, où f est AC et $c \succ d$,

$$\begin{array}{llll} P(f(f(a,d),f(b,c))) \ [\![\mathbb{T}]\!] & (1) & (f(a,b) \simeq b) \ [\![\mathbb{T}]\!] & (2) \\ \neg P(f(f(d,d),b)) \ [\![\mathbb{T}]\!] & (3) & (c \simeq d) \ [\![\mathbb{T}]\!] & (4) \end{array}$$

La clause (1) est simplifiée par (2) en $P(f(b,z_f)) \ [\![z_f =_{AC} f(c,d)]\!]$ (1). Mais alors, il n'y a aucun moyen de prouver l'incohérence dans les théories AC. Avec la définition précédente, la clause simplifiée est $P(f(b,f(c,d))) \ [\![\mathbb{T}]\!]$ (1), laquelle peut à nouveau être simplifiée par (4) en $P(f(b,f(d,d))) \ [\![\mathbb{T}]\!]$ (1). Une étape de résolution entre (1) et (3) dérive une contradiction. ♦

En reprenant l'exemple donné au début de cette section, l'application classique de ces règles de simplification consiste à remplacer la clause $L_2 \vee D_2$ par $L_2[r_1\rho]_{p_2} \vee D_2$. Mais, nous rencontrons le même problème que pour la subsomption ; la condition sur les co-domaines des substitutions est également nécessaire.

Définition 6.13 (Application des simplifications contraintes) *Si une clause $C_1 [\![c_1]\!]$ simplifie une clause $C_2 [\![c_2]\!]$ à une position p_2 et si $\mathcal{C}o\mathcal{D}om(\rho'_1) \cup \mathcal{C}o\mathcal{D}om(\sigma_1\rho'_1) \trianglelefteq_{AC} \mathcal{C}o\mathcal{D}om(\sigma_2)$, la clause $C_2 [\![t_2]\!]$ est remplacée par $L_2[r_1\rho_1]_{p_2} \vee D_2 \ [\![c_2 \wedge (l_1\sigma_1\rho_1 =^?_{AC} L_2|_{p_2}) \wedge \sigma_1|_{\mathcal{V}ar(r_1)}]\!]$ pour la règle de simplification, et $L_2[f(r_1,z)\rho_1]_{p_2} \vee D_2 \ [\![c_2 \wedge (f(l_1\sigma_1,z)\rho_1 =^?_{AC} L_2|_{p_2}) \wedge \sigma_1|_{\mathcal{V}ar(r_1)}]\!]$ pour la règle de simplification contextuelle.*

Cette définition de l'application des règles de simplification montre l'intérêt des différentes substitutions calculées. La substitution σ_1 représente une solution de l'ensemble de contraintes c_1 associé à la clause C_1. Ensuite, pour chaque solution σ_2 de c_2, il faut trouver un filtre ρ de $l_1\sigma_1$ dans $L_2|_{p_2}\sigma_2$. Nous avons vu que ce filtre devrait être appliqué dans la clause déduite sur r_1. Mais, comme nous vérifions que tout terme du co-domaine de σ_1 (auquel on applique ρ) est inclus dans le co-domaine de σ_2, il est inutile d'appliquer cette substitution $\sigma_1\rho$ sur r_1. Il s'agit en quelque sorte d'une propagation de contraintes ; nous gardons une trace de ces instanciations en ajoutant $\bigwedge_{x \in \mathcal{V}ar(r_1)} (x =^?_{AC} x\sigma_1)$ dans les contraintes de la clause déduite, sous la notation $\sigma_1|_{\mathcal{V}ar(r_1)}$.
Il reste donc à considérer les variables de r_1 instanciées directement par le filtre ρ. Nous décomposons la substitution ρ en deux substitutions ρ_1 et ρ'_1 : le domaine de ρ_1 est constitué des variables x de r_1 instanciées directement par ρ et telles que $x\rho$ n'est pas dans $\mathcal{C}o\mathcal{D}om(\sigma_2)$; $x\rho =_{AC} x\rho_1\rho'_1$, et ρ'_1 extrait les plus gros sous-termes de $x\rho$ qui appartiennent à $\mathcal{C}o\mathcal{D}om(\sigma_2)$.

Le raisonnement que nous venons de développer a été appliqué pour un filtre ρ, calculé pour une substitution σ_2. Il ne faut pas oublier cependant que l'ensemble de contraintes c_2 peut admettre plusieurs solutions σ_2. Une première possibilité serait de n'appliquer une règle de simplification que si chaque solution σ_2 engendre le même filtre ρ, ce qui est très restrictif. Nous avons opté pour une définition plus générale, ce qui explique pourquoi nous ne mentionnons pas l'existence de ce filtre ρ dans les règles. La substitution ρ_1 apparaissant avant le choix de σ_2 représente en fait la partie commune de tous les filtres ρ. La substitution ρ'_1 dépend de σ_2 et associe à une variable un terme du co-domaine de σ_2.

Pour la règle de simplification contextuelle, la substitution ρ_1 sert en plus à identifier le contexte de la simplification. Celui-ci doit donc être similaire pour chaque solution σ_2.

6.4.3 Simplification par transformation ou application locale

Nous avons vu que des conditions sur les co-domaines des substitutions devaient être respectées pour conserver la complétude de nos règles d'inférence. Mais, elles limitent énormément les simplifications. Pour compenser cela, nous donnons ci-dessous quelques modifications à appliquer sur les définitions de la subsomption et des simplifications pour pouvoir les appliquer même si l'inclusion des co-domaines n'est pas vérifiée. Il s'agit d'instancier partiellement la clause C_1 avec la substitution σ_1 jusqu'à ce que σ_1 soit réduit à une substitution σ'_1 respectant la condition d'inclusion des co-domaines. Cette solution a été proposée par L. Bachmair, H. Ganzinger, C. Lynch et W. Snyder dans [BGLS92].
Cependant, ce n'est pas suffisant car l'ensemble de contraintes c_1 peut avoir d'autres solutions que σ_1, et instancier partiellement C_1 avec σ_1 peut provoquer la perte des autres solutions.

Nous proposons donc de créer une nouvelle clause, qui est en fait l'instance de $C_1 [\![\sigma_1]\!]$, et de transformer $C_1 [\![c_1]\!]$ en $C_1 [\![c_1 - \sigma_1]\!]$ pour préciser que σ_1 ne doit plus être considérée comme une solution de c_1.

La notation $c_1 - \sigma_1$ peut être interprétée soit par de la AC-disunification, c'est-à-dire en ajoutant l'ensemble $\{\bigvee_{x \in \mathcal{D}om(\sigma_1)} x \neq^? x\sigma_1\}$ à c_1, soit par des contraintes d'ordre forçant les variables x du domaine de σ_1 à prendre d'autres valeurs que $x\sigma_1$, i.e. $\{\bigvee_{x \in \mathcal{D}om(\sigma_1)} (x \prec^? x\sigma_1) \vee (x \succ^? x\sigma_1)\}$. Mais cette seconde technique est difficile à gérer en raison des nouvelles variables introduites par l'algorithme d'unification AC. Notons que des contraintes de disunification ont déjà été utilisées par T.B. Baird, G.E. Peterson et R.W. Wilkerson [BPW89] ainsi que par J.-P. Jouannaud et C. Marché [JM90] pour effectuer de la complétion modulo des théories associatives, commutatives avec élément neutre. Ces contraintes permettent de limiter les applications de certaines règles de réécriture comme l'opposé d'une somme dans la théorie des groupes Abéliens, grâce à l'énoncé suivant de cette règle :

$$\text{si} \quad x \neq 0 \quad \text{et} \quad y \neq 0 \quad \text{alors} \quad -(x+y) \to (-x) + (-y)$$

où $+$ est AC et 0 est son élément neutre.

Ainsi, pour les règles de subsomption et de simplification, si la condition d'inclusion des termes du co-domaine de $\sigma_1\rho$ dans ceux de σ_2 n'est pas respectée, il est quand même possible d'appliquer la simplification mais à condition d'effectuer les actions suivantes :

- Créer la nouvelle clause $C_1\tau [\![\sigma'_1]\!]$, où σ'_1 est la plus grande substitution et τ la plus petite substitution vérifiant $\sigma_1 =_{AC} \tau\sigma'_1$ et $\mathcal{C}o\mathcal{D}om(\sigma'_1\rho) \trianglelefteq_{AC} \mathcal{C}o\mathcal{D}om(\sigma_2)$,
- Transformer la clause $C_1 [\![c_1]\!]$ en $C_1 [\![c_1 - \sigma_1]\!]$,
- Pour les règles de simplification et de simplification contextuelle, instancier dans la clause déduite le membre droit de l'équation r_1 par τ et utiliser σ'_1 au lieu de σ_1 dans les contraintes.

Nous appellerons ces règles des **règles de simplification par transformation**.

Les règles de simplification définies jusqu'à présent s'appliquent lorsque toutes les instances d'une clause peuvent être simplifiées par la même clause et à la même position. Ceci est très restrictif et nous pouvons imaginer que si seulement l'une des instances d'une clause peut être simplifiée, il serait quand même intéressant d'effectuer cette simplification. C'est ce que nous appelons des **simplifications locales**. L'idée de telles règles a été proposée par C. Kirchner, H. Kirchner et M. Rusinowitch dans [KKR90], puis développée par C. Lynch et W. Snyder dans [LS93].
Elles se traduisent par l'utilisation de $\exists \sigma_2 \in \mathcal{S}ol(c_2)$ au lieu de la quantification universelle. L'application d'une simplification locale consiste à créer une nouvelle clause, le résultat de l'étape de simplification, puis à transformer la clause $L_2 \vee D_2 \ [\![c_2]\!]$ en spécifiant que σ_2 n'est plus une solution de c_2.

Exemple 6.12 Une simplification locale de la clause $(a * b \simeq d) [\![\mathbb{T}]\!]$ (1) dans la clause $P(a * y, z) [\![y * z =^?_{AC} e * b]\!]$ (2), avec la substitution $\sigma_2 = \{y \mapsto b,\ z \mapsto e\}$, provoque la création de la clause $P(d, z) [\![(y =^?_{AC} b) \wedge (z =^?_{AC} e)]\!]$ (3), et le remplacement de la clause (2) par $P(a * y, z) [\![c]\!]$, où c est l'ensemble de contraintes $(y * z =^?_{AC} e * b) \wedge \{(y \neq^? b) \vee (z \neq^? e)\}$, qui se simplifie en $(y =^?_{AC} e) \wedge (z =^?_{AC} b)$. ♦

Avec le même principe, il est possible d'appliquer des subsomptions locales, qui consistent à ôter une solution d'un ensemble de contraintes si elle est subsumée.

6.4.4 Autres règles de simplification

Il existe une simplification très importante apportée par la stratégie contrainte ; elle s'énonce ainsi : étant donnée une clause contrainte $C [\![c]\!]$ et une solution σ de c, *si la clause $C [\![\sigma]\!]$ est simplifiable dans un terme de l'ensemble $\mathcal{C}o\mathcal{D}om(\sigma)$, σ est une solution redondante*. En effet, dans notre preuve de complétude de la stratégie contrainte, nous montrons que les instances closes de clauses, réductibles dans leur partie contraintes, sont inutiles. Cette propriété peut être appliquée localement ou bien sur une clause entière si toutes ses instances sont simplifiables dans leur partie contrainte, même par des clauses différentes.

Enfin, les autres règles de simplifications classiques, comme l'élimination de clauses triviales, la simplification clausale, la réflexion triviale ou les simplifications par remplacement peuvent être adaptées à la stratégie contrainte et appliquées localement ou non.

6.5 Preuve de complétude

Enonçons le théorème de complétude réfutationnelle des systèmes de règles d'inférence définis dans ce chapitre, en présence des règles de simplification.

Théorème 6.14 (Complétude de la déduction contrainte) *Soit S un ensemble AC-incohérent de clauses dont la partie contrainte est vide ($\mathbb{T}$). Alors, toute dérivation équitable de S engendre la clause vide.*

Ce théorème s'applique donc pour les stratégies de paramodulation et de superposition. Sa preuve ne diffère que très peu de la preuve dans le cas non contraint (Chapitre 4), adaptée aux théories AC. En effet, comme nous allons le voir dans ce qui suit, la preuve de complétude de la déduction contrainte impose de ne considérer que des substitutions irréductibles pour instancier les clauses. Or, cette condition est déjà imposée dans la stratégie non contrainte par le Lemme de Relèvement (Section 4.2). Aussi, nous ne détaillerons pas à nouveau ces preuves, mais nous nous contenterons de montrer les quelques propriétés supplémentaires nécessaires.

Dans l'énoncé de ce théorème, nous avons précisé que toute clause initiale doit contenir une partie contrainte vide. Cette condition ne s'applique que si le prédicat d'égalité fait partie de l'ensemble des opérateurs, comme l'ont montré R. Nieuwenhuis et A. Rubio [NR92b]. La présence de contraintes symboliques dans les clauses initiales, combinée avec la présence du prédicat d'égalité impose l'utilisation de techniques de surréduction comme celles décrites par J. Chabin [Cha94] et H. Kirchner et C. Ringeissen [KR94].

6.5.1 Technique des arbres sémantiques

La principale modification intervient dans la définition de l'arbre sémantique cohérent. Il faut en effet tenir compte d'une nouvelle condition : étant donnée un clause contrainte $C [\![c]\!]$, interdire tout remplacement dans sa partie contrainte signifie qu'il ne faut retenir que les substitutions closes σ, solution de c, dont le co-domaine est irréductible.

Définition 6.15 (Arbre sémantique cohérent) *Soit $\mathcal{T}$ un arbre sémantique transfini. Nous définissons l'arbre sémantique cohérent associé à un ensemble de clauses contraintes S, et noté $MCT(S)$, le sous-arbre maximal de $\mathcal{T}$ tel que, pour chaque nœud I dans $MCT(S)$,*

- *soit I est AC-consistant et, pour toute clause C', AC-égale à une instance close $C\sigma$ d'une clause $C [\![c]\!]$ de S, telle que σ appartient à $\mathcal{S}ol(c)$ et les atomes de C' appartiennent au domaine de I, $I(C') = V$ si*
  1. *pour chaque variable x d'un atome A de C,*
     (a) *si x est une variable d'extension liée à un opérateur AC f, alors pour chaque terme u, AC-égal à un élément de $\mathcal{H}terms(x\sigma, f)$, il n'existe pas d'équation $(l \simeq r)$ plus petite que A (où $l \succ r$) telle que $u \longrightarrow_I^{l\simeq r} u[r]$,*
     (b) *si x n'est pas une variable d'extension, alors pour chaque terme u, AC-égal à $x\sigma$, il n'existe pas d'équation $(l \simeq r)$ plus petite que A (où $l \succ r$) telle que $u \longrightarrow_I^{l\simeq r} u[r]$.*
- *soit I est AC-inconsistant, mais satisfait toute équation $(u \simeq u')$ de son domaine telle que $u =_{AC} u'$.*

La condition précédente 1 peut être interprétée comme l'irréductibilité de la substitution σ. Un nœud d'échec se définit alors par :

Définition 6.16 (Nœud d'échec) *Un nœud d'échec I est*

- *soit une interprétation AC-consistante falsifiant une clause C', AC-égale à une instance close d'une clause $C [\![c]\!]$ de S, où C' satisfait la condition précédente 1 ; I est alors étiqueté par C'.*

- *soit une interprétation AC-inconsistante falsifiant une équation $(u \simeq u')$ où $u =_{AC} u'$; I est dite étiqueté par $(u \simeq u')$.*

Si J est le dernier nœud d'un chemin de $MCT(S)$, alors toute extension de J est un nœud d'échec. Un *chemin maximal* dans $MCT(S)$ est un chemin dont les extensions ne sont pas dans $MCT(S)$; ces extensions sont donc des nœuds d'échec.

Le lemme suivant énonce que, si S est un ensemble AC-incohérent de clauses contraintes, aucun chemin de $MCT(S)$ n'est défini sur toute la base de Herbrand.

Lemme 6.17 *Soit S un ensemble de clauses contraintes ; S est AC-incohérent si et seulement si tout chemin maximal de $MCT(S)$ s'étend en un nœud d'échec.*

Pour montrer que les systèmes de règles d'inférence INF définis dans ce chapitre sont réfutationnellement complets en présence des règles de simplification (Théorème 6.14), nous raisonnons par contradiction : supposons que S_0 est un ensemble AC-incohérent de clauses dont les contraintes sont vides ($\mathbb{T}$), et que $S_0, S_1, S_2, \ldots$ est une dérivation équitable (Définition 2.20) ; S^* dénotant $\bigcup_{i \geq 0} S_i$, supposons que S^* ne contient pas la clause vide. L'arbre sémantique cohérent $MCT(S^*)$ n'est donc pas vide. Définissons les ensembles suivants :

$$\begin{array}{rcl} \mathcal{GS} & = & \{\ C' [\![\sigma]\!] \mid \exists C' [\![c]\!] \in S^*,\ \sigma \text{ close } \in \mathcal{S}ol(c)\ \} \\ \mathcal{AC} & = & \{\ (u \simeq u') [\![\mathbb{T}]\!] \mid u, u' \in \mathcal{T}(\mathcal{F}),\ u =_{AC} u'\ \} \end{array}$$

$\mathcal{GS}$ est l'ensemble de toutes les instances closes des clauses de S^*, et $\mathcal{AC}$ est l'ensemble de toutes les équations closes triviales pour les théories AC. Ces deux ensembles contiennent en fait toutes les clauses pouvant étiqueter un nœud d'échec au bout d'un chemin de $MCT(S^*)$. Par définition d'un arbre sémantique cohérent, $MCT(\mathcal{GS})$ est équivalent à $MCT(S^*)$; donc, $MCT(\mathcal{GS})$ est également non vide.

Comme pour la preuve de complétude dans le cas non contraint, nous avons besoin de définir des nœuds de quasi-échec (Définition 4.14), ainsi que des nœuds d'échec distants pour la stratégie de superposition (Définition 4.20). La branche droite Q_γ de l'arbre sémantique cohérent $MCT(\mathcal{GS})$ est construite comme dans les Définitions 4.15 et 4.21.
Dans ce qui suit, nous montrons en même temps la complétude des stratégies de paramodulation et de superposition.

6.5.2 Preuve de complétude

Pour montrer que l'arbre sémantique cohérent $MCT(\mathcal{GS})$ est vide, il faut montrer que la branche droite Q_γ est vide. Mais dans un premier temps, il faut s'assurer que les nœuds d'échec le long de cette branche droite sont bien persistants.

Proposition 6.18 *Soit K un nœud d'échec, extension d'un nœud Q_α de la branche droite, ou plus précisément extension droite de Q_α ou extension de Q_γ. Si K est étiqueté par une clause de $\mathcal{GS}$, alors K est un nœud d'échec persistant.*

La preuve de cette proposition est identique à celle de la Proposition 4.30. Le seul problème qui pourrait se poser vient de la définition des nœuds d'échec, et plus précisément de l'irréductibilité de la substitution. Cependant, nos règles de simplification (Section 6.4) sont définies de telle manière que si une clause $C_2 [\![\sigma_2]\!]$ étiquetant un nœud d'échec K est subsumée par une clause $C_1 [\![\sigma_1 \rho_1]\!]$, où σ_1 désigne une substitution close solution de l'ensemble de contraintes associé à C_1 et ρ_1 désigne une substitution close instanciant les variables de C_1 qui ne sont pas contraintes,

- soit tous les termes du co-domaine de σ_1 sont également dans le co-domaine de σ_2,
- soit la clause $C_1 [\![\sigma_1 \rho_1]\!]$ est transformée en une clause $C_1 \sigma_1' [\![\sigma_1'' \rho_1]\!]$ (où $\sigma_1' \sigma_1'' =_{AC} \sigma_1$) afin que cette propriété d'inclusion des co-domaines soit satisfaite par les substitutions σ_1'' et σ_2.

Nous avons isolé la substitution ρ_1 de la solution σ_1 des contraintes car, dans la règle de subsomption, l'inclusion des co-domaines n'est testée que sur les variables apparaissant dans les contraintes d'unification. Cependant, nous pouvons remarquer que cette substitution ρ_1 est également irréductible, car sinon la clause $C_1 \sigma_1 \rho_1'$ (où ρ_1' désigne la substitution irréductible issue de ρ_1) devrait étiqueter un nœud d'échec au niveau ou au-dessus de K.

