THÈSE

présentée à l'École Normale Supérieure de Cachan

pour obtenir le grade de

Docteur de l'École Normale Supérieure de Cachan

par : Stéphanie DELAUNE

Spécialité : INFORMATIQUE

Vérification des protocoles cryptographiques et propriétés algébriques

Soutenue le 20 juin 2006

Composition du Jury :

– Ahmed BOUAJJANI	président du jury
– Hubert COMON-LUNDH	directeur de thèse
– Francis KLAY	directeur de thèse
– Denis LUGIEZ	rapporteur
– Christopher LYNCH	invité
– Michaël RUSINOWITCH	rapporteur
– Mark RYAN	examinateur

Thèse préparée au sein du Laboratoire Spécification et Vérification (LSV) et du Laboratoire Middleware et Plates-formes Avancées (MAPS) de France Télécom R&D.

Résumé

Avec le développement des réseaux de communications comme Internet, et l'essor du commerce électronique, le besoin d'assurer la sécurité des échanges a considérablement augmenté. Les communications « sécurisées » sont réalisées par l'utilisation de petits programmes appelés protocoles cryptographiques qui peuvent être attaqués même en présence d'un chiffrement parfait. De telles failles, qualifiées de « failles logiques », sont souvent subtiles et difficiles à déceler à la simple vue du texte du protocole.

Le problème de la vérification des protocoles cryptographiques est un problème difficile pouvant être vu comme un cas particulier de model-checking où l'environnement considéré est un environnement hostile. Compte tenu de la difficulté du problème de vérification, l'hypothèse du *chiffrement parfait* a été introduite. Elle signifie que l'on suppose l'existence d'un chiffrement « incassable » et que, par conséquent, le seul moyen d'obtenir le contenu d'un message chiffré est de connaître la clef de déchiffrement. Cette hypothèse a permis d'obtenir de nombreux résultats, mais elle est trop forte dans certaines situations. En effet, certaines primitives cryptographiques possèdent des propriétés algébriques qu'un agent malveillant peut exploiter pour réaliser une attaque.

Dans cette thèse, nous poussons les limites de l'analyse des protocoles au-delà de cette hypothèse. En particulier, nous proposons des procédures de décision, pour le problème de la recherche d'attaques en présence d'opérateurs satisfaisant des propriétés algébriques.

Mots-Clefs

Vérification formelle, protocole cryptographique, théorie équationnelle, complexité

Abstract

Cryptographic protocols are small concurrent programs designed to guarantee the security of exchanges between participants using non-secure medium. Establishing the correctness of these protocols is crucial given the increasing number of applications, such as electronic commerce, that exchange information on the Internet. Unfortunately, the existence of cryptographic primitives such as encryption is not sufficient to ensure security. The security of exchanges is ensured by cryptographic protocols which are notoriously error prone.

The formal verification of cryptographic protocols is a difficult problem that can be seen as a particular model-checking problem in an hostile environment. To verify such protocols, a line of research consists in considering encryption as a black box and assuming that an adversary can't learn anything from an encrypted message except if he has the key. This is called the *perfect cryptography* assumption. Many results have been obtained under this assumption, but such an assumption is too strong in general. Some attacks exploit in a clever way the interaction between protocol rules and properties of cryptographic operators.

In this thesis, we relax the perfect cryptography assumption by taking into account some algebraic properties of cryptographic primitives. We give decision procedures for the security problem in presence of several algebraic operators.

Keywords

Formal verification, cryptographic protocol, equational theory, complexity

Remerciements

« L'habileté à formuler les remerciements caractérise le spécialiste de haut vol [...]. Toute recherche sans dette est une recherche suspecte [...]. »

Umberto ECO, *Comment voyager avec un saumon.*

JE tiens à dire que ces derniers mots posés sont loin d'être les plus évidents. Mais, comme tout travail de thèse est assisté par d'autres esprits et d'autres mains que ceux de son seul auteur, je vais donc me plier à cet exercice un peu particulier. En espérant que ce petit texte sera à la hauteur de la reconnaissance et de l'amitié que je porte aux nombreuses personnes qui ont contribué à ce travail.

En tout premier lieu, je tiens à remercier tous les membres de mon jury. Merci à Denis Lugiez et à Michaël Rusinowitch pour avoir accepté la lourde tâche de rapporteurs. Merci à Chris Lynch et Mark Ryan pour avoir participé à ce jury, et à Ahmed Bouajjani pour l'avoir présidé. Enfin, je remercie Hubert Comon-Lundh et Francis Klay pour m'avoir encadrée durant cette thèse. Francis a été mon directeur de thèse à France Télécom. Il a toujours été disponible pour répondre à mes questions, qu'elles soient administratives ou scientifiques. Hubert Comon-Lundh a été mon directeur de thèse au LSV à Cachan. Son enthousiasme, son soutien et ses conseils avisés ont facilité mon travail de thèse. Tout au long de ces trois années, il a su orienter mes recherches aux bons moments.

Je souhaite aussi remercier les autres chercheurs avec qui j'ai eu la chance de travailler. Merci aux enseignants de l'université Paris 7, ils ont su me montrer la voie de la recherche, et sans eux je n'en serais pas là. En particulier, je remercie Ahmed Bouajjani, pour m'avoir accueilli dans son équipe pendant mon stage de Maîtrise, et Michaela Sighireanu pour m'avoir encadrée. Je tiens également à remercier Florent Jacquemard pour m'avoir encadré pendant mon stage de DEA et pour sa disponibilité au début de cette thèse. Les voyages forment la jeunesse et au cours de cette thèse, j'ai eu l'occasion d'expérimenter cette maxime. Je remercie particulièrement Mark Ryan et Chris Lynch pour m'avoir accueilli dans leur équipe respective et pour m'avoir beaucoup appris pendant ces séjours.

Le laboratoire MAPS (France Télécom R&D) et le LSV (ENS de Cachan) sont deux endroits offrant un cadre de travail idéal. Je tiens à remercier tous leurs membres pour leur disponibilité et leur gentillesse. Merci en particulier aux organisateurs d'animations en tout genre : matchs de foot, pique-nique, goûters, batailles de boules de neige, ...

Enfin, pour ceux qui ne le savent pas, le LSV est également une réserve d'animaux en tout genre (singes, panthère, marmotte, mouton, lion, ...). Ces derniers sont parfois maltraités.

Je lance donc un appel pour que ces captures et ces tortures, déjà dénoncés dans [Cor03], cessent.

Finalement, un merci particulier à Nico, qui a eu la lourde tâche de me soutenir durant ces dernières années. Sa patience et ses encouragements m'ont aidé à mener à bien cette thèse. Merci du temps qu'il a consacré à redonner un peu de rigueur a ma plume. Je tiens aussi à remercier Totoy et Toyette pour avoir assisté aux nombreuses répétitions de mon exposé. Enfin, je n'oublie pas mes parents pour leur soutien et pour avoir organisé le plus important : le pot !

Table des matières

1 Introduction **9**

1.1 Enjeux . . . 9

1.2 Protocoles cryptographiques . . . 10

1.2.1 Cryptographie . . . 10

1.2.2 Protocoles . . . 11

1.3 Vérification des protocoles cryptographiques . . . 11

1.3.1 Modélisation . . . 12

1.3.2 Propriétés de sécurité . . . 12

1.3.3 Méthodes de vérification . . . 13

1.4 Affaiblissement de l'hypothèse du chiffrement parfait . . . 14

1.5 Contributions . . . 16

1.6 Plan de la thèse . . . 17

I Modélisation des protocoles cryptographiques **19**

2 Messages **21**

2.1 Primitives cryptographiques « classiques » . . . 22

2.1.1 Concaténation . . . 22

2.1.2 Chiffrement . . . 22

2.2 Propriétés algébriques . . . 23

2.2.1 Retour sur les primitives « classiques » . . . 23

2.2.2 Associativité (A) . . . 25

2.2.3 Commutativité (C) . . . 25

2.2.4 « Ou » exclusif (ACUN) . . . 26

2.2.5 Groupe abélien (AG) . . . 26

2.2.6 Signature en aveugle . . . 27

2.2.7 Propriété préfixe . . . 27

2.2.8 Homomorphisme . . . 28

2.2.9 Distributivité du chiffrement . . . 28

2.2.10 Diffie-Hellman (DH) . . . 29

2.3 Modélisation des messages . . . 29

2.3.1 Termes, sous-termes . . . 30

2.3.2 Unification . . . 31

2.3.3 Systèmes de réécriture . . . 32

3 Intrus, protocoles **35**
3.1 Intrus 36
3.1.1 Contrôle du réseau 36
3.1.2 Capacités de déduction 36
3.1.2.1 Modèle à base d'un système d'inférence avec filtrage 36
3.1.2.2 Modèle à base d'un système d'inférence simple 38
3.2 Protocoles 40
3.2.1 Modèles par rôles 41
3.2.1.1 Modèle par rôles avec filtrage 41
3.2.1.2 Modèle par rôles avec tests d'égalités 42
3.2.2 Rôles bien formés 43
3.2.2.1 Critère de décision dans le modèle avec tests d'égalités . . . 45
3.2.2.2 Critère de décision dans le modèle avec filtrage 45
3.3 Comparaison du point de vue de l'expressivité 47
3.3.1 Système d'inférence avec filtrage et système d'inférence simple 47
3.3.2 Modèle par rôles avec filtrage et modèle avec tests d'égalités 51

4 Problèmes de vérification dans les deux formalismes **55**
4.1 Intrus passif 56
4.1.1 Système d'inférence 57
4.1.2 Localité 58
4.1.3 Déductibilité en un pas 59
4.2 Intrus actif 60
4.2.1 Systèmes de contraintes symboliques 62
4.2.2 Construction du système de contraintes 63
4.2.2.1 Algorithme 63
4.2.2.2 Exemples 65
4.2.3 Systèmes de contraintes bien formés 69

II Vérification des protocoles cryptographiques **73**

5 Vérification dans le modèle par rôles avec tests d'égalités **75**
5.1 Cadre 76
5.1.1 Théories équationnelles considérées 76
5.1.2 Exemples 78
5.2 Problème de déduction de l'intrus 80
5.2.1 Système d'inférence 80
5.2.2 Résultat de localité 81
5.2.3 Déductibilité en un pas 83
5.3 Résolution de systèmes de contraintes bien formés 83
5.3.1 Procédure 84
5.3.2 Terminaison et Complexité 84
5.3.3 Correction 87
5.3.4 Complétude 88
5.3.5 NP-difficulté 95
5.3.6 Illustration de la procédure 96

5.4 Comparaisons . . . 99
5.4.1 Travaux de M. Abadi et V. Cortier . . . 99
5.4.2 Travaux de M. Baudet . . . 100
5.4.3 Travaux de Y. Chevalier et M. Rusinowitch . . . 100

6 Vérification dans le modèle par rôles avec filtrage 103
6.1 Cadre . . . 104
6.1.1 Théories équationnelles considérées : ACh, ACUNh et AGh . . . 104
6.1.2 Applications . . . 105
6.2 Problème de déduction de l'intrus . . . 108
6.2.1 Système d'inférence . . . 108
6.2.2 Résultat de localité . . . 110
6.2.3 Déductibilité en un pas . . . 112
6.2.3.1 Procédure . . . 112
6.2.3.2 Complexité . . . 114
6.2.4 Comparaison avec les travaux de P. Lafourcade *et al.* . . . 115
6.3 Résolution de systèmes de contraintes bien formés . . . 116
6.3.1 Existence d'une solution conservatrice . . . 116
6.3.2 De la résolution dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ à la résolution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$ . . . 121
6.3.3 Réduction de la signature . . . 125
6.3.4 À propos des systèmes bien formés . . . 126
6.3.4.1 Une nouvelle caractérisation des systèmes bien formés . . . 126
6.3.4.2 Bonne formation des systèmes obtenus après abstraction . . . 130
6.3.5 Résolution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$ sur signature réduite . . . 135
6.4 Résultats d'indécidabilité . . . 142
6.4.1 Théorie ACh . . . 142
6.4.2 Théorie AGh . . . 143
6.4.2.1 Indécidabilité dans $(\mathcal{I}_{\mathsf{M_E}}, \mathsf{AGh})$ . . . 144
6.4.2.2 Indécidabilité dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$ . . . 147
6.5 Discussion . . . 148
6.6 Comparaisons . . . 150
6.6.1 Travaux de J. Millen et V. Shmatikov . . . 150
6.6.2 Travaux de Y. Chevalier et M. Rusinowitch . . . 150

7 Réduction de la théorie équationnelle 153
7.1 Étude de cas : protocole de porte-monnaie électronique . . . 154
7.1.1 Description du protocole . . . 154
7.1.2 Modélisation du protocole . . . 156
7.2 Propriété de variants finis . . . 157
7.2.1 Préliminaires . . . 158
7.2.2 Première caractérisation . . . 161
7.2.3 Principale application . . . 163
7.3 Comment établir la propriété de variants finis ? . . . 166
7.3.1 Deuxième caractérisation . . . 166
7.3.2 En présence d'axiomes orientables . . . 168
7.3.3 En présence d'axiomes non-orientables . . . 169
7.4 Application à différentes théories équationnelles . . . 170

7.4.1 Variantes de Dolev-Yao . . . 170
7.4.2 Théorie « signature en aveugle » . . . 171
7.4.3 Théorie ACUN . . . 171
7.4.4 Théorie AG . . . 172
7.4.5 Théorie DH . . . 174
7.4.6 Théorie équationnelle de l'étude de cas . . . 176
7.5 Discussion . . . 178
7.5.1 Travaux sur l'unification . . . 178
7.5.2 Travaux de H. Comon-Lundh et V. Cortier . . . 178
7.5.3 Travaux de V. Bernat et H. Comon-Lundh . . . 179
7.5.4 De la difficulté à résoudre le cas AC . . . 179

8 Conclusion et perspectives **181**

Bibliographie **185**

Index **193**

Chapitre 1

Introduction

« J'ai inventé un code formidable, il nous a dit Geoffroy. C'est un code secret que nous serons seuls à comprendre, ceux de la bande. [...] Il y a des gestes différents pour toutes les lettres: on se gratte l'oreille, on se frotte le menton, on se donne des tapes sur la tête, comme ça jusqu'à « z », où on louche. Terrible ! »

René GOSCINNY, *Le petit Nicolas et les copains.*

1.1 Enjeux

PENDANT longtemps, du code de César à la machine Enigma, la cryptographie fut un domaine réservé aux militaires. Elle fait aujourd'hui partie de notre vie quotidienne. Il est très pratique de contrôler ses comptes, ou de commander son billet de train depuis chez soi, grâce à Internet. Souvent, nous donnons notre numéro de carte bancaire sans se soucier de quoi que ce soit. Circule-t-il en clair sur le réseau ? Est-ce bien au commerçant qu'il est transmis, et non à quelqu'un qui serait en train d'usurper son identité ?

Avec le développement des réseaux de communications comme Internet, et l'essor du commerce électronique, le besoin d'assurer la sécurité des échanges, alors même que les acteurs sur le réseau ne sont pas forcément fiables, a considérablement augmenté. Les transactions utilisant ce médium sont de plus en plus nombreuses (*e.g.* service bancaire, porte-monnaie électronique, vote électronique) et les aspects sécuritaires occupent une place importante dans toutes ces applications. Toutes ces applications demandent des garanties de sécurité élevées, portant sur des propriétés de secret et d'authenticité, mais aussi de nombreuses autres propriétés, parmi lesquelles, la non-duplication (des factures, dans l'intérêt du client), la non-révocation (des commandes, dans l'intérêt du commerçant),...

Selon une étude récente [Big05], réalisée par le Centre de Recherche pour l'Étude et l'Observation des Conditions de Vie, le commerce électronique représente près de la moitié de l'activité de l'ensemble de la vente à distance alors que celui-ci ne représentait que 8% de la vente à distance en France en l'an 2000. Mais, les Français s'inquiètent toujours de la sécurité des paiements sur Internet. Tout du moins, c'est le principal facteur qu'ils mettent en avant pour expliquer leurs hésitations à effectuer des achats via le réseau. Cette réponse arrive très largement devant les autres, recueillant 41% des suffrages.

Ces protocoles, destinés à « sécuriser » les échanges, interviennent dans des communications électroniques et ont des caractéristiques spécifiques. On les trouve en très grand nombre, souvent sous la forme de petites variantes d'un protocole connu et une faille de sécurité dans l'un de ces protocoles peut avoir des conséquences économiques graves, en particulier à cause de leur déploiement à grande échelle. Il est donc crucial de s'assurer que ces protocoles remplissent parfaitement leurs tâches.

1.2 Protocoles cryptographiques

Les protocoles cryptographiques sont des petits programmes informatiques. Ils régissent les échanges entre plusieurs intervenants, et ont pour but de sécuriser les communications. Pour assurer ces propriétés de sécurité, des moyens algorithmiques tels que les chiffrements et les fonctions à sens unique ont été mis au point.

1.2.1 Cryptographie

La cryptographie, véritable science régissant le codage de l'information, a connu une véritable explosion avec le développement des systèmes informatiques, passant d'une ère artisanale à des systèmes de très hautes technologies. Elle a connu un large essor avec le développement des réseaux de communications modernes et le besoin de protéger les données échangées pour respecter les individus.

Le fait de coder un message de façon à le rendre secret s'appelle *chiffrement*. La méthode inverse consistant à retrouver le message original, est appelée *déchiffrement*. Le chiffrement se fait généralement à l'aide d'une clef et on distingue deux types d'algorithmes de chiffrement. Le chiffrement *symétrique* ou *à clef secrète* où, les clefs utilisées pour chiffrer et déchiffrer sont identiques (*e.g.* DES, AES), et le chiffrement *asymétrique* où, les clefs sont différentes. Ce deuxième type d'algorithmes, plus difficile à mettre en œuvre, n'est apparu que bien plus tard (*e.g.* chiffrement RSA mis au point par R. Rivest, A. Shamir et L. Adleman [RSA78]).

Un peu d'histoire ...
La cryptographie existe depuis de nombreux siècles et a parfois pris une part importante dans notre histoire. Citons par exemple, la méthode de chiffrement très ancienne connue sous le nom de « chiffrement de César ». Cette méthode, encore utilisée à l'heure actuelle mais uniquement par les écoliers, consiste à décaler chaque lettre de l'alphabet d'un nombre convenu à l'avance constituant la clef de chiffrement. Pendant la conquête de la Gaule, cet algorithme permettait à Jules César d'envoyer ses instructions aux centurions sans que l'ennemi puisse les lire s'il arrivait à les intercepter. Ce codage, très simple, a fini par être connu et les empereurs suivants ont dû chercher des moyens plus élaborés pour coder leurs messages.

Au début des années 20, une machine à chiffrer du nom d'Enigma est apparue sur le marché. Lors de la Seconde Guerre Mondiale, elle va servir à chiffrer les messages confidentiels, laissant perplexes les alliés pendant plusieurs années. Ces derniers vont tout entreprendre pour la « casser ». Une fois la machine récupérée, c'est le britannique Alan Turing qui réussira à déjouer la machine Enigma.

Depuis la Seconde Guerre Mondiale, les besoins en cryptographie ont explosé. Les applications civiles du chiffrement (banques, télécommunications, informatique...) deviennent un moteur fondamental de progrès. Les algorithmes utilisés à l'heure actuelle reposent sur des propriétés mathématiques, issues en particulier de la théorie des nombres.

1.2.2 Protocoles

Les communications « sécurisées » sont assurées par des protocoles cryptographiques qui utilisent ces moyens algorithmiques mais sont constitués de plusieurs messages. La description d'un protocole cryptographique est en général assez simple, courte, constituée d'une dizaine d'échanges de messages au plus.

Depuis les années 80, on dispose d'algorithmes de chiffrement suffisamment sûrs, mais même si ces moyens algorithmiques remplissent parfaitement leur rôle, les propriétés de sécurité ne sont pas pour autant toujours satisfaites. Plusieurs exemples célèbres ont montré que ces protocoles peuvent être attaqués (« man-in-the-middle attaque », « attaque par rejeu », « attaque par dictionnaire », ...) même en présence d'un chiffrement parfait [CJ97, Spo]. Les failles qui permettent de telles attaques sont qualifiées de *failles logiques* et on s'accorde pour penser que l'étude des failles logiques des protocoles est orthogonale à l'étude des failles du système de cryptographie sous-jacent. On pourrait penser qu'il est aisé de s'assurer de la fiabilité d'un protocole, mais ces failles sont relativement subtiles, difficiles à déceler à la simple vue du texte du protocole. Un exemple célèbre est la fameuse attaque découverte par G. Lowe [Low96] sur le protocole de R. Needham et M. Schroeder [NS78], une quinzaine d'années après sa publication. Et pourtant, R. Needham et M. Schroeder étaient conscients de la difficulté de mettre au point de tels protocoles – ils concluaient leur article avec le paragraphe suivant :

> *« Finally, protocols such as those developed here are prone to extremely subtle errors that are unlikely to be detected in normal operation. The need for techniques to verify the correctness of such protocols is great, and we encourage those interested in such problems to consider this area. »*

1.3 Vérification des protocoles cryptographiques

La vérification des protocoles cryptographiques est un problème difficile pouvant être vu comme un cas particulier de model-checking où l'environnement considéré est un environnement hostile représenté par un intrus. Compte tenu de la difficulté du problème de vérification, l'hypothèse dite du *chiffrement parfait* a été introduite. Elle signifie que l'on suppose l'existence d'un algorithme de chiffrement « incassable », *i.e.* un algorithme avec lequel il est impossible d'obtenir le contenu d'un message chiffré sans connaître la clef de déchiffrement. Cette hypothèse ne correspond pas à la réalité, mais elle permet de distinguer la vérification de la robustesse des algorithmes de chiffrement, de la détection de failles logiques dans les protocoles. Elle a permis d'automatiser la vérification des protocoles et de découvrir de nombreuses failles logiques. D'autre part, cette hypothèse est aujourd'hui, au moins en partie, formellement justifiée [MW04].

Les dix dernières années ont vu une profusion de résultats dans le domaine de la vérification automatique des protocoles cryptographiques. Mais ces résultats, obtenus dans des modèles différents, sont parfois incomparables et il n'est pas rare qu'un protocole soit prouvé correct dans un modèle alors qu'il est montré attaquable dans un autre. Pour ajouter à cette confusion, les propriétés dépendent du modèle sous-jacent, et les définitions formelles associées à une même propriété sont souvent incomparables.

1.3.1 Modélisation

La première étape du processus de vérification d'un protocole est sa modélisation. Cette étape de modélisation est un passage obligé extrêmement délicat. C'est en effet sur elle que repose la confiance que l'on peut avoir dans le résultat de vérification formelle. De nombreux modèles de spécification des protocoles cryptographiques existent dans la littérature.

Certains modèles, et en particulier ceux utilisés dans cette thèse, utilisent une notation sous forme de règles de réécriture pour représenter les échanges de messages effectués par chacun des participants du protocole (*cf.* chapitre 3). Ce type de modèle, introduit par I. Cervesato *et al.* [CDL$^+$99], a été repris et largement utilisé par la suite (*e.g.* [RT03, CLS03, MS05]). Il est assez proche du modèle des *strand spaces* [THG99] très répandu dans la communauté américaine. Cependant, d'autres types de modélisation existent. Citons par exemple, les modèles à base de clauses de Horn (*e.g.* [Bla01]), le modèle de Millen-Rueß [MR00] et les algèbres de processus (*e.g.* [Sch96, AG97]). L'avantage de cette dernière approche est qu'elle permet d'exprimer des propriétés de sécurité sous forme d'*équivalence* (*i.e.* bisimulation entre processus).

1.3.2 Propriétés de sécurité

Compte tenu de la diversité des applications, il n'est pas surprenant qu'il existe de nombreuses propriétés de sécurité que l'on puisse vouloir exiger d'un protocole. Les propriétés listées ci-dessous ont pour but d'illustrer cette diversité et ne constituent en aucun cas une liste exhaustive.

Secret. C'est sans aucun doute la propriété la plus étudiée encore à l'heure actuelle. Nous dirons qu'une donnée est secrète si l'intrus n'est pas en mesure d'apprendre cette donnée. Cette notion de secret est généralement suffisante. Cependant, dans le cadre particulier des algèbres de processus (*e.g.* spi-calcul [AG97], pi-calcul appliqué [AF01]), ce que l'on appelle « secret » est en fait une propriété plus forte permettant d'assurer que deux exécutions particulières du protocole faisant intervenir, pour l'une, la donnée secrète s, et pour l'autre, une autre donnée s', sont indistinguables. Une telle propriété prend en compte, par exemple, la capacité de l'intrus à comparer les messages, ce qui n'est généralement pas le cas dans les autres modèles.

Par ailleurs, on peut vouloir qu'une donnée soit secrète tout le temps (on parle alors de *secret long-terme*) ou simplement que la confidentialité de cette donnée soit préservée pendant un laps de temps, par exemple jusqu'à la fin de l'exécution de l'instance du protocole (encore appelée « session ») ayant engendrée cette donnée. Ce deuxième type de secret, appelé *secret court-terme*, est généralement plus difficile à modéliser puisqu'il faut pouvoir parler du début et de la fin d'une session. Cependant, lorsque l'on se place dans le cadre d'un nombre borné de sessions, la modélisation d'un secret court-terme ne pose aucun problème particulier.

Authentification. L'authentification est une propriété très courante. Informellement, elle permet à un agent de s'assurer de l'identité de son correspondant. Le problème est que la formalisation de cette propriété dépend de ce que l'on entend par « vérifier l'identité de son interlocuteur ». Cette propriété est assez difficile à énoncer formellement et dès lors que l'on essaie de donner une formalisation précise, on obtient de nombreuses définitions [Low97] et le lien entre celles-ci n'est pas toujours clair.

Certaines formes d'authentification expriment simplement le fait qu'il n'est pas possible que l'initiateur du protocole termine son exécution avec un agent B alors que ce dernier n'a pas commencé à exécuter le protocole. On peut également vouloir exiger que l'exécution du protocole du point de vue de B ait bien eu lieu apparemment avec A. D'autre formes d'authentification existent et permettent d'assurer que les agents participant à l'exécution du protocole sont d'accord sur certaines données échangées au cours du protocole.

Anonymat. Cette propriété intervient, par exemple, dans les protocoles de vote électronique : un agent ne doit pas être en mesure de faire le rapprochement entre un électeur et son vote. Plusieurs définitions formelles de l'anonymat ont été proposées. Elles reposent, en général, sur une notion d'équivalence. Par exemple, dans [KR05a], un protocole est dit anonyme si deux processus particuliers sont bisimilaires. Le premier processus correspond à un protocole dans lequel les agents A et B votent respectivement a et b alors que dans le deuxième processus, les deux agents ont échangé leur vote. Intuitivement, si ces deux processus sont équivalents, alors un observateur extérieur ne peut pas rapprocher l'électeur A ou B de son vote puisqu'il n'est pas capable de distinguer les deux situations décrites ci-dessus.

Dans le cadre des protocoles de vote, les propriétés de sécurité sont très complexes. En effet, pour être utilisable en pratique, un protocole de vote doit non seulement préserver l'anonymat mais il doit également satisfaire un ensemble de propriétés pouvant même sembler à première vue contradictoire. Par exemple, un électeur doit être convaincu que son vote a effectivement été pris en compte (propriété de vérifiabilité individuelle), mais il ne doit pas être capable de prouver comment il a voté (le vote doit être « sans reçu »). Ces propriétés spécifiques ont fait l'objet d'études particulières pour permettre leur modélisation [DKR06].

Dans cette thèse, nous nous sommes intéressés aux propriétés dites *de traces*, parmi lesquelles le secret et certaines formes d'authentification, deux propriétés dont les définitions formelles sont à l'heure actuelle suffisamment abouties.

1.3.3 Méthodes de vérification

Une fois la modélisation du protocole et de la propriété à vérifier effectuées, la vérification peut commencer. Le caractère non borné d'un certain nombres de paramètres du système à vérifier rendent l'énumération de toutes les exécutions impossible : un tel algorithme ne terminerait pas. En fait, il s'avère que ce problème est indécidable en général (*e.g.* [DLMS99]). Pour obtenir des procédures automatiques, plusieurs approches sont alors possibles. La première consiste à faire du test et a essayer différents scénarios et regarder si l'exécution de l'un d'entre eux invalide une propriété. Mais, cette approche n'offre cependant aucune garantie de sécurité : elle ne permet pas de prouver une propriété, mais seulement de la réfuter. D'autres approches, plus satisfaisantes, sont décrites ci-dessous.

Preuves par Abstraction

Contrairement au test, ces méthodes ont pour but de garantir la sécurité d'un protocole. Pour cela, les approximations proposées doivent être correctes. De nombreuses abstractions sont possibles pour prouver le secret d'un protocole ; citons par exemple, les abstractions consistant à sur-approximer la connaissance de l'intrus et à ne pas prendre en compte l'ordre d'exécution des différentes règles composant le protocole. Cette approche, utilisée dans le cadre d'un nombre non-borné de sessions par T. Genet et F. Klay [GK00], a été reprise et améliorée par d'autres par la suite. Ainsi, dans un tel modèle, lorsque l'on prouve que le secret est préservé, alors le secret est également préservé dans le modèle concret. Bien entendu, ces abstractions ne sont en général pas complètes. Autrement dit, une réponse négative à l'issue de la procédure ne permet pas d'affirmer l'existence d'une attaque sur le protocole considéré. Si l'on souhaite avoir un algorithme à la fois correct et complet, il faut s'orienter vers la recherche de classes décidables.

Classes décidables

Comme nous l'avons déjà mentionné, le problème est indécidable en général, mais cela n'empêche pas l'obtention de résultats de décidabilité pour des classes de protocoles particulières. L'un des travaux les plus anciens est sans doute l'article de D. Dolev et A. Yao [DY81] dans lequel les auteurs considèrent une classe très restreinte de protocoles : les protocoles ping-pong. Depuis, d'autres résultats de décidabilité ont été établis (*cf.* [Cor05]). Ces résultats de décidabilité, dans le cadre d'un nombre non borné de sessions, sont obtenus en faisant des hypothèses souvent très fortes sur les protocoles. Dans le cadre d'un nombre borné de sessions, c'est-à-dire lorsque l'on considère un nombre fini d'instances du protocole, les résultats sont assez satisfaisants (*e.g.* [ALV02, Bor01, RT03]). Par exemple, dans un modèle, similaire à celui que nous utilisons dans cette thèse, M. Rusinowitch et M. Turuani établissent une procédure non-déterministe s'exécutant en temps polynomial et permettant de traiter une classe de protocoles très générale [RT03].

Ces travaux, pour la plupart, reposent sur l'hypothèse du chiffrement parfait et idéalisent les primitives cryptographiques. Cette hypothèse limite, parfois d'une façon dramatique, les capacités de l'intrus.

1.4 Affaiblissement de l'hypothèse du chiffrement parfait

Compte tenu de la complexité des problèmes de vérification, un certain nombres d'hypothèses simplificatrices ont été prises, permettant d'idéaliser les primitives cryptographiques. Ainsi, les modèles généralement utilisés pour la vérification formelle de protocoles cryptographiques font une série d'hypothèses.

Générateur aléatoire parfait

L'hypothèse du *générateur aléatoire parfait* confère un caractère complètement imprévisible aux nombres, encore appelés *nonces*, engendrés aléatoirement au cours de l'exécution d'un protocole. Or, certaines spécifications de protocoles, destinées à des systèmes aux ressources limitées (*e.g.* cartes à puces), suggèrent explicitement l'emploi de mécanisme peu coûteux pour la génération des nonces [SR96] tel que l'usage de compteurs. Le caractère prévisible de ce type de nonces rend possible de nouvelles attaques qui ne sont pas prises en compte par les procédures développées à l'heure actuelle.

Mots de passe et données de type faible.

Dans le même ordre d'idées, de nombreux protocoles font intervenir un mot de passe choisi par un utilisateur. Or, il est bien connu que lorsque l'on demande à un utilisateur de choisir un mot de passe, la probabilité qu'il choisissent un mot facile à retenir, et donc facile à deviner, est très grande. Cette faiblesse va pouvoir être exploitée par l'intrus pour réaliser une attaque. Ce type d'attaque, appelée « attaque par dictionnaire », consiste pour l'intrus à deviner quel mot a été utilisé en testant tous les mots d'un dictionnaire donné jusqu'à ce qu'il reconnaisse une information caractéristique. Ce type d'attaque est également réalisable sur des protocoles faisant intervenir des données prenant leur valeur dans un domaine fini connu de l'intrus, comme c'est souvent le cas pour les protocoles de vote référendaire. Des résultats récents proposent des procédures permettant de vérifier la résistance des protocoles aux attaques par dictionnaire [DJ04a, Bau05].

Propriétés algébriques des primitives cryptographiques

Au moment où cette thèse a commencé, la plupart des démonstrateurs ne prenaient pas en compte les propriétés algébriques des primitives cryptographiques. Ils travaillaient dans le modèle standard dit de *Dolev-Yao* [DY81]. Dans ce modèle, les capacités d'analyse de l'intrus sont très limitées. L'évolution des connaissances de l'intrus est définie par un ensemble de règles de déduction permettant à ce dernier de construire de nouveaux messages (c'est-à-dire une paire ou un chiffré), à partir des messages qu'il connaît, en utilisant les opérations de concaténation et de chiffrement. Il peut également récupérer les différents composants d'une paire. En revanche, la seule façon qu'il a d'obtenir le message en clair caché dans un message chiffré et de connaître la clef de déchiffrement. Cette hypothèse, appelée communément hypothèse du chiffrement parfait, est excessive dans beaucoup de cas. Typiquement, les protocoles de distribution de clefs s'appuient sur des propriétés d'associativité et de commutativité de certaines techniques de chiffrement. D'autre part, ces propriétés algébriques, même si elles ne sont pas utilisées lors d'une exécution normale du protocole, existent et peuvent être exploitées par un agent malveillant pour réaliser une attaque.

Considérons le protocole d'échange de clefs de W. Diffie et M. Hellman [DH76], décrit ci-dessous. Il permet l'échange d'une clef entre deux entités ne possédant aucun secret en commun au préalable. De plus, les deux participants sont à l'origine de cette clef de session. Cet échange est rendu possible grâce aux propriétés de l'exponentielle modulaire. L'initiateur du protocole engendre un nonce N_a et reçoit $\mathsf{exp}(\mathsf{g}, N_b)$. De son côté, l'autre agent engendre un nonce N_b et reçoit $\mathsf{exp}(\mathsf{g}, N_a)$. Ces deux échanges peuvent se représenter de la façon suivante :

$$\begin{array}{rcll} A & \to & B: & \mathsf{exp}(\mathsf{g}, N_a) \\ B & \to & A: & \mathsf{exp}(\mathsf{g}, N_b) \end{array}$$

Chacun dispose, à l'issue de cet échange, de suffisamment d'information pour calculer la clef de session $\mathsf{exp}(\mathsf{g}, N_a \times N_b)$ puisque $\mathsf{exp}(\mathsf{g}, N_a \times N_b) = \mathsf{exp}(\mathsf{exp}(\mathsf{g}, N_a), N_b) = \mathsf{exp}(\mathsf{exp}(\mathsf{g}, N_b), N_a)$. D'autre part, seuls les agents A et B, ayant participé à cet échange, sont en mesure de calculer $\mathsf{exp}(\mathsf{g}, N_a \times N_b)$. En effet, un agent récupérant les messages $\mathsf{exp}(\mathsf{g}, N_a)$ et $\mathsf{exp}(\mathsf{g}, N_b)$ ne peut pas en déduire le secret $\mathsf{exp}(\mathsf{g}, N_a \times N_b)$.

De nombreuses primitives cryptographiques fréquemment utilisées dans les protocoles possèdent des propriétés algébriques (*e.g.* exponentielle modulaire, « ou » exclusif, ...).

Citons par exemple, le standard 802.11, un protocole réseau sans fil de plus en plus utilisé pour les réseaux locaux. Il intègre en option un protocole de sécurité, le WEP (Wired Equivalent Privacy). Celui-ci, très simple à administrer et à utiliser, est malheureusement peu sûr. Il comporte des failles de sécurité liées aux propriétés de l'opérateur de « ou » exclusif [BGW01]. Un autre exemple est une faille dans le protocole dû à T. Tatebayashi, N. Matsuzaki et D. Newman [TMN89]. L'attaque, découverte par G. Simmons [Sim94], fait intervenir un intrus actif, interceptant et émettant des messages sur le réseau, qui exploite la propriété d'homomorphisme satisfaite par l'algorithme de chiffrement RSA.

Dans cette thèse, nous allons relâcher l'hypothèse du chiffrement parfait et prendre en compte les propriétés algébriques des primitives cryptographiques. L'un des plus ancien travaux sur le sujet est sans doute celui de S. Even *et al.* [EGS86] où les auteurs étudient les protocoles ping-pong utilisant du chiffrement RSA. Ils montrent qu'il est inutile de considérer les propriétés de RSA : si un protocole peut être attaqué en utilisant les faiblesses du chiffrement RSA, alors il existe une attaque qui n'utilise pas ces faiblesses. Depuis, en particulier au cours de ces dernières années, de nombreux résultats ont été établis. La plupart de ces résultats ont été établis dans le cadre d'un nombre borné de sessions, ils sont listés au chapitre 2. En revanche, dans le cadre d'un nombre non-borné de sessions, les résultats sont plus rares [CLC03, Ver03]. Le problème étant déjà indécidable sous l'hypothèse du chiffrement parfait, les procédures proposées doivent restreindre la classe des protocoles étudiée et/ou faire quelques abstractions.

1.5 Contributions

Dans cette thèse, nous nous intéressons au problème de la sécurité d'un protocole dans le cadre d'un nombre borné de sessions et nous considérons des modèles d'intrus plus réalistes que le modèle d'intrus standard classiquement utilisé dans le domaine. Pour cela, nous enrichissons le modèle d'intrus *de Dolev-Yao* [DY81] par l'ajout d'une règle de déduction permettant à ce dernier de réaliser des étapes de raisonnement équationnel. Nous relâchons ainsi l'hypothèse du chiffrement parfait en tenant compte des propriétés algébriques des primitives cryptographiques.

Sur le plan de la modélisation, un point important et plutôt satisfaisant est que nous n'introduisons pas de nouveaux modèles. Les travaux réalisés dans cette thèse utilisent deux variantes d'un modèle déjà bien établi dans le domaine : le modèle par rôles à base de règles de réécriture avec filtrage ou tests d'égalités. Bien que très proches, la comparaison du pouvoir d'expression de ces deux modèles s'est révélée très intéressante et a permis de mettre au jour quelques subtilités. Le passage d'un modèle à l'autre s'avère finalement assez difficile.

Sur le plan de la vérification, le problème de déduction de l'intrus (recherche d'attaques en présence d'un intrus passif, c'est-à-dire se contentant d'écouter les messages circulant sur le réseau) et la sécurité d'un protocole pour un nombre borné de sessions en présence d'un intrus actif ont été étudiés pour plusieurs théories équationnelles particulières comme par exemple le « ou » exclusif [CKRT03b, CLS03] et l'exponentielle modulaire [CKRT03a, MS05]. Mais pour de nombreuses théories équationnelles intéressantes, ces problèmes restent ouverts. Nous avons contribué sur ce point de deux manières différentes.

Nous avons d'une part menés l'étude de nouvelles théories équationnelles plus complexes que celles étudiées jusqu'à présent. Ils permettent de traiter des théories équationnelles complexes faisant intervenir un opérateur associatif-commutatif et une propriété d'homomorphisme sur cet opérateur. Une caractérisation algébrique des problèmes considérés a permis de les réduire à la résolution de systèmes d'équations, et d'établir des procédures de décision et des résultats d'indécidabilité.

D'autre part, nous nous sommes intéressés à obtenir des résultats génériques. Nous proposons des procédures permettant de traiter la classe des théories équationnelles *publique-effondrantes*. Les algorithmes proposés ont la même complexité théorique que les procédures connues dans le cas du modèle de Dolev-Yao standard. Il s'agissait, en 2004, du premier résultat de décidabilité du problème de la sécurité pour une classe de théorie équationnelle définie de manière syntaxique. Enfin, nous apportons quelques réponses pour l'obtention d'un résultat générique qui permettrait de traiter une classe très large de théories équationnelles. Nous définissons une propriété, vérifiée par un grand nombre de théories équationnelles et qui permet de simplifier la théorie équationnelle étudiée. Cette propriété, vérifiée par la plupart des théories pertinentes du point de vue de la vérification des protocoles cryptographiques, permet de réduire le problème de la vérification en présence d'une théorie équationnelle à l'étude de deux théories équationnelles : la théorie vide et la théorie associative et commutative.

1.6 Plan de la thèse

Cette thèse est composée de deux parties. Nous commençons par aborder les différents problèmes liés à la modélisation et nous définissons formellement les deux problèmes dont nous menons l'étude dans la deuxième partie.

Première partie : Modélisation.

Dans le **chapitre 2**, nous dressons une liste des propriétés algébriques pertinentes du point de vue de la vérification des protocoles crytographiques et nous en profitons pour donner les résultats existant aujourd'hui sur ces différentes théories. Nous nous limitons aux résultats concernant le problème de déduction de l'intrus et le problème de la sécurité d'un protocole pour un nombre borné de sessions. Nous introduisons également quelques notions classiquement utilisées en réécriture. Dans le **chapitre 3**, nous nous intéressons à la modélisation des protocoles proprement dits et à la modélisation de l'intrus. Ce dernier représente l'environnement hostile dans lequel le protocole évolue. Du point de vue de la modélisation des protocoles, nous introduisons deux modèles légèrement différents, le modèles par rôles avec filtrage et le modèle par rôles avec tests d'égalités. Une comparaison de ces deux modèles est également proposée dans ce chapitre. Enfin, au **chapitre 4**, nous définissons formellement les deux problèmes dont nous faisons l'étude dans cette thèse : le problème de déduction de l'intrus, et le problème de la sécurité pour un nombre borné de sessions. Nous montrons comment ces deux problèmes s'expriment en terme de résolution de contraintes.

Deuxième partie : Vérification.

Dans cette partie, nous présentons différents résultats de décidabilité et de complexité pour le problème de déduction de l'intrus et le problème de la sécurité en présence d'opérateurs ayant des propriétés algébriques. Dans le **chapitre 5**, nous présentons une classe de théories équationnelles, les théories équationnelles *publique-effondrantes*, et nous établissons une procédure générique permettant de traiter toute théorie équationnelle publique-effondrante.

Cette étude a été réalisée dans le modèle par rôle avec tests d'égalités. Dans le **chapitre 6**, un travail similaire est réalisé. Cette étude est réalisée dans le modèle par rôles avec filtrage et permet de conclure pour des théories équationnelles particulières (disjointes de celles étudiées au chapitre précédent).

Le **chapitre 7** permet de faire un pas supplémentaire vers l'obtention d'un résultat générique permettant de traiter un grand nombre de théories équationnelles. Pour cela, nous proposons une propriété, la *propriété des variants finis*, qui permet de se débarrasser de certains axiomes composant une théorie équationnelle. Nous montrons son intérêt dans le cadre de la vérification des protocoles cryptographiques. Cette propriété, satisfaite par un grand nombre de théories équationnelles, permet de réduire la théorie équationnelle. Cette réduction est valable dans les deux modèles présentés dans cette thèse.

L'état de l'art, présenté au chapitre 2, a été publié dans l'article [CDL06]. Le chapitre 5 a fait l'objet de l'article [DJ04a] et les résultats présentés dans le chapitre 6 ont fait l'objet des articles [DLLT06, Del06a, Del06b]. Les résultats décrits dans le chapitre 7 ont été publiés dans l'article [CD05]. Par souci d'homogénéité, nous avons choisi de ne présenter que les résultats contribuant à l'affaiblissement de l'hypothèse du chiffrement parfait par la prise en compte de propriétés algébriques. Nous ne présenterons donc pas les résultats publiés dans les articles [DJ04b, DJ06] concernant la prise en compte des données de type faibles et des attaques par dictionnaires. De même, nous ne parlerons pas des propriétés d'anonymat, s'exprimant sous forme d'équivalence entre processus, dont nous avons fait l'étude dans le cadre des protocoles de vote [DKR06].

Première partie

Modélisation des protocoles cryptographiques

Chapitre 2

Messages

Sommaire

2.1 Primitives cryptographiques « classiques » 22
2.1.1 Concaténation 22
2.1.2 Chiffrement 22
2.2 Propriétés algébriques 23
2.2.1 Retour sur les primitives « classiques » 23
2.2.2 Associativité (A) 25
2.2.3 Commutativité (C) 25
2.2.4 « Ou » exclusif (ACUN) 26
2.2.5 Groupe abélien (AG) 26
2.2.6 Signature en aveugle 27
2.2.7 Propriété préfixe 27
2.2.8 Homomorphisme 28
2.2.9 Distributivité du chiffrement 28
2.2.10 Diffie-Hellman (DH) 29
2.3 Modélisation des messages 29
2.3.1 Termes, sous-termes 30
2.3.2 Unification 31
2.3.3 Systèmes de réécriture 32

Le but de ce chapitre est d'introduire la modélisation que nous allons utiliser pour représenter les messages échangés au cours de l'exécution d'un protocole. Ces messages sont obtenus par application d'opérations (concaténation, chiffrement, ...) sur des données (dans la réalité, des suites de bits). Nous les modéliserons par des termes. Ces opérateurs utilisés lors de la création des messages possèdent des propriétés algébriques. Ainsi, définir un message par la suite d'opérations permettant de l'obtenir aboutit à une *représentation non canonique* : deux messages identiques peuvent être représentés différemment. Une manière classique de traiter ce problème est de raisonner modulo une théorie équationnelle, et de considérer un représentant pour chacune des classes d'équivalences induites par cette théorie.

Nous commencerons par décrire les opérations de concaténation et de chiffrement (*cf.* partie 2.1), et nous dresserons une liste de propriétés algébriques pertinentes du point de vue de l'étude des protocoles de sécurité (*cf.* partie 2.2). Dans la partie 2.3, nous introduirons quelques notions et notations classiques en réécriture très utiles pour la manipulation des termes.

2.1 Primitives cryptographiques « classiques »

Nous présentons ici deux primitives couramment utilisées dans les protocoles cryptographiques : la concaténation et le chiffrement.

2.1.1 Concaténation

Cet opérateur est généralement modélisé par un symbole binaire, noté $\langle \cdot, \cdot \rangle$[1]. Ainsi le message $\langle m_1, \langle m_2, m_3 \rangle\rangle$ représente la concaténation du message m_1 avec le message $\langle m_2, m_3 \rangle$. Il est différent du message représenté par le terme $\langle\langle m_1, m_2 \rangle, m_3\rangle$. Nous pouvons cependant modéliser l'associativité de cet opérateur de concaténation, encore appelé *paire*, en ajoutant la théorie équationnelle de l'associativité du symbole $\langle ., . \rangle$, c'est-à-dire :

$$\langle x, \langle y, z \rangle\rangle = \langle\langle x, y \rangle, z\rangle.$$

2.1.2 Chiffrement

Le chiffrement est un algorithme prenant en entrée un message m et une clef k, appelée *clef de chiffrement*, et qui retourne un message, généralement noté $\{m\}_k$, que l'on appelle le *chiffré de m par k*. Ce message chiffré est tel qu'il est très difficile, voire impossible, de retrouver le message d'origine à partir du chiffré sans connaître la clef de déchiffrement. Cette hypothèse, souvent appelée *hypothèse du chiffrement parfait*, ne correspond pas à la réalité. Nous verrons (*cf.* partie 2.2) qu'il existe de nombreux algorithmes de chiffrement, ayant des propriétés algébriques (*e.g.* chiffrement RSA [RSA78]).