Nos autres règles de simplification vérifient que la clause $C_2' [\![\sigma_2']\!]$ résultant de la simplification de $C_2 [\![\sigma_2]\!]$ satisfait : $\mathcal{C}o\mathcal{D}om(\sigma_2') \trianglelefteq_{AC} \mathcal{C}o\mathcal{D}om(\sigma_2)$. Si cette propriété n'est pas vérifiée, la clause $C_1 [\![\sigma_1 \rho_1]\!]$ utilisée pour la simplification est transformée en $C_1 \sigma_1' [\![\sigma_1'' \rho_1]\!]$, comme pour la subsomption.

Proposition 6.19 *Si $MCT(\mathcal{GS})$ n'est pas vide, alors sa branche droite Q_γ est non vide et AC-consistante.*

La preuve de cette proposition est identique à celle du cas non contraint (Propositions 4.18 et 4.24). Il reste cependant plusieurs cas à vérifier. La preuve faisant appel à l'application de règles d'inférence, il faut s'assurer qu'à chaque fois la clause déduite étiquette bien un nœud d'échec coupant la branche droite. La preuve montre que la clause déduite est falsifiée par Q_α, mais il faut vérifier en plus que la substitution représentant ses contraintes est irréductible (condition 1 dans la Définition 6.15). Montrons dans un premier temps cette propriété pour la règle de paramodulation étendue contrainte.

Proposition 6.20 *Soit Q_β une interprétation AC-consistante, restriction de la branche droite Q_γ. Soit Q_{α_1} et Q_{α_2} deux restrictions de Q_β, admettant un nœud d'échec (ou d'échec distant) comme extension droite. Si une paramodulation étendue est appliquée entre les clauses étiquetant ces nœuds d'échec, et si la clause déduite est falsifiée par Q_β, alors elle étiquette un nœud d'échec coupant la branche droite.*

Preuve : *Soient $C_1 = (l_1 \simeq r_1) \vee D_1$ et $C_2 = (l_2 \simeq r_2) \vee D_2$ les clauses étiquetant les nœuds d'échec, extensions droites de Q_{α_1} et Q_{α_2}. Pour la stratégie de paramodulation, cela signifie que $(l_1 \simeq r_1)$ et $(l_2 \simeq r_2)$ sont irréductibles ; pour la stratégie de superposition, cela signifie que les termes l_1 et l_2 sont irréductibles.*
Soient $C_1' = (l_1' \simeq r_1') \vee D_1' [\![c_1]\!]$ et $C_2' = (l_2' \simeq r_2') \vee D_2' [\![c_2]\!]$ les deux clauses contraintes de S^* telle que : C_1 est AC-égal à $C_1' \sigma_1$ et C_2 est AC-égal à $C_2' \sigma_2$, où les substitutions σ_1 et σ_2 sont closes, irréductibles (condition 1 dans la définition de MCT), et solution de c_1 et c_2 respectivement. Une étape de paramodulation étendue entre les clauses closes $C_1' [\![\sigma_1]\!]$ et $C_2' [\![\sigma_2]\!]$ engendre une nouvelle clause contrainte $C' [\![\sigma]\!]$.
Si nous supposons que l'interprétation Q_β falsifie une clause C, AC-égale à $C'\sigma$, la seule

condition à vérifier pour qu'un nœud d'échec (étiqueté par C) coupe la branche droite est l'irréductibilité de la substitution σ.

Pour chaque variable x du domaine de σ :

1. *Si x appartient au domaine d'un σ_i ($i \in [1,2]$), $x\sigma_i$ est égal à $x\sigma$ car σ_i est close, est donc tout terme AC-égal à $x\sigma$ est irréductible. Si x est une variable d'extension x_f sous un opérateur AC f dans C'_i, elle apparaît également sous f dans C, et la condition d'irréductibilité de ses $\mathcal{H}terms$ reste valide.*
2. *Si x est l'une des variables x_{r_1} et x_{r_2}, x_{r_1} par exemple, introduites par l'étape d'inférence dans le cas de la stratégie de paramodulation (Définition 6.6) : elle apparaît dans les contraintes par $x =^?_{AC} r'_1\sigma_1$.*

   Donc, $x\sigma$ est le terme $r'_1\sigma$, AC-égal à r_1. Il nous faut montrer qu'il n'existe pas de terme r''_1 AC-égal à r_1 et réductible par une équation $(g \simeq d)$ (où $g \succ d$).

   Si un tel terme existait, l'équation $(l_1 \simeq r''_1)$ serait réductible par $(g \simeq d)$ en $(l_1 \simeq r''_1[d])$. Comme Q_β est AC-consistante, elle satisfait les atomes $(l_1 \simeq r_1)$, $(l_1 \simeq r''_1)$ et donc $(l_1 \simeq r''_1[d])$. Mais, $(l_1 \simeq r_1) \succ (l_1 \simeq r''_1[d])$ et ainsi $(l_1 \simeq r_1)$ devrait être réductible en $(r''_1[d] \simeq r_1)$; cela contredit l'irréductibilité de $(l_1 \simeq r_1)$.

   Ainsi, chaque terme AC-égal à r_1 est irréductible. Le raisonnement est identique pour x_{r_2} et r_2.
3. *Le dernier cas est que la variable x est l'une des variables z_1 et z_2 (cf. Définition 6.6). Supposons que x est égal à z_1. Par définition de cette variable, $l_1 =_{AC} f(u, z_1\sigma)$, où f est l'opérateur AC au sommet de l_1, et u un terme clos. S'il existe un terme t, AC-égal à $z_1\sigma$, et réductible par une équation $(g \simeq d)$, où $g \succ d$, le terme $f(u,t)$ (AC-égal à l_1) est lui aussi réductible par $(g \simeq d)$. Cela signifie que le nœud droit de $(l_1 \simeq r_1)$ devrait être un nœud de quasi-échec au lieu d'un nœud d'échec ou d'échec distant. Donc, chaque terme AC-égal à $z_1\sigma$ est irréductible par une équation $(g \simeq d)$ où $g \succ d$.*
   Le raisonnement est identique pour $z_2\sigma$.

Ainsi, un nœud d'échec étiqueté par la clause C coupe la branche droite de $MCT(\mathcal{GS})$. $\square$

Le cas de la paramodulation étendue étant traité, il reste à étudier les autres inférences possibles. Rappelons auparavant la proposition énonçant que la branche droite Q_γ de $MCT(\mathcal{GS})$ falsifie une clause de $\mathcal{GS}$.

Proposition 6.21 *Le dernier nœud Q_γ de la branche droite de $MCT(\mathcal{GS})$ falsifie une clause de $\mathcal{GS}$.*

Sa preuve est identique à celle des stratégies de AC-paramodulation et AC-superposition pour le cas non contraint (Propositions 4.19 et 4.25).

Le dernier nœud de la branche droite falsifiant une clause de $\mathcal{GS}$, il reste à montrer que cette clause peut effectivement étiqueter un nœud d'échec. En effet, comme nous l'avons déjà fait pour la règle de paramodulation étendue (Proposition 6.20), il faut vérifier que la substitution représentant la partie contrainte de cette clause falsifiée est bien irréductible.

Proposition 6.22 *Si une étape d'inférence est appliquée entre des clauses de $\mathcal{GS}$ étiquetant des nœuds d'échec ou d'échec distants, extensions de nœuds de la branche droite Q_γ de $MCT(\mathcal{GS})$, et si la clause déduite est falsifiée par Q_γ, alors un nœud d'échec coupe cette branche droite.*

Preuve: *Soient $C_1, \ldots, C_n$ ($1 \leq n \leq 2$) des clause étiquetant des nœuds d'échec (ou d'échec distants). Cela signifie qu'il existe des clauses $C'_1 [\![c_1]\!], \ldots, C'_n [\![c_n]\!]$ dans S^* telles que chaque C_i est AC-égal à $C'_i\sigma_i$, où σ_i est une substitution close irréductible (condition 1 dans la définition de MCT), solution de c_i. L'étape d'inférence est appliquée entre les clauses contraintes $C'_1 [\![\sigma_1]\!], \ldots, C'_n [\![\sigma_n]\!]$, et engendre une clause contrainte $C' [\![\sigma]\!]$. Si nous supposons que l'interprétation Q_γ falsifie une clause C, AC-égale à $C'\sigma$, la seule condition restant à vérifier pour que C étiquette un nœud d'échec dans la branche droite est l'irréductibilité de la substitution σ.*

Supposons que les σ_i ont des domaines disjoints (obtenus en renommant les variables). Alors, pour chaque variable x du domaine de σ :

1. *Si x appartient au domaine d'un σ_i, $x\sigma_i$ est égal à $x\sigma$ car σ_i est close, et donc chaque terme AC-égal à $x\sigma$ est irréductible. Si x est une variable d'extension x_f, comme C_i étiquette un nœud d'échec, les $\mathcal{H}terms$ de $x_f\sigma_i$ doivent être irréductibles.*
2. *Si x est la variable d'extension x_f introduite par la règle de paramodulation (ou superposition) contextuelle, et si un terme t de $\mathcal{H}terms(x_f\sigma, f)$ est réductible par une équation $(g \simeq d)$ $(g \succ d)$, alors une autre étape de paramodulation (ou superposition), éventuellement contextuelle, de la clause étiquetant le nœud droit de $(g \simeq d)$ dans C_2 (à une position dans le terme t) engendre une clause falsifiée par Q_γ étiquetant un nœud d'échec dans la branche droite.*
   Ce raisonnement par récurrence est bien fondé, car cette dernière inférence est appliquée dans une position plus interne que la première.
3. *La dernière possibilité est la suivante : la variable est introduite par l'étape d'inférence dans la clause déduite. Seules trois règles d'inférence créent de nouvelles variables ; il s'agit des règles de paramodulation, paramodulation contextuelle et paramodulation étendue. Le cas de la paramodulation étendue ayant déjà été traité dans la Proposition 6.20, étudions les deux autres règles.*
   Les clauses C'_1 et C_1 sont respectivement notées $(l' \simeq r') \vee D'_1$ et $(l \simeq r) \vee D_1$; x est décrite par la contrainte $x =^?_{AC} r'$, produite par l'étape d'inférence ; donc, $x\sigma$ est égal à $r'\sigma_1$ (et AC-égal à r). Nous devons montrer qu'il n'existe pas de terme r'' AC-égal à r et réductible par une équation $(g \simeq d)$ (où $g \succ d$). Le raisonnement est identique à celui effectué dans le cas 2 de la Proposition 6.20.

Nous venons donc de montrer que tous les termes du co-domaine de la substitution σ sont irréductibles. Un nœud d'échec peut donc être placé au-dessus de Q_γ dans la branche droite. $\square$

Pour résumer, nous avons montré que la branche droite de l'arbre sémantique cohérent $MCT(\mathcal{GS})$ est vide, car sinon elle est coupée par un nœud d'échec, ce qui contredit sa construction. Donc, l'arbre sémantique cohérent $MCT(\mathcal{GS})$ est également vide, de même que $MCT(S^*)$. La clause vide appartient donc à S^*.

6.6 Simplification de contraintes

Nous consacrons cette section à la description de règles permettant de simplifier les contraintes symboliques manipulées.

6.6.1 Simplification des contraintes d'unification AC

Les contraintes d'unification AC peuvent être simplifiées grâce à des règles inspirées des algorithmes d'unification AC [JK91], combinées avec l'aplatissement (Définition 1.7) systématique des termes. Nous décrivons dans un premier temps des règles de décomposition, élimination, simplification et instanciation. Il faut remarquer que le symbole $=^?_{AC}$ est commutatif, ce qui signifie qu'il faut parfois permuter ses arguments pour pouvoir appliquer l'une des règles suivantes.

Décomposition	$\dfrac{(f(s_1,\ldots,s_n)=^?_{AC} f(t_1,\ldots,t_n))\wedge c}{(s_1=^?_{AC} t_1)\wedge\ldots\wedge(s_n=^?_{AC} t_n)\wedge c}$	si $f\notin\mathcal{F}_{AC}$
Elimination	$\dfrac{(s=^?_{AC} t)\wedge c}{c}$	si $s=_{AC} t$
Simplification 1	$\dfrac{(f(s_1,\ldots,s_i,\ldots,s_m)=^?_{AC} f(t_1,\ldots,t_j,\ldots,t_n))\wedge c}{(f(s_1,\ldots,\ldots,s_m)=^?_{AC} f(t_1,\ldots,\ldots,t_n))\wedge c}$	si $f\in\mathcal{F}_{AC}$, $m>1$, $n>1$ et $s_i=_{AC} t_j$
Simplification 2	$\dfrac{(f(s)=^?_{AC} t)\wedge c}{(s=^?_{AC} t)\wedge c}$	si $f\in\mathcal{F}_{AC}$
Instanciation	$\dfrac{(x=^?_{AC} t)\wedge c}{(x=^?_{AC} t)\wedge c\{x\mapsto t\}}$	si $x\notin\mathcal{V}ar(t)$ et $x\in\mathcal{V}ar(c)$

Ces règles peuvent être appliquées sans aucune stratégie particulière.

- La règle *Décomposition* consiste à propager le problème d'unification sur les sous-termes, si les deux membres de la contrainte considérée possèdent le même opérateur (non AC) au sommet.
- La règle *Elimination* consiste à ôter une contrainte dont les deux membres sont AC-égaux.
- Si les deux membres d'une contrainte ont le même opérateur AC au sommet et s'ils ont des arguments AC-égaux, alors la règle *Simplification 1* ôte ces arguments (un par un, afin de respecter leur nombre d'occurrences). Cette règle ne s'applique que si les opérateurs AC ont au moins deux arguments, afin d'éviter la suppression de tous les arguments.
- La règle précédente ôtant des arguments d'un opérateur AC, il peut arriver qu'un tel opérateur se retrouve avec un seul argument. La règle *Simplification 2* supprime donc cet opérateur pour ne conserver que l'argument.
- La règle *Instanciation* consiste à remplacer une variable x par un terme t dans les autres contraintes, si la contrainte considérée est $(x=^?_{AC} t)$, à condition que x n'ait pas d'occurrence dans t.

D'autre règles peuvent être définies pour détecter d'éventuelles incohérences dans les contraintes.

Cycle 1	$\dfrac{(x =^?_{AC} t) \wedge c}{\mathbb{F}}$	si $x \in \mathcal{V}ar(t)$, $x \neq t$ et $\mathcal{H}ead(t) \notin \mathcal{F}_{AC}$
Cycle 2	$\dfrac{(x =^?_{AC} f(t_1, \ldots, t_n)) \wedge c}{\mathbb{F}}$	si $f \in \mathcal{F}_{AC}$, $\exists i,\ x \in \mathcal{V}ar(t_i)$ et $n > 1$
Conflit	$\dfrac{(f(s_1, \ldots, s_m) =^?_{AC} g(t_1, \ldots, t_n)) \wedge c}{\mathbb{F}}$	si $f \neq g$ et - si $f \in \mathcal{F}_{AC}$, $m > 1$ - si $g \in \mathcal{F}_{AC}$, $n > 1$

Ces incohérences proviennent soit de cycles, soit de conflits de symboles :

- La règle *Cycle 1* s'applique si une variable doit être unifiée avec un terme contenant cette même variable à une position interne, mais à condition que le terme n'ait pas un opérateur AC au sommet.
- La règle *Cycle 2* complète la règle *Cycle 1*, en précisant que si l'opérateur au sommet du terme à unifier avec la variable est AC, alors il doit posséder au moins deux arguments. Nous avons vu en effet que la règle *Simplification 1* peut engendrer des termes avec un opérateur AC à un seul argument.
- La règle *Conflit* s'applique si l'on essaie d'unifier deux termes dont les opérateurs de tête sont différents. Pour la même raison que la règle précédente, si un opérateur de tête est AC, il doit posséder au moins deux arguments.

Si aucun conflit n'est trouvé, ces règles de simplification engendrent une conjonction de contraintes $(s_1 =^?_{AC} t_1) \wedge \ldots \wedge (s_n =^?_{AC} t_n)$ telle que, $\forall i \in [1, n]$,

- soit s_i est une variable n'ayant aucune occurrence dans les termes s_1, …, s_{i-1}, s_{i+1}, …, s_n, t_1, …, t_n.
- soit t_i est une variable n'ayant aucune occurrence dans les termes s_1, …, s_n, t_1, …, t_{i-1}, t_{i+1}, …, t_n.
- soit s_i et t_i sont deux termes avec le même opérateur AC au sommet.

6.6.2 Simplification des contraintes d'ordre

Il est également possible de définir des règles de simplification pour les contraintes d'ordre, en appliquant aussi loin que possible l'algorithme de comparaison des termes ou atomes. Ces règles dépendent donc de l'ordre utilisé : interprétation polynômiale, ordre récursif sur les chemins, … (cf. Section 1.2).

Nous ne donnons pas de règles pour simplifier ces contraintes d'ordre comme nous l'avons fait pour les contraintes d'unification, car à part éliminer des contraintes triviales ou détecter des conflits, la décomposition de contraintes d'ordre fait appel à un langage de contraintes plus complet, mêlant conjonctions et disjonctions de contraintes pour traduire la notion de complémentarité.

Cependant, pour un ordre récursif sur les chemins, il est assez simple de transformer l'algorithme afin d'utiliser les informations fournies par les autres contraintes d'ordres accompagnant celle qui est testée. La stratégie serait alors de chercher à évaluer les plus petites contraintes (par la taille) en premier.

6.6.3 Combinaison de contraintes

La combinaison de contraintes d'ordre et d'unification AC est un problème très complexe. Cependant, chaque type de contrainte peut aider à simplifier l'autre.

En effet, lorsque des contraintes d'unification AC sont simplifiées par les règles décrites précédemment, rien n'interdit la présence de contraintes d'un type différent dans la conjonction ; ces dernières n'interviennent pas dans la simplification, mais elles peuvent être concernées par l'application de la substitution dans la règle *Instanciation*. Cela peut permettre de simplifier ces autres contraintes, ou bien de détecter un conflit.

De même, une contrainte du type $s \succ^? t$ peut servir dans un problème d'unification, car elle signifie que s et t ne sont pas AC-unifiables, grâce à la propriété d'AC-compatibilité de l'ordre $\succ$.

Très peu de travaux ont été réalisés sur ce sujet de combinaison de résolveurs de contraintes symboliques. Nous ne citerons que ceux de H. Kirchner et C. Ringeissen [KR94, Rin93], qui ont proposé un cadre formel pour la combinaison de résolveurs de contraintes symboliques.

6.7 Stratégie basique

Nous avons décrit dans ce chapitre une stratégie contrainte, consistant à associer à chaque clause des contraintes d'ordre et d'unification AC. Ces dernières provoquent une très sérieuse limitation du nombre de clauses déduites, et par conséquent de l'espace de recherche. En contrepartie, l'un des inconvénients majeurs de cette stratégie contrainte est la difficulté pour appliquer les règles de simplification, comme nous l'avons souligné dans la Section 6.6. De telles règles sont cependant indispensables pour l'efficacité d'un démonstrateur de théorèmes.

Aussi, il est intéressant de définir une stratégie intermédiaire, conservant la limitation des positions dans lesquelles les remplacements peuvent être effectués, mais facilitant l'application des règles de simplification. Cette stratégie n'est autre que la *stratégie basique*, qui consiste à résoudre les problèmes d'unification AC et à engendrer une clause par solution de ces problèmes ; cependant les substitutions solutions ne sont pas appliquées aux clauses, mais conservées comme contraintes.

Par exemple, la règle de paramodulation devient :

$$\frac{(l_1 \simeq r_1) \vee D_1 \ [\![c_1]\!] \qquad L_2 \vee D_2 \ [\![c_2]\!]}{L_2[p_2 \leftarrow x_{r_1}] \vee D_1 \vee D_2 \ [\![c_1 \wedge c_2 \wedge c_3 \wedge c_4]\!]}$$

où x_{r_1} est une nouvelle variable, $p_2 \in \mathcal{MP}os(L_2)$, $\sigma \in \mathcal{S}ol(L_2|_{p_2} =^?_{AC} l_1)$,
c_3 est $(x_{r_1} =^?_{AC} r_1) \wedge \{\bigwedge_{x \in \mathcal{D}om(\sigma)} (x =^?_{AC} x\sigma)\}$,
c_4 est $(l_1 \succ^? r_1) \wedge ((l_1 \simeq r_1) \succ^? D_1) \wedge (L_2 \succ^? D_2) \wedge (L_2 \succeq^? (l_1 \simeq r_1))$

La stratégie basique est théoriquement beaucoup moins intéressante que la stratégie contrainte, mais nous verrons dans le Chapitre 7, sur l'implantation de ces techniques, qu'elle est beaucoup plus efficace dans la pratique, car d'autres paramètres viennent à son secours, comme le fait de ne pas conserver toutes les solutions des problèmes d'unification AC.