Enfin, on distingue généralement deux grandes classes d'algorithmes de chiffrement : le chiffrement *symétrique* et le chiffrement *asymétrique*. Les algorithmes de chiffrement symétrique sont ceux pour lesquels la clef utilisée lors du déchiffrement est identique à celle utilisée lors du chiffrement. Pour le chiffrement asymétrique, les deux clefs utilisées sont distinctes. On parle souvent de chiffrement *à clefs publiques*, car le chiffrement asymétrique est généralement utilisé en publiant la clef de chiffrement (clef publique, notée $\mathsf{pub}(\cdot)$) et en gardant la clef de déchiffrement secrète (clef privée, notée $\mathsf{priv}(\cdot)$). Il semble naturel de représenter ces deux types d'algorithmes de chiffrement par des symboles différents (comme par exemple $\{m\}_k^a$ et $\{m\}_k^s$). Cependant, nous utiliserons parfois le même symbole. L'algorithme utilisé dépend alors implicitement du type de la clef utilisée.

Dans le cas du modèle d'intrus standard, dit de *Dolev-Yao*, il a été établit que le problème de déduction de l'intrus est décidable en temps polynomial et que le problème de la sécurité d'un protocole pour un nombre borné de sessions est co-NP-complet [RT03].

Dans cette thèse, nous affaiblissons l'hypothèse du chiffrement parfait par la prise en compte de certaines propriétés algébriques. Il y a plusieurs raisons à cela. Tout d'abord, l'exécution même de certains protocoles est basée sur le fait que certaines primitives ont des propriétés algébriques. En faire abstraction donne un protocole non exécutable sur lequel on ne peut plus raisonner. D'autre part, même si le protocole n'utilise pas ces propriétés

[1] Parfois, nous noterons simplement m_1, m_2 au lieu de $\langle m_1, m_2 \rangle$.

algébriques lorsqu'il s'exécute normalement, certaines attaques reposent sur l'utilisation de ces propriétés. Ainsi, le protocole présenté dans [Pau97] a été montré correct par L. Paulson [Pau97] en considérant l'opérateur « ou » exclusif comme un symbole libre. Mais P. Ryan et S. Schneider ont trouvé une attaque sur ce protocole [RS98] : cette attaque utilise les propriétés algébriques du « ou » exclusif. Il est donc nécessaire de prendre en compte les propriétés algébriques des opérateurs lors de la vérification des protocoles.

L'ajout de théories équationnelles permet de mieux représenter les propriétés algébriques des primitives cryptographiques, mais elles rendent le problème de la vérification des protocoles beaucoup plus difficiles.

2.2 Propriétés algébriques

Certaines propriétés algébriques du chiffrement, telles que la propriété d'homomorphisme du chiffrement RSA [RSA78] ou les propriétés induites par les méthodes de chiffrement par blocs, sont très utilisées dans les protocoles. De nombreuses attaques exploitent ces propriétés. Il est donc important d'en tenir compte. Récemment, un certain nombre de procédures, en particulier dans le cadre d'un nombre borné de sessions, ont été proposées pour vérifier les protocoles en présence de propriétés algébriques.

Dans cette partie, nous dressons une liste des théories équationnelles pertinentes dans le cadre de la vérification des protocoles. Nous présentons également un certain nombre de résultats existants relatifs au problème de déduction de l'intrus et au problème de la vérification dans le cadre d'un nombre borné de sessions. Tout ces résultats contribuent à l'affaiblissement de l'hypothèse du chiffrement parfait et sont résumés dans la figure 2.1.

2.2.1 Retour sur les primitives « classiques »

On peut choisir de représenter les primitives de bases accompagnées de leurs destructeurs et de représenter les propriétés algébriques entre ces opérateurs à l'aide d'une théorie équationnelle. Cela nous conduit à considérer :

- un symbole dec pour représenter l'algorithme de déchiffrement,
- des symboles proj_1 et proj_2 pour représenter les projections permettant d'accéder aux composantes d'une paire,
- la théorie équationnelle suivante pour exprimer le lien entre ces différentes primitives :

$$\mathsf{E}_{\mathsf{DY}} := \left\{ \begin{array}{rcl} \mathsf{dec}(\{x\}_y, y) & = & x, \\ \mathsf{proj}_1(\langle x_1, x_2 \rangle) & = & x_1, \\ \mathsf{proj}_2(\langle x_1, x_2 \rangle) & = & x_2. \end{array} \right.$$

Parfois, on considère en plus l'équation $\{\mathsf{dec}(x,y)\}_y = x$. Cette dernière permet de modéliser une propriété de « commutativité » entre les algorithmes de chiffrement et de déchiffrement. Ce changement dans la modélisation des primitives classiques n'est pas anodin. En effet, avec cette nouvelle modélisation, il est désormais possible de parler du terme $\mathsf{dec}(m,k)$. Autrement dit, il est possible d'appliquer l'algorithme de déchiffrement à un message qui n'est pas un chiffré et de considérer le résultat obtenu comme étant un message valide. Ainsi, un agent s'attendant à recevoir un message de la forme $\{x\}_k$ acceptera en fait n'importe quel message m puisque celui-ci est bien de la forme attendue (on a $m = \{\mathsf{dec}(m,k)\}_k$).

	Problème de déduction de l'intrus	**Problème de la sécurité nombre borné de sessions**
Commutativité du chiffrement	*PTIME [CKRT04]*	*co-NP-complet [CKRT04]*
Associativité - Commutativité	*Décidable [AC05]*[2]	*?*
+		
homomorphisme	*NP-complet [LLT05a]*	*Indécidable* (car l'unification est indécidable [Nar96])
« Ou » exclusif (ACUN)	*PTIME [CKRT03b]*	*Décidable [CLS03, MS05]* *co-NP-complet [CKRT03b]*
+		
homomorphisme	*PTIME [Thèse]*	*Décidable [Thèse]*
Groupe abélien (AG)	*NPTIME [CLS03]* *PTIME [CKRT03a]*	*Décidable [MS05]*
+		
homomorphisme	*PTIME [Thèse]*	*Indécidable [Thèse]*
Signature en aveugle	*PTIME [Ber06]*	*Décidable [Ber06]*
Préfixe	*PTIME [CKRT03b]*	*co-NP-complet [CKRT03b]*
Distributivité du chiffrement		
- sur la paire	*PTIME [CLT03]*	*?*
- sur un symbole AC	*NP-complet [LLT05a]*	*?*
- sur un symbole ACUN	*EXPTIME [LLT05a, LLT05b]*	*?*
- sur un symbole AG	*EXPTIME [LLT05a]*	*?*
Distributivité et Commutativité du chiffrement sur ACUN	*EXPTIME [Laf05]*	*?*
Diffie-Hellman (DH)	*PTIME [CKRT03a]*	*Décidable [MS05]* *co-NP-complet [CKRT03a]*

[2] Ce problème est en fait NP-complet en utilisant la technique décrite dans [LLT05a] pour la théorie ACh.

Dans le cas d'un nombre borné de sessions, les résultats cités ne considèrent pas toujours exactement la même classe de protocoles. Ceci explique les différences dans les résultats de complexité obtenus.

Figure 2.1 - Tableau récapitulatif.

Dans [Mil03], J. Millen justifie le fait de ne pas considérer ces opérateurs (déchiffrement, projections) et de travailler dans l'algèbre libre (sans théorie équationnelle). Il montre que dès lors que le protocole vérifie un certain nombre de propriétés syntaxiques raisonnables, ces deux modèles sont en fait équivalents. C. Lynch et C. Meadows étendent ce résultat au cas du chiffrement asymétrique [LM04]. Dans le chapitre 5 de cette thèse, nous regardons le problème sous un angle différent. Nous montrons qu'il n'est pas plus difficile, d'un point de vue théorique, de vérifier les protocoles en présence de ces destructeurs. Nous proposons une procédure générique de recherche d'attaques NP-complète (comme dans le cas du modèle standard de *Dolev-Yao*) permettant de résoudre le problème de la sécurité d'un protocole dans le cas d'un nombre borné de sessions. La classe de théories équationnelles comprend en particulier la théorie proposée ci-dessus ainsi que quelques variantes. En 2004, il s'agissait du premier résultat générique s'appliquant à une classe de théories équationnelles définie de façon syntaxique. À l'heure actuelle, d'autres résultats génériques existent pour les deux problèmes de vérification auxquelles nous nous sommes intéressés dans cette thèse. Concernant le problème de déduction de l'intrus, V. Cortier et M. Abadi ont montré que ce problème était décidable en temps polynomial pour une classe plus générale que la nôtre : la classe des théories sous-termes convergentes [AC04]. Ils ont également mis en place un ensemble de conditions suffisantes permettant d'obtenir la décidabilité du problème de déduction de l'intrus et ils ont montré qu'un certain nombre de théories (*e.g.* « ou » exclusif, AC) satisfont ces conditions [AC05]. Depuis, M. Baudet a montré la décidabilité du problème de la recherche d'attaques pour un nombre borné de sessions, dans le cas des théories sous-termes convergentes [Bau05]. Nous reviendrons sur ces travaux à la fin du chapitre 5.

2.2.2 Associativité (A)

L'associativité est une propriété pouvant être modélisée par $f(f(x,y),z) = f(x,f(y,z))$. Par exemple, on peut vouloir modéliser l'associativité de la somme par $(x+y)+z = x+(y+z)$. À notre connaissance, il n'existe pas de résultat théorique s'intéressant à cette propriété seule. Un autre exemple intéressant, mentionné ci-dessus, est l'associativité de la concaténation. Cette propriété est prise en compte, mais seulement en partie, par l'outil Casrul [JRV00] et permet de trouver beaucoup d'attaques *par confusion de type*. Ces attaques reposent sur le fait que les agents exécutant le protocole confondent deux messages de type très différents, comme par exemple un nonce et une paire formée d'un nonce et de l'identité d'un agent. Ce type d'attaque n'est pas très réaliste car la taille d'un nonce et d'une telle paire sont en réalité très différentes.

2.2.3 Commutativité (C)

Dans [CKRT04], Y. Chevalier *et al.* ont proposé une procédure de décision NP-complète pour le problème de la recherche d'attaques en présence d'un symbole de chiffrement commutatif, c'est-à-dire satisfaisant la propriété :

$$\{\{x\}_y\}_z = \{\{x\}_z\}_y.$$

Cette propriété est satisfaite par le chiffrement RSA lorsque les clefs utilisent le même module. Une forme restreinte de commutativité est satisfaite par le chiffrement RSA lorsque l'on suppose que toutes les clefs utilisées sont engendrées avec des modules différents. On a alors une propriété de commutation, mais uniquement entre la clef et son inverse. Cette propriété peut être prise en compte par les deux équations suivantes :

$$\{\{x\}_k\}_{k^{-1}} = x \text{ et } \{\{x\}_{k^{-1}}\}_k = x.$$

Ce type de théorie entre dans la classe des théories équationnelles *publiques-effondrantes* dont nous faisons l'étude au chapitre 5. Dans [RT03], M. Rusinowitch et M. Turuani considère aussi un tel opérateur en donnant à l'intrus la capacité de retrouver le message m lorsqu'il connaît $\{\{m\}_k\}_k$. Cependant, ce codage ne permet pas de traiter l'ensemble de la théorie représentée par l'équation $\{\{m\}_k\}_k = m$.

2.2.4 « Ou » exclusif (ACUN)

Le symbole $+$ dénote ici l'opérateur binaire appelé « ou » exclusif. Il est également souvent noté $\oplus$. Les propriétés algébriques de cet opérateur sont les suivantes :

$$\begin{array}{rcll} x+(y+z) & = & (x+y)+z & \text{Associativité (A)} \\ x+y & = & y+x & \text{Commutativité (C)} \\ x+0 & = & x & \text{Unité (U)} \\ x+x & = & 0 & \text{Nilpotence (N)} \end{array}$$

Cette opération est utilisée dans de nombreux protocoles et a suscité beaucoup d'intérêt ces dernières années. H. Comon-Lundh et V. Shmatikov ont montré que le problème de la vérification d'un protocole était décidable pour cette théorie [CLS03]. Ce résultat est également montré dans [MS05]. Dans [CKRT03b], Y. Chevalier *et al.* améliorent ce résultat en considérant des règles d'oracles (règles de déduction satisfaisant certaines conditions) et en montrant que la théorie du « ou » exclusif est une instance de ce cadre général. Ils obtiennent ainsi que le problème de la sécurité pour un nombre borné de sessions est co-NP-complet en présence de la théorie équationnelle du « ou » exclusif. Pour obtenir un tel résultat, ils ont dû considérer une classe de protocole raisonnable mais plus restrictive que celle traitée dans [CLS03, MS05].

2.2.5 Groupe abélien (AG)

Le symbole $\times$ dénote ici un opérateur binaire satisfaisant les propriétés suivantes :

$$\begin{array}{rcll} x\times(y\times z) & = & (x\times y)\times z & \text{Associativité (A)} \\ x\times y & = & y\times x & \text{Commutativité (C)} \\ x\times 1 & = & x & \text{Unité (U)} \\ x\times x^{-1} & = & 1 & \text{Inverse (I)} \end{array}$$

On préférera parfois la notation additive, et on utilisera alors le symbole $+$, le symbole unaire $-$ pour l'inverse et la constante 0 pour représenter l'élément neutre. Le premier résultat concernant le problème de déduction de l'intrus en présence d'un opérateur AG est une procédure dans NP. Ce premier résultat, obtenu par H. Comon-Lundh et V. Shmatikov [CLS03], a été amélioré par Y. Chevalier *et al.*. Ces derniers ont proposé une procédure polynomiale [CKRT03a].

Dans le cas d'un intrus actif, le problème de la sécurité, en présence d'un tel opérateur, a été étudié par J. Millen et V. Shmatikov [MS05]. Ils utilisent une technique basée sur la résolution de contraintes symboliques et ils réduisent le problème de la sécurité d'un protocole à la résolution de systèmes d'équations quadratiques dans $\mathbb{Z}$ [MS03]. V. Shmatikov montre alors que la satisfaisabilité de tels systèmes est décidable en exploitant les particularités des systèmes générés [Shm04]. Cependant, la procédure proposée n'est pas correcte et un certain nombre de lemmes cruciaux permettant d'établir la correction et la complétude de leur procédure sont faux. La technique est néanmoins intéressante et astucieuse. Elle est à la base de la procédure que nous avons développée (cf. chapitre 6), pour résoudre le problème de la sécurité dans le cas de la théorie du « ou » exclusif en combinaison avec l'axiome d'homomorphisme $\mathsf{h}(x+y) = \mathsf{h}(x)+\mathsf{h}(y)$ (*cf.* partie 2.2.8 ci-après). Notre procédure permet également de traiter la théorie ACUN (théorie du « ou » exclusif sans l'axiome d'homomorphisme), ainsi que la théorie AG. Nous reviendrons sur ces applications à la fin du chapitre 6 lorsque nous aurons présenté notre procédure permettant de traiter la théorie ACUNh.

2.2.6 Signature en aveugle

Les schémas de signature en aveugle ont été introduits par D. Chaum [Cha84] dans les années 1980. Ils permettent à une entité d'obtenir d'une autre la signature d'un message sans que le signataire n'apprenne quoi que ce soit sur le message qu'elle a pourtant signé. Ce mécanisme est très utilisé dans les protocoles de vote électronique [FOO92, Tra05] : chaque électeur doit, avant de déposer son vote dans l'« urne », obtenir la signature de son vote par une autorité. Cette autorité effectue un certain nombre de vérification avant d'apposer sa signature mais elle ne doit rien apprendre sur le vote de l'électeur en question afin de préserver l'anonymat du vote. Ce mécanisme de signature en aveugle peut se modéliser à l'aide de la théorie équationnelle suivante :

$$\begin{array}{rcl}
\mathsf{checksign}(\mathsf{sign}(x, \mathsf{priv}(y)), \mathsf{pub}(y)) & = & x \\
\mathsf{unblind}(\mathsf{blind}(x, y), y) & = & x \\
\mathsf{unblind}(\mathsf{sign}(\mathsf{blind}(x, y), z), y) & = & \mathsf{sign}(x, z)
\end{array}$$

L'électeur désirant obtenir la signature de son vote v par l'autorité A commencera par camoufler son vote à l'aide de la fonction blind, et il présentera le message $\mathsf{blind}(v, r)$ à l'autorité chargé d'apposer sa signature. L'électeur obtient ainsi $\mathsf{sign}(\mathsf{blind}(v, r))$ et, connaissant r, il peut récupérer $\mathsf{sign}(v, r)$ en appliquant la fonction unblind.

Cette théorie équationnelle a été utilisée par S. Kremer et M. Ryan [KR05a] pour mener l'étude du protocole de vote dû à A. Fujioka *et al.* [FOO92]. Cette théorie a également fait l'objet de quelques résultats théoriques. Le problème de déduction de l'intrus est décidable en temps polynomial [Ber06]. Dans le cas d'un intrus actif, le problème de la sécurité pour un nombre borné de sessions est décidable. Ce résultat a été obtenu comme instance d'un résultat générique développé dans la thèse de V. Bernat [Ber06].

2.2.7 Propriété préfixe

Cette propriété permet à l'intrus de récupérer, à partir d'un message chiffré, le chiffré de n'importe quel préfixe. À partir d'un message $\{x, y\}_z$, il peut déduire le message $\{x\}_z$. Cette propriété est utilisée dans le cas du chiffrement par blocs avec *chaînage* : le chiffrement d'un bloc dépend du bloc précédent. Avec une telle méthode de chiffrement, le chiffrement d'un message $P_1 P_2 \ldots P_n$ (où chacun des P_i constitue un bloc) avec la clef K est un message $C_0C_1C_2\dots C_n$ où $C_0 = I$ (bloc d'initialisation) et $C_i = \{C_{i-1} + P_i\}_K$. Le chiffrement par bloc avec chaînage a la propriété suivante :

$$\text{si } C_0C_1\dots C_iC_{i+1}\cdots C_n = \{P_1\dots P_iP_{i+1}\cdots P_n\}_K \text{ alors } C_0C_1\dots C_i = \{P_1\dots P_i\}_K.$$

Autrement dit, un attaquant peut obtenir $\{x\}_z$ à partir de $\{x,y\}_z$ si la longueur de x est un multiple de la longueur des blocs utilisé par l'algorithme de chiffrement. Dans [KR05b], S. Kremer et M. Ryan remarquent que ce type de propriétés est également vraie pour les suffixes. Autrement dit, $C_{i+1}\dots C_n$, extrait du chiffré $C_0C_1C_2\dots C_n$, est un chiffré valide. Il suffit d'initialiser le bloc d'initialisation avec C_i pour obtenir un chiffré valide correspondant au chiffrement des $n-i$ derniers blocs du message de départ.

Le cadre développé dans [CKRT03b] par Y. Chevalier *et al.* pour étudier le « ou » exclusif s'applique également à ce type de propriétés. Ils ont montré que le problème de la sécurité est également co-NP-complet en présence d'un opérateur de chiffrement par blocs.

2.2.8 Homomorphisme

Nous considérons un symbole h satisfaisant la propriété d'homomorphisme suivante :

$$\mathsf{h}(x+y) = \mathsf{h}(x) + \mathsf{h}(y). \quad (\mathsf{h})$$

Le chapitre 6 de cette thèse est consacré à l'étude de cet axiome d'homomorphisme où $+$ représente un opérateur AC, ACUN (« ou » exclusif) ou AG (groupe abélien). Dans [LLT05a], P. Lafourcade *et al.* ont étudié le problème de déduction de l'intrus pour les théories ACh, ACUNh et AGh, et ont obtenu une procédure EXPTIME pour les théories ACUNh et AGh. Le problème est NP-complet dans le cas de la théorie ACh.

Nous reviendrons sur ces trois théories équationnelles au chapitre 6 et nous améliorerons les résultats de complexité obtenus par P. Lafourcade *et al.* en proposant une nouvelle approche pour traiter le symbole d'homomorphisme. Nous détaillerons un peu plus les travaux de P. Lafourcade *et al.* à cette occasion (*cf.* partie 6.2.4). Cette nouvelle approche, dans le traitement du symbole d'homomorphisme, va nous permettre d'obtenir une procédure de décision pour le problème de la sécurité d'un protocole, en s'inspirant des techniques utilisées par J. Millen et V. Shmatikov dans le cas de la théorie AG [MS05].

2.2.9 Distributivité du chiffrement

Dans [CLT03], H. Comon-Lundh et R. Treinen ont mis en place des conditions sur la théorie équationnelle permettant d'assurer la décidabilité du problème de déduction de l'intrus. En particulier, cela leur a permis d'établir que le problème de déduction de l'intrus est décidable en temps polynomial en présence de la propriété d'homomorphisme suivante :

$$\{\langle u,v\rangle\}_k = \langle\{u\}_k, \{v\}_k\rangle.$$

Cette propriété est particulièrement intéressante puisqu'elle est satisfaite par les algorithmes de chiffrements utilisant le chiffrement par blocs *sans chaînage*. Cette technique consiste à chiffrer les messages en procédant à un découpage en blocs et en chiffrant séparant chacun des blocs.

Dans [LLT05a, LLT05b], P. Lafourcade *et al.* considèrent le problème de déduction de l'intrus en présence de la propriété d'homomorphisme $\{x+y\}_k = \{x\}_k + \{y\}_k$. Ils montrent que dans le cas d'un opérateur AC, le problème est NP-complet, alors qu'il est EXPTIME lorsque $+$ est un opérateur ACUN ou AG. Dans [Laf05], P. Lafourcade obtient un résultat similaire pour le cas d'un opérateur ACUN en présence d'un opérateur de chiffrement distributif et commutatif.

2.2.10 Diffie-Hellman (DH)

Le symbole $\times$ représente un opérateur AG, et le symbole, noté exp, est utilisé pour représenter l'exponentielle modulaire. Ce nouvel opérateur satisfait les propriétés suivantes :

$$\begin{aligned} \mathsf{exp}(\mathsf{exp}(x,y),z) &= \mathsf{exp}(x, y \times z) \\ \mathsf{exp}(x,1) &= x \end{aligned}$$

Cette théorie équationnelle permet de prendre en compte quelques propriétés simples du produit et de l'exponentielle modulaire. Ces opérateurs sont utilisés dans le chiffrement RSA [RSA78], et dans l'échange Diffie-Hellman [DH76] (d'où le nom de la théorie). Ce dernier permet, grâce aux propriétés de l'exponentielle modulaire décrite ci-dessus, l'établissement d'une clef de session entre deux participants. L'originalité de cet échange est que les deux participants sont à l'origine de cette clef de session. L'initiateur du protocole génère N_a et reçoit $\mathsf{exp}(g, N_b)$. De son côté, l'autre agent génère N_b et reçoit $\mathsf{exp}(g, N_a)$. Chacun dispose alors de suffisamment d'information pour calculer la clef de session $\mathsf{exp}(g, N_a \times N_b)$. En effet,

$$\mathsf{exp}(g, N_a \times N_b) = \mathsf{exp}(\mathsf{exp}(g, N_a), N_b) = \mathsf{exp}(\mathsf{exp}(g, N_b), N_a).$$

Les résultats obtenus pour cette théorie suppose que le symbole $\times$ est utilisé uniquement dans les exposants. En particulier, les exponentielles ne sont pas multipliées entre elles.

Des résultats partiels ont été obtenus par M. Boreale et M.G. Buscemi dans [BB03] et par Y. Chevalier *et al.* dans [CKRT03a]. La procédure de décision de [BB03] nécessite le calcul d'une borne sur le nombre de facteurs pouvant apparaître dans chacun des produits et ils ne donnent pas de résultats de complexité. Y. Chevalier *et al.* [CKRT03a] montrent que le problème de la sécurité est co-NP-complet en présence d'une telle théorie équationnelle pour une classe de protocole « raisonnable ». Dans [MS05] le problème est réduit à la satisfaisabilité d'un problème équivalent dans le cas d'un opérateur AG, ce qui permet d'obtenir un résultat de décision pour une classe de protocoles plus grande que celle traitée dans [CKRT03a].

2.3 Modélisation des messages

Dans le cadre de l'hypothèse du chiffrement parfait, les messages sont représentés par des termes et l'égalité entre messages correspond en fait à l'égalité syntaxique sur les termes. Ainsi, les deux messages chiffrés $\{m_1\}_{k_1}$ et $\{m_2\}_{k_2}$ sont égaux si et seulement si $m_1 = m_2$ et $k_1 = k_2$. L'introduction d'une théorie équationnelle permet de prendre en compte les propriétés algébriques de certains opérateurs, et ainsi de tenir compte des égalités entre messages pourtant construits différemment. Par exemple, si $+$ représente un opérateur associatif et commutatif (AC), les messages $a + (b + c)$ et $(b + a) + c$ sont indistinguables et sont donc représentés par la même suite de bits. Cette égalité se retrouve au niveau de la modélisation par le fait que $a + (b + c) =_{\mathsf{AC}} (b + a) + c$: ces deux termes sont égaux modulo AC.

Dans la suite, nous allons considérer des théories équationnelles décrites sous forme de systèmes de réécriture convergents (éventuellement modulo AC). Cela nous permettra de raisonner sur les représentants des classes d'équivalences. Pour cela, nous allons introduire quelques notions et notations classiques en réécriture (*cf.* [DJ90]).

2.3.1 Termes, sous-termes

On suppose donnée une signature $\mathcal{F}$ et un ensemble infini de variables $\mathcal{X}$. L'ensemble des termes construits à partir des symboles de $\mathcal{F}$ et des variables de $\mathcal{X}$ est noté $\mathcal{T}(\mathcal{F},\mathcal{X})$. L'ensemble des termes clos (sans variables) est noté $\mathcal{T}(\mathcal{F})$. Nous notons $vars(t)$ l'ensemble des variables apparaissant dans un terme t.

Les termes sont souvent représentés par des arbres dont les feuilles sont étiquetées par des constantes ou des variables, et les nœuds internes par des symboles de fonctions. Dans cette représentation, on suppose qu'il y a un mécanisme pour distinguer les différents fils issus d'un même nœud. Les positions d'un terme t sont représentées par des suites d'entiers et sont notées $\mathcal{O}(t)$. La séquence vide, notée Λ, représente la racine de l'arbre. Si p est une position de t, alors $t|_p$ représente le sous-terme de t à la position p et $t[s]_p$ représente le terme obtenu en remplaçant $t|_p$ par le terme s. L'ensemble des positions non-variables de t, noté $\bar{\mathcal{O}}(t)$, est défini par $\bar{\mathcal{O}}(t) = \{p \in \mathcal{O}(t) \mid t|_p \notin vars(t)\}$. On appelle symbole de tête d'un terme t, le symbole à la position Λ dans l'arbre représentant le terme t : on le note $\mathsf{head}(t)$.

Définition 2.1 (sous-terme syntaxique)
Soit $t \in \mathcal{T}(\mathcal{F},\mathcal{X})$. L'ensemble $St(t)$ des sous-termes syntaxiques *de t est le plus petit ensemble tel que :*
- *$t \in St(t)$,*
- *si $f(t_1,\ldots,t_n) \in St(t)$ alors $t_1,\ldots,t_n \in St(t)$.*

Exemple 2.2
Considérons $t = \{\langle a,b\rangle\}_k$. Les sous-termes syntaxiques du terme t sont le terme t lui-même, le terme $\langle a,b\rangle$ et les constantes a, b et k.

Définition 2.3 (taille d'un terme, $\|t\|$)
La taille *d'un terme t, notée $\|t\|$, est définie par le nombre de sommets de l'arbre représentant t.*

L'inconvénient de cette représentation est qu'une partie de l'information est dupliquée. La représentation d'un terme par un graphe orienté acyclique (DAG) permet d'éviter cet écueil en partageant le maximum d'information (*cf.* Figure 2.2).

Définition 2.4 (taille DAG d'un terme, $\|t\|_{DAG}$)
La taille DAG *d'un terme[3] t, notée $\|t\|_{DAG}$, est le nombre de sous-termes syntaxiques de t, c'est-à-dire $|St(t)|$.*

Pour les théories équationnelles impliquant un opérateur AC, la taille d'un terme est définie un peu différemment. Nous considérons le symbole AC comme un symbole variadique et nous nous restreignons à des termes *aplatis*.

Exemple 2.5
Le terme aplati correspondant à $(\langle a,b\rangle + c) + \langle a,b\rangle$ est $\langle a,b\rangle + \langle a,b\rangle + c$ que l'on notera également $2\langle a,b\rangle + c$.

[3] Cette taille correspond au nombre de sommets du DAG minimal représentant t.

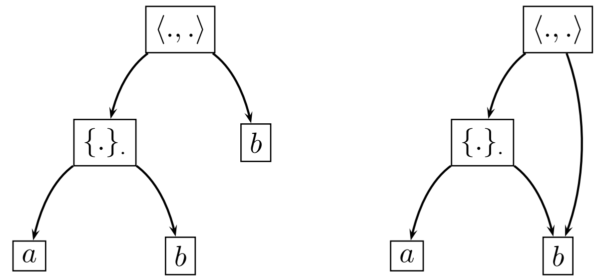

Figure 2.2 - Représentation des termes.

La taille (resp. taille DAG) d'un terme doit alors tenir compte de la taille (resp. taille DAG) de chacun des « facteurs » du terme (dans l'exemple 2.5 les facteurs sont $\langle a, b \rangle$ et c), mais aussi de la taille des coefficients devant ces facteurs (dans l'exemple 2.5, les coefficients des différents facteurs sont respectivement 2 et 1). La taille de ces derniers est le nombre de bits nécessaires pour écrire leur représentation binaire. Nous reviendrons sur ces notions au chapitre 6 lors de l'étude des théories impliquant un opérateur associatif-commutatif.

2.3.2 Unification

Soit E un ensemble fini d'équations (théorie équationnelle), nous notons $sig(\mathsf{E})$ l'ensemble des symboles de fonctions apparaissant dans E et par $=_\mathsf{E}$ la plus petite relation d'équivalence sur $\mathcal{T}(\mathcal{F}, \mathcal{X})$ telle que $u\sigma =_\mathsf{E} v\sigma$ pour toute paire $u = v \in \mathsf{E}$ et pour toute substitution σ.

Le domaine d'une substitution σ est défini par $\{x \mid x \in \mathcal{X}$ et $x\sigma \neq x\}$, on le note $dom(\sigma)$.

Définition 2.6 (E-unificateur)
Deux termes s et t sont dits E-unifiables *s'il existe une substitution σ telle que $s\sigma =_\mathsf{E} t\sigma$. Une telle substitution est appelée* E-unificateur *de s et de t.*

Soit E une théorie équationnelle et $\mathcal{X}$ un ensemble de variables. Une substitution σ est *plus générale* qu'une substitution θ sur $\mathcal{X}$ s'il existe une substitution τ telle que $\theta \leq_\mathsf{E}^\mathcal{X} \sigma\tau$, c'est-à-dire telle que $x\theta =_\mathsf{E} x\sigma\tau$ pour tout $x \in \mathcal{X}$.

Définition 2.7 (ensemble complet d'E-unificateurs, E-mgu)
Soit P un problème d'unification (un ensemble fini d'équations) tel que $vars(P) = \mathcal{X}$. Un ensemble complet d'E-unificateurs *de P est un ensemble de substitutions Θ tel que :*
- *pour toute substitution $\theta \in \Theta$, θ est un* E-unificateur *de P,*
- *pour tout* E-unificateur *σ de P, il existe $\theta \in \Theta$ tel que $\theta \leq_\mathsf{E}^\mathcal{X} \sigma$.*

*Si $\{\theta\}$ est un ensemble complet d'*E-unificateur *de P, la substitution θ est appelée* plus général E-unificateur *de P, et est notée* E-mgu.

Si pour tout problème d'unification P, il existe un ensemble complet et fini d'E-unificateurs de P, la théorie équationnelle E est dite *finitaire*. Si de plus, ces ensembles sont réduits à un unique élément, elle est dite *unitaire*.

Exemple 2.8
Considérons la théorie équationnelle $\mathsf{E} = \mathsf{ACUN}$ *et les termes* $s = x + a$ *et* $t = y + b + c$. *La substitution* $\theta = \{x \mapsto b, y \mapsto a + c\}$ *est un* E-*unificateur de* s *et* t. *En effet :*

$$s\theta \;\; =_{\mathsf{E}} \;\; a + b \;\; =_{\mathsf{E}} \;\; a + b + c + c \;\; =_{\mathsf{E}} \;\; t\theta.$$

La substitution $\sigma = \{x \mapsto y + a + b + c\}$ *est également un* E-*unificateur de* s *et de* t. *La substitution* σ *est plus générale que* θ. *En effet, posons* $\tau = \{y \mapsto a + c\}$, *on a :*
- $x\sigma\tau \; = \; (y + a + b + c)\tau \; =_{\mathsf{E}} \; b =_{\mathsf{E}} x\theta$,
- $y\sigma\tau \; = \; a + c \; = \; y\theta$.

2.3.3 Systèmes de réécriture

Un système de réécriture est un ensemble fini de règle de réécriture $l \to r$ avec $l \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ et $r \in \mathcal{T}(\mathcal{F}, vars(l))$. Un terme $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ se réécrit en t par un système de réécriture $\mathcal{R}$, noté $s \to_{\mathcal{R}} t$ s'il existe une règle $l \to r \in \mathcal{R}$, une position $p \in \mathcal{O}(s)$ et une substitution σ tels que $s|_p = l\sigma$ et $t = s[r\sigma]_p$. Le terme $l\sigma$ est appelé un *radical*. On dit que s *se réécrit en* t *en contractant le radical* $l\sigma$.

Définition 2.9 (radical profond, stratégie de l'intérieur vers l'extérieur)
Un radical r *est dit* profond *si pour toute position* $p \in \mathcal{O}(r)$ *telle que* $p \neq \Lambda$, *le terme* $r|_p$ *n'est pas un radical. Une séquence de réduction suit une* stratégie de l'intérieur vers l'extérieur *si lors de chacune des étapes de réécriture le radical* r *mis en jeu est un radical profond.*

Notation : Nous notons $\xrightarrow{*}_{\mathcal{R}}$ la fermeture réflexive-transitive de $\to_{\mathcal{R}}$ et $\xleftrightarrow{*}_{\mathcal{R}}$ sa fermeture réflexive, transitive et symétrique.

Définition 2.10 (réécriture modulo E, $\to_{\mathcal{R}/\mathsf{E}}$**)**
Soit $\mathcal{R}$ *un ensemble fini de règles de réécriture et* E *un ensemble fini d'équations (typiquement* AC*), la* réécriture modulo E *est la relation* $\to_{\mathcal{R}/\mathsf{E}}$ *définie par :* $s \to_{\mathcal{R}/\mathsf{E}} t$ *s'il existe une position* $p \in \mathcal{O}(s)$, *une règle* $l \to r \in \mathcal{R}$ *et une substitution* σ *telles que* $s =_{\mathsf{E}} u[l\sigma]_p$ *et* $t =_{\mathsf{E}} u[r\sigma]_p$.

Notation : Nous notons $\xrightarrow{*}_{\mathcal{R}/\mathsf{E}}$ la fermeture réflexive-transitive de $\to_{\mathcal{R}/\mathsf{E}}$ et $\xleftrightarrow{*}_{\mathcal{R}/\mathsf{E}}$ sa fermeture réflexive, transitive et symétrique.

On dit qu'un système de réécriture $\mathcal{R}/\mathsf{E}$ représente une théorie équationnelle E' si les relations $\xleftrightarrow{*}_{\mathcal{R}/\mathsf{E}}$ et $=_{\mathsf{E}'}$ sont égales.

Exemple 2.11
Considérons le système de réécriture $\mathcal{R}$ *suivant :*

$$\mathcal{R} := \left\{ \begin{array}{lcl} x + x & \to & 0 \\ x + 0 & \to & x \end{array} \right.$$

Le système de réécriture $\mathcal{R}/\mathsf{AC}$ *est un système représentant la théorie* ACUN.

Définition 2.12 (E-convergent)
Un système de réécriture est E-confluent *si pour tous termes* s, t *tels que* $s =_{\mathcal{R}/\mathsf{E}} t$, *il existe* s' *et* t' *tels que* $s \xrightarrow{*}_{\mathcal{R}/\mathsf{E}} s'$, $t \xrightarrow{*}_{\mathcal{R}/\mathsf{E}} t'$ *et* $s' =_{\mathsf{E}} t'$. *Il est dit* E-convergent *si, de plus, la relation* $\to_{\mathcal{R}/\mathsf{E}}$ *termine.*

Définition 2.13 (forme normale, substitution normalisée)
Un terme t est dit en forme normale *(par rapport à $\mathcal{R}/\mathsf{E}$) s'il n'existe pas de terme s tel que $t \rightarrow_{\mathcal{R}/\mathsf{E}} s$. Si $t \xrightarrow{*}_{\mathcal{R}/\mathsf{E}} s$ et s est en forme normale alors s est* une forme normale de t. *Une substitution σ est dite* normalisée *si pour tout $x \in dom(\sigma)$, le terme $x\sigma$ est en forme normale.*

Notation : Lorsque cette forme normale est unique ($\mathcal{R}$ est convergent modulo E), nous notons $s =_{\mathsf{E}} t{\downarrow}_{\mathcal{R}/\mathsf{E}}$ ou $s = t{\downarrow}$. Nous étendons cette notation à un ensemble de terme S, l'ensemble $S{\downarrow}$ représente l'ensemble $\{s{\downarrow} \mid s \in S\}$.

Dans la suite, lorsque nous étudierons une théorie équationnelle E, nous commencerons par calculer un système de réécriture convergent ou AC-convergent représentant E et nous travaillerons sur les termes en formes normales. Ainsi, dans le cas des systèmes convergents, nous pourrons considérer que deux termes sont égaux si leurs formes normales sont égales syntaxiquement, et dans le cas d'opérateurs plus complexes (*e.g.* « ou » exclusif, groupe abélien), nous travaillerons modulo AC : deux termes sont alors égaux si leurs formes normales sont égales modulo AC.

Pour les théories équationnelles étudiées dans le cadre de la vérification de protocoles cryptographiques, et en particulier dans cette thèse, il est relativement facile d'obtenir un système convergent associé à une théorie équationnelle donnée. Parfois, et c'est le cas pour la théorie AG, il n'est pas suffisant d'orienter les équations, il faut ajouter des règles de réécriture (conséquences logiques des règles déjà présentes) pour obtenir un système convergent.

Exemple 2.14
Dans le cas de la théorie équationnelle AG, *un système* AC*-convergent représentant* AG *peut être obtenu en orientant les axiomes $x \times 1 = x$ (U) et $x \times x^{-1} = 1$ (N) de gauche à droite et en ajoutant les conséquences suivantes :*

$$1^{-1} \;\rightarrow\; 1, \qquad (x^{-1})^{-1} \;\rightarrow\; x, \quad \textit{et} \quad (x \times y)^{-1} \;\rightarrow\; y^{-1} \times x^{-1}$$

Soit $\mathcal{R}_{\mathsf{AG}}$ le système de réécriture ainsi obtenu. Soit $t = (1 \times b) \times (a \times b^{-1})$ où a et b sont des constantes. Nous avons :

$$(1 \times b) \times (a \times b^{-1}) \;\rightarrow_{\mathcal{R}_{\mathsf{AG}}/\mathsf{AC}}\; b \times (a \times b^{-1}) \;\rightarrow_{\mathcal{R}_{\mathsf{AG}}/\mathsf{AC}}\; a.$$

Théories équationnelles ACh, ACUNh et AGh. Dans cette thèse, nous nous sommes particulièrement intéressés aux théories ACh, ACUNh et AGh. Les systèmes de réécriture AC-convergents correspondant à chacune de ces théories sont présentés à la figure 2.3. Par souci d'homogénéité, nous avons choisi la représentation additive pour la théorie groupe abélien.

Exemple 2.15
Considérons le terme $t = \mathsf{h}(\langle a, b\rangle + a + b) + -(\mathsf{h}(a)) + b + \mathsf{h}(-\langle a, b\rangle)$. Nous avons :

$$\begin{array}{lll} t & \rightarrow_{\mathcal{R}_{\mathsf{AGh}/\mathsf{AC}}} & \mathsf{h}(\langle a, b\rangle) + \mathsf{h}(a) + \mathsf{h}(b) + -(\mathsf{h}(a)) + b + \mathsf{h}(-\langle a, b\rangle) \\ & \rightarrow_{\mathcal{R}_{\mathsf{AGh}/\mathsf{AC}}} & \mathsf{h}(\langle a, b\rangle) + \mathsf{h}(b) + b + \mathsf{h}(-\langle a, b\rangle) \\ & \rightarrow_{\mathcal{R}_{\mathsf{AGh}/\mathsf{AC}}} & \mathsf{h}(\langle a, b\rangle) + \mathsf{h}(b) + b + -(\mathsf{h}(\langle a, b\rangle)) \\ & \rightarrow_{\mathcal{R}_{\mathsf{AGh}/\mathsf{AC}}} & \mathsf{h}(b) + b \end{array}$$

$$\mathcal{R}_{\mathsf{ACh}}: \quad \mathsf{h}(x+y) \rightarrow \mathsf{h}(x)+\mathsf{h}(y)$$

$$\mathcal{R}_{\mathsf{ACUNh}}: \quad \begin{array}{rcl} \mathsf{h}(x+y) & \rightarrow & \mathsf{h}(x)+\mathsf{h}(y) \\ x+0 & \rightarrow & x \end{array} \qquad \begin{array}{rcl} \mathsf{h}(0) & \rightarrow & 0 \\ x+x & \rightarrow & 0 \end{array}$$

$$\mathcal{R}_{\mathsf{AGh}}: \quad \begin{array}{rcl} \mathsf{h}(x+y) & \rightarrow & \mathsf{h}(x)+\mathsf{h}(y) \\ x+0 & \rightarrow & x \\ -(x+y) & \rightarrow & -(y)+-(x) \\ \mathsf{h}(-(x)) & \rightarrow & -(\mathsf{h}(x)) \end{array} \qquad \begin{array}{rcl} \mathsf{h}(0) & \rightarrow & 0 \\ x+-(x) & \rightarrow & 0 \\ -0 & \rightarrow & 0 \\ -(-x) & \rightarrow & x \end{array}$$

Figure 2.3 - Systèmes de réécriture AC-convergents associés aux théories ACh, ACUNh et AGh.

Le système de réécriture AC-convergent associé à une théorie, lorsqu'il existe, n'est pas nécessairement unique. Certains axiomes peuvent s'orienter aussi bien dans un sens que dans l'autre. C'est le cas, par exemple, de la théorie AG, où l'on peut éventuellement choisir d'orienter la conséquence $(x \times y)^{-1} = y^{-1} \times x^{-1}$ de droite à gauche. Une telle représentation, moins classique, peut s'avérer être très utile. Nous l'utiliserons au chapitre 7.

Chapitre 3

Intrus, protocoles

Sommaire

3.1 Intrus **36**
3.1.1 Contrôle du réseau 36
3.1.2 Capacités de déduction 36
3.1.2.1 Modèle à base d'un système d'inférence avec filtrage 36
3.1.2.2 Modèle à base d'un système d'inférence simple 38
3.2 Protocoles **40**
3.2.1 Modèles par rôles 41
3.2.1.1 Modèle par rôles avec filtrage 41
3.2.1.2 Modèle par rôles avec tests d'égalités 42
3.2.2 Rôles bien formés 43
3.2.2.1 Critère de décision dans le modèle avec tests d'égalités . . 45
3.2.2.2 Critère de décision dans le modèle avec filtrage 45
3.3 Comparaison du point de vue de l'expressivité **47**
3.3.1 Système d'inférence avec filtrage et système d'inférence simple . . . 47
3.3.2 Modèle par rôles avec filtrage et modèle avec tests d'égalités 51

Les protocoles cryptographiques sont des règles d'échanges de messages ayant pour but de sécuriser les communications. Pour procéder à la vérification automatique de ces protocoles, nous avons besoin d'une représentation non-ambiguë. La notation Alice-Bob, utilisée en introduction, n'est pas adaptée : elle n'indique pas clairement quelles sont les vérifications faites par un agent lors de la réception d'un message. Nous choisissons de modéliser un protocole par un ensemble de rôles : chaque rôle est un « programme » décrivant les actions réalisées par l'agent qui exécute ce programme. Cette approche est détaillée dans la partie 3.2. D'autre part, nous devons tenir compte du fait que les échanges de messages ont lieu sur un réseau de communication non sûr (*e.g.* Internet) : toutes les communications ont lieu en présence d'un attaquant, encore appelé intrus. Nos décrivons dans la partie 3.1 deux approches possibles permettant de représenter les capacités de l'intrus. Enfin, dans la partie 3.3, nous comparons les différentes modélisations de l'intrus et des protocoles introduites dans ce chapitre.

3.1 Intrus

Comme annoncé en introduction, nous supposons que toutes les communications ont lieu en présence d'un intrus. Il représente un adversaire essayant de profiter d'une éventuelle faille du protocole pour obtenir une information secrète : la clef privée d'un agent, le code confidentiel d'une carte de paiement, ...

3.1.1 Contrôle du réseau

Les premiers intrus considérés étaient *passifs* : ils se contentaient d'écouter les messages circulant sur le réseau et d'utiliser leur pouvoir de déduction pour analyser les messages ainsi obtenus.

Les intrus considérés à l'heure actuelle sont *actifs*. Ils peuvent écouter les messages circulant sur le réseau, mais aussi les intercepter, en synthétiser et en émettre de nouveaux. Pour simplifier, on considère que l'intrus intercepte systématiquement les messages envoyés sur le réseau. Ce n'est pas une restriction puisque l'intrus peut toujours intercepter un message et l'émettre sur le réseau sans le modifier. Ainsi, toutes les communications ont lieu dans un environnement hostile, représenté par l'intrus. L'intrus actif permet de modéliser le contrôle d'un attaquant sur un réseau comme Internet, où il n'y a pas de mécanisme implicite d'authentification. Sur un tel réseau, il est possible lors de l'envoi d'un message de falsifier son en-tête pour lui donner une adresse d'origine différente de l'adresse réelle d'envoi. Cette capacité à falsifier la provenance d'un message est modélisée par le fait que l'intrus peut toujours émettre un message sur le réseau en se faisant passer pour un agent honnête. Personne ne sera en mesure de remonter la provenance du message émis et de voir s'il s'agit d'un message retransmis par l'intrus ou d'un message émis par un agent honnête.

3.1.2 Capacités de déduction

Dans notre approche, les actions de l'intrus sont décrites par un ensemble de règles de déduction où apparaissent la capacité de l'intrus à former des nouveaux messages, à déchiffrer des messages lorsqu'il connaît la clef de déchiffrement, ... De tels modèles sont souvent appelés « modèles à la Dolev-Yao » en référence aux premiers modèles de D. Dolev et A.C. Yao [DY81]. Dans ces modèles, la capacité de mémoire et de calcul de l'intrus ne sont pas bornées. Ainsi, considérer un seul intrus et non plusieurs n'est pas une restriction : ce que deux intrus peuvent apprendre, un seul intrus peut le faire seul puisque ni sa capacité de calcul ni la taille de sa mémoire ne sont bornées. Afin de décrire précisément les capacités de l'intrus, on utilise un système de règles d'inférence.