7
Le système DATAC

Nous décrivons dans ce chapitre l'implantation des résultats sur les théories associatives et commutatives présentés dans ce document. Le logiciel se nomme `DATAC`, pour *Déduction Automatique dans des Théories Associatives et Commutatives*. Ses fonctionnalités sont de

- démontrer des théorèmes par réfutation,
- compléter des systèmes de clauses,

en présence d'opérateurs commutatifs, ou associatifs et commutatifs.

Il est écrit en CAML Light [LM93] (environ 15000 lignes de code), langage fonctionnel de la famille ML, et possède une interface graphique [1] réalisée en Tcl/Tk [Ous94] (X11 Toolkit basé sur le langage Tcl). `DATAC` est portable sur stations SUN, HP et IBM PC.

L'algorithme d'unification AC est celui de M.E. Stickel [Sti81], muni de la technique de résolution des équations diophantiennes de A. Fortenbacher [For89]. Deux algorithmes de filtrage AC sont disponibles : le premier est inspiré de l'algorithme de J.-M. Hullot [Hul80] ; le second est dû à S. Eker [Eke93] (écrit en langage C).

La première section de ce chapitre sera consacrée à la description du logiciel et de son interface. Nous présenterons ensuite quelques exemples d'exécution, puis comparerons les différentes stratégies proposées dans le logiciel `DATAC`.

7.1 Description du logiciel

L'interface du système `DATAC` offre la possibilité de saisir la spécification d'un problème de manière très conviviale. Ces spécifications sont également stockées dans un fichier, qui peut être modifié indépendamment du logiciel. Nous décrivons dans cette section les données d'une spécification, puis les différents paramètres pouvant influencer le déroulement de l'exécution. Enfin, nous mentionnons les stratégies d'exécution possibles.

7.1.1 Données principales

Les principales données pour spécifier un problème sont les clauses. Dans l'implantation, elles sont représentées sous la forme de séquents : une clause $\neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n$

1 L'interface graphique a été réalisée par V. Bardelang, étudiant à l'Ecole Supérieure d'Informatique Appliquée de Lorraine.

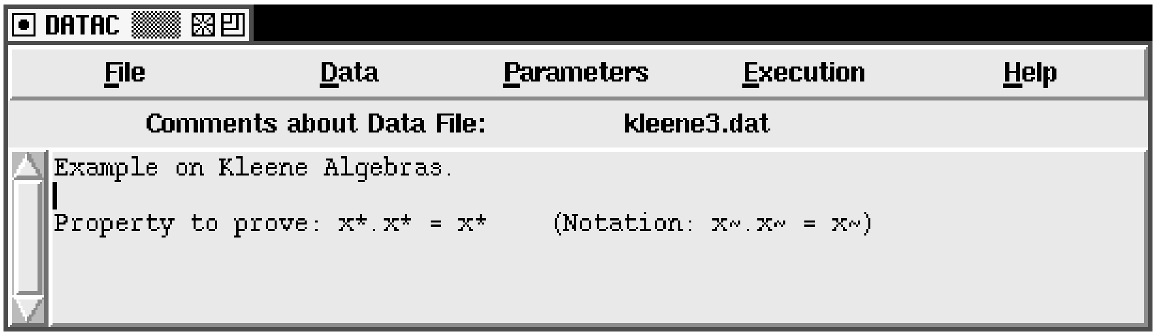


Figure 7.1: Fenêtre principale de l'interface

est représentée par $A_1, \ldots, A_m \Rightarrow B_1, \ldots, B_n$. En conséquence, chaque règle d'inférence de paramodulation ou de superposition est décomposée en deux règles : une règle s'appliquant à gauche du symbole $\Rightarrow$ (c'est-à-dire dans les littéraux négatifs), et une autre s'appliquant à droite de ce symbole (c'est-à-dire dans les littéraux positifs).

Parmi les données initiales, nous distinguons deux ensembles de clauses :

- les clauses décrivant la théorie,
- les clauses représentant la négation de la conjecture à montrer.

Un autre ensemble de données très important concerne les opérateurs. L'ordre utilisé sur les termes étant l'APO de L. Bachmair et D. Plaisted [BP85a], avec des adaptations de C. Delor et L. Puel [DP93], il est possible de définir pour chaque opérateur,

- une propriété : commutativité (C), ou associativité et commutativité (AC),
- un statut : lexicographique gauche-droite ou droite-gauche, ou multi-ensemble (par convention, les opérateurs AC et C ont le statut multi-ensemble),
- une liste d'opérateurs inférieurs en précédence,
- une liste d'opérateurs égaux en précédence.

En plus de cela, un opérateur peut être désigné minimal. Cette information est utilisée dans l'algorithme comparant des termes dans le sens où une variable est supérieure ou égale à cet opérateur.

7.1.2 Paramètres d'exécution

Lorsque la spécification du problème a été définie (clauses, opérateurs), l'utilisateur a la possibilité de faire varier de nombreux paramètres pour l'exécution.

- *Nombre maximal de clauses.* Il est obligatoire de préciser un nombre maximal de clauses à engendrer, afin d'assurer l'arrêt de l'exécution.

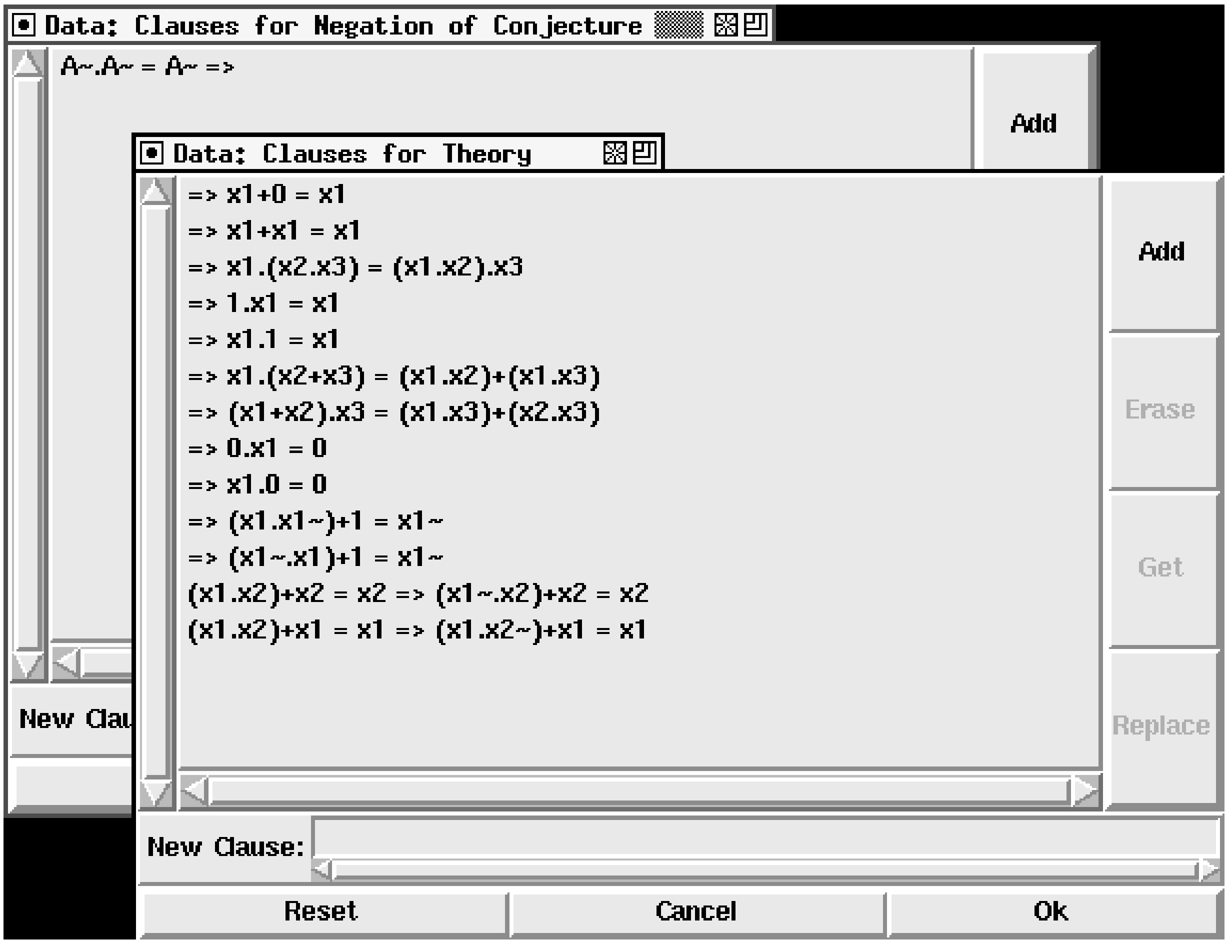


Figure 7.2: Description des clauses initiales

- *Unification AC optimisée*. L'algorithme d'unification AC implanté offre la possibilité de ne pas calculer toutes les solutions minimales. Les solutions « oubliées » font partie de celles qui ajoutent de nouvelles variables. Que cet algorithme soit utilisé dans sa version optimisée ou non, nous y avons intégré les conditions sur les $\mathcal{H}terms$ apparaissant dans la règle de paramodulation étendue (Définition 5.3).

- *Nombre maximal de AC-mgus*. Ce paramètre précise le nombre de solutions minimales d'un problème d'unification AC à conserver.

- *Règles de simplification*. Booléen indiquant si les règles de simplification standard sont autorisées.

- *Algorithme de Subsomption*. Si la règle de subsomption est permise, deux algorithmes sont proposés. Le premier est une extension de celui de G. Gottlob et A. Leitsch [GL85]. Le second est un simple appel au filtrage AC après un petit traitement des clauses.

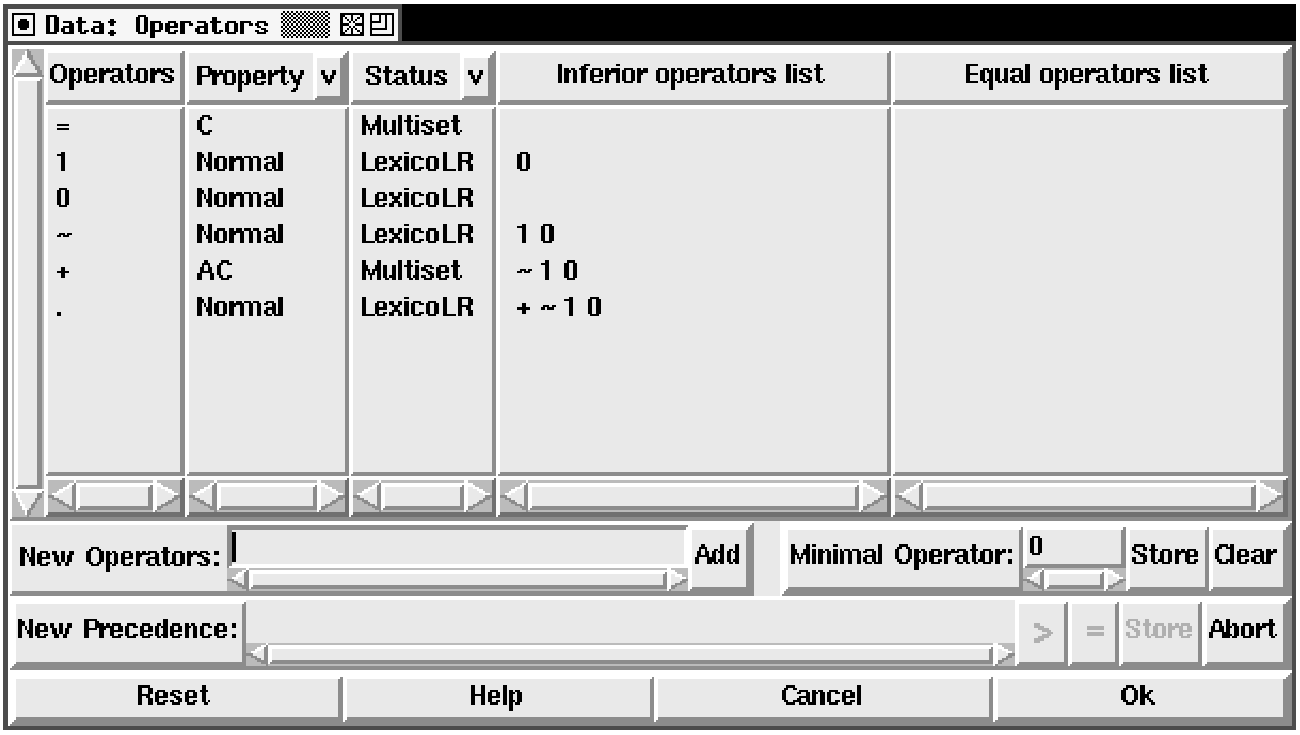


Figure 7.3: Description des opérateurs

- *Autres règles de simplification*. Booléen indiquant si les règles de simplification, comme la simplification clausale, la réflexion triviale ou la simplification par remplacement peuvent être utilisées.

- *Taille limite des clauses*. Il est possible de préciser une taille maximale pour les clauses à engendrer.

- *Degré de trace*. Lors d'une exécution, les informations affichées peuvent être soit pratiquement nulles, soit restreintes aux inférences réussies, soit très détaillées (mention de tous les essais de déduction).

- *Stratégie de déduction*. Il s'agit de préciser s'il faut utiliser la stratégie de paramodulation ou de superposition. En plus, il faut choisir entre une stratégie dite de *Résolution*, qui coupe quelques branches de l'arbre de recherche, et une stratégie dite de *Complétion*, qui n'oublie aucun chemin.

- *Préférence dans le choix des clauses*. Il s'agit de placer d'éventuels poids pour avantager le choix des clauses issues de la négation de la conjecture à prouver, de la théorie, ou possédant des littéraux négatifs ; le troisième poids peut être combiné avec les deux autres.

- *Simplification avant Subsomption*. Booléen indiquant s'il faut d'abord essayer de simplifier une clause avant de tester si elle est subsumée.

- *Littéraux non équationnels.* Booléen indiquant s'il faut commencer par un pré-traitement des clauses possédant des littéraux non équationnels.
- *Préférence pour les inférences.* Saisie d'un ordre de préférence pour appliquer les règles d'inférence.

7.1.3 Stratégies et modes d'exécution

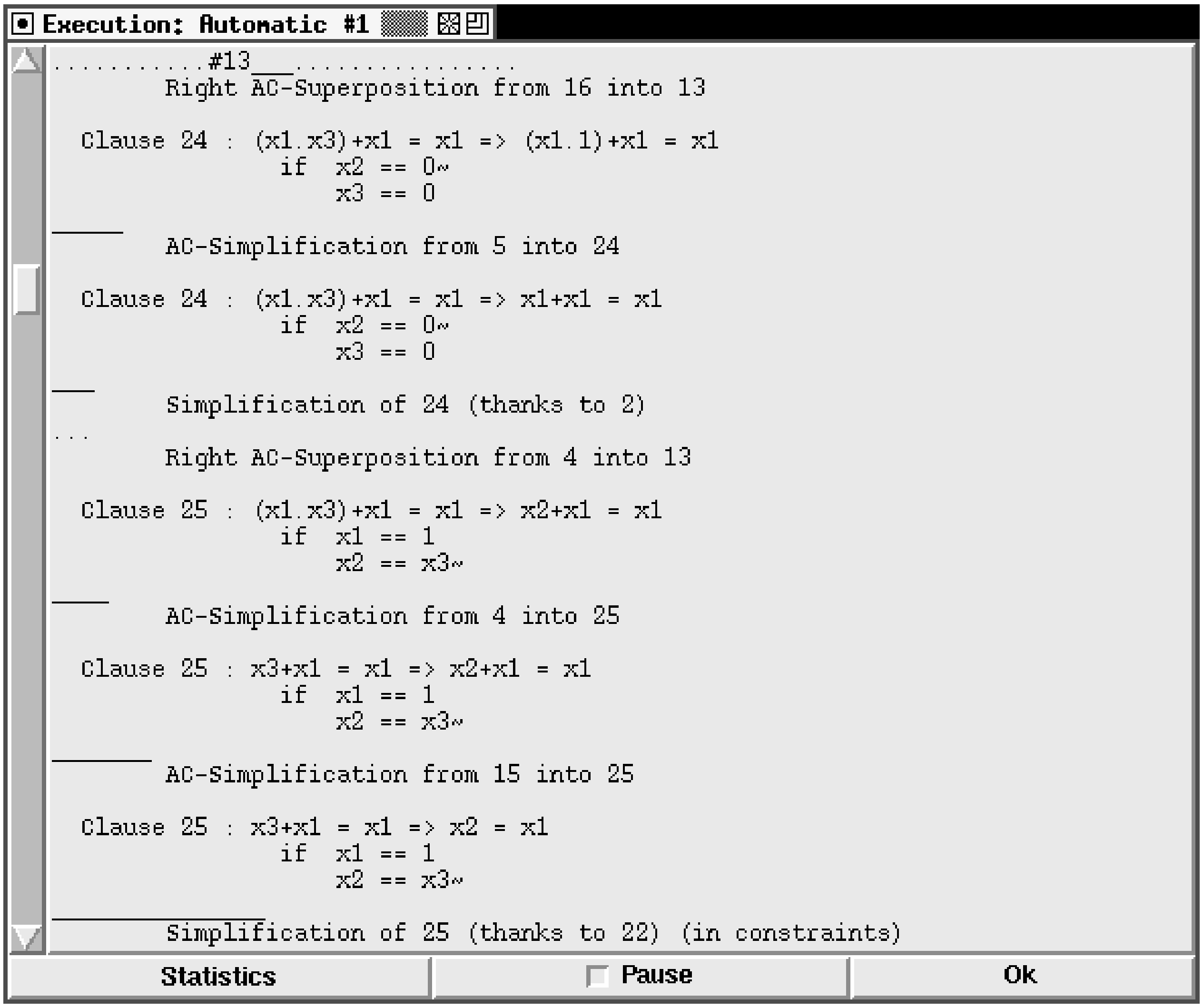


Figure 7.4: Fenêtre d'exécution

Avant de lancer une exécution, l'utilisateur a la possibilité de choisir la combinaison de plusieurs stratégies :

- *Stratégie ordonnée* ou *Stratégie positive*. La stratégie ordonnée vérifie que chaque inférence est effectuée entre les littéraux maximaux des clauses, alors que la stratégie positive oblige à utiliser au moins une clause positive, c'est-à-dire sans littéraux négatifs.

- *Stratégie standard*, *basique* ou *contrainte*. La stratégie standard applique la stratégie de AC-paramodulation ou de AC-superposition en conservant les contraintes d'ordre non résolues. La stratégie basique, en plus de conserver les contraintes d'ordre, n'applique pas les substitutions solution des problèmes d'unification AC ; elles sont conservées dans la partie contrainte associée à chaque clause. La stratégie contrainte se contente de décomposer les problèmes d'unification AC et de vérifier qu'ils ont au moins une solution.

Lorsque les stratégies d'exécution sont fixées, le lancement du programme peut se faire de deux façons :

- *Mode automatique*. L'exécution est alors lancée sans que l'utilisateur puisse intervenir par la suite. L'arrêt de l'exécution peut être provoqué
  - soit par la découverte d'une contradiction (clause vide),
  - soit par le dépassement du nombre maximum de clauses à engendrer,
  - soit par l'impossibilité d'appliquer une déduction.
- *Mode manuel*. Chaque inférence doit alors être donnée par l'utilisateur. En plus des règles d'inférence et de simplification classiques, il est offert la possibilité de simplifier complètement une clause, de supprimer une clause, ou d'ajouter une nouvelle clause. Des statistiques sont également disponibles pendant l'exécution : ensemble de clauses courant, liste des inférences ayant permis d'engendrer une clause donnée, liste de toutes les inférences ayant déduit une nouvelle clause, statistiques globales sur les inférences échouées, réussies, le nombre d'appels aux algorithmes d'unification et de filtrage.

En plus, l'utilisateur a la possibilité de stopper définitivement une exécution, d'interrompre momentanément le défilement dans la fenêtre (mode automatique). Lorsque l'exécution est finie, tout son contenu peut être sauvegardé dans un fichier.

7.2 Exemples de résolution avec la stratégie basique

Détaillons deux exemples de résolution et un exemple de complétion avec la stratégie basique.

7.2.1 Treillis modulaires

Nous détaillons ci-dessous deux exemples de résolution de problèmes dans les treillis modulaires. Pour représenter ces treillis, nous utilisons la formulation équationnelle de W. McCune [McC88].