3.1.2.1 Modèle à base d'un système d'inférence avec filtrage

Parmi les symboles de fonctions, noté $\mathcal{F}$, certains sont accessibles à l'intrus (comme la paire ou le chiffrement) alors que d'autres sont privés (comme le symbole constructeur des clefs privées priv). Cela nous conduit à décomposer l'ensemble $\mathcal{F}$ en deux sous-ensembles disjoints : $\mathcal{PF}$, l'ensemble des *symboles privés* et $\mathcal{VF}$, l'ensemble des *symboles publics* ou *visibles*.

On représente la connaissance de l'intrus par un ensemble T de termes clos. L'expression « $T \vdash u$ » signifie que l'intrus peut déduire u à partir de T. Autrement dit, l'intrus est capable, à partir de sa connaissance T, de déduire le message u en utilisant ses capacités de déduction. Le modèle couramment utilisé pour représenter les capacités de déduction de l'intrus est décrit à la figure 3.1 dans le cas du chiffrement symétrique. Il est connu sous le nom de *modèle de Dolev-Yao*.

$$\text{Projection (Proj}_1) \quad \frac{T \vdash \langle u, v\rangle}{T \vdash u} \qquad \text{Projection (Proj}_2) \quad \frac{T \vdash \langle u, v\rangle}{T \vdash v} \qquad \text{Déchiffrement (D)} \quad \frac{T \vdash \{u\}_v \quad T \vdash v}{T \vdash u}$$

$$\text{Composition (C)} \quad \frac{T \vdash u_1 \ \ldots \ T \vdash u_n}{T \vdash f(u_1, \ldots, u_n)} \quad \text{avec } f \in \mathcal{VF}$$

Figure 3.1 - Modèle de Dolev-Yao : $\mathcal{I}_{\mathsf{DY}}$.

Ce modèle permet à l'intrus de construire de nouveaux messages. Il peut par exemple concaténer deux messages qu'il sait déduire (règle C). Il peut également extraire un sous-message d'un message déjà connu. Plus précisément, il peut déchiffrer un message s'il possède la clef de déchiffrement correspondante (règle D). Il peut aussi récupérer la première (resp. deuxième) composante d'un message, si ce dernier est une paire, en utilisant la règle Proj_1 (resp. Proj_2).

Définition 3.1 (arbre de preuve)
Soit $\mathcal{I}$ un système d'inférence. Un arbre de preuve *de $T \vdash u$ dans $\mathcal{I}$ est un arbre tel que :*
- *chaque feuille est étiquetée par $T \vdash v$ avec $v \in T$,*
- *pour chaque nœud étiqueté par $T \vdash v$ ayant n fils étiquetés $T \vdash v_1$, ..., $T \vdash v_n$, il existe une instance d'une règle d'inférence de $\mathcal{I}$ ayant pour conclusion $T \vdash v$, et pour hypothèses $T \vdash v_1$, ..., $T \vdash v_n$,*
- *la racine est étiquetée par $T \vdash u$.*

On dit alors que u est déductible de *T* dans *$\mathcal{I}$ ou que $T \vdash u$* dans *$\mathcal{I}$.*

Définition 3.2 (taille d'une preuve, preuve minimale)
La taille *d'une preuve P, notée $|P|$, est le nombre de nœuds distincts dans P. Une preuve P de $T \vdash u$ est dite* minimale *s'il n'existe pas de preuve P' de $T \vdash u$ tel que $|P'| < |P|$.*

Exemple 3.3
Considérons l'ensemble T composé des messages $\{secret\}_{\langle a,b\rangle}$, a et b. L'arbre ci-dessous est un arbre de preuve de $T \vdash secret$ dans $\mathcal{I}_{\mathsf{DY}}$, mettant en évidence le fait que l'intrus peut déduire $secret$ à partir de la connaissance des messages $\{secret\}_{\langle a,b\rangle}$, a et b.

$$\dfrac{T \vdash \{secret\}_{\langle a,b\rangle} \qquad \dfrac{T \vdash a \qquad T \vdash b}{T \vdash \langle a, b\rangle}\,(\mathsf{C})}{T \vdash secret}\,(\mathsf{D})$$

Ce modèle, relativement simple, repose sur l'hypothèse du chiffrement parfait et ne correspond pas toujours à la réalité. Dans cette thèse, nous poussons les limites de la vérification des protocoles au-delà de cette hypothèse en prenant en compte les propriétés algébriques des primitives cryptographiques. Pour cela, nous avons besoin d'étendre le modèle standard à base de système d'inférence par une règle d'inférence particulière permettant de tenir compte de ces propriétés algébriques. Cette nouvelle règle, noté (Eq), est décrite à la figure 3.2. Nous notons $\mathcal{I}_{\mathsf{E}}$ le système d'inférence composé de cette unique règle.

$$\text{Égalité (Eq)} \quad \frac{T \vdash u}{T \vdash v} \quad \text{avec } u =_{\mathsf{E}} v$$

Figure 3.2 - Système d'inférence $\mathcal{I}_{\mathsf{E}}$.

Exemple 3.4

Soit $T = \{a \oplus b, b\}$ où $\oplus$ est un symbole visible représentant l'opérateur « ou » exclusif. Le terme b est déductible de T dans $\mathcal{I}_{\mathsf{DY}} \cup \mathcal{I}_{\mathsf{ACUN}}$, ce qui n'est pas le cas dans $\mathcal{I}_{\mathsf{DY}}$.

$$\dfrac{\dfrac{T \vdash a \oplus b \qquad T \vdash b}{T \vdash (a \oplus b) \oplus b}\,(\mathsf{C})}{T \vdash a}\,(\mathsf{Eq})$$

Notation : Dans la suite, nous noterons $(\mathcal{I}, \mathsf{E})$ le système d'inférence $\mathcal{I} \cup \mathcal{I}_{\mathsf{E}}$. Un arbre de preuve dans $(\mathcal{I}, \mathsf{E})$ est donc un arbre de preuve dans $\mathcal{I} \cup \mathcal{I}_{\mathsf{E}}$.

3.1.2.2 Modèle à base d'un système d'inférence simple

Une autre approche consiste à « tout mettre » dans la théorie équationnelle et à représenter les primitives classiques accompagnées de leurs *destructeurs* (*cf.* partie 2.2.1). Dans le cas du chiffrement, cela signifie :

- l'introduction d'un nouveau symbole public, noté dec, afin de modéliser l'opération de déchiffrement,
- l'ajout de l'équation $\mathsf{dec}(\{x\}_y, y) = x$ pour modéliser les propriétés des algorithmes de chiffrement et de déchiffrement.

Dans le cas de la concaténation, l'idée est essentiellement la même. Nous avons besoin d'ajouter deux symboles publics, notés généralement proj_1 et proj_2, ainsi que les équations $\mathsf{proj}_1(\langle x_1, x_2\rangle) = x_1$ et $\mathsf{proj}_2(\langle x_1, x_2\rangle) = x_2$.

L'article de D. Dolev et A.C. Yao [DY81], souvent cité comme référence dans le domaine, utilise le symbole de déchiffrement de façon explicite. Par la suite, ce modèle a un peu été délaissé au profit du modèle $\mathcal{I}_{\mathsf{DY}}$ permettant de raisonner dans l'algèbre libre (sans théorie équationnelle). Nous verrons, dans la partie 3.3.1, que le modèle décrit ci-dessus est en fait « équivalent » au modèle $\mathcal{I}_{\mathsf{DY}}$ présenté précédemment : ces deux modèles donnent à l'intrus le même pouvoir de déduction. Bien que naturelle, la traduction utilisée ci-dessus ne permet pas toujours de traduire un modèle à base d'un système d'inférence quelconque en un modèle à base d'un système d'inférence simple équivalent (*cf.* partie 3.3.1).

L'avantage de cette approche, consistant à « coder » les règles de déduction dans la théorie équationnelle, est son uniformité. Mise à part la règle d'inférence $\mathcal{I}_{\mathsf{E}}$ permettant de prendre en compte les propriétés algébriques des primitives cryptographiques, les règles d'inférences restantes constituent un *système d'inférence simple*, c'est-à-dire comprenant uniquement la règle de composition (C). Elle permet à l'intrus d'appliquer un symbole public à des messages qu'il sait déduire.

On remarquera qu'un système d'inférence simple est entièrement déterminé par l'ensemble des symboles visibles. Étant donné un système d'inférence simple $\mathcal{I}$, on notera $\mathcal{VF}_{\mathcal{I}}$ l'ensemble des symboles visibles qui lui est associé. De même, étant donné un ensemble de symboles de fonctions $\mathcal{H}$, on notera $\mathcal{I}_{\mathcal{H}}$ le système d'inférence réduit à la règle composition (C) pour les symboles de $\mathcal{H}$.

La notion d'arbre de preuve (Définition 3.1) est bien sûr toujours valable. Mais on peut aussi voir un arbre de preuve comme l'application d'un *contexte* public correspondant à l'application d'un certain nombre d'opérations à la disposition de l'intrus.

Définition 3.5 ($\mathcal{H}$-contexte)
*Soit $\mathcal{H}$ un ensemble de symboles de fonctions. Un $\mathcal{H}$-*contexte *est un λ-terme $\lambda x_1, \dots, x_n.t$ avec $t \in \mathcal{T}(\mathcal{H}, \{x_1, \dots, x_n\})$, noté aussi $t[x_1, \dots x_n]$. L'application du contexte $t[x_1, \dots x_n]$ à des termes $u_1, \dots, u_n$ s'écrit $t[u_1, \dots, u_n]$.*

Notation : Étant donné une théorie équationnelle E, nous appellerons E-contexte un contexte construit à partir des symboles de fonction de $sig(\mathsf{E})$.

Nous pouvons maintenant définir la notion de preuve par contexte.

Définition 3.6 (preuve par contexte)
Soient $\mathcal{I}$ un système d'inférence simple et E *une théorie équationnelle. Soit T un ensemble de termes, et posons $T = \{t_1, \dots, t_n\}$. Une* preuve par contexte *de $T \vdash u$ dans $\mathcal{I} \cup \mathcal{I}_{\mathsf{E}}$ est un $\mathcal{VF}_{\mathcal{I}}$-contexte $C[x_1, \dots, x_n]$ tel que $C[t_1, \dots, t_n] =_{\mathsf{E}} u$.*

Pour les systèmes d'inférence simples, les notions d'arbre de preuve et de preuve par contexte coïncident. Nous avons le lemme suivant :

Lemme 3.7
Soient $\mathcal{I}$ un système d'inférence simple et E *une théorie équationnelle. Soient $T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$. Il existe une preuve par contexte de $T \vdash u$ dans $(\mathcal{I}, \mathsf{E})$ si, et seulement si, il existe un arbre de preuve de $T \vdash u$ dans $(\mathcal{I}, \mathsf{E})$.*

Démonstration.
($\Rightarrow$) Notons $t_1, \dots, t_n$ les termes de T. Par hypothèse, il existe $C \in \mathcal{T}(\mathcal{VF}_{\mathcal{I}}, \{x_1, \dots, x_n\})$ tel que $C[t_1, \dots, t_n] =_{\mathsf{E}} u$. L'arbre de preuve appliquant un à un les symboles de fonctions de C et se terminant par l'instance de (Eq) écrit ci-dessous convient :

$$\frac{T \vdash C[t_1, \dots, t_n]}{T \vdash u} \;(\mathsf{Eq})$$

($\Leftarrow$) Soit P un arbre de preuve de $T \vdash u$ dans $(\mathcal{I}, \mathsf{E})$. Nous allons montrer par induction sur P qu'il existe un contexte $C \in \mathcal{T}(\mathcal{VF}_{\mathcal{I}}, \{x_1, \dots, x_n\})$ tel que $C[t_1, \dots, t_n] =_{\mathsf{E}} u$.

Cas de base : Si P se réduit à une feuille, alors le contexte vide convient.

Étape d'induction : Nous distinguons deux cas suivant la dernière règle d'inférence de P.

– Si P se termine par une instance de (Eq), alors P est de la forme :

$$\frac{T \vdash v}{T \vdash u}\ (\mathsf{Eq})$$

avec $u =_{\mathsf{E}} v$. Par hypothèse d'induction, il existe C' tel que $C'[t_1, \ldots, t_n] =_{\mathsf{E}} v$. Le contexte C' convient puisque $v =_{\mathsf{E}} u$.

– Si P se termine par une instance de (C), alors P est de la forme :

$$\frac{T \vdash u_1 \quad \ldots \quad T \vdash u_k}{T \vdash f(u_1, \ldots, u_k)}\ (\mathsf{C})$$

avec $f \in \mathcal{VF}_{\mathcal{I}}$. Par hypothèse d'induction, il existe des contextes $C_1, \ldots, C_k$ tels que $C_i[t_1, \ldots, t_n] =_{\mathsf{E}} u_i$ pour tout $i \leq k$. Le contexte $C' = f(C_1, \ldots, C_k)$ convient. En effet, on a $C'[t_1, \ldots, t_n] = f(C_1[t_1, \ldots, t_n], \ldots, C_k[t_1, \ldots, t_n]) =_{\mathsf{E}} f(u_1, \ldots, u_k)$.

□

La notion de preuve par contexte n'a pas de sens dans le cas d'un système d'inférence quelconque. Dans la suite, nous appellerons *preuve*, aussi bien un arbre de preuve qu'une preuve par contexte.

3.2 Protocoles

Pour représenter les protocoles, nous avons jusqu'ici utilisé la représentation intuitive utilisée dans [CJ97]. Considérons par exemple une variante du protocole de Denning-Sacco [DS81] :

$$\begin{array}{lcll} A & \rightarrow & B : & A, \{\{\langle\langle A, B\rangle, K_{ab}\rangle\}_{\mathsf{priv}(A)}\}_{\mathsf{pub}(B)} \\ B & \rightarrow & A : & \{secret\}_{K_{ab}} \end{array}$$

Ce protocole a pour but d'établir une clef de session K_{ab} (clef symétrique) entre les agents A et B. À la première étape du protocole, l'agent A envoie d'une part son nom A, et d'autre part un triplet contenant en particulier la clef de session K_{ab} qu'il vient d'engendrer. Ce triplet est chiffré par un algorithme de chiffrement asymétrique avec la clef privée de A (notée $\mathsf{priv}(A)$), puis avec la clef publique de B (notée $\mathsf{pub}(B)$). À la deuxième étape du protocole, l'agent B reçoit le message $A, \{\{\langle\langle A, B\rangle, K_{ab}\rangle\}_{\mathsf{priv}(A)}\}_{\mathsf{pub}(B)}$. Il déchiffre la deuxième partie du message à l'aide de sa clef privée $\mathsf{priv}(B)$. Il effectue ensuite un déchiffrement avec la clef publique de l'agent dont le nom correspond à la première composante du message reçu, ici $\mathsf{pub}(A)$, ce qui lui permet de récupérer la dernière composante du triplet : K_{ab}. L'obtention de cette clef va lui permettre d'envoyer son secret, $secret$, chiffré par un algorithme de chiffrement symétrique avec la clef K_{ab}.

Cette représentation simple du protocole est en fait ambiguë. Que fait l'agent B lorsqu'il reçoit le message censé correspondre au message $A, \{\{\langle\langle A, B\rangle, K_{ab}\rangle\}_{\mathsf{priv}(A)}\}_{\mathsf{pub}(B)}$? Il peut déchiffrer la deuxième partie du message avec sa clef privée, puis déchiffrer le résultat obtenu avec la clef publique de l'agent figurant dans la première composante du message. Il récupère ainsi K_{ab} et peut alors émettre $\{secret\}_{K_{ab}}$ sur le réseau. Il peut aussi faire un certain nombre de vérification supplémentaires. Il peut vérifier que le message obtenu après les deux déchiffrements est un triplet dont :

1. la première composante est le nom d'un agent : celui dont il a utilisé la clef publique pour réaliser le deuxième déchiffrement,
2. la deuxième composante est son nom.

Cet exemple illustre bien que cette représentation intuitive des protocoles ne décrit pas précisément toutes les actions des agents, mais se contente de donner une « trace d'exécution normale » du protocole. L'étude des protocoles cryptographiques commence donc par une première étape de modélisation.

3.2.1 Modèles par rôles

De nombreux modèles pour spécifier les protocoles cryptographiques existent dans la littérature. Nous allons nous intéresser au *modèle par rôles*. Ce type de modèle, introduit par I. Cervesato *et al.* [CDL+99], a été largement utilisé par la suite (*e.g.* [RT03, CLS03, MS05]). Ce modèle est assez proche du modèle des strand spaces [THG99]. Les actions des différents acteurs du protocole sont regroupées au sein d'un *rôle*, *i.e.* une séquence d'instructions. Un protocole est un ensemble de rôles. La différence essentielle entre les deux modèles présentés ci-dessous réside dans la définition de ce qu'est une instruction. Nous effectuons une comparaison de ces deux modèles dans la partie 3.3.2.

3.2.1.1 Modèle par rôles avec filtrage

Dans ce modèle, chaque pas de protocole représente une action élémentaire, et associe à un message reçu par un agent la réponse correspondante émise par ce même agent. L'ensemble des messages acceptés par l'agent, à un pas de protocole donné, est spécifié par un filtre. Un pas de protocole est donc une paire de termes u, v (éventuellement non clos), notée $\mathsf{recv}(u); \mathsf{send}(v)$. L'ensemble des messages acceptés par l'agent est spécifié par le filtre u. Cette approche est assez naturelle et décrit clairement, pas à pas, les actions des agents, sans entrer dans le détail des opérations réellement effectuées par ces derniers.

Exemple 3.8
Nous décrivons ici, dans le modèle avec filtrage, *le rôle B, noté $R(z_b)$, de la variante du protocole de Denning-Sacco, présentée en introduction de la partie 3.2.*

$$\nu\ secret\ .\ \mathit{recv}(x_A, \{\{\langle\langle x_A, z_b\rangle, x_{K_{ab}}\rangle\}_{\mathsf{priv}(x_A)}\}_{\mathsf{pub}(z_b)});\ \mathit{send}(\{secret\}_{x_{K_{ab}}})$$

Cette modélisation correspond au cas où l'agent z_b (z_b est appelé *paramètre du rôle*) jouant le rôle B effectue toutes les vérifications supplémentaires. En effet, le motif utilisé lors du filtrage, c'est-à-dire $x_A, \{\{\langle\langle x_A, z_b\rangle, x_{K_{ab}}\rangle\}_{\mathsf{priv}(x_A)}\}_{\mathsf{pub}(z_b)}$, assure que la première composante x_A du message est le nom d'un agent, celui-là même dont la clef privée est utilisée pour réaliser le chiffrement $\mathsf{priv}(x_A)$ et que l'on retrouve comme première composante du triplet obtenu après déchiffrement. L'agent, lorsqu'il reçoit le message, est capable de voir que la deuxième composante du message est un chiffré avec sa clef publique $\mathsf{pub}(z_b)$, et de vérifier que le triplet chiffré contient bien son nom en deuxième composante.

Dans le modèle par rôles avec filtrage, un rôle est défini formellement de la façon suivante :

Définition 3.9 (Rôle dans le modèle avec filtrage)
Un rôle dans le modèle avec filtrage *est la donnée :*
– *d'un ensemble de paramètres, noté $z_1, \ldots, z_p$,*

– *d'un ensemble de nonces, noté $\tilde{n}$, et*
– *d'une séquence d'instructions de la forme recv(u_1); send$(v_1); \ldots;$ recv(u_k); send(v_k).*
On le note :

$$R(z_1, \ldots, z_p) = \nu\tilde{n}.\mathit{recv}(u_1);\ \mathit{send}(v_1); \ldots; \mathit{recv}(u_k);\ \mathit{send}(v_k).$$

Les termes u_i et v_i ne sont pas nécessairement clos.

L'exécution d'un tel rôle par un agent a consiste à engendrer un ensemble de nonces (les $\tilde{n}$), puis à exécuter la séquence d'instructions décrite au sein du rôle. L'exécution d'une instruction « $\mathsf{recv}(u);\ \mathsf{send}(v)$ » par un agent consiste en la réception d'un message m accepté par le filtre u, et l'envoi d'un message correspondant au terme v. Par exemple, si $u = \langle x_1, x_2 \rangle$, alors le message reçu, pour être accepté par le filtre et déclencher le pas de protocole, devra être une paire, c'est-à-dire un message de la forme $\langle m_1, m_2 \rangle$. L'agent remplacera, dans tout son programme, les occurrences de la variable x_1 par m_1 et les occurrences de x_2 par m_2. Il émettra ensuite le message correspondant au terme v sur le réseau. Les variables du rôle sont ainsi instanciées au fur et à mesure de la réception des différents messages.

Cette notion de rôle est trop générale et permet de spécifier des protocoles non réalistes demandant à un agent d'émettre un message m correspondant à un terme v avant que ce dernier soit totalement instancié. Nous restreindrons donc cette notion de rôle dans la partie 3.2.2 en introduisant la notion de rôles *normalisés* et *bien formés*.

3.2.1.2 Modèle par rôles avec tests d'égalités

Dans la modélisation précédente du rôle B, il est supposé implicitement qu'un certain nombre de vérifications sont effectuées. Dans le modèle par rôles avec tests d'égalités, les tests effectués sont rendus plus explicites. Une instruction est un triplet constitué d'une paire de termes et d'un ensemble d'équations. Ce modèle ne permet plus de faire du filtrage sur le message reçu. L'ensemble des messages acceptés par un agent à un pas de protocole donné sont ceux satisfaisant les différents tests spécifiés par les équations. Ce modèle nécessite l'utilisation des destructeurs explicites afin de pouvoir accéder aux différentes composantes d'un message. Auparavant cette opération était effectuée par l'intermédiaire du filtrage.

Exemple 3.10
Nous décrivons, dans le modèle avec tests d'égalité, le rôle B, noté $R(z_b)$, de la variante du protocole de Denning-Sacco, présentée en introduction de la partie 3.2.

$$\begin{array}{l}
\nu\ secret\ .\\
\mathit{recv}(x);\\
z_b = \mathsf{proj}_2(\mathsf{proj}_1(\mathsf{dec}(\mathsf{dec}(\mathsf{proj}_2(x), \mathsf{priv}(z_b)), \mathsf{pub}(\mathsf{proj}_1(x)))));\\
\mathsf{proj}_1(x) = \mathsf{proj}_1(\mathsf{proj}_1(\mathsf{dec}(\mathsf{dec}(\mathsf{proj}_2(x), \mathsf{priv}(z_b)), \mathsf{pub}(\mathsf{proj}_1(x)))));\\
\mathit{send}\big(\{secret\}_{\mathsf{proj}_2(\mathsf{dec}(\mathsf{dec}(\mathsf{proj}_2(x), \mathsf{priv}(z_b)), \mathsf{pub}(\mathsf{proj}_1(x))))}\big)
\end{array}$$

Les deux tests décrits précédemment sont rendus explicites par l'utilisation des deux égalités. Le premier test $z_b = \mathsf{proj}_2(\mathsf{proj}_1(\mathsf{dec}(\mathsf{dec}(\mathsf{proj}_2(x), \mathsf{priv}(z_b)), \mathsf{pub}(\mathsf{proj}_1(x)))))$ correspond au fait que l'agent z_b jouant le rôle B vérifie que son nom se retrouve en deuxième composante du triplet. Le second test vérifie que la première composante du message reçu, c'est à dire $\mathsf{proj}_1(x)$, est égale à la première composante du triplet obtenu après avoir fait les déchiffrements. On peut aussi vouloir spécifier un agent plus laxiste, n'effectuant pas ces vérifications. Il suffit pour cela, de ne pas écrire ces deux égalités.

Définition 3.11 (Rôle dans le modèle avec tests d'égalités)
Un rôle dans le modèle avec tests d'égalités *est la donnée :*

- *d'un ensemble de paramètres, noté $z_1, \ldots, z_p$,*
- *d'un ensemble de nonces, noté $\tilde{n}$, et*
- *d'une séquence d'instructions : $\mathit{recv}(x_1);\ \mathcal{E}_1;\ \mathit{send}(v_1); \ldots ; \mathit{recv}(x_k);\ \mathcal{E}_k;\ \mathit{send}(v_k)$.*

On le note :

$$R(z_1, \ldots, z_p) = \nu\tilde{n}.\mathit{recv}(x_1);\ \mathcal{E}_1\ \mathit{send}(v_1); \ldots ; \mathit{recv}(x_k);\ \mathcal{E}_k\ \mathit{send}(v_k).$$

Les termes v_i ne sont pas nécessairement clos, les x_i sont des variables et les $\mathcal{E}_i$ des ensembles d'équations.

Dans cette nouvelle modélisation, aucun filtrage n'est réalisé au moment de la réception d'un message. En revanche, pour que la $i^{\text{ème}}$ instruction soit déclenchée lors de la réception d'un message m, il faut que ce message satisfasse les différentes conditions spécifiées par le système d'équations $\mathcal{E}_i$.

Dans cette thèse, nous nous sommes intéressés au problème de la sécurité d'un protocole pour un nombre borné de sessions, c'est-à-dire en présence d'un ensemble fini d'instances de rôles. Cette notion se définit formellement de la façon suivante :

Définition 3.12 (instance d'un rôle)
Soit R un rôle ayant pour paramètres $z_1, \ldots, z_p$, pour ensemble de nonces $n_1, \ldots, n_l$ et pour séquence d'instructions S. L'instance $R(q_1, \ldots, q_p)$ du rôle R est la séquence d'instructions $S\sigma$ où σ est la substitution définie par $\{z_1 \mapsto q_1; \ldots, z_p \mapsto q_p\}$.

Dans la suite, nous serons amenés à considérer plusieurs instances, et nous supposerons toujours que deux instances différentes n'ont aucune variable et aucun nonce en commun. Ceci peut toujours s'obtenir en effectuant un renommage.

3.2.2 Rôles bien formés

La notion de rôle, et donc de protocole, que nous venons de présenter est trop générale. Elle permet de modéliser des protocoles non réalistes en permettant à un rôle d'utiliser une variable x dans un envoi de message sans en connaître la valeur. L'agent serait alors capable d'émettre un message utilisant la variable x alors que cette variable n'a pas été instanciée lors d'un pas de protocole précédent. De plus, nous souhaitons prendre en compte les propriétés algébriques des primitives cryptographiques et pour cela, nous voulons un modèle de protocoles où ni les agents, ni l'intrus, ne peuvent différencier deux termes appartenant à la même classe d'équivalence. Pour ces raisons, nous supposerons toujours que les messages échangés et les termes apparaissant dans les rôles sont normalisés et nous nous intéresserons uniquement à la classe des rôles dits *bien formés*.

Remarque : Dans la suite de ce chapitre, nous considérons implicitement que les termes sont normalisés par rapport au système de réécriture (AC)-convergent associé à la théorie équationnelle E que nous étudions, et nous écrivons u (resp. $u\theta$) à la place de $u\downarrow$ (resp. $u\theta\downarrow$).

Une première restriction raisonnable est de demander que chacune des variables apparaissant dans une instruction send(v) apparaisse « avant » dans une instruction recv(u). L'idée est d'assurer que lorsque l'agent devra envoyer le message correspondant à v, toutes les variables de v auront été instanciées. Cette condition syntaxique simple est cependant trop générale pour les rôles avec filtrage : elle permet de spécifier des rôles non-réalistes.

Exemple 3.13
Considérons par exemple, le pas de protocole suivant recv($x \oplus y$); send($\langle x, y\rangle$) où $\oplus$ est l'opérateur « ou » exclusif.

Ce pas de protocole est « non-déterministe » et par conséquent non-réaliste. En effet, lorsque l'agent exécutant ce rôle reçoit le message 0, il peut envoyer en réponse à cette réception différents messages, comme par exemple $\langle a, a\rangle$ ou $\langle b, b\rangle$.

Nous devons donc définir une notion de bonne formation plus restrictive. La définition 3.14 ci-dessous est valable pour les deux types de rôles que nous venons de présenter. Cette notion de bonne formation est, dans le cas d'un rôle avec tests d'égalités, indépendante des équations présente dans la séquence d'instructions.

Définition 3.14 (Rôle bien formé)
Un rôle R ayant pour paramètres $z_1, \ldots, z_p$ est dit bien formé *si la séquence d'instructions recv(u_1) ; send(v_1) ; $\ldots$; recv(u_k) ; send(v_k) qui lui est associée satisfait la propriété suivante :*

$$\text{pour toute substitution } \theta, \text{ pour tout } i \leq k, \text{ pour tout } x \in vars(v_i\theta),$$
$$\text{on a } x \in vars(\{z_1\theta, \ldots, z_p\theta, u_1\theta, \ldots, u_i\theta\}).$$

Cette classe de protocole très générale a été introduite par J. Millen et V. Shmatikov [MS03]. Elle englobe la classe des protocoles étudiée par Y. Chevalier *et al.* dans le cas de la théorie du « ou » exclusif [CKRT03b] ou de l'exponentielle modulaire [CKRT03a]. Ces derniers avaient préféré une condition syntaxique : dans un recv, une somme introduit au plus une nouvelle variable. Cependant, comme illustré par l'exemple ci-dessous, cette condition syntaxique écarte un certain nombre de protocoles réalistes.

Exemple 3.15
Considérons le rôle suivant où $\oplus$ est l'opérateur « ou » exclusif.

$$\text{recv}(x \oplus y)\ ;\ \text{send}(0)\ ;\ \text{recv}(x)\ ;\ \text{send}(y)$$

Ce rôle ne vérifie pas la condition syntaxique de bonne formation définie par Y. Chevalier et al. dans [CKRT03b] puisque le premier message $x \oplus y$ contient deux nouvelles variables. En revanche, ce rôle est bien formé selon la définition 3.14.

Il est important de remarquer que la notion de bonne formation que nous venons d'introduire sur les rôles est très générale. Nous ne nous sommes pas restreints à considérer des rôles « exécutables ». Autrement dit, nous ne nous sommes pas intéressés à savoir si l'agent, qui devra exécuter le rôle, a la connaissance et le pouvoir suffisants pour réaliser les actions qui lui sont demandées.

Exemple 3.16
Le rôle composé de l'unique instruction recv($\{s\}_x$); send(x) est un rôle bien formé bien qu'il ne soit pas « exécutable ».

3.2.2.1 Critère de décision dans le modèle avec tests d'égalités

Dans le cas du modèle par rôles avec tests d'égalités, il est facile de savoir si un rôle est bien formé. En effet, la condition de bonne formation que nous avons énoncée (Définition 3.14) est équivalente à la condition syntaxique, énoncée à la page 44, sur l'ordre d'apparition des variables. Nous avons le lemme suivant :

Lemme 3.17
Soit R un rôle dans le modèle avec tests d'égalités ayant pour paramètres $z_1, \ldots, z_p$, et pour séquence d'instructions $\mathit{recv}(x_1); \mathcal{E}_1; \mathit{send}(v_1); \ldots; \mathit{recv}(x_k);\ \mathcal{E}_k; \mathit{send}(v_k)$. Le rôle R est bien formé si, et seulement si, pour tout $i \leq k$, on a $vars(v_i) \subseteq \{z_1, \ldots, z_p, x_1, \ldots, x_i\}$.

Démonstration.
($\Rightarrow$) Par hypothèse, pour toute substitution θ, pour tout $i \leq k$, pour tout $x \in vars(v_i\theta)$ nous avons $x \in vars(\{z_1\theta, \ldots, z_p\theta, x_1\theta, \ldots, x_i\theta\})$. Cette propriété est en particulier vraie si θ est l'identité, d'où le résultat.
($\Leftarrow$) Soient θ une substitution et i un entier compris entre 1 et k. Soit $x \in vars(v_i\theta)$. Nous avons :

1. soit $x \in vars(v_i)$ et $x \notin dom(\theta)$,
2. soit il existe $y \in vars(v_i)$ tel que $x \in vars(y\theta)$.

Par hypothèse, $vars(v_i) \subseteq \{z_1, \ldots, z_p, x_1, \ldots, x_i\}$. Dans le premier cas, nous en déduisons qu'il existe $j \leq p$ tel que $x = z_j$, ou alors qu'il existe $j \leq i$ tel que $x = x_j$. Dans le deuxième cas, nous pouvons appliquer le même raisonnement sur la variable y : soit il existe $j \leq p$ tel que $y = z_j$, soit il existe $j \leq i$ tel que $y = x_j$. Dans les deux cas, nous concluons. $\square$

3.2.2.2 Critère de décision dans le modèle avec filtrage

Dans le cas d'un rôle dans le modèle avec filtrage, il n'est pas aussi aisé de décider si un rôle est bien formé. La condition syntaxique, énoncé dans le lemme 3.17 (dans le cas du modèle avec tests d'égalités), ne suffit pas. Il est possible de construire un rôle qui ne soit pas bien formé et qui respecte cette condition sur l'ordre d'apparition des variables (une variable apparaît dans un recv avant d'apparaître dans un send).

Exemple 3.18
Reprenons le rôle décrit dans exemple 3.13, c'est-à-dire $\mathit{recv}(x \oplus y); \mathit{send}(\langle x, y\rangle)$. La condition syntaxique sur l'ordre d'apparition des variables x et y est respectée. En revanche, ce rôle n'est pas bien formé. En effet, considérons la substitution $\theta : x \mapsto y$. Nous obtenons, $(x \oplus y)\theta = 0$ et $\langle x, y\rangle\theta = \langle y, y\rangle$. Nous avons donc $y \in vars(\langle x, y\rangle\theta)$ alors que $y \notin vars((x \oplus y)\theta)$.

Dans le modèle avec filtrage, la notion de rôle bien formé (*cf.* Définition 3.14), introduite par J. Millen et V. Shmatikov [MS03], est très générale. Elle permet de modéliser tous les protocoles *déterministes*. Intuitivement, un rôle est déterministe si à chaque envoi de message, l'agent exécutant ce rôle sait très précisément quel est le message qu'il doit envoyer compte tenu des messages qu'il a déjà reçus. Cette notion de rôle déterministe se définit formellement de la façon suivante :

Définition 3.19 (Rôle déterministe)
Soit $R(z_1, \ldots, z_p) = \nu\tilde{n}.\mathsf{recv}(u_1); \mathcal{E}_1; \mathsf{send}(v_1); \ldots; \mathsf{recv}(u_k); \mathcal{E}_k; \mathsf{send}(v_k)$ un rôle. Le rôle R est dit déterministe à l'étape i *si pour tous* E-*unificateurs* θ_1, θ_2 *de* $\bigcup_{j=1}^{i} \mathcal{E}_j$*, nous avons :*

$$(z_1\theta_1 =_{\mathsf{E}} z_1\theta_2 \wedge \ldots \wedge z_p\theta_1 =_{\mathsf{E}} z_p\theta_2 \wedge u_1\theta_1 =_{\mathsf{E}} u_1\theta_2 \wedge \ldots \wedge u_i\theta_1 =_{\mathsf{E}} u_i\theta_2) \Rightarrow v_i\theta_1 =_{\mathsf{E}} v_i\theta_2$$

Exemple 3.20
Reprenons l'exemple 3.13. Le pas de protocole $\mathsf{recv}(x \oplus y); \mathsf{send}(\langle x, y\rangle)$, où $\oplus$ est l'opérateur « ou » exclusif, n'est pas déterministe. Les substitutions $\theta_a : x, y \mapsto a$ et $\theta_b : x, y \mapsto b$ ne satisfont pas l'implication. Nous avons $\langle x, y\rangle\theta_a \neq \langle x, y\rangle\theta_b$ alors que $(x \oplus y)\theta_a = (x \oplus y)\theta_b = 0$.

L'avantage d'une telle notion est que l'on peut décider si un protocole est déterministe dès lors que le fragment existentiel de la théorie du premier ordre modulo la théorie équationnelle E considérée est décidable. Nous obtenons ainsi des procédures permettant de décider si un rôle est déterministe dans le cas de la théorie ACUNh [DLLT06], mais aussi pour les théories équationnelles ACUN, AG et DH [CD05].

Dans le cas du modèle avec filtrage, la classe des protocoles déterministes est incluse dans la classe des protocoles bien formés.

Lemme 3.21
Soit $R(z_1, \ldots, z_p) = \nu\tilde{n}.\mathsf{recv}(u_1);\ \mathsf{send}(v_1); \ldots\ ; \mathsf{recv}(u_k);\ \mathsf{send}(v_k)$ un rôle dans le modèle avec filtrage. Si R est déterministe alors R est bien formé.

Démonstration.
Supposons que R ne soit pas bien formé. Il existe alors une substitution θ, un indice i et une variable x tels que $x \in vars(v_i\theta)$ et $x \notin vars(\{z_1\theta, \ldots, z_p\theta, u_1\theta, \ldots, u_i\theta\})$. Nous allons montrer que R n'est pas déterministe à l'étape i. Soient a et b deux nouvelles constantes. Posons $\theta_1 = \theta \circ \{x \mapsto a\}$ et $\theta_2 = \theta \circ \{x \mapsto b\}$. Nous avons $z_j\theta_1 = z_j\theta_2$ pour tout $j \leq p$ et $u_j\theta_1 = u_j\theta_2$ pour tout $j \leq i$. Or nous avons $v_i\theta_1 \neq v_i\theta_2$ puisque $x \in vars(v_i\theta)$. Nous en déduisons donc que le rôle R n'est pas déterministe. □

Ainsi, nous avons un critère permettant d'assurer qu'un protocole est bien formé ce qui nous permet d'appliquer la procédure de décision que nous avons développée pour traiter les protocoles bien formés. Dans le cas de théories équationnelles telles que ACUN, AG, ACUNh, ... nous ne savons pas si la classe des protocoles déterministes est strictement plus grande que la classe des protocoles bien formés. Nous conjecturons que non.

Cependant, comme l'illustre l'exemple 3.22, la classe des protocoles déterministes est strictement plus grande si l'on s'intéresse à des théories equationelles un peu plus exotiques.

Exemple 3.22
Considérons la théorie équationnelle E *composée des deux axiomes $f(a, a) = c$ et $f(b, b) = c$ (a, b et c sont des constantes). Considérons le rôle $\mathsf{recv}(f(x, x));\ \mathsf{send}(x)$. Ce rôle n'est pas déterministe. En effet, considérons $\theta_1 = \{x \mapsto a\}$ et $\theta_2 = \{x \mapsto b\}$. On a $f(x, x)\theta_1 =_{\mathsf{E}} f(x, x)\theta_2$ et $x\theta_1 \neq_{\mathsf{E}} x\theta_2$. En revanche, ce rôle est bien formé.*

Notons également que le lemme 3.21 est faux dans le modèle par rôle avec tests d'égalités (*cf.* exemple 3.23). Ceci est dû au fait que les équations ne sont pas prises en compte lors de la vérification de la bonne formation du rôle, alors qu'elles jouent un rôle prépondérant pour déterminer si un rôle est déterministe ou non.

Exemple 3.23
Considérons le rôle recv$(x); y = 0;$ *send*(y)*. Ce rôle est déterministe. En effet, quel que soit le message reçu, l'agent exécutant ce rôle va émettre le message* 0 *sur le réseau. Ce rôle n'est pourtant pas bien formé.*

Les liens entre différentes classes de protocoles (bien formé, déterministe, ...) que nous avons introduites dans cette partie 3.2.2 sont résumés dans la figure 3.3. Dans la suite, nous considérons la classe des protocoles bien formés.

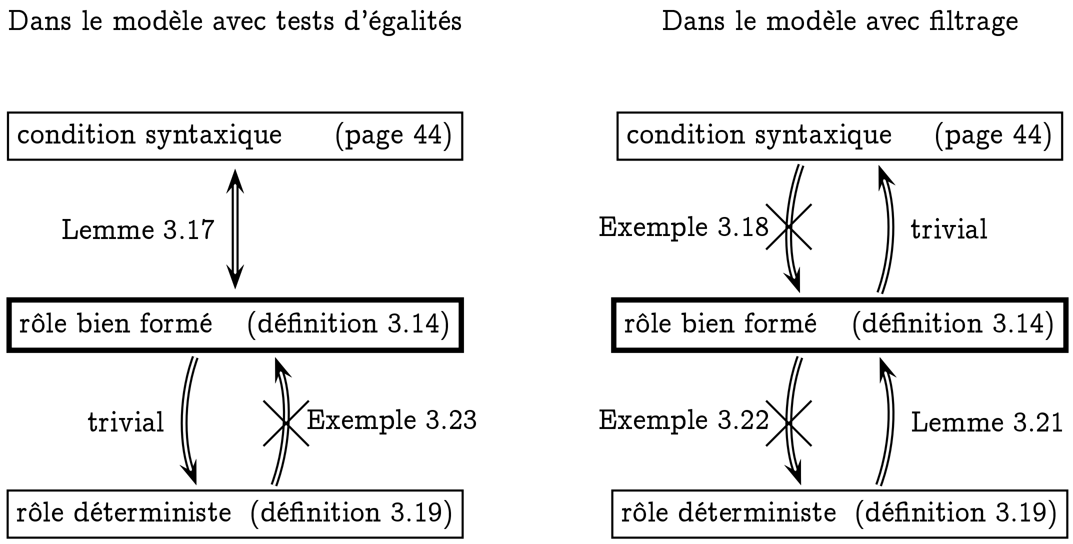

Figure 3.3 - Relation entre les différentes classes de protocoles.

Nous reverrons cette notion de bonne formation dans le chapitre 4 lorsque nous traduirons le problème de vérification d'un protocole en un système de contraintes symboliques. Ces restrictions énoncées au niveau des rôles ont en fait pour but d'assurer que les systèmes de contraintes que nous allons générer possèdent de bonnes propriétés dans le but d'obtenir des procédures de décision pour la résolution de tels systèmes.

3.3 Comparaison du point de vue de l'expressivité

3.3.1 Système d'inférence avec filtrage et système d'inférence simple

Nous avons présenté deux approches permettant de modéliser le pouvoir de déduction de l'intrus. La première consiste en un système d'inférence permettant de réaliser du filtrage sur les prémisses, accompagné d'une théorie équationnelle. La seconde, *a priori* plus restrictive, nous oblige à considérer un système d'inférence simple (règles de composition) également accompagné d'une théorie équationnelle.

Tout d'abord, nous allons montrer comment transformer un système d'inférence $\mathcal{I}$ avec filtrage (mais sans théorie équationnelle) en un système d'inférence simple $\mathcal{I}'$ et une théorie équationnelle $\mathsf{E}_{\mathcal{I}}$ de telle sorte que ces deux modèles représentent un intrus ayant des capacités de déduction « équivalentes ». Plus précisément, nous montrons que :

Proposition 3.24
Soient $T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$. Nous avons :

$$T \vdash u \textit{ dans } \mathcal{I} \Leftrightarrow T \vdash u \textit{ dans } (\mathcal{I}', \mathsf{E}_{\mathcal{I}}).$$

Démonstration.
Nous construisons $\mathcal{I}'$ et $\mathsf{E}_{\mathcal{I}}$ de la façon suivante : Soit n le nombre de règles d'inférence dans $\mathcal{I}$. Soit $\mathcal{G} = \{g_1, \ldots, g_n\}$, un ensemble de symboles de fonctions tels que $\mathcal{F} \cap \mathcal{G} = \varnothing$. Soit $\dfrac{T \vdash u_1 \quad \ldots \quad T \vdash u_k}{T \vdash u}\ (\mathsf{I}_i)$ la $i^{\text{ème}}$ règle de $\mathcal{I}$, On ajoute :

- l'équation $g_i(u_1, \ldots, u_k) = u$ dans $\mathsf{E}_{\mathcal{I}}$,
- et la règle $\mathsf{R_i}$, $\dfrac{T \vdash x_1 \quad \ldots \quad T \vdash x_k}{T \vdash g_i(x_1, \ldots, x_k)}\ (\mathsf{R_i})$ dans $\mathcal{I}'$.

Notation : Nous notons $\rightarrow_{\mathsf{E}_{\mathcal{I}}}$ la relation associée au système de réécriture obtenu en orientant les équations de $\mathsf{E}_{\mathcal{I}}$ de gauche à droite.

Nous pouvons maintenant établir l'équivalence entre les systèmes d'inférence $\mathcal{I}$ et $(\mathcal{I}', \mathsf{E}_{\mathcal{I}})$.
($\Rightarrow$) Soient $T \subseteq \mathcal{T}(\mathcal{F})$, $u \in \mathcal{T}(\mathcal{F})$ et P un arbre de preuve de $T \vdash u$ dans $\mathcal{I}$. Nous allons montrer par induction sur l'arbre de preuve P que l'on peut construire P', un arbre de preuve de $T \vdash u$ dans $(\mathcal{I}', \mathsf{E}_{\mathcal{I}})$. Si P est réduit à une feuille, alors il existe $t \in T$ tel que $t = u$, ce qui nous permet de conclure. Sinon, on considère la dernière règle d'inférence de P. La preuve P se termine par une instance d'une règle d'inférence I_i de $\mathcal{I}$, et est donc de la forme :

$$\frac{T \vdash u_1 \quad \ldots \quad T \vdash u_k}{T \vdash u}\ (\mathsf{I}_i).$$

Par hypothèse d'induction, il existe des preuves $P'_1, \ldots, P'_k$ de $T \vdash u_1, \ldots, T \vdash u_k$ dans $(\mathcal{I}', \mathsf{E}_{\mathcal{I}})$, et nous pouvons « simuler » cette instance de I_i par :

$$\cfrac{\cfrac{T \vdash u_1 \quad \ldots \quad T \vdash u_k}{T \vdash g_i(u_1, \ldots, u_k)}\ (\mathsf{R_i})}{T \vdash u}\ (\mathsf{Eq})$$

Ceci nous permet de conclure.
($\Leftarrow$) Soient $T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$ tels que $T \vdash u$ dans $(\mathcal{I}', \mathsf{E}_{\mathcal{I}})$. Soit P un arbre de preuve de $T \vdash u$ dans $(\mathcal{I}', \mathsf{E}_{\mathcal{I}})$. Nous allons définir une transformation ϕ_1 sur les termes et nous montrerons comment cette transformation permet d'obtenir un arbre de preuve dans $\mathcal{I}$ à partir d'un arbre de preuve dans $(\mathcal{I}', \mathsf{E}_{\mathcal{I}})$.

La transformation ϕ_1 est définie de la façon suivante (t_0 représente un terme dans la connaissance de l'intrus, c'est à dire $t_0 \in T$) :

$$\phi_1(f(t_1,\ldots,t_n)) = \begin{cases} f(\phi_1(t_1),\ldots,\phi_1(t_n)) & \text{si } f \notin \mathcal{G} \\ u & \text{si } f \in \mathcal{G} \text{ et} \\ & f(\phi_1(t_1),\ldots,\phi_1(t_n)) \xrightarrow{\Lambda}_{\mathsf{E}_\mathcal{I}} u \\ t_0 & \text{sinon} \end{cases}$$

Nous construisons à partir d'une preuve P dans ($\mathcal{I}'$,$\mathsf{E}_\mathcal{I}$) une preuve P' de $T \vdash u$ dans $\mathcal{I}$ en procédant de la façon suivante :

- on applique ϕ_1 sur tous les termes de P,
- on supprime les instances des règles (Eq),
- on élague l'arbre ainsi obtenu en remplaçant les sous-arbres étiquetés $T \vdash t_0$ par une feuille de même étiquette.