Les opérateurs habituellement appelés *min* et *max* sont notés " $\cdot$ " et " $+$ " respectivement ; ils sont tous les deux associatifs et commutatifs. Le symbole de prédicat *COMP*, dénotant la propriété de complémentarité, est commutatif. L'élément maximal est noté 1, et l'élément minimal 0. Les treillis sont caractérisés par les clauses *1* à *8*, la modularité par la clause *9* ; la complémentarité est décrite par les clauses *10* à *12*.

Clause 1: $\Rightarrow\ (x_1 \cdot x_1 \simeq x_1)$
Clause 2: $\Rightarrow\ (x_1 + x_1 \simeq x_1)$

Clause 3 : $\Rightarrow \ (x_1 \cdot (x_1 + x_2) \simeq x_1)$
Clause 4 : $\Rightarrow (x_1 + (x_1 \cdot x_2) \simeq x_1)$
Clause 5 : $\Rightarrow \ (x_1 \cdot 0 \simeq 0)$
Clause 6 : $\Rightarrow (x_1 + 0 \simeq x_1)$
Clause 7 : $\Rightarrow \ (x_1 \cdot 1 \simeq x_1)$
Clause 8 : $\Rightarrow (x_1 + 1 \simeq 1)$
Clause 9 : $(x_1 \cdot x_2 \simeq x_1) \ \Rightarrow \ (x_2 \cdot (x_1 + x_3) \simeq x_1 + (x_3 \cdot x_2))$
Clause 10 : $COMP(x_1, x_2) \ \Rightarrow \ (x_1 \cdot x_2 \simeq 0)$
Clause 11 : $COMP(x_1, x_2) \ \Rightarrow \ (x_1 + x_2 \simeq 1)$
Clause 12 : $(x_1 \cdot x_2 \simeq 0), (x_1 + x_2 \simeq 1) \ \Rightarrow \ COMP(x_1, x_2)$

I Premier exemple

Le but de ce premier exemple est de montrer la propriété suivante :

si x et *max(u,v)* sont complémentaires, et y et *min(u,v)* sont complémentaires,
alors u et *max(x,min(y,v))* sont complémentaires.

La négation skolémisée de cette propriété est exprimée par les clauses *13* à *15*. La précédence sur les symboles de prédicat est $COMP >_{\mathcal{P}} \simeq$ et celle sur les symboles de fonction est $\cdot >_{\mathcal{F}} + >_{\mathcal{F}} B >_{\mathcal{F}} A >_{\mathcal{F}} R_2 >_{\mathcal{F}} R_1 >_{\mathcal{F}} 1 >_{\mathcal{F}} 0$.

Clause 13 : $\Rightarrow \ COMP(R_1, A + B)$
Clause 14 : $\Rightarrow \ COMP(R_2, A \cdot B)$
Clause 15 : $COMP(A, R_1 + (R_2 \cdot B)) \ \Rightarrow$

>> Résolution entre 10 et 13
Clause 16 : $\Rightarrow \ (z_1 \cdot z_2 \simeq 0) \quad [\![\ (z_1 =^?_{AC} R_1) \ \wedge \ (z_2 =^?_{AC} A + B) \ ]\!]$
>> Résolution entre 11 et 13
Clause 17 : $\Rightarrow \ (z_1 + z_2 \simeq 1) \quad [\![\ (z_1 =^?_{AC} R_1) \ \wedge \ (z_2 =^?_{AC} A + B) \ ]\!]$
>> Résolution entre 10 et 14
Clause 18 : $\Rightarrow \ (z_1 \cdot z_2 \simeq 0) \quad [\![\ (z_1 =^?_{AC} R_2) \ \wedge \ (z_2 =^?_{AC} A \cdot B) \ ]\!]$
>> Résolution entre 11 et 14
Clause 19 : $\Rightarrow \ (z_1 + z_2 \simeq 1) \quad [\![\ (z_1 =^?_{AC} R_2) \ \wedge \ (z_2 =^?_{AC} A \cdot B) \ ]\!]$
>> Résolution entre 12 et 15
Clause 20 : $(z_1 \cdot z_2 \simeq 0), (z_1 + z_2 \simeq 1) \ \Rightarrow \quad [\![\ (z_1 =^?_{AC} A) \ \wedge \ (z_2 =^?_{AC} R_1 + (R_2 \cdot B)) \ ]\!]$
>> Superposition droite de 19 dans 9
Clause 75 : $(z_1 \cdot x_1 \simeq z_1) \ \Rightarrow \ (z_1 + (z_2 \cdot x_1) \simeq x_1 \cdot 1) \quad [\![\ (z_1 =^?_{AC} A \cdot B) \ \wedge \ (z_2 =^?_{AC} R_2) \ ]\!]$
>> Simplification droite de 7 dans 75
Clause 75 : $(z_1 \cdot x_1 \simeq z_1) \ \Rightarrow \ (z_1 + (z_2 \cdot x_1) \simeq x_1) \quad [\![\ (z_1 =^?_{AC} A \cdot B) \ \wedge \ (z_2 =^?_{AC} R_2) \ ]\!]$
>> Superposition étendue entre 3 et 9
Clause 110 : $(x_1 \cdot z_1 \simeq x_1) \ \Rightarrow \ ((x_1 + (x_4 \cdot z_1)) \cdot x_3 \simeq x_3 \cdot z_2)$
$[\![\ (z_1 =^?_{AC} x_2 + x_3) \ \wedge \ (z_2 =^?_{AC} x_4 + x_1) \ \wedge \ \mathcal{H}terms(x_3, \cdot) \cap^? \{x_4 + x_1\} =^? \emptyset \ ]\!]$

>> Superposition contextuelle gauche de 3 dans 110

Clause 151 : $(x_1 \cdot z. \simeq z_3) \Rightarrow ((z_3 + (x_4 \cdot z_1)) \cdot x_3 \simeq x_3 \cdot z_2)$
$[\![(z_1 =^?_{AC} x_2 + x_3) \wedge (z_2 =^?_{AC} x_4 + (x_1 \cdot x_2)) \wedge (z_3 =^?_{AC} x_1 \cdot x_2) \wedge (z. =^?_{AC} x_2)$
$\wedge\ \mathcal{H}terms(x_3, \cdot) \cap^? \{x_4 + (x_1 \cdot x_2)\} =^? \emptyset \wedge \{x_2 + x_3\} \not\subseteq^? \mathcal{H}terms(x_1, \cdot)$
$\wedge\ ((x_1 \cdot x_2) \cdot (x_2 + x_3) \simeq x_1 \cdot x_2)$
$\succ^? (((x_1 \cdot x_2) + (x_4 \cdot (x_2 + x_3))) \cdot x_3 \simeq x_3 \cdot (x_4 + (x_1 \cdot x_2)))]\!]$

>> Réflexion triviale dans 151

Clause 151 : $\Rightarrow ((z_3 + (x_4 \cdot z_1)) \cdot x_3 \simeq x_3 \cdot z_2)$
$[\![(z_1 =^?_{AC} x_2 + x_3) \wedge (z_2 =^?_{AC} x_4 + (x_1 \cdot x_2)) \wedge (z_3 =^?_{AC} x_1 \cdot x_2)$
$\wedge\ \mathcal{H}terms(x_3, \cdot) \cap^? \{x_4 + (x_1 \cdot x_2)\} =^? \emptyset \wedge \{x_2 + x_3\} \not\subseteq^? \mathcal{H}terms(x_1, \cdot)$
$\wedge\ ((x_1 \cdot x_2) \cdot (x_2 + x_3) \simeq x_1 \cdot x_2)$
$\succ^? (((x_1 \cdot x_2) + (x_4 \cdot (x_2 + x_3))) \cdot x_3 \simeq x_3 \cdot (x_4 + (x_1 \cdot x_2)))]\!]$

>> Superposition contextuelle gauche de 1 dans 75

Clause 193 : $(z_3 \cdot z. \simeq z_1) \Rightarrow (z_1 + (z_2 \cdot z_3) \simeq z_3)$
$[\![(z_1 =^?_{AC} A \cdot B) \wedge (z_2 =^?_{AC} R_2) \wedge (z_3 =^?_{AC} B) \wedge (z. =^?_{AC} A)]\!]$

>> Réflexion triviale dans 193

Clause 193 : $\Rightarrow (z_1 + (z_2 \cdot z_3) \simeq z_3) \quad [\![(z_1 =^?_{AC} A \cdot B) \wedge (z_2 =^?_{AC} R_2) \wedge (z_3 =^?_{AC} B)]\!]$

>> Superposition droite de 16 dans 151

Clause 352 : $\Rightarrow (z_1 \cdot z_2 \simeq (z_3 + 0) \cdot z_1)$
$[\![(z_1 =^?_{AC} A) \wedge (z_2 =^?_{AC} R_1 + (x_1 \cdot B)) \wedge (z_3 =^?_{AC} x_1 \cdot B)$
$\wedge\ \{A + B\} \not\subseteq^? \mathcal{H}terms(x_1, \cdot)]\!]$

>> Simplification droite de 6 dans 352

Clause 352 : $\Rightarrow (z_1 \cdot z_2 \simeq z_3 \cdot z_1)$
$[\![(z_1 =^?_{AC} A) \wedge (z_2 =^?_{AC} R_1 + (x_1 \cdot B)) \wedge (z_3 =^?_{AC} x_1 \cdot B)$
$\wedge\ \{A + B\} \not\subseteq^? \mathcal{H}terms(x_1, \cdot)]\!]$

>> Simplification gauche de 352 dans 20

Clause 20 : $(z_3 \cdot z_1 \simeq 0), (z_1 + z_2 \simeq 1) \Rightarrow$
$[\![(z_1 =^?_{AC} A) \wedge (z_2 =^?_{AC} R_1 + (R_2 \cdot B)) \wedge (z_3 =^?_{AC} R_2 \cdot B)]\!]$

>> Simplification clausale dans 20 grâce à 18

Clause 20 : $(z_1 + z_2 \simeq 1) \Rightarrow \quad [\![(z_1 =^?_{AC} A) \wedge (z_2 =^?_{AC} R_1 + (R_2 \cdot B))]\!]$

>> Superposition étendue entre 4 et 193

Clause 457 : $\Rightarrow (z_1 + z_2 \simeq z_3 + z_1) \quad [\![(z_1 =^?_{AC} A) \wedge (z_2 =^?_{AC} R_2 \cdot B) \wedge (z_3 =^?_{AC} B)]\!]$

>> Simplification contextuelle gauche de 457 dans 20

Clause 20 : $(z_1 + z_2 + z_3 \simeq 1) \Rightarrow \quad [\![(z_1 =^?_{AC} A) \wedge (z_2 =^?_{AC} B) \wedge (z_3 =^?_{AC} R_1)]\!]$

>> Simplification clausale dans 20 grâce à 17

Clause 20 : $\square \quad [\![\mathbb{T}]\!]$

>> Le système initial est AC-incohérent <<

Voici quelques statistiques sur le déroulement de cette preuve. A partir de 15 clauses initiales, 240 applications de règles d'inférence et 901 simplifications, 443 clauses ont été engendrées mais seules 114 clauses apparaissent dans l'ensemble final. Les autres clauses ont été supprimées soit parce qu'elles étaient réductibles dans leur partie contrainte (112), soit parce qu'elles ont été réduites à des clauses triviales (232). De plus, l'algorithme de subsomption et la détection de clauses triviales ont évité d'engendrer 149 clauses.

II Second exemple : Le Lemme de SAM

Ce second exemple est le célèbre Lemme de SAM, posé en 1965 par Bumcroft comme un problème ouvert dans la théorie des treillis, et extrait de l'article *Semi-Automated Mathematics* de Guard et al. [GOBS69]. A notre connaissance, la seule preuve entièrement automatique de cette version a été réalisée par RRL [ZK88]. L'énoncé du lemme est le suivant :

si x et *max(u,v)* sont complémentaires, et y et *min(u,v)* sont complémentaires,
alors *min(max(x,min(y,v)),max(x,min(y,u)))* est égal à x.

La négation skolémisée de ce lemme est exprimée par les clauses *13* à *15*. La précédence sur les symboles de prédicat est $COMP >_{\mathcal{P}} \simeq$ et celle sur les symboles de fonction est $\cdot >_{\mathcal{F}} + >_{\mathcal{F}} B >_{\mathcal{F}} A >_{\mathcal{F}} R_2 >_{\mathcal{F}} R_1 >_{\mathcal{F}} 1 >_{\mathcal{F}} 0$.

Clause 13 : $\Rightarrow\ COMP(R_1, A+B)$
Clause 14 : $\Rightarrow\ COMP(R_2, A\cdot B)$
Clause 15 : $((R_1+(R_2\cdot B))\cdot(R_1+(R_2\cdot A)) \simeq R_1)\ \Rightarrow$

>> Résolution entre 10 et 13

Clause 16 : $\Rightarrow\ (z_1\cdot z_2 \simeq 0)\quad [\![\ (z_1 =^?_{AC} R_1)\ \wedge\ (z_2 =^?_{AC} B+A)\ ]\!]$

>> Résolution entre 10 et 14

Clause 18 : $\Rightarrow\ (z_1\cdot z_2 \simeq 0)\quad [\![\ (z_1 =^?_{AC} R_2)\ \wedge\ (z_2 =^?_{AC} B\cdot A)\ ]\!]$

>> Superposition étendue entre 3 et 9

Clause 106 : $(x_1\cdot z_1 \simeq x_1)\ \Rightarrow\ ((x_1+(x_4\cdot z_1))\cdot x_3 \simeq x_3\cdot z_2)$
$[\![\ (z_1 =^?_{AC} x_2+x_3)\ \wedge\ (z_2 =^?_{AC} x_4+x_1)\ \wedge\ \mathcal{H}terms(x_3,\cdot)\cap^? \{x_4+x_1\} =^? \emptyset\ ]\!]$

>> Superposition contextuelle gauche de 3 dans 106

Clause 150 : $(x_1\cdot z. \simeq z_3)\ \Rightarrow\ ((z_3+(x_4\cdot z_1))\cdot x_3 \simeq x_3\cdot z_2)$
$[\![\ (z_1 =^?_{AC} x_2+x_3)\ \wedge\ (z_2 =^?_{AC} x_4+(x_1\cdot x_2))\ \wedge\ (z_3 =^?_{AC} x_1\cdot x_2)\ \wedge\ (z. =^?_{AC} x_2)$
$\wedge\ \mathcal{H}terms(x_3,\cdot)\cap^? \{x_4+(x_1\cdot x_2)\} =^? \emptyset\ \wedge\ \{x_2+x_3\}\ \not\subseteq^?\ \mathcal{H}terms(x_1,\cdot)$
$\wedge\ ((x_1\cdot x_2)\cdot(x_2+x_3) \simeq x_1\cdot x_2)$
$\succ^?\ (((x_1\cdot x_2)+(x_4\cdot(x_2+x_3)))\cdot x_3 \simeq x_3\cdot(x_4+(x_1\cdot x_2)))\ ]\!]$

>> Réflexion triviale dans 150

Clause 150 : $\Rightarrow\ ((z_3+(x_4\cdot z_1))\cdot x_3 \simeq x_3\cdot z_2)$
$[\![\ (z_1 =^?_{AC} x_2+x_3)\ \wedge\ (z_2 =^?_{AC} x_4+(x_1\cdot x_2))\ \wedge\ (z_3 =^?_{AC} x_1\cdot x_2)$
$\wedge\ \mathcal{H}terms(x_3,\cdot)\cap^? \{x_4+(x_1\cdot x_2)\} =^? \emptyset\ \wedge\ \{x_2+x_3\}\ \not\subseteq^?\ \mathcal{H}terms(x_1,\cdot)$
$\wedge\ ((x_1\cdot x_2)\cdot(x_2+x_3) \simeq x_1\cdot x_2)$
$\succ^?\ (((x_1\cdot x_2)+(x_4\cdot(x_2+x_3)))\cdot x_3 \simeq x_3\cdot(x_4+(x_1\cdot x_2)))\ ]\!]$

>> Superposition gauche de 9 dans 15

Clause 239 : $(z_1\cdot z_2 \simeq z_1)\ ,\ (z_1+(z_3\cdot z_2) \simeq z_1)\ \Rightarrow$
$[\![\ (z_1 =^?_{AC} R_1)\ \wedge\ (z_2 =^?_{AC} R_1+(R_2\cdot B))\ \wedge\ (z_3 =^?_{AC} R_2\cdot A)\ ]\!]$

>> Simplification clausale dans 239 grâce à 3

Clause 239 : $(z_1 + (z_3 \cdot z_2) \simeq z_1) \Rightarrow$
$[\![\ (z_1 =^?_{AC} R_1) \ \wedge \ (z_2 =^?_{AC} R_1 + (R_2 \cdot B)) \ \wedge \ (z_3 =^?_{AC} R_2 \cdot A) \ ]\!]$

>> Superposition droite de 16 dans 150

Clause 260 : $\Rightarrow (z_1 \cdot z_2 \simeq (z_3 + 0) \cdot z_1)$
$[\![\ (z_1 =^?_{AC} A) \ \wedge \ (z_2 =^?_{AC} R_1 + (x_1 \cdot B)) \ \wedge \ (z_3 =^?_{AC} x_1 \cdot B)$
$\wedge \ \{A + B\} \ \not\subseteq^? \ \mathcal{H}terms(x_1, \cdot) \ ]\!]$

>> Simplification droite de 6 dans 260

Clause 260 : $\Rightarrow (z_1 \cdot z_2 \simeq z_3 \cdot z_1)$
$[\![\ (z_1 =^?_{AC} A) \ \wedge \ (z_2 =^?_{AC} R_1 + (x_1 \cdot B)) \ \wedge \ (z_3 =^?_{AC} x_1 \cdot B)$
$\wedge \ \{A + B\} \ \not\subseteq^? \ \mathcal{H}terms(x_1, \cdot) \ ]\!]$

>> Simplification contextuelle gauche de 260 dans 239

Clause 239 : $(z_1 + ((z_3 \cdot z_2) \cdot z_4) \simeq z_1) \Rightarrow$
$[\![\ (z_1 =^?_{AC} R_1) \ \wedge \ (z_2 =^?_{AC} A) \ \wedge \ (z_3 =^?_{AC} R_2 \cdot B) \ \wedge \ (z_4 =^?_{AC} R_2) \ ]\!]$

>> Simplification contextuelle gauche de 1 dans 239

Clause 239 : $(z_1 + (z_4 \cdot (z_2 \cdot z_3)) \simeq z_1) \Rightarrow$
$[\![\ (z_1 =^?_{AC} R_1) \ \wedge \ (z_4 =^?_{AC} R_2) \ \wedge \ (z_3 =^?_{AC} B) \ \wedge \ (z_2 =^?_{AC} A) \ ]\!]$

>> Simplification gauche de 18 dans 239

Clause 239 : $(z_1 + 0 \simeq z_1) \Rightarrow \ [\![\ (z_1 =^?_{AC} R_1) \ ]\!]$

>> Simplification clausale dans 239 grâce à 6

Clause 239 : $\Box$

>> Le système initial est AC-incohérent <<

Pour donner quelques statistiques sur cette preuve, à partir de 15 clauses initiales, 132 applications de règles d'inférence et 475 simplifications, 246 clauses ont été engendrées mais seules 66 clauses apparaissent dans l'ensemble final. Les autres clauses ont été supprimées soit parce qu'elles étaient réductibles dans leur partie contrainte (54), soit parce qu'elles ont été réduites à des clauses triviales (141). De plus, l'algorithme de subsomption et la détection de clauses triviales ont évité d'engendrer 106 clauses.

7.2.2 Groupes Abéliens

Détaillons maintenant un exemple de complétion, les groupes Abéliens. Ils sont caractérisés par deux axiomes :

$$\Rightarrow (x_1 + 0 \simeq x_1)$$
$$\Rightarrow (x_1 + i(x_1) \simeq 0)$$

L'opérateur + est AC, et la précédence sur les opérateurs est $i >_{\mathcal{F}} + >_{\mathcal{F}} 0$.

Si nous complétons ces axiomes avec la stratégie standard, nous obtenons les 3 clauses supplémentaires suivantes :

$$\Rightarrow (i(0) \simeq 0)$$
$$\Rightarrow (i(i(x_1)) \simeq x_1)$$
$$\Rightarrow (i(x_1 + x_2) \simeq i(x_1) + i(x_4))$$

Si nous utilisons la stratégie basique pour compléter l'ensemble initial de clauses, les trois clauses finales sont les suivantes :

$$\begin{array}{lll} \Rightarrow & (x_1 \simeq x_2) & [\![(x_1 =^?_{AC} i(0)) \;\wedge\; (x_2 =^?_{AC} 0)]\!] \\ \Rightarrow & (x_1 \simeq x_2) & [\![(x_1 =^?_{AC} i(i(x_2)))]\!] \\ \Rightarrow & (x_1 \simeq x_2 + x_3) & [\![(x_1 =^?_{AC} i(x_4 + x_5)) \;\wedge\; (x_2 =^?_{AC} i(x_4)) \;\wedge\; (x_3 =^?_{AC} i(x_5))]\!] \end{array}$$

La première chose que nous remarquons est la forte proportion de termes bloqués dans la partie contrainte. Cependant, il ne faut pas perdre de vue que la stratégie basique que nous avons implantée ne vérifie pas toutes les conditions lors de l'application d'étapes de simplification. Ces trois clauses ne devraient donc certainement pas apparaître sous une forme aussi contrainte.

Cependant, la présence de termes bloqués dans l'ensemble final de clauses peut être très utile lorsque celles-ci serviront à normaliser un terme. En effet, les conditions d'inclusion des termes bloqués dans les clauses pour appliquer une règle de simplification guident la mise en forme normale. Si nous reprenons les trois équations déduites ci-dessus, leur application impose une stratégie en profondeur d'abord.

7.3 Comparaison des différentes stratégies

Dans cette section, nous comparons les différentes stratégies d'exécution, puis celles de remplacement.

7.3.1 Stratégies d'exécution

Le logiciel `DATAC` propose trois stratégies : standard, basique ou contrainte. Détaillons les avantages et inconvénients de chacune d'entre elles, puis donnons quelques statistiques décrivant leur effet.

I Stratégie standard

Cette stratégie applique les règles d'inférence et de simplification définies dans le Chapitre 2 et adaptées aux théories associatives et commutatives (Chapitre 5). Ces règles font appel à un algorithme d'unification AC. Et à chaque déduction, pour ne pas perdre la propriété de complétude, elles doivent engendrer autant de clauses qu'il y a de solutions minimales au problème d'unification traité. Le nombre de clauses augmente donc très rapidement, ce qui a pour conséquence d'accroître le nombre de déductions possibles.

Des limitations non négligeables sont cependant apportées dans l'espace de recherche. Elles proviennent d'une part de la stratégie de sélection des clauses et des termes (stratégies ordonnées et positives), et d'autre part du contrôle très strict lors de l'application des règles simulant l'utilisation de clauses étendues (paramodulations contextuelles et étendues, Définitions 5.2 et 5.3). La manipulation de règles de simplification permet également de supprimer ou de réduite de nombreuses clauses redondantes.

II Stratégie basique

Cette stratégie consiste à résoudre les problèmes d'unification et à engendrer une clause par solution, comme dans la stratégie précédente, mais les substitutions solution ne sont pas appliquées dans les clauses déduites. Elles sont conservées sous la forme d'une contrainte associée à la clause concernée. Ainsi, cette technique engendre un grand nombre de clauses, mais le nombre de déductions applicables avec ces clauses est largement diminué ; les substitutions n'étant pas appliquées, de nombreux sous-termes ne peuvent être atteints par les règles effectuant des remplacements.

Nous n'avons implanté que partiellement les règles de simplification, dans le sens où nous ne vérifions pas l'inclusion des co-domaines des substitutions utilisées. La conséquence principale est la perte de la complétude du système d'inférence. Cependant, dans la pratique, nous n'avons jamais eu d'exemple de preuve de propriété échouant pour cette raison.

III Stratégie contrainte

Cette stratégie est la meilleure en théorie, car elle a l'énorme avantage de n'engendrer qu'une seule clause par déduction, et de ne nécessiter qu'un algorithme de décidabilité des problèmes d'unification. Cependant, dans la pratique, cette stratégie est très difficilement utilisable dans l'état dans lequel nous l'avons décrite, car ce qu'elle apporte en limitation d'espace mémoire et de calculs est perdu lorsque l'on veut appliquer des règles de simplification. En effet, si nous essayons de simplifier une clause $C_2 [\![c_2]\!]$ à l'aide d'une clause $C_1 [\![c_1]\!]$, il faut dans un premier temps choisir une instance de C_1, ce qui oblige à résoudre l'ensemble de contraintes c_1. De plus, soit la simplification peut être appliquée sur toutes les instances de C_2, ce qui est assez rare et demande quand même de résoudre c_2 afin de vérifier les conditions d'inclusion des co-domaines des substitutions, soit la simplification est appliquée pour une solution de c_2, ce qui oblige également à résoudre c_2, mais aussi à créer une nouvelle clause (la clause simplifiée) et à transformer $C_2 [\![c_2]\!]$ en précisant que la solution utilisée dans la simplification ne doit plus être prise en compte dans c_2.

Ces différents calculs sont relativement complexes, mais ils doivent en plus être réitérés à chaque essai de simplification. La version implantée de cette stratégie contrainte n'effectue des simplifications que si elles sont triviales, c'est-à-dire s'il n'est pas besoin de calculer les solutions des problèmes d'unification.

Donnons maintenant quelques statistiques sur ces trois stratégies d'exécution. L'exemple considéré porte sur les Algèbres de Kleene décrites par 13 clauses (voir Section 5.4). Nous avons effectué deux étapes de complétion linéaire à partir de ces clauses. Cela signifie que, dans un premier temps, toutes les déductions possibles entre ces clauses initiales ont été effectuées (colonnes #1). Dans un second temps (colonnes #2), nous avons refait la même opération mais en partant de l'ensemble de clauses obtenu après la première étape de complétion linéaire. Notons que nous avons utilisé la stratégie de superposition positive, et la version optimisée de l'algorithme d'unification AC, afin d'obtenir des résultats dans un temps raisonnable.

Les résultats obtenus sont décrits dans la Figure 7.5, et reflètent complètement les observations faites sur d'autres exemples. Les statistiques portent sur le nombre d'inférences appliquées, le nombre de clauses engendrées et le nombre de clauses finales. L'interprétation de ces résultats

Complétion linéaire (Algèbres de Kleene)	*Standard* #1	*Standard* #2	*Basique* #1	*Basique* #2	*Contrainte* #1	*Contrainte* #2
Nbre clauses initiales	13	31	13	31	13	52
Nbre inférences	51	296	51	231	61	845
Nbre clauses engendrées	64	652	64	455	64	1271
Nbre clauses finales	31	143	31	91	52	1070

Figure 7.5: Comparaison des stratégies de déduction

est la suivante :

- Si nous comparons les trois colonnes #1, nous remarquons qu'il n'y a aucune différence entre les stratégies standard et basique. Cela s'explique simplement par le fait que chaque clause de l'ensemble initial est sans contraintes ; il n'existe donc aucun sous-terme bloqué dans ces clauses. Pour la stratégie contrainte, un plus grand nombre de règles d'inférence a pu être appliqué, car la redondance de certaines clauses déduites n'a pu être montrée.

- La comparaison des colonnes #2 permet d'observer l'effet de la stratégie basique : comme certains sous-termes ont été bloqués dans les clauses issues de la première étape de complétion linéaire, le nombre d'inférence possibles lors de la seconde étape est diminué, ce qui entraîne une diminution du nombre de clauses déduites, et donc du nombre de clauses finales. Les résultats obtenus pour la stratégie contrainte confirment la difficulté d'application des règles de simplification.

7.3.2 Stratégies de remplacement

Dans le logiciel `DATAC` comme dans ce document, nous proposons deux techniques de remplacement : la paramodulation et la superposition. La seconde étant un raffinement de la première, elle est plus efficace pour la déduction non contrainte. Cependant, pour les stratégies basique et contrainte, la différence entre ces deux techniques est moins évidente :

- La règle de paramodulation permet d'appliquer des remplacements dans les plus petits membres d'équations, mais compense cela en n'insérant pas le membre droit de l'équation utilisée pour le remplacement dans la clause déduite. Ce membre droit est abstrait par une nouvelle variable, ce qui interdit tout futur remplacement dans celui-ci.

- La règle de superposition n'effectue que des remplacements dans les membres maximaux d'équations.

Nous avons ainsi dans chacune de ces règles la notion de remplacement interdit dans un membre droit d'une équation. Cependant, si nous étudions plus précisément le comportement de ces deux règles, la superposition reste la plus efficace. Montrons cela sur un exemple très schématique.

Exemple 7.1 Supposons que l'on dispose d'une clause C et de deux équations $(l_1 \simeq r_1)$ et $(l_2 \simeq r_2)$, telles que :

- un remplacement peut être effectué de l'équation $(l_1 \simeq r_1)$ dans la clause C, pour déduire $C[r_1]$,
- un remplacement peut être effectué de l'équation $(l_2 \simeq r_2)$ dans le terme r_1 de l'équation $(l_1 \simeq r_1)$, pour déduire $(l_1 \simeq r_1[r_2])$.

Avec la stratégie de superposition, la clause $C[r_1]$ sera déduite, puis la clause $C[r_1[r_2]]$. Par contre, l'équation $(l_1 \simeq r_1[r_2])$ ne sera pas déduite car tout remplacement dans un membre droit est interdit.

Avec la stratégie de paramodulation, la clause $C[r_1]$ sera déduite, mais pas $C[r_1[r_2]]$ car le sous-terme r_1 sera bloqué dans la partie contrainte, et ne pourra être atteint pour effectuer le remplacement avec $(l_2 \simeq r_2)$. Cependant, l'équation $(l_1 \simeq r_1[r_2])$ sera engendrée, et elle servira à déduire la clause $C[r_1[r_2]]$ directement à partir de C.

Pour conclure cet exemple, la stratégie de superposition a engendré deux clauses alors que la stratégie de paramodulation en a engendré trois. ♦

Conclusion

Nous nous sommes concentrés au cours de ce travail sur l'étude de techniques de déduction automatique dans la logique du premier ordre, modulo un ensemble d'équations. Nous avons étudié différentes améliorations à apporter, en particulier lors de la gestion des extensions. Ces techniques, appliquées aux théories associatives et commutatives, ont permis la définition de règles d'inférence optimales en intégrant une stratégie contrainte. Les conséquences de cette stratégie sont de ne plus résoudre les problèmes d'unification AC, et de conserver une trace de toutes les autres conditions d'application des règles d'inférence qui n'ont pu être résolues.

Apports

La déduction automatique modulo une théorie équationnelle étant restreinte au cadre purement équationnel, nous avons étendu cela au cadre clausal, et avons montré comment gérer les clauses étendue par l'ajout de règles d'inférence. Nous avons également défini une procédure de calcul des extensions utiles pour une théorie équationnelle E, qui est exécuté avant même de connaître les clauses du système à résoudre.

Nous avons proposé des systèmes d'inférence très variés, puisque nous offrons plusieurs stratégies de choix de clauses et de termes pour effectuer des déductions (stratégies ordonnées et positives), ainsi que plusieurs stratégies de remplacement (paramodulation et superposition). Nous avons démontré que ces systèmes d'inférence sont réfutationnellement complets si la théorie E est régulière. Ce résultat généralise les travaux de U. Wertz [Wer92] et E. Paul [Pau94], car la classe des théories régulières est beaucoup plus grande que celle qu'ils imposaient.

Le cas des théories associatives et commutatives a été approfondi, et nous avons montré que des restrictions très importantes sur les règles d'inférence peuvent être ajoutées, afin de diminuer le nombre de déductions possibles.

Pour ces théories associatives et commutatives, nous avons défini une amélioration très forte, qui consiste à ajouter des contraintes symboliques représentant les conditions des règles d'inférence. Ainsi, nous n'avons plus besoin de résoudre les problèmes d'unification AC, mais simplement de tester leur satisfaisabilité. Le gain est considérable, car la complexité du calcul de l'ensemble des substitutions minimales solution d'un problème d'unification AC est doublement exponentiel, alors que le test de satisfaisabilité d'un tel problème n'est qu'exponentiel. La stratégie contrainte ainsi définie simule la stratégie basique qui précise que tout remplacement dans un terme créé dans une clause par une déduction antérieure est inutile.

Les résultats présentés dans ce document sur les stratégies associatives et commutatives ont été implantés dans un logiciel appelé `DATAC`. Les résultats obtenus avec ce logiciel permettent de comparer les différentes stratégies, et de cerner les avantages et inconvénients de chacune.

Perspectives

Les systèmes d'inférence présentés dans ce document, concernant la déduction modulo une théorie équationnelle E, ont été démontrés réfutationnellement complets, mais nous allons nous attacher à montrer qu'ils peuvent également servir pour effectuer de la complétion, ainsi que de l'unification, grâce à la technique de surréduction. De même, les règles de simplification définies permettent d'éliminer un grand nombre de clauses redondantes, mais nous avons dû montrer la compatibilité de chacun d'entre elles, séparément. Nous allons étudier une notion plus générale de redondance, qui permettra ainsi de manipuler toute règle de simplification satisfaisant les conditions associées à cette notion, comme l'ont fait U. Wertz et E. Paul dans [Wer92, Pau94].

Dans nos règles d'inférence, nous avons besoin d'un ordre de simplification E-compatible pour comparer les termes. De tels ordres existent pour un nombre très restreint de théories équationnelles, comme nous l'avons précisé dans la Section 1.2. Une de nos préoccupations est donc de définir des ordres compatibles avec une classe plus grande de théories E.

La définition d'une stratégie contrainte n'a été appliquée que pour des théories associatives et commutatives. Nous sommes sur le point de généraliser ce résultat aux théories équationnelles régulières, ce qui apportera un progrès considérable. En effet, nous pourrons traiter des théories dont l'unification est soit indécidable, soit infinitaire, car nous n'aurons besoin que de la satisfaisabilité des problèmes d'unification.

Lors de la définition de cette stratégie contrainte, nous avons souligné le problème posé par les règles de simplification. Ces règles sont difficilement applicables, mais comme elles sont indispensables pour l'efficacité des déductions, leur étude doit être approfondie.
Plus généralement, l'étude de la redondance de clauses est un sujet important, qui fait appel à des mécanismes de résolution et de propagation de contraintes. En particulier, il serait intéressant de définir des procédures incrémentales de résolution et de propagation de contraintes, ceci afin de minimiser les temps de calcul. D'autre part, ces procédures devront mettre en œuvre des techniques de combinaison de contraintes, car il ne s'agit pas seulement d'étudier des contraintes d'unification, mais aussi des contraintes d'ordre.

Enfin, le logiciel `DATAC` est un outil appelé à évoluer, et donc à expérimenter ces techniques de résolution et de propagation de contraintes. De même, traitant actuellement des théories commutatives et associatives-commutatives, il serait intéressant de permettre l'utilisation d'autres théories équationnelles.

Appendice A
Preuves de complétude

Lors des preuves de complétude des stratégies de paramodulation (Section 4.3) et superposition (Section 4.4) modulo une théorie équationnelle E, nous avons montré comment sélectionner une branche de l'arbre sémantique (appelée *branche droite*), à l'aide de nœuds d'échec, de quasi-échec et d'échec distants. Nous avons également montré que, si cette branche est consistante pour la théorie E, elle ne peut être que vide, ce qui signifie qu'une contradiction (la clause vide) est toujours engendrée par nos systèmes de règles d'inférence si l'ensemble de clauses initial est E-incohérent.

Cependant, nous n'avons pas détaillé la preuve de consistance modulo E de cette branche droite. C'est ce que nous allons faire dans cet appendice, pour les stratégies de paramodulation et de superposition.

A.1 Stratégie de paramodulation

Commençons par rappeler la situation dans laquelle nous nous trouvons. A partir d'un ensemble de clauses E-incohérent S, nous avons défini un ensemble $\mathcal{GS}$, contenant l'ensemble des instances closes des clauses pouvant être engendrées à partir de S, par nos règles d'inférence (définies dans la Section 2.2.1). Les clauses de $\mathcal{GS}$ servent à couper toutes les branches de l'arbre sémantique au niveau de nœuds appelés *nœuds d'échec*. Le sous-arbre élagué est appelé *arbre sémantique cohérent*, et noté $MCT(\mathcal{GS})$. Pour montrer que notre ensemble de règles d'inférences est réfutationnellement complet, il faut montrer que ce sous-arbre est vide. Ceci est réalisé en sélectionnant une branche de ce sous-arbre, appelée *branche droite* (Définition 4.15), et en démontrant qu'elle ne peut être que vide (Proposition 4.19).

Cependant, cette branche droite doit être E-consistante (Définition 4.3). Cette propriété a été annoncée dans la Proposition 4.18, et nous allons détailler sa preuve dans cette section, sous la forme de plusieurs lemmes.

Soit Q_γ le dernier nœud de la branche droite. Par abus de langage, Q_γ désignera cette branche droite toute entière. Dans la preuve de la Proposition 4.18, nous avons montré que Q_γ ne peut être vide si $MCT(\mathcal{GS})$ n'est pas vide. Supposons que Q_γ est E-inconsistant ; il existe un ordinal minimal α tel que Q_α est E-inconsistant. α n'est pas un ordinal limite et admet donc un prédécesseur α^-. Notons K l'interprétation partielle Q_{α^-} et B l'atome A_{α^-}. Par minimalité de α, K est E-consistant, et

$$\exists A_\beta =_E B \ \text{ tel que } \begin{array}{l} \bullet \ \text{soit } A_\beta \in W_\alpha \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta) \\ \bullet \ \text{soit } A_\beta \notin W_\alpha,\ A_\beta \longrightarrow_K^{l \simeq r} A_\beta[r] \text{ où } l \succ r, \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta[r]) \end{array}$$

Montrons dans un premier temps que les atomes B et A_β sont équationnels et qu'ils possèdent un même membre droit irréductible.

Lemme A.1 *B est une équation $(u_2 \simeq v)$ et A_β est une équation $(u_1 \simeq v)$, où $u_1 =_E u_2$, $u_2 \succ v$ et v est irréductible.*

Preuve : *Décomposons la preuve de ce lemme en quatre faits :*

1. ***$\mathcal{H}ead(B)$* est "$\simeq$"** *; si ce n'est pas le cas, B est un atome $P(s_1, \ldots, s_n)$, A_β est un atome $P(t_1, \ldots, t_n)$, et $s_i =_E t_i$ pour tout i. Cependant, B et A_β sont interprétés différemment implique qu'il existe au moins un indice j tel que $K(s_j \simeq t_j) = F$; donc, K serait E-inconsistant, ce qui est faux par hypothèse.*
   Ainsi, nous pouvons noter que : $B = (u_2 \simeq v_2)$ et $A_\beta = (u_1 \simeq v_1)$, avec $u_2 \geq v_2$, $u_1 \geq v_1$, $u_1 =_E u_2$ et $v_1 =_E v_2$.
2. **$u_2 \neq_E v_2$** *; en effet, si u_2 et v_2 sont E-égaux, les atomes B et A_β sont tous les deux des équations triviales pour la théorie E, dont l'une est falsifiée :*
   - *si $Q_\alpha(B) = F$ ou bien, $Q_\alpha(B) = V$ et $A_\beta \in W_\alpha$, Q_α ne devrait pas appartenir à $MCT(\mathcal{GS})$, car elle falsifie une équation de $\mathcal{E}$,*
   - *si $Q_\alpha(B) = V$ et $A_\beta \notin W_\alpha$, l'atome A_β est réductible par une équation $(l \simeq r)$ $(l \succ r)$; supposons que cette réduction s'applique dans le terme u_1 (le raisonnement serait identique pour v_1) : $K(v_1 \simeq u_1[r]) = F$. L'atome $(u_1[l] \simeq u_1[r])$ est satisfait par K car il se réduit en un atome trivial ; $(u_2 \simeq u_1[r])$ l'est aussi car il lui est E-égal. Cependant, $(u_2 \simeq v_2)$ étant valide pour Q_α, cela signifie que $(v_2 \simeq u_1[r])$ l'est également, ce qui est en contradiction avec la E-consistance de K, puisque K falsifie $(v_1 \simeq u_1[r])$.*

   Donc, $u_2 \succ v_2$, et par E-compatibilité, $u_1 \succ v_1$.
3. **On peut choisir A_β tel que $v_1 = v_2$** *; si v_1 et v_2 sont différents, comme K est E-consistant, B ne peut être réduit en un atome E-égal et ainsi $v_1 > v_2$, $K(v_1 \simeq v_2) = V$ et $(u_1 \simeq v_1) \longrightarrow_K^{v_1 \simeq v_2} (u_1 \simeq v_2)$. Donc les atomes B et $(u_1 \simeq v_2)$ sont toujours interprétés différemment et il est possible d'utiliser $(u_1 \simeq v_2)$ à la place de A_β pour marquer l'inconsistance avec B.*
   Il faut remarquer que l'atome $(u_1 \simeq v_2)$ est plus petit que l'atome initialement choisi A_β. C'est pour cette raison que nous supposerons dans toutes les preuves à venir que A_β a été choisi minimal. Notons v les termes v_1 et v_2.
4. **v est irréductible** *; si le terme v est réductible en un terme v', les équations $(u_1 \simeq v)$ et $(u_2 \simeq v)$ sont réductibles en $(u_1 \simeq v')$ et $(u_2 \simeq v')$; or, ces atomes sont toujours E-égaux et interprétés différemment par K ; cela signifierait que K est E-inconsistant.*

Donc, B est une équation $(u_2 \simeq v)$ et A_β une équation $(u_1 \simeq v)$ telles que : $u_1 =_E u_2$, $u_2 \succ v$ et v est irréductible. □

Le lemme suivant est très important pour prouver la E-consistance de la branche droite. Il établit que, pour un nœud E-consistant K de cette branche droite, si deux atomes E-égaux et différemment interprétés sont réductibles par des équations différentes à des positions différentes, alors un nœud d'échec devrait couper la branche droite. Pour démontrer ce lemme, nous nous sommes inspiré de la preuve d'une propriété similaire effectuée par E. Paul dans [Pau94].