Montrons que l'arbre P' ainsi obtenu est une preuve de $T \vdash u$ dans $\mathcal{I}$:

1. Les feuilles de P' sont étiquetées par $\phi_1(T) \vdash \phi_1(v)$ avec $v \in T$, d'où $\phi_1(v) \in \phi_1(T)$.
2. La racine de P' est étiquetée par $T \vdash u$ puisque $\phi_1(T) = T$ et $\phi_1(u) = u$.
3. Regardons comment ϕ_1 transforme les instances des règles d'inférence de $\mathcal{I}' \cup \mathcal{I}_{\mathsf{E}_\mathcal{I}}$. Nous distinguons deux cas :
   - Si l'instance considérée est une instance de (Eq) $\in \mathcal{I}_{\mathsf{E}_\mathcal{I}}$, alors elle est de la forme :

     $$\frac{T \vdash u_1}{T \vdash u_2}\ (\mathsf{Eq})$$

     avec $u_1 =_{\mathsf{E}_\mathcal{I}} u_2$. Le lemme 3.25, énoncé et montré ci-dessous, permet d'assurer l'égalité syntaxique $\phi_1(u_1) = \phi_1(u_2)$. Nous pouvons donc supprimer cette instance.
   - Si l'instance considérée est une instance de $\mathcal{I}'$, alors elle est de la forme :

     $$\frac{T \vdash u_1 \ \ \ldots \ \ T \vdash u_k}{T \vdash g_i(u_1,\ldots,u_k)}\ (\mathsf{R}_i).$$

   Nous allons montrer que :
   - soit $\phi_1(g_i(u_1,\ldots,u_k)) = t_0$,
   - soit $\dfrac{T \vdash \phi_1(u_1) \ \ \ldots \ \ T \vdash \phi_1(u_k)}{T \vdash \phi_1(g_i(u_1,\ldots,u_k))}$ (R_i) est une instance d'une règle d'inférence de $\mathcal{I}$.

   Supposons que $\phi_1(g_i(u_1,\ldots,u_k)) \neq t_0$. Cela signifie qu'il existe un terme u tel que $g_i(\phi_1(u_1),\ldots,\phi_1(u_k)) \xrightarrow{\Lambda}_{\mathsf{E}_\mathcal{I}} u$. L'étape $\rightarrow_{\mathsf{E}_\mathcal{I}}$ a lieu en-tête avec la règle de réécriture $g_i(w_1,\ldots,w_k) \rightarrow w$. Il existe donc une substitution σ tel que $w\sigma = u$ et pour tout j compris entre 1 et k, $w_j\sigma = \phi_1(u_j)$. L'inférence proposée est bien une instance de la règle d'inférence $\mathsf{I}_i \in \mathcal{I}$. □

Le lemme ci-dessous est utilisé dans la preuve de la proposition 3.24.

Lemme 3.25
Soient $u, v \in \mathcal{T}(\mathcal{F} \cup \mathcal{G})$.

$$u \ =_{\mathsf{E}_\mathcal{I}} \ v \implies \phi_1(u) = \phi_1(v).$$

Démonstration.
Ce résultat se montre par induction sur la longueur de la dérivation. Il nous suffit en fait de montrer qu'il est vrai pour les dérivations de longueur 1, c'est-à-dire que :

$$u \leftrightarrow_{\mathsf{E}_\mathcal{I}} v \implies \phi_1(u) = \phi_1(v).$$

Ce résultat se montre par induction sur la position p où a lieu l'étape de réécriture. Soit $l = r \in \mathsf{E}_\mathcal{I}$, l'équation utilisée pour réaliser cette étape de réécriture.

Cas de base $p = \Lambda$: Supposons (sans perte de généralité) que $l = g(l_1, \ldots, l_k)$ avec $g \in \mathcal{G}$. Il existe une substitution σ telle que $u = g(l_1, \ldots, l_k)\sigma$ et $v = r\sigma$ (resp. $v = g(l_1, \ldots, l_k)\sigma$ et $u = r\sigma$). Nous avons $\phi_1(v) = r\phi_1(\sigma)$ (resp. $\phi_1(u) = r\phi_1(\sigma)$) et $\phi_1(l_i\sigma) = l_i\phi_1(\sigma)$. D'où $g(\phi_1(l_1\sigma), \ldots, \phi_1(l_k\sigma)) \xrightarrow{\Lambda} r\phi_1(\sigma)$. Donc $\phi_1(u) = r\phi_1(\sigma)$ (resp. $\phi_1(v) = r\phi_1(\sigma)$). Dans les deux cas, nous avons $\phi_1(u) = \phi_1(v)$.

Étape d'induction $p = i.p'$:
Nous avons $u = f(u_1, \ldots, u_p)$ et $v = f(v_1, \ldots, v_p)$ avec $f \in \mathcal{F} \cup \mathcal{G}$. En appliquant l'hypothèse d'induction, nous obtenons $\phi_1(u_i) = \phi_1(v_i)$. De plus, pour tout $j \neq i$, nous avons $\phi_1(u_j) = \phi_1(v_j)$. Nous distinguons 2 cas :
- $f \notin \mathcal{G}$: $\phi_1(u) = f(\phi_1(u_1), \ldots, \phi_1(u_p)) = f(\phi_1(v_1), \ldots, \phi_1(v_p)) = \phi_1(v)$.
- $f \in \mathcal{G}$: Tout d'abord si $\phi_1(u) = t_0$ et $\phi_1(v) = t_0$, alors on a $\phi_1(u) = \phi_1(v)$. Supposons que $\phi_1(u)$ (ou $\phi_1(v)$) soit différent de t_0. Nous avons $f(\phi_1(u_1), \ldots, \phi_1(u_p)) = f(\phi_1(v_1), \ldots, \phi_1(v_p))$. De plus, $f(\phi_1(u_1), \ldots, \phi_1(u_p)) \xrightarrow{\Lambda}_{\mathsf{E}_\mathcal{I}} u'$. On en déduit donc que $f(\phi_1(v_1), \ldots, \phi_1(v_p)) \xrightarrow{\Lambda}_{\mathsf{E}_\mathcal{I}} u'$. Et donc que $\phi_1(u) = \phi_1(v)$ $(= u')$. □

Remarque : Cette traduction générique appliquée à $\mathcal{I}_{\mathsf{DY}}$ (modèle standard de Dolev-Yao) permet d'obtenir la théorie équationnelle E_{DY}. Nous obtenons ainsi le modèle d'intrus avec destructeurs explicites que nous avons présenté au début de la partie 3.1.2.2.

En revanche, cette traduction ne permet pas de transformer un système d'inférence $\mathcal{I}$ avec filtrage lorsque celui-ci est accompagné d'une théorie équationnelle E. En effet, les axiomes $\mathsf{E}_\mathcal{I}$ que l'on ajoute lors de la traduction peuvent interagir avec la théorie équationnelle E de départ et être à l'origine d'effets indésirables.

Exemple 3.26
Considérons le modèle d'intrus ($\mathcal{I}$,E) suivant :

$$\mathcal{I} : \quad \frac{T \vdash \langle x, y\rangle}{T \vdash x} \qquad \mathsf{E} : \quad \langle x, x\rangle = \langle \{x\}_x, \{x\}_x\rangle.$$

Avec la traduction proposée précédemment, on obtiendrait le modèle d'intrus ($\mathcal{I}'$, $\mathsf{E} \cup \mathsf{E}_\mathcal{I}$) :

$$\mathcal{I}' : \quad \frac{T \vdash x}{T \vdash g_1(x)} \qquad \begin{array}{lrcl} \mathsf{E} : & \langle x, x\rangle & = & \langle \{x\}_x, \{x\}_x\rangle \\ \mathsf{E}_\mathcal{I} : & g_1(\langle x, y\rangle) & = & x \end{array}$$

Ces deux modèles d'intrus ne sont pas équivalents. En effet, $\{a\}_a$ n'est pas déductible à partir de a dans ($\mathcal{I}$, E) alors qu'il l'est dans ($\mathcal{I}'$, $\mathsf{E} \cup \mathsf{E}_\mathcal{I}$) en utilisant uniquement le raisonnement équationnel modulo $\mathsf{E} \cup \mathsf{E}_\mathcal{I}$:

$$a =_{\mathsf{E}_\mathcal{I}} g_1(\langle a, a\rangle) =_{\mathsf{E}} g_1(\langle \{a\}_a, \{a\}_a\rangle) =_{\mathsf{E}_\mathcal{I}} \{a\}_a.$$

3.3.2 Modèle par rôles avec filtrage et modèle avec tests d'égalités

L'utilisation des destructeurs explicites et des équations permet de modéliser les pas d'un protocole d'une façon beaucoup plus naturelle. En particulier pour les protocoles nécessitant de décomposer des messages *a posteriori*, c'est-à-dire bien après les avoir reçus. En effet, il se peut qu'un agent reçoive un message qu'il ne soit pas capable d'analyser à un moment donné (sa connaissance ne lui permettant pas), mais qu'il puisse plus tard effectuer cette analyse. Le modèle par filtrage permet de décomposer un message au moment de sa réception, mais ne permet pas d'effectuer cette décomposition plus tard.

Considérons le protocole décrit ci-dessous (*cf.* [Tur03]). Supposons que l'agent jouant le rôle de B ne connaisse pas la clef K initialement.

$$\begin{array}{lcll} A & \rightarrow & B: & \{M,B\}_K \\ B & \rightarrow & A: & B \\ A & \rightarrow & B: & K \\ B & \rightarrow & A: & M \end{array}$$

Le message $\{M,B\}_K$ est vu comme une variable par l'agent jouant le rôle B au premier pas du protocole. Par la suite, il reçoit la clef K. Il est alors en mesure de récupérer le message M se trouvant dans le message chiffré qu'il a reçu à la première étape du protocole. Mais le modèle par rôles avec filtrage autorise le filtrage uniquement au moment de la réception d'un message. Le rôle B nécessite la décomposition d'un message *a posteriori*, et ne semble pas pouvoir s'exprimer, d'une façon naturelle, dans le modèle par rôles avec filtrage. Si l'on suppose que le rôle s'exécute jusqu'au bout, une modélisation (non-naturelle) (*cf.* exemple 3.28) consiste a effectuer la décomposition en amont, lors de la réception du premier message. Cette modélisation n'est pas naturelle puisqu'elle laisse croire que l'agent jouant le rôle de B est capable de récupérer le message M dès la première étape du protocole.

En revanche, dans le modèle par rôles avec tests d'égalité, la modélisation est beaucoup plus aisée.

Exemple 3.27
Modélisation du rôle B dans le modèle par rôle avec tests d'égalités.

$$R(z_b) = \left\{ \begin{array}{c} \mathit{recv}(x);\ \mathit{send}(z_b) \\ \mathit{recv}(y);\ \mathsf{proj}_2(\mathsf{dec}(x,y)) = z_b;\ \mathit{send}(\mathsf{proj}_1(\mathsf{dec}(x,y))) \end{array} \right.$$

Nous avons introduit deux approches permettant de représenter un rôle. Il est naturel de comparer l'expressivité de ces deux approches. Nous venons de voir que le modèle par rôle avec tests d'égalités permet une modélisation plus naturelle de certains rôles. Cela ne signifie pas que ce modèle est plus expressif. En particulier, si la théorie équationnelle E considérée est unitaire (c'est-à-dire que tout problème d'unification admet un unique unificateur le plus général), il est relativement facile de traduire un rôle écrit dans le modèle avec tests d'égalités dans le modèle avec filtrage. Il suffit pour cela de résoudre le problème d'unification formé par les différentes équations obtenues après instanciation des différents paramètres du rôle. On obtient une substitution θ que l'on peut appliquer à chacun des termes du rôle. Les différents tests codés par les équations sont alors effectués en amont dès la réception du message, ce qui ne change pas fondamentalement le protocole si on assure que le rôle s'exécutera jusqu'au bout. De plus, le rôle ainsi obtenu à partir d'un rôle avec tests d'égalités bien formé est également un rôle bien formé : c'est une conséquence directe du fait que la définition d'un rôle bien formé est stable par substitution.

Exemple 3.28
Considérons le rôle décrit dans l'exemple 3.27. Le problème d'unification que l'on doit considérer est $\mathsf{proj}_2(\mathsf{dec}(x,y)) = p_b$. *Ce problème admet un unique mgu :* $\theta = \{x \mapsto \langle \{z\}_y, p_b\rangle\}$. *Dans le modèle par rôles avec filtrage, le rôle* B *s'écrit donc de la façon suivante :*

$$R(z_b) = \left\{ \begin{array}{c} \mathit{recv}(\langle \{z\}_y, z_b\rangle);\ \mathit{send}(z_b) \\ \mathit{recv}(y);\ \mathit{send}(z) \end{array} \right.$$

Maintenant, on peut se demander si le modèle par rôle avec filtrage n'est pas plus expressif. Ce modèle ne permettrait-il pas d'exprimer des pas de protocoles que l'on ne peut pas exprimer dans le modèle par rôles avec tests d'égalités ? Si l'on ne se restreint pas à la classe des rôles bien formés, la réponse est non. Il suffit en effet de remplacer chacune des instructions $\mathsf{recv}(u)$ par l'instruction $\mathsf{recv}(x)$ (où x est une nouvelle variable) suivi de l'équation $u = x$. Le problème est qu'une telle transformation ne préserve pas la bonne formation du rôle (*cf.* exemple 3.29).

Exemple 3.29
Considérons le rôle $\mathit{recv}(\langle x_1, x_2\rangle);\ \mathit{send}(x_1)$ *dans le modèle avec filtrage. Ce rôle est bien formé. Dans le modèle avec tests d'égalités, ce rôle « se traduit » en* $\mathit{recv}(x);\ \langle x_1, x_2\rangle = x;\ \mathit{send}(x_1)$. *Ce rôle n'est pas bien formé puisque la variable* x_1 *n'apparaît pas dans une instruction* $\mathit{recv}(.)$.

Maintenant, si l'on souhaite se restreindre à la classe des rôles bien formés, la réponse semble alors être oui.

Exemple 3.30
Considérons le rôle $\mathit{recv}(\{x_1\}_{x_2});\ \mathit{send}(x_2)$ *et la théorie équationnelle composée de l'unique équation* $\mathsf{dec}(\{x\}_y, y) = x$. *Les symboles* $\mathsf{dec}(.,.)$ *et* $\{.\}_.$ *sont publics.*

Ce rôle, bien que non réaliste, est un rôle bien formé. Ce rôle est même déterministe. Si l'on souhaite traduire ce rôle en un rôle bien formé dans le modèle avec tests d'égalités, il semble naturel d'introduire un nouveau destructeur, d^2_{dec}, *et l'axiome* $d^2_{\mathsf{dec}}(\mathsf{dec}(x,y)) = y$, *dans la théorie équationnelle. Ce nouveau symbole privé nous permet de « traduire » le pas de protocole* $\mathit{recv}(\{x_1\}_{x_2});\ \mathit{send}(x_2)$ *en le pas de protocole suivant :*

$$\mathit{recv}(y);\ \mathit{send}(d^2_{\mathsf{dec}}(y)).$$

Malheureusement, l'ajout de l'axiome $d^2_{\mathsf{dec}}(\mathsf{dec}(x,y)) = y$ *dans la théorie équationnelle de départ* $\mathsf{dec}(\{x\}_y, y) = x$, *confère un pouvoir « tout-puissant » à l'intrus. En effet, la nouvelle théorie équationnelle, obtenue à partir de cet exemple, rend tous les termes égaux.*

$$a = d^2_{\mathsf{dec}}(\mathsf{dec}(\{c\}_a, a)) = d^2_{\mathsf{dec}}(c) = d^2_{\mathsf{dec}}(\mathsf{dec}(\{c\}_b, b)) = b.$$

Cette traduction, assez naturelle, consistant à ajouter des destructeurs privés pour atteindre les différentes composantes d'un message, ne semble donc pas marcher. La situation s'avère même plus délicate en présence d'opérateurs AC. En effet, le modèle par rôle avec filtrage semble être très expressif : de nombreux rôles bien formés ne semblent pas pouvoir se traduire facilement dans le modèle avec destructeurs explicites. Considérons par exemple la théorie AG et les deux rôles suivants :

$$\mathsf{recv}(x+x); \mathsf{send}(x), \qquad \text{et} \qquad \mathsf{recv}(x+x+x); \mathsf{send}(x).$$

Afin de pouvoir traduire de tels rôles, nous allons devoir ajouter un nombre infini d'axiomes à notre théorie équationnelle (*e.g.* $div_2(x+x) = x$, $div_3(x+x+x) = x$, ...)

Enfin, considérons le rôle $\mathsf{recv}(\{x\}_k + x); \mathsf{send}(x)$. Ce rôle est un rôle bien formé, pour lequel une traduction en un rôle bien formé dans le modèle par rôles avec tests d'égalité ne semble pas évidente (à moins d'ajouter un destructeur d dédié à ce calcul, c'est-à-dire vérifiant l'égalité $\mathsf{d}(\{x\}_k + x) = x$).

À la vue de tous ces exemples, il semblerait que le modèle par rôles avec filtrage soit plus expressif que le modèle par rôles avec tests d'égalités. Le principe du modèle par filtrage est de décrire précisément (de façon non ambiguë) les actions des agents, sans entrer dans le détail des opérations effectuées. Les agents exécutant un rôle ont alors un pouvoir de déduction très puissant qui leur est donné par l'utilisation de l'algorithme de filtrage. Ils sont capables de récupérer les valeurs associées aux variables du filtre. La difficulté est de « coder » ce mécanisme de filtrage dans le modèle par rôles avec tests d'égalités, et cela en conservant des rôles bien formés.

Chapitre 4

Problèmes de vérification dans les deux formalismes

Sommaire

4.1 Intrus passif **56**
4.1.1 Système d'inférence 57
4.1.2 Localité 58
4.1.3 Déductibilité en un pas 59
4.2 Intrus actif **60**
4.2.1 Systèmes de contraintes symboliques 62
4.2.2 Construction du système de contraintes 63
4.2.2.1 Algorithme 63
4.2.2.2 Exemples 65
4.2.3 Systèmes de contraintes bien formés 69

Comme mentionné en introduction, il existe de nombreuses propriétés de sécurité qu'un protocole doit garantir. Parmi toutes ces propriétés, nous allons nous concentrer sur les propriétés dite *de trace*. Il s'agit essentiellement de la propriété de secret et de certaines formes d'authentification. En effet, la propriété de secret n'est pas satisfaite dès lors qu'il existe une séquence de messages (c'est-à-dire une trace) se terminant par l'émission du secret. De même, certaines formes d'authentification peuvent s'exprimer comme une séquence de messages. Par exemple, une propriété d'authentification n'est pas satisfaite si un agent honnête est convaincu de parler avec un autre agent alors que ce dernier n'a jamais participé au protocole. D'autre part, nous nous plaçons dans le cadre d'un nombre borné de sessions. Autrement dit, nous considérons un nombre borné d'instances de rôles s'exécutant en parallèle.

Le problème est donc, étant donné un nombre fini d'instances de rôles, de déterminer si une donnée particulière reste secrète ou non. Dans le cas d'un intrus passif, cela revient à se demander s'il existe une exécution des différentes instances des rôles considérées à l'issue de laquelle le secret est déductible par l'intrus à partir de sa connaissance initiale et des messages qu'il a récupérés sur le réseau au cours de l'exécution du protocole (*cf.* partie 4.1). En présence d'un intrus actif, le problème est un peu différent. L'intrus peut émettre des messages sur le réseau et interagir avec les agents exécutant les différents rôles. Dans ce cas, savoir si un protocole est sûr revient à se demander si une séquence particulière de messages représentant une attaque est accessible. Nous verrons que ce problème revient en fait à résoudre un système de contraintes symboliques (*cf.* partie 4.2).

4.1 Intrus passif

Dans cette partie, nous nous intéressons au problème de la vérification pour un nombre borné de sessions en présence d'un intrus passif. Ce problème se résout en deux phases. Dans un premier temps, on considère les différentes exécutions possibles des instances des rôles considérés. L'intrus n'intervient pas dans cette première phase, il se contente d'écouter les messages circulant sur le réseau. Si le protocole étudié respecte quelques règles simples de bonne formation, les agents exécutant les différents rôles ne peuvent pas réceptionner un message qui ne leur est pas destiné, ou confondre des messages provenant de sessions différentes. Quelque soit l'ordre d'exécution, les messages qui ont été émis sur le réseau sont alors les mêmes. Il reste ensuite à se demander si le secret est déductible par l'intrus à partir de sa connaissance initiale et de la connaissance qu'il a acquise au cours de l'exécution du protocole. Étant donné un système d'inférence $\mathcal{I}$ représentant le pouvoir de déduction de l'intrus, ce problème, appelé *problème de déduction de l'intrus*, s'énonce formellement de la façon suivante :

Problème de déduction de l'intrus

Entrées :
- un ensemble fini T de termes clos (la connaissance de l'intrus),
- un terme clos s (le secret).

Sortie : Est-ce que s est déductible de T dans $\mathcal{I}$? Autrement dit, est-ce qu'il existe une preuve de $T \vdash s$ dans $\mathcal{I}$?

Outre le fait de permettre la résolution du problème de la sécurité d'un protocole dans le cas d'un intrus passif, le problème de déduction de l'intrus correspond à la résolution d'une contrainte de déduction close (sans variable) et constitue généralement (et c'est le cas des algorithmes proposés dans cette thèse) une étape importante pour la résolution du problème de vérification en présence d'un intrus actif.

L'objectif de cette partie est d'introduire la technique de preuve utilisée, en règle générale, pour résoudre le problème de déduction de l'intrus. La plupart des résultats existants (*cf.* chapitre 2) sont en fait obtenus en transformant le système d'inférence de départ en un système d'inférence équivalent ayant les deux propriétés suivantes :

1. La propriété de *localité* pour une « bonne notion » de sous-termes.
   Cette propriété a pour but d'assurer que si u est déductible à partir de T, alors il existe une preuve dont les nœuds sont étiquetés par des termes appartenant à un ensemble (si possible petit) calculable à partir de T et de u. Cette propriété permet de réduire l'espace de recherche d'une preuve de $T \vdash u$.
2. La propriété de *déductibilité en un pas*.
   Elle permet de tester si un terme est déductible en une étape à partir d'un ensemble de termes T donné, dans le système d'inférence considéré. Cette propriété est satisfaite par les règles ayant un nombre fixé de prémisses (*e.g.* déchiffrement (D)) dès que l'on sait décider si une inférence est une instance d'une règle d'inférence donnée. Cependant,

nous verrons que la transformation que l'on fait subir au système d'inférence de départ est à l'origine de schémas de règles pour lesquels la propriété de déductibilité en un pas n'est pas toujours triviale à établir.

4.1.1 Système d'inférence

Le nouveau système, équivalent au système de départ $(\mathcal{I}, \mathsf{E})$, s'obtient en général en supprimant la règle d'inférence (Eq) permettant le raisonnement équationnel modulo E et en travaillant sur les termes normalisés par rapport à $\mathcal{R}_{\mathsf{E}}$ (système convergent représentant E). Notons $(\mathcal{I}, \mathcal{R}_{\mathsf{E}})$ le système d'inférence ainsi obtenu. Comme illustré par l'exemple 4.1, cette transformation ne permet pas, en général, d'obtenir un système d'inférence équivalent. C'est néanmoins ce type de transformation que l'on utilisera aux chapitres 5 et 6 pour transformer notre système de départ. Nous établirons alors le résultat d'équivalence entre les systèmes $(\mathcal{I}, \mathsf{E})$ et $(\mathcal{I}, \mathcal{R}_{\mathsf{E}})$.