Lemme A.2 *Soient $(u_1 \simeq v)$ et $(u_2 \simeq v)$ deux atomes ayant les propriétés énoncées dans le Lemme A.1 et tels que*

$$(u_1 \simeq v) \longrightarrow_K^{g_1 \simeq d_1} (u_1[d_1] \simeq v) \quad et \quad g_1 \succ d_1$$

$$(u_2 \simeq v) \longrightarrow_K^{g_2 \simeq d_2} (u_2[d_2] \simeq v) \quad et \quad g_2 \succ d_2$$

Si K est E-consistant et $K(u_1[d_1] \simeq v) \neq K(u_2[d_2] \simeq v)$, alors K falsifie une clause de $\mathcal{GS}$.

Preuve : *Le Théorème de la Réduction nous permet de dire qu'il existe des équations irréductibles réduisant $(u_1 \simeq v)$ et $(u_2 \simeq v)$. La Proposition 4.16 nous permet même d'affirmer qu'il existe des atomes E-égaux à $(u_1 \simeq v)$ et $(u_2 \simeq v)$, conservant les interprétations, réductibles par des équations dont les nœuds droits sont des nœuds d'échec étiquetés par des clauses de $\mathcal{GS}$. Alors, considérons que les atomes vérifiant ces propriétés sont $(w_1 \simeq v)$, $(w_2 \simeq v)$, $(l_1 \simeq r_1)$ et $(l_2 \simeq r_2)$.*

$$(w_1 \simeq v) \longrightarrow_K^{p_1,\ l_1 \simeq r_1} (w_1[r_1]_{p_1} \simeq v) \quad et \quad l_1 \succ r_1$$

$$(w_2 \simeq v) \longrightarrow_K^{q_1,\ l_2 \simeq r_2} (w_2[r_2]_{q_1} \simeq v) \quad et \quad l_2 \succ r_2$$

$$K(u_1[d_1] \simeq v) \;=\; K(w_1[r_1] \simeq v) \;\neq\; K(w_2[r_2] \simeq v) \;=\; K(u_2[d_2] \simeq v)$$

Comme $K(w_1[r_1] \simeq v) \neq K(w_2[r_2] \simeq v)$, K falsifie l'atome $(w_1[r_1] \simeq w_2[r_2])$.

— *Etudions la propriété de E-fermeture suivante :*

$$\begin{tikzcd}
w_2 \arrow[rr, leftrightarrow, "*", "E"'] \arrow[d, "\mathcal{C}_{l_2},E"'] & & w_1 \arrow[d, dotted, "\mathcal{C}_{l_2},E"'] \\
w_2[r_2] \arrow[d, dotted, "\mathcal{C}_{l_2},E"', "*"] & & w_1[r_2^e] \arrow[d, dotted, "\mathcal{C}_{l_2},E"', "*"] \\
w_2' \arrow[rr, leftrightarrow, dotted, "*", "E"'] & & w_1'
\end{tikzcd}$$

où $\mathcal{C}_{l_2}$ désigne l'ensemble des contextes (e_2, o_2) du terme l_2 tels que, soit $e_2 = l_2$ et $o_2 = \epsilon$, soit il existe un contexte (e_2', o_2, c_2) de $\mathcal{C}ont(l_2)$ et une substitution close σ, solution de c_2, tels que $e_2 = e_2'\sigma$.

Dans le diagramme précédent, la réduction de w_2 en $w_2[r_2]$ est faite par l'équation $(l_2 \simeq r_2)$ à la position q_1. La théorie E étant régulière, le terme w_1 est réductible à la position p_2 par une équation $(l_2^e \simeq r_2^e)$, extension de $(l_2 \simeq r_2)$ pour un contexte (e_2, o_2) de $\mathcal{C}_{l_2}$ (o_2 peut être la position vide) : $l_2^w =_E l_2^e$, où l_2^w désigne $w_1|_{p_2}$.

Les atomes $(w_1[r_2^e] \simeq v)$, $(w_1' \simeq v)$, $(w_2' \simeq v)$ et $(w_2[r_2] \simeq v)$ appartiennent tous au domaine de K, et comme cette interprétation est E-consistante, ils sont tous interprétés de la même *manière. Cependant, cela signifie que les atomes* $(w_1[r_2^e] \simeq v)$ *et* $(w_1[r_1] \simeq v)$ *sont interprétés différemment, ce qui n'est possible (par construction de l'arbre sémantique) que si l'atome* $(w_1 \simeq v)$ *n'est pas réductible par* $(l_2^w \simeq r_2^e)$*, c'est-à-dire si* $(l_2^w \simeq r_2^e) > (w_1 \simeq v)$*. Nous en concluons que les termes* l_2^w *et* w_1 *sont* E*-égaux.*

Dans la suite de la preuve, nous allons étudier l'extension $(l_2^e \simeq r_2^e)$*. Mais, pour faire le lien avec le terme* w_1*, il faut montrer que* $K(w_1[r_2^e] \simeq r_2^e) = V$*. U. Wertz [Wer92] et E. Paul [Pau94] ont résolu le problème en obligeant la théorie* E *à vérifier* $w_1[r_2^e] =_E r_2^e$ *(cf. Section 2.5). Nous allons montrer que cette restriction sur la théorie* E *est inutile grâce à l'étude de la propriété de* E*-fermeture suivante :*

$$\begin{tikzcd} w_1 \arrow[r, leftrightarrow, "*", "E"'] \arrow[d, "\mathcal{C}_{l_2^e},E"'] & l_2^e \\ w_1[r_2^e] & \end{tikzcd}$$

En effet, si $w_1[r_2^e]$ *et* r_2^e *ne sont pas* E*-égaux, au vu des preuves des Lemmes 3.2 et 3.3 (*E*-fermeture locale et* E*-fermeture de la relation* $\longrightarrow_{\mathcal{C}_{l_2^e},E}$*), la* E*-fermeture du graphe précédent ne peut être réalisée que de deux manières :*

- *Soit à l'aide d'une réduction au sommet de* l_2^e *par* $\longrightarrow_{\mathcal{C}_{l_2^e},E}$ *(Cas 3a), pour obtenir un terme* E*-égal à* $w_1[r_2^e]$*. Comme nous étudions le cas à* $w_1[r_2^e]$ *et* r_2^e *ne sont pas* E*-égaux, le contexte* (e'_2, o'_2) *utilisé dans la réduction précédente est un vrai contexte, i.e.* $o'_2 \neq \epsilon$*.*
  Soit C_2 *la clause de* $\mathcal{GS}$ *étiquetant le nœud droit de l'atome* $(l_2 \simeq r_2)$*. Soit* $C'_2 = (l'_2 \simeq r'_2) \vee D'_2$ *la clause de* $INF^*(S)$ *telle que :* $C_2 =_E C'_2\sigma$*, pour une substitution close* σ*.*
  - *Si le contexte* (e_2, o_2) *est égal à* (l_2, ϵ)*,* $l_2^e = l_2$*, une* E*-paramodulation contextuelle de* $C'_2\sigma$ *dans* $C'_2\sigma$*, avec le contexte* (e'_2, o'_2)*, engendre une clause* E*-égale à* $(e'_2[r'_2\sigma]_{o'_2} \simeq r'_2\sigma) \vee D'_2\sigma \vee D'_2\sigma$*, où* $e'_2[r'_2\sigma]$ *est* E*-égal à* $w_1[r_2^e]$*.*
  - *Si la position* o_2 *du contexte* (e_2, o_2) *n'est pas vide, alors une* E*-paramodulation étendue appliquée entre* $C'_2\sigma$ *et* $C'_2\sigma$*, avec les contextes* (e_2, o_2) *et* (e'_2, o'_2)*, engendre une clause* E*-égale à* $(e_2[r'_2\sigma]_{o_2} \simeq e'_2[r'_2\sigma]_{o'_2}) \vee D'_2\sigma \vee D'_2\sigma$*, où* $e_2[r'_2\sigma] =_E r_2^e$ *et* $e'_2[r'_2\sigma] =_E w_1[r_2^e]$*.*

  Si l'atome $(w_1[r_2^e] \simeq r_2^e)$ *est falsifié par* K*, cette interprétation falsifie la clause déduite dans chacun des cas, ce qui contredit la construction de la branche droite.*
- *Soit à l'aide d'une réduction dans un sous-terme strict de* l_2^e*, correspondant à l'occurrence d'une variable lors d'une étape de* E*-égalité entre* w_1 *et* l_2^e *(Cas 3(b)i). Le terme déduit est* E*-égal à* $w_1[r_2^e]$ *(propriétés de l'ordre sur les termes). Dans ce cas, nous pouvons étudier la* E*-fermeture du graphe formé par cette réduction et la* E*-égalité entre* l_2^e *et le sous-terme dans lequel la réduction a été appliquée,* E*-égalité impliquée par les propriétés de l'ordre sur les termes.*
  Ce raisonnement récursif est bien fondé, car nous considérons un sous-terme strict de l_2^e*, terme de taille finie.*

En conclusion, l'atome $(w_1[r_2^e] \simeq r_2^e)$ *est valide pour* K*. Nous en déduisons que l'atome* $(r_2^e \simeq v)$ *est interprété différemment de l'atome* $(w_1 \simeq v)$*.*

— *Etudions la propriété de E-fermeture suivante :*

$$\begin{tikzcd}
w_1 \arrow[r, leftrightarrow, "*", "E"'] \arrow[d, "\mathcal{C}_{l_1},E"'] & l_2^e \arrow[d, dotted, "\mathcal{C}_{l_1},E"'] \\
w_1[r_1] \arrow[d, dotted, "\mathcal{C}_{l_1},E"', "*"] & l_2^e[r_1^e] \arrow[d, dotted, "\mathcal{C}_{l_1},E"', "*"] \\
w_1' \arrow[r, dotted, leftrightarrow, "*", "E"'] & w_2'
\end{tikzcd}$$

où $\mathcal{C}_{l_1}$ désigne l'ensemble des contextes (e_1, o_1) du terme l_1 tels que, soit $e_1 = l_1$ et $o_1 = \epsilon$, soit il existe un contexte (e_1', o_1, c_1) de $\mathcal{C}ont(l_1)$ et une substitution close σ, solution de c_1, tels que $e_1 = e_1'\sigma$.

Dans le diagramme précédent, la réduction de w_1 en $w_1[r_1]$ est faite par l'équation $(l_1 \simeq r_1)$ à la position p_1. La théorie E étant régulière, le terme l_2^e est réductible à la position q_2 par une équation $(l_1^e \simeq r_1^e)$, extension de $(l_1 \simeq r_1)$ pour un contexte (e_1, o_1) de $\mathcal{C}_{l_1}$: $l_1^w =_E l_1^e$, où l_1^w désigne $l_2^e|_{q_2}$. o_1 ne peut être égal à ϵ, car le nœud droit de $(l_1 \simeq r_1)$ devrait alors être un nœud de quasi-échec au lieu d'un nœud d'échec (réductible modulo E par $l_2 \simeq r_2$). Il faut noter que les termes w_1' et w_2' n'ont aucun lien avec les termes appelés ainsi plus haut.

Les atomes $(l_2^e[r_1^e] \simeq v)$, $(w_2' \simeq v)$, $(w_1' \simeq v)$ et $(w_1[r_1] \simeq v)$ appartiennent tous au domaine de K, et cette interprétation étant E-consistante, ils sont tous interprétés de la même manière. Comme $K(w_1[r_1] \simeq v) \neq K(r_2^e \simeq v)$, l'atome $(l_2^e \simeq v)$ ne peut être réductible par $(l_1^w \simeq r_1^e)$ (par construction de l'arbre sémantique); l'atome $(l_1^w \simeq r_1^e)$ est donc plus grand que l'atome $(l_2^e \simeq v)$, et les termes l_1^w et l_2^e (ainsi que l_1^e) sont E-égaux.

Comme nous l'avons montré auparavant, dans cette situation, il est possible de montrer que l'interprétation K satisfait l'atome $(l_2^e[r_1^e] \simeq r_1^e)$. Cela implique que l'atome $(r_1^e \simeq v)$ est interprété comme $(w_1 \simeq v)$, et donc différemment de $(r_2^e \simeq v)$. D'où, $K(r_1^e \simeq r_2^e) = F$.

— *Pour résumer, nous disposons d'une extension $(l_1^e \simeq r_1^e)$ de $(l_1 \simeq r_1)$, et d'une extension $(l_2^e \simeq r_2^e)$ de $(l_2 \simeq r_2)$, telles que : $l_1^e =_E l_2^e$ et $K(r_1^e \simeq r_2^e) = F$. Notons (e_1', o_1, c_1) et (e_2', o_2, c_2) les contextes de $\mathcal{C}ont(l_1)$ et $\mathcal{C}ont(l_2)$ respectivement, vérifiant $e_1'\sigma = e_1$ et $e_2'\sigma = e_2$, pour une substitution close σ, solution de c_1 et c_2.*

Soient C_1 et C_2 les clauses de $\mathcal{GS}$ étiquetant les nœuds droits de $(l_1 \simeq r_1)$ et $(l_2 \simeq r_2)$. Soient $C_1' = (l_1' \simeq r_1') \vee D_1'$ et $C_2' = (l_2' \simeq r_2') \vee D_2'$ les clauses de $INF^*(S)$ telles que : $C_1 =_E C_1'\sigma$ et $C_2 =_E C_2'\sigma$, pour une substitution close σ. Une E-paramodulation étendue peut être appliquée entre ces clauses $C_1'\sigma$ et $C_2'\sigma$. La clause déduite est $C = (e_1'\sigma[r_1'\sigma]_{o_1} \simeq e_2'\sigma[r_2'\sigma]_{o_2}) \vee D_1'\sigma \vee D_2'\sigma$, appartenant donc à $\mathcal{GS}$.

L'atome $(e_1'\sigma[r_1'\sigma] \simeq e_2'\sigma[r_2'\sigma])$, ou plutôt l'atome $(e_1[r_1] \simeq e_2[r_2])$ qui lui est E-égal, est falsifié par K, car $e_1[r_1] = r_1^e$ et $e_2[r_2] = r_2^e$. Ainsi, l'interprétation K falsifie la clause C de $\mathcal{GS}$, produite par cette étape de E-paramodulation étendue. □

A partir de maintenant, supposons que A_β est le plus petit atome E-égal à B permettant de détecter la E-inconsistance de Q_α. Etudions chacune des trois possibilités pour l'interprétation Q_α.

I Si K a exactement un successeur (Q_α)

Si l'interprétation partielle K admet pour unique successeur Q_α, nous allons montrer en trois lemmes que Q_α ne peut pas appartenir à la branche droite. Le premier prouve que B est réductible, le second que A_β est irréductible, et le dernier soulève la contradiction finale après avoir montré que A_β était falsifié par K.

Lemme A.3 $B = (u_2 \simeq v) \longrightarrow_K^{g \simeq d} (u_2[d] \simeq v)$ *où* $g \succ d$.

Preuve: *Démontrons ce lemme en deux étapes :*

1. ***B* est réductible par ($g \simeq d$)** *; par construction de l'arbre sémantique (Théorème 4.2), K n'a qu'une extension implique que soit B est un atome ($u \simeq u$) et $Q_\alpha(B) = V$, soit B est réductible. Or, comme nous savons que $u_2 \succ v$, B ne peut être que réductible. Notons ($g \simeq d$) une équation irréductible (Théorème de la Réduction) réduisant B ; de plus, v étant irréductible, B est réductible dans u_2 : $(u_2 \simeq v) \longrightarrow_K^{g \simeq d} (u_2[d] \simeq v)$.*
2. **$g \neq_E d$** *car K étant E-consistant, B ne peut être réduit en un atome E-égal.*

Ainsi, B est réductible par une équation ($g \simeq d$) où $g \succ d$. □

Lemme A.4 *A_β est irréductible.*

Preuve: *Maintenant, nous allons étudier l'atome A_β, prouvant qu'il est irréductible.*
Si A_β est réductible, il est réductible par une équation ($l \simeq r$) telle que $l \succ r$, car l et r ne peuvent être E-égaux, A_β ayant été choisi minimal (Proposition 4.18). Cependant, nous savons que B est également réductible par une équation ($g \simeq d$) où $g \succ d$, et dans ce cas le Lemme A.2 nous permet de dire que K falsifie une clause de $\mathcal{GS}$, ce qui contredit le fait qu'il appartient à $MCT(\mathcal{GS})$.
Nous en déduisons que A_β est irréductible et ainsi qu'il appartient à W_α (Définition 4.3 de la E-inconsistance d'une interprétation). □

Lemme A.5 *K falsifie A_β et Q_α ne peut pas être un nœud de la branche droite.*

Preuve: *Nous allons montrer qu'un nœud de quasi-échec devrait être placé au niveau de A_β, et donc que Q_α ne peut pas appartenir à la branche droite. Mais, montrons d'abord que A_β est falsifié par K, car A_β appartient au domaine de K.*
Si nous supposons que A_β est valide dans K, et donc que B est falsifié par Q_α, le nœud droit R de A_β est

- *soit un nœud d'échec étiqueté par une clause $C_1 = (u_1 \simeq v) \vee D_1$ de $\mathcal{GS}$ (car $u_1 \neq_E v$) ; B étant réductible par une équation ($g \simeq d$) où $g \succ d$, la Proposition 4.16 nous permet de trouver un atome E-égal à B et réductible par une équation irréductible dont le nœud droit est un nœud d'échec étiqueté par une clause C_2 de $\mathcal{GS}$. Alors, par la Proposition 4.17, une E-paramodulation (contextuelle) peut être appliquée de C_2 dans C_1 pour produire une clause de $\mathcal{GS}$ falsifiée par Q_γ, ce qui contredit l'appartenance de Q_γ à $MCT(\mathcal{GS})$.*

- *soit un nœud de quasi-échec étiqueté par une équation* $(u_1 \simeq u_3)$ *telle que* $u_1 =_E u_3$ *;* A_β *ayant été choisi minimal, l'atome* $(u_3[l] \simeq v)$ *est réductible par une équation* $(l \simeq r)$*, où* $l \succ r$*, en* $(u_3[r] \simeq v)$*. De plus, par définition d'un nœud de quasi-échec,* $R(u_3[r] \simeq v)$ *est différent de* $R(u_1 \simeq v)$*, c'est-à-dire* $R(u_3[r] \simeq v) = V$*. Mais, nous avons donc deux atomes* E*-égaux* B *et* $(u_3 \simeq v)$ *interprétés différemment et réductibles. Nous pouvons appliquer le Lemme A.2 qui nous dit que* K *falsifie une clause de* $\mathcal{GS}$*, et donc ne peut pas appartenir à la branche droite.*

Finalement, A_β *est irréductible et* $K(A_\beta) = F$*. Mais, comme* B *est réductible par* $(g \simeq d)$*,* $g \succ d$ *et* $Q_\alpha(B) = V$*,* R *devrait être un nœud de quasi-échec, et donc* Q_α *ne devrait pas appartenir à la branche droite.* □

II Si K a exactement deux successeurs et Q_α est son successeur gauche

Si K admet deux extensions L et R, $L(B) = V$ et $R(B) = F$, et Q_α est l'extension gauche, c'est-à-dire L, alors nous montrons ci-dessous que Q_α ne peut pas appartenir à la branche droite.

Lemme A.6 Q_α *ne peut pas être un nœud de la branche droite.*

Preuve : *Comme* Q_α *est l'extension gauche de* K*,* $Q_\alpha(B) = V$*,* $K(A_\beta) = F$*, et l'extension droite* R *est soit un nœud d'échec, soit un nœud de quasi-échec. Alors, nous pouvons en déduire les faits suivants :*

1. $\boldsymbol{A_\beta}$ **est réductible** *: si l'atome* A_β *est irréductible, il appartient au domaine de* K *et*
   - *soit* R *est un nœud de quasi-échec, et un même nœud de quasi-échec devrait couper la branche droite au niveau du nœud droit de* A_β*,*
   - *soit* R *est un nœud d'échec étiqueté par une clause* $(u_2 \simeq v) \vee D_2$ *de* $\mathcal{GS}$*, et un nœud d'échec devrait couper la branche droite au niveau du nœud droit de* A_β*, étiqueté par la clause* $(u_1 \simeq v) \vee D_2$ *car elle appartient également à* $\mathcal{GS}$*.*

   Donc, l'atome A_β *est réductible par une équation* $(l \simeq r)$ *en* $(u_1[r] \simeq v)$*, atome falsifié par* K*, et* $l \succ r$ *par minimalité de* A_β*. Par la Proposition 4.16, il existe un atome* E*-égal à* A_β *qui est réductible par une équation dont le nœud droit est étiqueté par une clause* C_1 *de* $\mathcal{GS}$*.*
2. $\boldsymbol{R}$ **est un nœud de quasi-échec** *: si* R *est un nœud d'échec étiqueté par une clause* $C_2 = B \vee D_R$ *de* $\mathcal{GS}$*, la Proposition 4.17 permet d'en conclure que* K *falsifie une clause de* $\mathcal{GS}$*, produite par une* E*-paramodulation ou une* E*-paramodulation contextuelle de* C_1 *dans* C_2*.*

   Ainsi, R *est un nœud de quasi-échec étiqueté par une équation* $(u_2 \simeq u_3)$ *où* u_2 *et* u_3 *sont* E*-égaux ; le cas où l'atome* $(u_3 \simeq v)$ *est irréductible est impossible, car* K *serait alors* E*-inconsistant, validant* $(u_3 \simeq v)$ *et falsifiant* $(u_1 \simeq v)$*. Donc,* $(u_3 \simeq v) \longrightarrow_K^{g \simeq d} (u_3[d] \simeq v)$*,* $g \succ d$ *et* $K(u_3[d] \simeq v) = V$*.*

Pour résumer, nous savons que :

- $(u_1 \simeq v) \longrightarrow_K^{l \simeq r} (u_1[r] \simeq v)$ *et* $l \succ r$*,*
- $(u_3 \simeq v) \longrightarrow_K^{g \simeq d} (u_3[d] \simeq v)$ *et* $g \succ d$*,*

- $K(u_1[r] \simeq v) \neq K(u_3[d] \simeq v)$.

Le Lemme A.2 peut alors être appliqué, et nous en déduisons que K falsifie une clause de $\mathcal{GS}$ et ne devrait donc par appartenir à $MCT(\mathcal{GS})$. En conséquence, Q_α ne peut être un nœud de la branche droite. □

III Si K a exactement deux successeurs et Q_α est son successeur droit

Si K admet deux extensions L et R, $L(B) = V$ et $R(B) = F$, et Q_α est l'extension droite, c'est-à-dire R, nous montrons dans le lemme ci-dessous que Q_α ne peut pas appartenir à la branche droite.

Lemme A.7 *Q_α n'est pas un nœud de la branche droite.*

Preuve : *Q_α étant l'extension droite de K, $Q_\alpha(B) = F$ et $K(A_\beta) = V$. Nous avons alors deux cas à étudier :*

- **Si A_β est réductible** *par une équation $(g \simeq d)$ en $(u_1[d] \simeq v)$, g et d peuvent être choisis tels que $g \succ d$, par minimalité de A_β et par définition de la E-inconsistance de Q_α. Alors, Q_α devrait être un nœud de quasi-échec étiqueté par $(u_2 \simeq u_1[l])$ et ainsi ne pas appartenir à la branche droite.*
- **Si A_β est irréductible**, *Q_α devrait encore être un nœud de quasi-échec étiqueté par $(u_2 \simeq u_1)$ (premier cas de la définition d'un nœud de quasi-échec).*

Dans les deux cas, Q_α ne peut être un nœud de la branche droite. □

A.2 Stratégie de superposition

Commençons par rappeler la situation dans laquelle nous nous trouvons. A partir d'un ensemble de clauses E-incohérent S, nous avons défini un ensemble $\mathcal{GS}$, contenant l'ensemble des instances closes des clauses pouvant être engendrées à partir de S, par nos règles d'inférence (définies dans la Section 2.2.2). Les clauses de $\mathcal{GS}$ servent à couper toutes les branches de l'arbre sémantique au niveau de nœuds appelés *nœuds d'échec*. Le sous-arbre élagué est appelé *arbre sémantique cohérent*, et noté $MCT(\mathcal{GS})$. Pour montrer que notre ensemble de règles d'inférences est réfutationnellement complet, il faut montrer que ce sous-arbre est vide. Ceci est réalisé en sélectionnant une branche de ce sous-arbre, appelée *branche droite* (Définition 4.21), principale différence avec la preuve de la stratégie de E-paramodulation, et en démontrant qu'elle ne peut être que vide (Proposition 4.25).

Cependant, cette branche droite doit être E-consistante (Définition 4.3). Cette propriété a été annoncée dans la Proposition 4.24, et nous allons détailler sa preuve dans cette section, sous la forme de plusieurs lemmes. Notons que nous utiliserons quelques lemmes démontrés pour la stratégie de E-paramodulation (Section A.1).

Soit Q_γ le dernier nœud de la branche droite. Par abus de langage, Q_γ désignera cette branche droite toute entière. Dans la preuve de la Proposition 4.24, nous avons montré que Q_γ ne peut être vide si $MCT(\mathcal{GS})$ n'est pas vide. Supposons que Q_γ est E-inconsistant ; il existe un ordinal minimal α tel que Q_α est E-inconsistant. α n'est pas un ordinal limite et admet donc un prédécesseur α^-. Notons K l'interprétation partielle Q_{α^-} et B l'atome A_{α^-}. Par minimalité de α, K est E-consistant, et

$$\exists A_\beta =_E B \text{ tel que } \begin{array}{l} \bullet \text{ soit } A_\beta \in W_\alpha \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta) \\ \bullet \text{ soit } A_\beta \notin W_\alpha,\ A_\beta \longrightarrow_K^{l \simeq r} A_\beta[r] \text{ où } l \succ r, \text{ et } Q_\alpha(B) \neq Q_\alpha(A_\beta[r]) \end{array}$$

Dans le Lemme A.1, nous avons montré que les atomes B et A_β sont des équations $(u_2 \simeq v)$ et $(u_1 \simeq v)$ respectivement, où $u_1 =_E u_2$, $u_2 \succ v$ et v est irréductible. Ce lemme reste valable pour la stratégie de E-superposition. A partir de maintenant, supposons que A_β est le plus petit atome E-égal à B permettant de détecter la E-inconsistance de Q_α.

Le lemme suivant, d'énoncé identique au lemme A.2 pour la stratégie de paramodulation, reste très important pour prouver la E-consistance de la branche droite.

Lemme A.8 *Soient $(u_1 \simeq v)$ et $(u_2 \simeq v)$ deux atomes tels que $u_1 =_E u_2$, $u_1 \succ v$ et*

$$(u_1 \simeq v) \longrightarrow_K^{g_1 \simeq d_1} (u_1[d_1] \simeq v) \quad et \quad g_1 \succ d_1$$

$$(u_2 \simeq v) \longrightarrow_K^{g_2 \simeq d_2} (u_2[d_2] \simeq v) \quad et \quad g_2 \succ d_2$$

Si K est E-consistant et $K(u_1[d_1] \simeq v) \neq K(u_2[d_2] \simeq v)$, alors K falsifie une clause de $\mathcal{GS}$.

Sa preuve est similaire à celle du Lemme A.2 car elle utilise une étape de E-superposition étendue, règle qui est en fait identique à la E-paramodulation étendue ; la Proposition 4.22 ainsi que les propriétés des nœuds d'échec et d'échec distants citées au début de cette Section nous garantissent que les conditions de maximalité des littéraux sont bien respectées ; la seule différence à prendre en compte dans la preuve est que les équations irréductibles $(l_1 \simeq r_1)$ et $(l_2 \simeq r_2)$ peuvent avoir un nœud droit qui est un nœud d'échec distant au lieu d'un nœud d'échec. Si c'est le cas par exemple pour $(l_1 \simeq r_1)$, sa branche droite se termine par un nœud d'échec étiqueté par une clause C_1, au niveau d'un atome $(l_1 \simeq r_3)$. La preuve devra donc utiliser cette équation $(l_1 \simeq r_3)$ au lieu de $(l_1 \simeq r_1)$, ainsi que la clause C_1.

Etudions chacune des trois possibilités pour l'interprétation Q_α.

I Si K a exactement un successeur (Q_α)

Si l'interprétation K admet pour unique successeur Q_α, nous allons montrer que Q_α ne peut pas appartenir à la branche droite. Grâce à la preuve de complétude de la stratégie de paramodulation, nous savons déjà que l'atome $(u_2 \simeq v)$ (B) est réductible en $(u_2[d] \simeq v)$ par une équation $(g \simeq d)$ où $g \succ d$ (Lemme A.3), et $(u_1 \simeq v)$ (A_β) est irréductible (Lemme A.4) et appartient donc au domaine de K.

Lemme A.9 *K falsifie A_β et Q_α ne peut pas être un nœud de la branche droite.*

Preuve : *Nous allons montrer qu'un nœud de quasi-échec devrait être placé au niveau de A_β, et donc que Q_α ne peut pas appartenir à la branche droite. Mais, montrons d'abord que A_β est falsifié par K, car A_β appartient au domaine de K.*
Si nous supposons que A_β est valide dans K, et donc que B est falsifié par Q_α, le nœud droit R de A_β est

- *soit un nœud d'échec étiqueté par une clause* $C_1 = (u_1 \simeq v) \vee D_1$ *de* $\mathcal{GS}$ *(car* $u_1 \neq_E v$*) ;* B *étant réductible par une équation* $(g \simeq d)$ *où* $g \succ d$ *et* K *étant* E*-consistante, la Proposition 4.22 nous permet de trouver un atome* E*-égal à* B *et réductible par une équation irréductible dont le nœud droit est un nœud d'échec ou de quasi-échec étiqueté par une clause* C_2 *de* $\mathcal{GS}$*. Alors, par la Proposition 4.23, une* E*-superposition (contextuelle) peut être appliquée de* C_2 *dans* C_1 *pour produire une clause de* $\mathcal{GS}$ *falsifiée par* Q_γ*, ce qui contredit l'appartenance de* Q_γ *à* $MCT(\mathcal{GS})$*.*
  Cette étape d'inférence ne peut être faite que si le littéral $(u_1 \simeq v)$ *est maximal dans la clause* C_1*. Or, nous avons vu au début de cette Section que cette condition est vérifiée si l'atome* $(u_1 \simeq v)$ *est irréductible et s'il est défini par une interprétation* E*-consistante. Ces deux conditions sont vérifiées (la seconde par* K*).*
- *soit un nœud d'échec distant au niveau d'un atome* $(u_1 \simeq w)$*, étiqueté par une clause* $C_1 = (u_1 \simeq w) \vee D_1$ *de* $\mathcal{GS}$ *en une interprétation* M_1*. Comme* B *est réductible et* K *étant* E*-consistante, la Proposition 4.22 nous permet de trouver un atome,* E*-égal à* B *et interprété comme* B*, qui est réductible par une équation dont le nœud droit est soit un nœud d'échec soit un nœud d'échec distant. Soit* C_2 *la clause étiquetant ce nœud. Alors, par la Proposition 4.23,* M_1 *falsifie une clause de* $\mathcal{GS}$*, obtenue par* E*-superposition de* C_2 *dans* C_1*.*
  Comme dans le cas précédent, la condition de maximalité de $(u_1 \simeq w)$ *dans* C_1 *est respectée grâce aux propriétés des nœuds d'échec distants.*
- *soit un nœud de quasi-échec étiqueté par une équation* $(u_1 \simeq u_3)$ *telle que* $u_1 =_E u_3$ *;* A_β *ayant été choisi minimal, l'atome* $(u_3[l] \simeq v)$ *est réductible par une équation* $(l \simeq r)$*, où* $l \succ r$*, en l'atome* $(u_3[r] \simeq v)$*. De plus, par définition d'un nœud de quasi-échec,* $R(u_3[r] \simeq v)$ *est différent de* $R(u_1 \simeq v)$*, c'est-à-dire* $R(u_3[r] \simeq v) = V$*. Nous avons ainsi deux atomes* E*-égaux* B *et* $(u_3 \simeq v)$ *interprétés différemment et réductibles. L'application du Lemme A.8 nous dit que* K *falsifie une clause de* $\mathcal{GS}$*, et qu'il ne peut donc pas appartenir à la branche droite.*

Finalement, A_β *est irréductible et* $K(A_\beta) = F$*. Mais, comme* B *est réductible par* $(g \simeq d)$*,* $g \succ d$ *et* $Q_\alpha(B) = V$*,* R *devrait être un nœud de quasi-échec, et donc* Q_α *ne devrait pas appartenir à la branche droite.* □

II Si K a exactement deux successeurs et Q_α est son successeur gauche

Si K admet deux extensions L et R, $L(B) = V$ et $R(B) = F$, et Q_α est l'extension gauche, c'est-à-dire L, alors nous montrons ci-dessous que Q_α ne peut pas appartenir à la branche droite.

Lemme A.10 *Q_α ne peut pas être un nœud de la branche droite.*

Preuve : *Comme* Q_α *est l'extension gauche de* K*,* $Q_\alpha(B) = V$*,* $K(A_\beta) = F$*, et l'extension droite* R *est soit un nœud d'échec, soit un nœud d'échec distant, soit un nœud de quasi-échec. Nous pouvons en déduire les faits suivants :*

1. $\boldsymbol{A_\beta}$ **est réductible** *: si l'atome* A_β *est irréductible, il appartient au domaine de* K *et*
   - *soit* R *est un nœud de quasi-échec, et un même nœud de quasi-échec devrait couper la branche droite au niveau du nœud droit de* A_β*,*

- *soit R est un nœud d'échec étiqueté par une clause $(u_2 \simeq v) \vee D_2$ de $\mathcal{GS}$, et un nœud d'échec devrait couper la branche droite au niveau du nœud droit de A_β, étiqueté par la clause $(u_1 \simeq v) \vee D_2$ car elle appartient également à $\mathcal{GS}$,*
- *soit R est un nœud d'échec distant étiqueté par une clause $(u_2 \simeq w) \vee D_2$ de $\mathcal{GS}$; alors, un nœud d'échec devrait être placé au-dessus de $(u_2 \simeq w)$, au niveau de l'atome $(u_1 \simeq w)$, car la clause $(u_1 \simeq w) \vee D_2$ appartient également à $\mathcal{GS}$; notons que dans ce cas la branche droite serait coupée par un nœud d'échec distant au niveau du nœud droit de A_β.*

Donc, l'atome A_β est réductible par une équation $(l \simeq r)$ en $(u_1[r] \simeq v)$, atome falsifié par K, et $l \succ r$ par minimalité de A_β. Par la Proposition 4.22, il existe un atome E-égal à A_β qui est réductible par une équation dont le nœud droit est étiqueté par une clause C_1 de $\mathcal{GS}$.

2. ***R* est un nœud de quasi-échec** *: si R est un nœud d'échec ou d'échec distant étiqueté par une clause $C_2 = (u_2 \simeq w) \vee D_R$ de $\mathcal{GS}$ ($w = v$ si c'est un nœud d'échec), la Proposition 4.23 permet d'en conclure que K falsifie une clause de $\mathcal{GS}$, produite par une E-superposition ou une E-superposition contextuelle de C_1 dans C_2. Cette inférence est possible car le littéral $(u_2 \simeq w)$ est maximal dans C_2 selon les propriétés citées au début de cette Section.*

   Ainsi, R est un nœud de quasi-échec étiqueté par une équation $(u_2 \simeq u_3)$ où u_2 et u_3 sont E-égaux ; le cas où l'atome $(u_3 \simeq v)$ est irréductible est impossible, car K serait alors E-inconsistant, validant $(u_3 \simeq v)$ et falsifiant $(u_1 \simeq v)$. Donc, $(u_3 \simeq v) \longrightarrow_K^{g \simeq d} (u_3[d] \simeq v)$, $g \succ d$ et $K(u_3[d] \simeq v) = V$.

Pour résumer, nous savons que :

- $(u_1 \simeq v) \longrightarrow_K^{l \simeq r} (u_1[r] \simeq v)$ *et* $l \succ r$,
- $(u_3 \simeq v) \longrightarrow_K^{g \simeq d} (u_3[d] \simeq v)$ *et* $g \succ d$,
- $K(u_1[r] \simeq v) \neq K(u_3[d] \simeq v)$.

Le Lemme A.8 peut alors être appliqué, et nous en déduisons que K falsifie une clause de $\mathcal{GS}$ et ne devrait donc par appartenir à $MCT(\mathcal{GS})$. D'où Q_α ne peut être un nœud de la branche droite. □

III Si K a exactement deux successeurs et Q_α est son successeur droit

Si K admet deux extensions L et R, $L(B) = V$ et $R(B) = F$, et Q_α est l'extension droite, c'est-à-dire R, nous montrons dans le lemme ci-dessous que Q_α ne peut pas appartenir à la branche droite.

Lemme A.11 *Q_α n'est pas un nœud de la branche droite.*

Preuve : *Q_α étant l'extension droite de K, $Q_\alpha(B) = F$ et $K(A_\beta) = V$. Nous avons alors deux cas à étudier :*

- **Si A_β est réductible** *par une équation $(g \simeq d)$ en $(u_1[d] \simeq v)$, g et d peuvent être choisis tels que $g \succ d$, par minimalité de A_β et par définition de la E-inconsistance de*

Q_α. *Alors,* Q_α *devrait être un nœud de quasi-échec étiqueté par* $(u_2 \simeq u_1[l])$ *et ainsi ne pas appartenir à la branche droite.*

- **Si A_β est irréductible**, Q_α *devrait encore être un nœud de quasi-échec étiqueté par* $(u_2 \simeq u_1)$, *par définition d'un tel nœud.*

Dans les deux cas, Q_α *ne peut être un nœud de la branche droite.* $\square$

Bibliographie

[AHM89] S. Anantharaman, J. Hsiang, and J. Mzali. SbReve2: A Term Rewriting Laboratory with (AC-)Unfailing Completion. In N. Dershowitz, editor, *Proceedings 3rd Conference on Rewriting Techniques and Applications, Chapel Hill (N.C., USA)*, volume 355 of *Lecture Notes in Computer Science*, pages 533–537. Springer-Verlag, April 1989.

[BC86] A. Ben Cherifa. *Preuves de terminaison de systèmes de réécriture. Un outil fondé sur les interprétations polynomiales.* Thèse de Doctorat d'Université, Université de Nancy 1, October 1986.

[BCL87] A. Ben Cherifa and P. Lescanne. Termination of Rewriting Systems by Polynomial Interpretations and its Implementation. *Science of Computer Programming*, 9(2):137–160, October 1987.

[BD89] L. Bachmair and N. Dershowitz. Completion for Rewriting Modulo a Congruence. *Theoretical Computer Science*, 67(2-3):173–202, October 1989.

[BG90] L. Bachmair and H. Ganzinger. On Restrictions of Ordered Paramodulation with Simplification. In M. E. Stickel, editor, *Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, volume 449 of *Lecture Notes in Computer Science*, pages 427–441. Springer-Verlag, July 1990.

[BG93] L. Bachmair and H. Ganzinger. Associative-Commutative Superposition. Technical report MPI-I-93-267, Max Planck Institut für Informatik, Saarbrücken, 1993.

[BG94] L. Bachmair and H. Ganzinger. Rewrite-based Equational Theorem Proving with Selection and Simplification. *Journal of Logic and Computation*, 4(3):1–31, 1994.

[BGLS92] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic Paramodulation and Superposition. In *Proceedings 11th International Conference on Automated Deduction, Saratoga Springs (N.Y., USA)*, pages 462–476, 1992.

[BHK+88] H.-J. Bürckert, A. Herold, D. Kapur, J. Siekmann, M. E. Stickel, M. Tepp, and H. Zhang. Opening the AC-Unification Race. *Journal of Automated Reasoning*, 4(1):465–474, 1988.

[BL79] A. M. Ballantyne and D. S. Lankford. The Refutation Completeness of Blocked Permutative Narrowing and Resolution. In *Proceedings 4th Workshop on Automated Deduction, Austin (Tex., USA)*, 1979.

[BP85a] L. Bachmair and D. Plaisted. Associative Path Orderings. In *Proceedings 1st Conference on Rewriting Techniques and Applications, Dijon (France)*, volume 202 of *Lecture Notes in Computer Science*. Springer-Verlag, 1985.

[BP85b] L. Bachmair and D. A. Plaisted. Termination Orderings For Associative-Commutative Rewriting Systems. *Journal of Symbolic Computation*, 1:329–349, 1985.

[BPW89] T. B. Baird, G. E. Peterson, and R. W. Wilkerson. Complete Sets of Reductions Modulo Associativity, Commutativity and Identity. In N. Dershowitz, editor, *Proceedings 3rd Conference on Rewriting Techniques and Applications, Chapel Hill (N.C., USA)*, volume 355 of *Lecture Notes in Computer Science*. Springer-Verlag, April 1989.

[Bra75] D. Brand. Proving Theorems with the Modification Method. *SIAM Journal of Computing*, 4:412–430, 1975.

[Buc65] B. Buchberger. *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-dimensional Polynomial Ideal*. PhD thesis, University of Innsbruck (Austria), 1965. (in German).

[Buc79] B. Buchberger. A Criterion for Detecting Unnecessary Reductions in the Construction of Gröbner Bases. In *Proceedings of EUROSAM'79*, volume 72 of *Lecture Notes in Computer Science*, pages 3–21. Springer-Verlag, 1979.

[Bün91] R. Bündgen. Simulating Buchberger's Algorithm by Knuth-Bendix Completion. In R. V. Book, editor, *Proceedings 4th Conference on Rewriting Techniques and Applications, Como (Italy)*, volume 488 of *Lecture Notes in Computer Science*, pages 386–397. Springer-Verlag, April 1991.

[Bür91] H.-J. Bürckert. *A Resolution Principle for a Logic with Restricted Quantifiers*, volume 568 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1991.

[Cha94] J. Chabin. *Unification Générale par Surréduction Ordonnée Contrainte et Surréduction Dirigée*. Thèse de Doctorat d'Université, Université d'Orléans, January 1994.

[Com90] H. Comon. Solving Symbolic Ordering Constraints. *International Journal of Foundations of Computer Sciences*, 1(4):387–411, 1990.

[Com92] H. Comon. Completion of Rewrite Systems with Membership Constraints. In W. Kuich, editor, *Proceedings 18th ICALP Conference, Madrid (Spain)*, volume 623 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992.

[CZ92] R. Caferra and N. Zabel. A Method for Simultaneous Search for Refutations and Models by Equational Constraint Solving. *Journal of Symbolic Computation*, 13:613–641, 1992.

[Der82] N. Dershowitz. Orderings for Term-Rewriting Systems. *Theoretical Computer Science*, 17:279–301, 1982.

[DJ90] N. Dershowitz and J.-P. Jouannaud. Rewrite Systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*. Elsevier Science Publishers B. V. (North-Holland), 1990.

[Dom91] E. Domenjoud. *Outils pour la déduction automatique dans les théories associatives-commutatives*. Thèse de Doctorat d'Université, Université de Nancy 1, September 1991.

[Dom92] E. Domenjoud. A Technical Note on AC-Unification. The Number of Minimal Unifiers of the Equation $\alpha x_1 + \cdots + \alpha x_p \doteq_{AC} \beta y_1 + \cdots + \beta y_q$. *Journal of Automated Reasoning*, 8:39–44, 1992. Also as research report CRIN 89-R-2.

[DP93] C. Delor and L. Puel. Extension of the Associative Path Ordering to a Chain of Associative Symbols. In C. Kirchner, editor, *Proceedings 5th Conference on Rewriting Techniques and Applications, Montreal (Canada)*, volume 690 of *Lecture Notes in Computer Science*, pages 389–404. Springer-Verlag, 1993.

[Eke93] S. Eker. Some ideas on elementary AC matching. note technique, January 1993.

[For89] A. Fortenbacher. *Effizientes Rechnen in AC-Gleichungstheorien*. PhD thesis, Universität Karlsruhe (Germany), February 1989.

[GL85] G. Gottlob and A. Leitsch. On the Efficiency of Subsumption Algorithms. *Journal of the ACM*, 32(2):280–295, April 1985.

[GL86] I. Gnaedig and P. Lescanne. Proving Termination of Associative Rewriting Systems by Rewriting. In J. Siekmann, editor, *Proceedings 8th International Conference on Automated Deduction, Oxford (UK)*, volume 230 of *Lecture Notes in Computer Science*, pages 52–61. Springer-Verlag, 1986.

[Gna86] I. Gnaedig. *Preuves de terminaison des systèmes de réécriture associatifs-commutatifs : une méthode fondée sur la réécriture elle-même*. Thèse de Doctorat de Troisième Cycle, Université de Nancy 1, 1986.

[GOBS69] J. R. Guard, F. C. Oglesby, J. H. Bennett, and Settle. Semi-Automated Mathematics. *Journal of the ACM*, 16:49–62, 1969.

[Hab90] M. Haberstrau. About the Completeness of Strict Superposition. Internal report 156 RR 595, Université de Paris Sud, Paris 11. Orsay, 1990.

[Her30] J. Herbrand. Recherches sur la Théorie de la Démonstration. *Travaux de la Soc. des Sciences et des Lettres de Varsovie, Classe III*, 33(128), 1930.

[HR87] J. Hsiang and M. Rusinowitch. On Word Problem in Equational Theories. In T. Ottmann, editor, *Proceedings of 14th International Colloquium on Automata, Languages and Programming, Karlsruhe (Germany)*, volume 267 of *Lecture Notes in Computer Science*, pages 54–71. Springer-Verlag, 1987.

[HR91] J. Hsiang and M. Rusinowitch. Proving Refutational Completeness of Theorem Proving Strategies: The Transfinite Semantic Tree Method. *Journal of the ACM*, 38(3):559–587, July 1991.

[Hue72] G. Huet. *Constrained Resolution: A Complete Method for Higher Order Logic.* PhD thesis, Case Western Reserve University, 1972.

[Hul80] J.-M. Hullot. *Compilation de Formes Canoniques dans les Théories équationelles.* Thèse de Doctorat de Troisième Cycle, Université de Paris Sud, Orsay (France), 1980.

[JK86] J.-P. Jouannaud and H. Kirchner. Completion of a Set of Rules Modulo a Set of Equations. *SIAM Journal of Computing*, 15(4):1155–1194, 1986. Preliminary version in Proceedings 11th ACM Symposium on Principles of Programming Languages, Salt Lake City (USA), 1984.

[JK91] J.-P. Jouannaud and C. Kirchner. Solving Equations in Abstract Algebras: a Rule-based Survey of Unification. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic. Essays in honor of Alan Robinson*, chapter 8, pages 257–321. MIT Press, Cambridge (MA, USA), 1991.

[JL87] J. Jaffar and J.-L. Lassez. Constraint Logic Programming. In *Proceedings of the 14th Annual ACM Symposium on Principles Of Programming Languages, Munich (Germany)*, pages 111–119, 1987.

[JM90] J.-P. Jouannaud and C. Marché. Completion modulo Associativity, Commutativity and Identity (AC1). In A. Miola, editor, *Proceedings of DISCO'90*, volume 429 of *Lecture Notes in Computer Science*, pages 111–120. Springer-Verlag, April 1990.

[JO91] J.-P. Jouannaud and M. Okada. Satisfiability of Systems of Ordinal Notations with the Subterm Property is Decidable. In J. Leach Albert, B. Monien, and M. Rodríguez Artalejo, editors, *Proceedings 18th ICALP Conference, Madrid (Spain)*, volume 510 of *Lecture Notes in Computer Science*. Springer-Verlag, 1991.

[KB70] D. E. Knuth and P. B. Bendix. Simple Word Problems in Universal Algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, Oxford, 1970.

[Kir85] H. Kirchner. *Preuves par complétion dans les variétés d'algèbres.* Thèse de Doctorat d'Etat, Université de Nancy 1, 1985.

[KK89] C. Kirchner and H. Kirchner. Constrained Equational Reasoning. In *Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation, Portland (Oregon)*, pages 382–389. ACM Press, July 1989. Report CRIN 89-R-220.

[KKR90] C. Kirchner, H. Kirchner, and M. Rusinowitch. Deduction with Symbolic Constraints. *Revue d'Intelligence Artificielle*, 4(3):9–52, 1990. Special issue on Automatic Deduction.

[KMN85] D. Kapur, D. R. Musser, and P. Narendran. Only Prime Superpositions Need be Considered in the Knuth-Bendix Procedure, 1985. Computer Science Branch, Corporate Research and Development, General Electric, Schenectady, New York.

[KN92] D. Kapur and P. Narendran. Double-exponential Complexity of Computing a Complete Set of AC-unifiers. In *Proceedings 7th IEEE Symposium on Logic in Computer Science, Santa-Cruz (California, USA)*, June 1992.

[Koz91] D. Kozen. A Completeness Theorem for Kleene Algebras and the Algebra of Regular Events. In *Proceedings 6th IEEE Symposium on Logic in Computer Science, Los Alamitos (California, USA)*, pages 214–225, July 1991.

[KR87] E. Kounalis and M. Rusinowitch. On Word Problem in Horn Logic. In J.-P. Jouannaud and S. Kaplan, editors, *Proceedings 1st International Workshop on Conditional Term Rewriting Systems, Orsay (France)*, volume 308 of *Lecture Notes in Computer Science*, pages 144–160. Springer-Verlag, July 1987. See also the extended version published in Journal of Symbolic Computation, 11(1 & 2), 1991.

[KR94] H. Kirchner and C. Ringeissen. Constraint Solving by Narrowing in Combined Algebraic Domains. In P. Van Hentenryck, editor, *Proceedings 11th International Conference on Logic Programming*, pages 617–631. The MIT press, 1994.

[Lai89] M. Lai. On How to Move Mountains 'Associatively and Commutatively'. In N. Dershowitz, editor, *Proceedings 3rd Conference on Rewriting Techniques and Applications, Chapel Hill (N.C., USA)*, volume 355 of *Lecture Notes in Computer Science*, pages 187–202. Springer-Verlag, April 1989.

[Lan79] D. S. Lankford. On Proving Term Rewriting Systems are Noetherian. Technical report, Louisiana Tech. University, Mathematics Dept., Ruston LA, 1979.

[LB77a] D. S. Lankford and A. Ballantyne. Decision Procedures for Simple Equational Theories with Permutative Axioms: Complete Sets of Permutative Reductions. Technical report, Univ. of Texas at Austin, Dept. of Mathematics and Computer Science, 1977.

[LB77b] D. S. Lankford and A. Ballantyne. Decision Procedures for Simple Equational Theories with Associative Commutative Axioms: Complete Sets of Associative Commutative Reductions. Technical report, Univ. of Texas at Austin, Dept. of Mathematics and Computer Science, 1977.

[Les90] P. Lescanne. Implementation of Completion by Transition Rules + Control: ORME. In H. Kirchner and W. Wechler, editors, *Proceedings 2nd International Conference on Algebraic and Logic Programming, Nancy (France)*, volume 463 of *Lecture Notes in Computer Science*, pages 262–269. Springer-Verlag, 1990.

[LM93] X. Leroy and M. Mauny. The Caml Light System, Release 0.5. Documentation and Users' Manual. Technical report, INRIA Rocquencourt, September 1993.

[LMO+90] E. Lusk, W. McCune, R. Overbeek, S. Winker, and L. Wos. Automated Reasoning Contributes to Mathematics and Logic. In M. E. Stickel, editor, *Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, volume 449 of *Lecture Notes in Computer Science*, pages 485–499. Springer-Verlag, July 1990.

[Lov78] D. Loveland. *Automatic Theorem Proving.* Elsevier Science Publishers B. V. (North-Holland), 1978.

[LS93] C. Lynch and W. Snyder. Redundancy Criteria for Constrained Completion. In C. Kirchner, editor, *Proceedings 5th Conference on Rewriting Techniques and Applications, Montreal (Canada)*, volume 690 of *Lecture Notes in Computer Science*, pages 2–16. Springer-Verlag, 1993.

[Mar93] C. Marché. *Réécriture modulo une théorie présentée par un système convergent et décidabilité du problème du mot dans certaines classes de théories équationnelles.* Thèse de Doctorat d'Université, Université de Paris-Sud, Orsay (France), October 1993.

[McC88] W. McCune. Challenge Equality Problems in Lattice Theory. In E. Lusk and R. Overbeek, editors, *Proceedings 9th International Conference on Automated Deduction, Argonne (Ill., USA)*, volume 310 of *Lecture Notes in Computer Science*, pages 704–709. Springer-Verlag, 1988.

[MN90] U. Martin and T. Nipkow. Ordered Rewriting and Confluence. In M. E. Stickel, editor, *Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, volume 449 of *Lecture Notes in Computer Science*, pages 366–380. Springer-Verlag, 1990.

[NR91] P. Narendran and M. Rusinowitch. Any Ground Associative-Commutative Theory Has a Finite Canonical System. In R. V. Book, editor, *Proceedings 4th Conference on Rewriting Techniques and Applications, Como (Italy)*. Springer-Verlag, 1991.

[NR92a] R. Nieuwenhuis and A. Rubio. Basic Superposition is Complete. In B. Krieg-Brückner, editor, *Proceedings of ESOP'92*, volume 582 of *Lecture Notes in Computer Science*, pages 371–389. Springer-Verlag, 1992.

[NR92b] R. Nieuwenhuis and A. Rubio. Theorem Proving with Ordering Constrained Clauses. In D. Kapur, editor, *Proceedings 11th International Conference on Automated Deduction, Saratoga Springs (N.Y., USA)*, volume 607 of *Lecture Notes in Computer Science*, pages 477–491. Springer-Verlag, 1992.

[NR94] R. Nieuwenhuis and A. Rubio. AC-Superposition with Constraints: no AC-unifiers Needed. In A. Bundy, editor, *Proceedings 12th International Conference on Automated Deduction, Nancy (France)*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 545–559. Springer-Verlag, June 1994.

[NRS90] W. Nutt, P. Réty, and G. Smolka. Basic Narrowing Revisited. In C. Kirchner, editor, *Unification*, pages 517–540. Academic Press, London, 1990.

[Ous94] J. K. Ousterhout. *Tcl and the Tk Toolkit*, volume ISBN 0.201.63337.X. Addison-Wesley, 1994.

[Pau92] E. Paul. A General Refutational Completeness Result for an Inference Procedure Based on Associative-Commutative Unification. *Journal of Symbolic Computation*, 14(6):577–618, 1992.

[Pau94] E. Paul. E-Semantic Tree. Unpublished paper (E-mail: etienne.paul@issy.cnet.fr), 70 pages, 1994.

[Pet83] G. E. Peterson. A Technique for Establishing Completeness Results in Theorem Proving with Equality. *SIAM Journal of Computing*, 12(1):82–100, 1983.

[Pet90] G. E. Peterson. Complete Sets of Reductions with Constraints. In M. E. Stickel, editor, *Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, volume 449 of *Lecture Notes in Computer Science*, pages 381–395. Springer-Verlag, 1990.

[Pet91] U. Petermann. Building in Equational Theories into the Connection Method. In P. Jorrand and J. Kelemen, editors, *Fundamental of Artificial Intelligence Research*, volume 535 of *Lecture Notes in Computer Science*, pages 156–169. Springer-Verlag, 1991.

[Plo72] G. Plotkin. Building-in Equational Theories. *Machine Intelligence*, 7:73–90, 1972.

[PP91] J. Pais and G. E. Peterson. Using Forcing to Prove Completeness of Resolution and Paramodulation. *Journal of Symbolic Computation*, 11(1 & 2):3–19, 1991.

[PS81] G. E. Peterson and M. E. Stickel. Complete Sets of Reductions for Some Equational Theories. *Journal of the ACM*, 28:233–264, 1981.

[Rin93] C. Ringeissen. *Combinaison de Résolutions de Contraintes.* Thèse de Doctorat d'Université, Université de Nancy 1, December 1993.

[RN93] A. Rubio and R. Nieuwenhuis. A Precedence-Based Total AC-Compatible Ordering. In C. Kirchner, editor, *Proceedings 5th Conference on Rewriting Techniques and Applications, Montreal (Canada)*, volume 690 of *Lecture Notes in Computer Science*, pages 374–388. Springer-Verlag, 1993.

[Rob65] J. A. Robinson. A Machine-oriented Logic Based on the Resolution Principle. *Journal of the ACM*, 12:23–41, 1965.

[Rub94] A. Rubio. *Automated Deduction with Constrained Clauses.* PhD thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, April 1994.

[Rus89] M. Rusinowitch. *Démonstration automatique — Techniques de réécriture.* InterEditions, 1989.

[Rus91] M. Rusinowitch. Theorem-proving with Resolution and Superposition. *Journal of Symbolic Computation*, 11:21–49, 1991.

[RV91] M. Rusinowitch and L. Vigneron. Automated Deduction with Associative Commutative Operators. In P. Jorrand and J. Kelemen, editors, *Fundamental of Artificial Intelligence Research*, volume 535 of *Lecture Notes in Computer Science*, pages 185–199. Springer-Verlag, 1991.

[RV95] M. Rusinowitch and L. Vigneron. Automated Deduction with Associative-Commutative Operators. *Applicable Algebra in Engineering, Communication and Computation*, 6(1):23–56, January 1995. Also available as INRIA Research Report 1896, or CRIN Research Report 93-R-252.

[RW69] G. A. Robinson and L. T. Wos. Paramodulation and First-order Theorem Proving. In B. Meltzer and D. Mitchie, editors, *Machine Intelligence*, volume 4, pages 135–150. Edinburgh University Press, 1969.

[SL91] W. Snyder and C. Lynch. Basic Paramodulation. In H. Comon, editor, *Proceedings 5th International Workshop on Unification, Barbizon (France)*, July 1991.

[Sla67] J. R. Slagle. Automated Theorem-Proving with Renamable and Semantic Resolution. *Journal of the ACM*, 14:687–697, 1967.

[Sla74] J. R. Slagle. Automated Theorem-Proving for Theories with Simplifiers, Commutativity and Associativity. *Journal of the ACM*, 21(4):622–642, 1974.

[Sti81] M. E. Stickel. A Unification Algorithm for Associative-Commutative Functions. *Journal of the ACM*, 28:423–434, 1981.

[Sti84] M. E. Stickel. A Case Study of Theorem Proving by the Knuth-Bendix Method: Discovering that $x^3 = x$ Implies Ring Commutativity. In R. Shostak, editor, *Proceedings 7th International Conference on Automated Deduction, Napa Valley (Calif., USA)*, volume 170 of *Lecture Notes in Computer Science*, pages 248–258. Springer-Verlag, 1984.

[Vig93] L. Vigneron. Basic AC-Paramodulation. In F. Orejas, editor, *Proceedings of the 2nd CCL Workshop, La Escala (Spain)*, September 1993.

[Vig94a] L. Vigneron. Associative-Commutative Deduction with Constraints. In A. Bundy, editor, *Proceedings 12th International Conference on Automated Deduction, Nancy (France)*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 530–544. Springer-Verlag, June 1994.

[Vig94b] L. Vigneron. Superposition in AC Theories: Proof of Completeness by Semantic Trees. Technical report 94-R-045, Centre de Recherche en Informatique de Nancy, Vandœuvre-lès-Nancy, 1994.

[Wer92] U. Wertz. First-Order Theorem Proving Modulo Equations. Technical Report MPI-I-92-216, Max Planck Institut für Informatik, April 1992.

[Win84] F. Winkler. *The Church-Rosser Property in Computer Algebra and Special Theorem Proving: an Investigation of Critical-pairs/Completion Algorithms.* PhD thesis, Universität Linz, Austria, 1984.

[ZK88] H. Zhang and D. Kapur. First-order Theorem Proving Using Conditional Rewrite Rules. In E. Lusk and R. Overbeek, editors, *Proceedings 9th International Conference on Automated Deduction, Argonne (Ill., USA)*, volume 310 of *Lecture Notes in Computer Science*, pages 1–20. Springer-Verlag, 1988.

Liste des Figures

Index