Exemple 4.1
Considérons la théorie équationnelle $\mathsf{E} = \{\langle \{x_1\}_y, \{x_2\}_y \rangle = \{\langle x_1, x_2 \rangle\}_y\}$. *En orientant de gauche à droite cette équation, on obtient un système convergent. Considérons la règle* (Eq) *et le système d'inférence* $\mathcal{I}$ *constitué des deux règles d'inférence suivantes :*

$$\frac{T \vdash \langle x_1, x_2 \rangle}{T \vdash x_1}\,(\mathsf{Proj}_1) \qquad\qquad \frac{T \vdash \langle x_1, x_2 \rangle}{T \vdash x_2}\,(\mathsf{Proj}_2)$$

Le système d'inférence $(\mathcal{I}, \mathcal{R}_{\mathsf{E}})$ *n'est pas équivalent au système* $(\mathcal{I}, \mathsf{E})$. *En effet,* $\{a\}_k$ *est déductible à partir de* $\{\langle a, b \rangle\}_k$ *dans* $\mathcal{I} \cup \mathcal{I}_{\mathsf{E}}$. *On a :*

$$\frac{\dfrac{T \vdash \{\langle a, b \rangle\}_k}{T \vdash \langle \{a\}_k, \{b\}_k \rangle}\,(\mathsf{Eq})}{T \vdash \{a\}_k}\,(\mathsf{Proj}_1)$$

En revanche, il n'est pas possible de faire une preuve de $\{\langle a, b \rangle\}_k \vdash \{a\}_k$ *dans* $(\mathcal{I}, \mathcal{R}_{\mathsf{E}})$ *: si l'on se place dans l'algèbre libre, le terme* $\{\langle a, b \rangle\}_k$ *ne filtre pas le motif* $\langle x_1, x_2 \rangle$.

D'autre part, afin d'obtenir une notion de localité pour une « bonne » notion de sous-termes, on introduit en général des schémas de règles. Un schéma de règles est une méta-règle représentant un ensemble infini de règles. Ils permettent de faire en une étape, ce qui nécessitait plusieurs étapes dans l'ancien système de déduction. Ainsi, nous pouvons considérer une notion de sous-termes « plus petite » et obtenir de meilleures résultats de complexité. Mais attention, l'introduction de schémas de règles n'est intéressante que si l'on sait décider efficacement la déductibilité en un pas sur ces schémas. Cette technique consistant à introduire des schémas est utilisée en particulier en présence d'opérateurs AC. Par exemple, dans le cas du « ou » exclusif, on remplace la règle binaire

$$\frac{T \vdash x_1 \quad T \vdash x_2}{T \vdash x_1 \oplus x_2}$$

par le schéma $(\mathsf{M}_\oplus)$ suivant :

$$\frac{T \vdash x_1 \;\; \ldots \;\; T \vdash x_n}{T \vdash x_1 \oplus \ldots \oplus x_n}\,(\mathsf{M}_\oplus)$$

Alors qu'il existe une preuve de $a_1, a_2, a_2 \vdash a_1 \oplus a_2 \oplus a_3$ en une étape en utilisant le schéma $(\mathsf{M}_\oplus)$, l'utilisation de la règle binaire nous obligerait à faire une preuve en deux étapes et à passer par un terme intermédiaire (*e.g.* $a_1 \oplus a_2$). Si l'on souhaite obtenir un résultat de localité avec la règle binaire, nous sommes obligés de considérer une notion de sous-termes faisant intervenir les sommes partielles. Une telle notion de sous-termes est « mauvaise » : un terme t a un nombre exponentiel de tel sous-termes. L'introduction du schéma proposé ci-dessus permet de ne pas tenir compte des sommes partielles et d'obtenir une notion de sous-termes polynomiale en la taille de t. L'introduction de schémas facilite l'obtention de la propriété de localité, mais le problème reste entier tant que nous n'avons pas de procédure efficace permettant de décider le problème de la déductibilité en un pas pour chacun des schémas introduits.

4.1.2 Localité

La notion de localité a été introduite par D. McAllester [McA93] pour caractériser les systèmes de déduction finis pour lesquels il existe un algorithme polynomial permettant de décider si un terme u est déductible d'un ensemble fini T de termes. Un système local permet d'assurer (lorsque u est déductible de T) l'existence d'une preuve *locale*, c'est-à-dire une preuve dont tous les termes intermédiaires sont des sous-termes syntaxiques de T ou de u. Nous généralisons cette notion de localité à différentes notions de sous-termes afin de pouvoir traiter les opérateurs AC.

Définition 4.2 (notion de sous-termes)
Une notion de sous-termes *est une fonction $\mathcal{ST} : \mathcal{T}(\mathcal{F}, \mathcal{X}) \to \mathcal{P}(\mathcal{T}(\mathcal{F}, \mathcal{X}))$ associant à un terme t un ensemble fini de termes, appelé* sous-termes de t.

Exemple 4.3
La notion de sous-termes syntaxiques, notée St, introduite au chapitre 2, est une notion de sous-termes.

Notation : Soit S une notion de sous-termes. Nous notons $\mathcal{S}(T)$ l'ensemble $\bigcup_{t \in T} \mathcal{S}(t)$, et $\mathcal{S}(T, u)$ l'ensemble $\mathcal{S}(T \cup \{u\})$.

Définition 4.4 (Système local)
Soit $\mathcal{I}$ un système d'inférence. Le système $\mathcal{I}$ est dit local par rapport à une notion de sous-termes S *si pour tout $T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$ tel que $T \vdash u$ dans $\mathcal{I}$, il existe une preuve P de $T \vdash u$ dans $\mathcal{I}$ dont les nœuds sont étiquetés par des expressions de la forme « $T \vdash v$ » avec $v \in S(T, u)$.*

Dans le cas d'un système d'inférence fini, la notion de localité suffit à obtenir un algorithme permettant de résoudre le problème de déduction de l'intrus. En effet, l'espace de recherche d'une preuve P de $T \vdash u$ est borné :
- la longueur d'un chemin de la racine à une feuille est bornée par le nombre de sous-termes dans $\mathcal{S}(T, u)$,
- le branchement de l'arbre P est borné.

En présence de schémas de règles, ce raisonnement n'est plus possible : le branchement de l'arbre n'est pas borné. L'idée est alors de résoudre le problème de déduction de l'intrus par un algorithme de saturation (*cf.* Algorithme 4.1).

```
Entrées: T, u
Sortie: oui /non

Algorithme:
  S' := S(T,u)
  T' := ∅
  tant que T ≠ T' faire
       T' := T
       pour tout t ∈ S'  faire
       si t est déductible en un pas de T' alors T := T ∪ {t}
       fin pour tout
  fin tant que

  si u ∈ T alors retourner oui
             sinon retourner non
```

Algorithme 4.1 - Problème de déduction de l'intrus.

L'idée de cet algorithme est de saturer la connaissance de l'intrus en ajoutant parmi les termes de $\mathcal{S}(T,u)$ ceux qui sont déductibles en un pas à partir de sa connaissance actuelle. Après au plus $|\mathcal{S}(T,u)|$ itérations de la boucle « tant que », cet algorithme s'arrête. Il suffit alors de regarder si le terme u est dans cet ensemble.
Cet algorithme est polynomial si :
- la notion de sous-termes $\mathcal{S}$ est telle que $|\mathcal{S}(T,u)|$ est polynomial en la taille de T et de u, et
- si le problème de la déductibilité en un pas est décidable en temps polynomial.

4.1.3 Déductibilité en un pas

Soit $\mathcal{I}$ un système d'inférence. Le problème de la déductibilité en un pas s'énonce formellement de la façon suivante :

Problème de déductibilité en un pas

Entrées :
- un ensemble fini T de termes clos,
- un terme clos u.

Sortie : Est-ce que u est déductible de T en (au plus) un pas ? Autrement dit, a-t-on $u \in T$ ou est-ce qu'il existe $u_1, \ldots, u_n$ tels que $\dfrac{T \vdash u_1 \quad \ldots \quad T \vdash u_n}{T \vdash u}$ est un arbre de preuve de $T \vdash u$ dans $\mathcal{I}$?

Notation : On note $T \vdash_1 u$ (resp. $T \not\vdash_1 u$) si u est déductible (resp. n'est pas déductible) en un pas à partir de T.

Comme nous l'avons déjà fait remarquer, ce problème est en fait trivial pour les règles d'inférence « classique ». Dans la suite, nous nous intéresserons à ce problème uniquement pour les schémas de règle.

Exemple 4.5
Soient $T = \{a_1 \oplus a_2,\ a_1 \oplus a_3 \oplus a_4,\ a_2 \oplus a_4\}$ *et* $\mathsf{M}_\oplus$ *le schéma suivant :*

$$\frac{T \vdash x_1 \qquad \ldots \qquad T \vdash x_n}{T \vdash x_1 \oplus \ldots \oplus x_n}\ (\mathsf{M}_\oplus)$$

Le terme a_3 *est déductible en un pas à partir de* T*, autrement dit on a* $T \vdash_1 a_3$*. En effet :*

$$\frac{T \vdash a_1 \oplus a_3 \oplus a_4 \quad T \vdash a_2 \oplus a_4 \quad T \vdash a_1 \oplus a_2}{T \vdash a_3}\ (\mathsf{M}_\oplus)$$

est un arbre de preuve de $T \vdash a_3$*.*

En fait, le problème de la déductibilité en un pas, pour le schéma $(\mathsf{M}_\oplus)$ (schéma introduit pour traiter la théorie du « ou » exclusif) a été résolu par Y. Chevalier *et al.* [CKRT03b]. Le problème de la déductibilité en un pas se réduit au problème de la satisfaisabilité d'un système d'équations linéaires dans $\mathbb{Z}/2\mathbb{Z}$. La résolution de tels systèmes d'équations est connue pour être décidable en temps polynomiale [Sch86]. Nous utiliserons, au chapitre 6, une technique similaire pour résoudre ce problème de la déductibilité en un pas dans le cas de théories plus complexes telles que ACh, ACUNh et AGh.

4.2 Intrus actif

Le problème de la vérification en présence d'un intrus actif (pour un nombre borné de sessions) consiste à générer l'ensemble fini des séquences d'instructions décrivant des attaques potentielles, et à regarder si, parmi ces *traces symboliques*, il en existe une *accessible*. Les messages d'une trace symbolique peuvent contenir des variables. Une variable représente un message dont la valeur est initialement inconnue du récepteur. L'intrus est ainsi libre d'instancier cette variable comme il l'entend, en restant cependant cohérent lors des prochaines instanciations.

Définition 4.6 (trace symbolique)
Une trace symbolique *est une séquence d'instructions* $\mathsf{instr}_1; \ldots; \mathsf{instr}_\ell$ *dans laquelle chacune des instruction* instr_i *est :*
- *soit une instruction de la forme « recv*(u) *» ou « send*(u) *», avec* $u \in \mathcal{T}(\mathcal{F}, \mathcal{X})$*,*
- *ou alors, une équations de la forme «* $u = v$ *», avec* $u, v \in \mathcal{T}(\mathcal{F}, \mathcal{X})$*.*

Dans la suite, nous serons amenés à considérer des traces ayant quelques particularités. Ces particularités sont dues au fait que les traces que nous allons considérer sont construites à partir de séquences d'instructions provenant d'instances de rôles.

Définition 4.7 (trace sans équation, trace simple)
Une trace sans équation *est une trace dans laquelle toutes les instructions sont de la forme « recv*(u) *» ou « send*(u) *». Une trace* $\mathsf{instr}_1; \ldots; \mathsf{instr}_\ell$ *est dite* simple *si toutes les instructions de la forme « recv*$(.)$ *» sont de la forme « recv*(x) *» avec* $x \in \mathcal{X}$*.*

Le principal problème consiste à décider si une trace symbolique donnée est accessible, c'est-à-dire s'il existe une instanciation de cette trace pour laquelle l'intrus est capable de construire, à partir de sa connaissance et des messages précédemment émis sur le réseau, les messages que les différents agents s'attendent à recevoir.

Définition 4.8 (trace accessible)
Une trace $\mathsf{instr}_1; \ldots, \mathsf{instr}_\ell$ *est* accessible à partir de T_0 dans $(\mathcal{I}, \mathsf{E})$ *s'il existe une substitution* σ *telle que pour tout* i $(1 \leq i \leq \ell)$*, on a :*

- *si* instr_i *est de la forme « $u = v$ » alors* $u\sigma =_{\mathsf{E}} v\sigma$*,*
- *si* instr_i *est de la forme « $\mathsf{recv}(u)$ » alors :*

$$u\sigma \text{ est déductible de } T_0 \cup \bigcup_{j=1}^{i-1} \{v\sigma \mid \mathsf{instr}_j \text{ est de la forme } \mathsf{send}(v)\} \text{ dans } (\mathcal{I}, \mathsf{E}).$$

Nous souhaitons pouvoir décider si une trace est accessible ou non. Nous nous limiterons à l'étude des traces sans équations et des traces simples.

Exemple 4.9
Considérons l'intrus standard $\mathcal{I}_{\mathsf{DY}}$*, et la trace sans équation suivante :*

$$T := \mathsf{recv}(\{x_1\}_{k_1}); \mathsf{send}(x_1); \mathsf{recv}(\{x_2\}_{k_2}); \mathsf{send}(k_s); \mathsf{recv}(s)$$

La trace T *est accessible à partir de* $T_0 = \{\{k_2\}_{k_1},\ m,\ \{s\}_{k_s}\}$ *dans* $\mathcal{I}_{\mathsf{DY}}$*. En effet, la substitution* $\sigma = \{x_1 \mapsto k_2, x_2 \mapsto m\}$ *convient : on a* $T_0 \vdash \{k_2\}_{k_1}$ *;* $T_0, k_2 \vdash \{m\}_{k_2}$ *et* $T_0, k_2, k_s \vdash s$ *dans* $\mathcal{I}_{\mathsf{DY}}$*.*

Les propriétés de sécurité dont la violation correspond à l'existence d'une trace symbolique accessible sont appelées *propriétés de trace*. Parmi ces propriétés, on trouve entre autres la propriété de secret et certaines formes d'authentification.

En effet, la propriété de secret n'est pas satisfaite dès lors qu'il existe une trace symbolique, correspondant à un ordonnancement des différentes instructions composants les instances des rôles considérés, se terminant par l'émission du secret. Ce secret peut être une donnée atomique (*i.e.* une constante) ou un terme composé. D'autre part, il est également possible que cette donnée soit un terme non clos, c'est-à-dire un terme contenant des variables apparaissant ou non dans la trace symbolique. Cette souplesse permet d'énoncer des propriétés de secret aussi bien sur des données émises que sur des données reçues, ou même de se demander si l'intrus est en mesure, à l'issue de l'exécution des différents rôles, de construire un message respectant certaines conditions.

Certaines formes d'authentification sont relativement aisées à modéliser. Si l'on considère la forme d'authentification la plus faible, celle où l'on se contente de vérifier que l'initiateur du protocole ne peut pas avoir achevé l'exécution de son rôle avec un agent b si ce dernier n'a exécuté aucune instruction auparavant, il suffit de considérer les traces symboliques où les actions composant le rôle d'initiateur sont entièrement exécutées alors qu'aucune instruction provenant d'un rôle joué par l'agent b n'a été exécutée. Ensuite il faut regarder si, parmi ces traces symboliques, l'une d'entre elles est accessible, ce qui mettrait en évidence une faille sur le protocole.

Si l'on souhaite en plus garantir que l'agent b a exécuté une action d'un rôle fixé, disons B, et que son interlocuteur est bien l'agent a, il faut alors considérer les traces symboliques composées d'une exécution complète du rôle A joué par a avec l'agent b et précédée d'aucune action composant le rôle B joué par b avec a. Si aucune de ces traces symboliques n'est accessible, alors la propriété d'authentification est satisfaite.

Les formes d'authentification plus complexes exprimant le fait que les agents a et b soient d'accord, à l'issue de la session, sur un certain nombres de données échangées au cours de cette session peut se modéliser en ayant recours à des tests d'inégalités permettant d'exprimer leur désaccord. Ce genre de propriétés sort du cadre de notre étude.

Dans la partie 4.2.1, nous définissons les différents types de contraintes symboliques que nous allons utiliser. Nous montrons ensuite (*cf.* partie 4.2.2) comment générer les différentes traces symboliques ainsi que les systèmes de contraintes correspondant, dont la satisfaisabilité permet de décider si la trace considérée est accessible ou non. Suivant le formalisme utilisé pour décrire les rôles, les traces symboliques obtenues auront quelques particularités (traces sans équation, traces simples), et il en sera de même pour le système de contraintes qui leur est associé (système de contraintes de déduction, système de contraintes de déduction simple avec équations). Enfin, nous présentons, dans la partie 4.2.3, la classe des systèmes dont nous ferons l'étude dans les chapitres 5 et 6 : les systèmes de contraintes bien formés. Nous montrerons que cette propriété de bonne formation est satisfaite par tout système de contraintes généré à partir d'instances de rôles bien formés.

4.2.1 Systèmes de contraintes symboliques

Définition 4.10 (contrainte de déduction, en un pas, simple)
Une contrainte de déduction *(resp.* contrainte de déduction en un pas*) est une expression de la forme « $T \Vdash u$ » (resp. « $T \Vdash_1 u$ ») où T est un ensemble fini de termes et u un terme. Une contrainte de déduction est dite* simple *si u est une variable.*

Suivant le formalisme considéré (modèle par rôles avec filtrage ou avec égalités) nous sommes amenés à considérer deux types de systèmes de contraintes. Dans le cas du modèle par rôle avec filtrage, on obtient un *système de contraintes de déduction.*

Définition 4.11 (système de contraintes de déduction, en un pas)
Un système $\mathcal{C}$ de contraintes de déduction *(resp.* contraintes de déduction en un pas*) est un ensemble de contraintes de déduction (resp. contraintes de déduction en un pas). Étant donné un système d'inférence $(\mathcal{I}, \mathsf{E})$, une solution de $\mathcal{C}$ est une substitution σ telle que :*

- *pour tout $T \Vdash u \in \mathcal{C}$ (resp. $T \Vdash_1 u \in \mathcal{C}$) , $T\sigma \vdash u\sigma$ (resp. $T\sigma \vdash_1 u\sigma$) dans $(\mathcal{I}, \mathsf{E})$.*

Dans le cas du modèle par rôle avec tests d'égalités, on obtient un système de contraintes simple avec équations.

Définition 4.12 (système de contraintes simple avec équations)
Un système $\mathcal{B}$ de contraintes simples avec équations est composé d'un ensemble $\mathcal{C}$ de contraintes de déduction simple et un ensemble $\mathcal{E}$ d'équations. Étant donné un système d'inférence simple $\mathcal{I}$ et une théorie équationnelle E, une solution de $\mathcal{B}$ est une substitution σ telle que :

- *pour tout $T \Vdash u \in \mathcal{C}$, $T\sigma \vdash u\sigma$ dans $(\mathcal{I}, \mathsf{E})$*
- *pour tout $u = v \in \mathcal{E}$, $u\sigma =_{\mathsf{E}} v\sigma$.*

4.2.2 Construction du système de contraintes

Dans cette partie, nous montrons quelles sont les différentes traces symboliques que nous devons considérer et comment construire le système de contraintes associé à une telle trace. La construction est relativement similaire quel que soit le formalisme utilisé (modèle avec filtrage ou avec tests d'égalités). Nous allons donner une construction générique valable pour ces deux formalismes, mais compte tenu des spécificités de chacun de ces modèles, nous obtiendrons des systèmes de contraintes légèrement différents : système de contraintes de déduction dans le cas du modèle avec filtrage, et système de contraintes de déduction simple avec équations dans le cas du modèle avec tests d'égalités.

4.2.2.1 Algorithme

Nous commençons par deviner un *ordonnancement*, c'est-à-dire l'ordre dans lequel les différentes instances des rôles considérés s'exécutent. Nous supposons que ces différentes instances ne partagent pas de variables, et que les constantes utilisées pour représenter les nonces générés par les agents exécutant les différents rôles sont tous distincts.

Définition 4.13 (ordonnancement)
Soient $R_1, \ldots, R_m$ un ensemble fini d'instances de rôles où chaque R_i est une séquence d'instructions de la forme : recv(u_1^i); $\mathcal{E}_1^i$; send(v_1^i); $\ldots$; recv$(u_{k_i}^i)$; $\mathcal{E}_{k_i}^i$; send$(v_{k_i}^i)$.

Un ordonnancement *de $R_1, \ldots, R_m$ de longueur ℓ est une fonction $\pi : [1, \ldots, \ell] \mapsto [1, \ldots, m]$ telle que pour tout $i \leq m$, on a $|\{j \mid \pi(j) = i\}| \ \leq\ k_i$.*

Un ordonnancement permet de définir l'ordre dans lequel les instructions issues des différents rôles s'entrelacent. Une instruction est un couple de la forme send(u); recv(v) (ou un triplet dans le cas du modèle avec tests d'égalités). Nous faisons le choix de traiter le couple send(u); recv(v) comme une instruction atomique, ce qui revient en fait à exécuter les instructions du type send(.) dès que possible. Retarder l'exécution des instructions send(.) revient à retarder le moment où l'on va divulguer une information à l'intrus, et conduit à générer des traces « inutiles ». En effet, s'il existe une attaque sur une telle trace, il en existe aussi une sur la trace correspondante dans laquelle les instructions send(.) ont été exécutées prioritairement.

Nous sommes particulièrement intéressés par la propriété de secret : celle-ci n'est pas satisfaite dès lors qu'il existe une trace accessible à l'issue de laquelle l'intrus est capable d'émettre le secret. C'est pour cette raison que nous ajoutons à la fin de la trace associée à un ordonnancement, une instruction « recv(s) ». Cette dernière instruction assure que le secret s est reçu par un agent, ce qui signifie qu'il a été émis sur le réseau, et qu'il est donc connu de l'intrus. Dans le cas d'une trace simple, nous ajouterons les instructions recv(y) et $y = s$ afin de conserver le caractère simple de la trace.

L'algorithme 4.2 permet de construire la trace symbolique associée à un ordonnancement et à un terme. Lorsque les rôles considérés sont des rôles exprimés dans le modèle avec filtrage, l'algorithme retourne une trace symbolique sans équation. Lorsque les rôles sont exprimés dans le modèle avec tests d'égalités, l'algorithme retourne une trace symbolique simple.

Étant donné un système d'inférence $\mathcal{I}$ et une théorie équationnelle E, le *problème de la sécurité* pour un nombre borné de sessions s'énonce de la façon suivante :

Entrées: $R_1 = \mathsf{recv}(u_1^1);\ \mathcal{E}_1^1;\ \mathsf{send}(v_1^1), \ldots, \mathsf{recv}(u_{k_1}^1);\ \mathcal{E}_{k_1}^1;\ \mathsf{send}(v_{k_1}^1)$
...
$R_m = \mathsf{recv}(u_1^m);\ \mathcal{E}_1^m;\ \mathsf{send}(v_1^m), \ldots, \mathsf{recv}(u_{k_m}^m);\ \mathcal{E}_{k_m}^m;\ \mathsf{send}(v_{k_m}^m)$,
π et s
Sortie: une trace symbolique

```
Algorithme:
   //intialisation
   //T est stockée sous forme d'une liste
   T := []
   pour i := 1 à m faire c[i] := 0

   //construction de la trace
   pour i := 1 à ℓ faire
     c[π(i)] := c[π(i) + 1]
     T := T @ [recv(u^{π(i)}_{c[π(i)]}); E^{π(i)}_{c[π(i)]}; send(v^{π(i)}_{c[π(i)]})]
   fin pour

   si (modèle avec filtrage)  alors
     T := T @ [recv(s)]
   si (modèle avec tests d'égalités) alors
     T := T @ [recv(y));y  =  s]

   retourner T.
```

Algorithme 4.2 - Construction d'une trace symbolique.

Problème de la sécurité (pour un nombre borné de sessions)

Entrées :
- un ensemble fini $R_1, \ldots, R_m$ d'instances de rôles,
- un ensemble fini T_0 de termes clos (la connaissance de l'intrus),
- un terme clos s (le secret).

Sortie : Est-ce qu'il existe un ordonnancement de $R_1, \ldots, R_m$ et s tel que la trace associée à cet ordonnancement par l'algorithme 4.2 soit accessible à partir de T_0 dans $(\mathcal{I}, \mathsf{E})$?

Puisque dans le cadre d'un nombre borné de sessions, le nombre d'ordonnancement possible est fini, l'unique difficulté réside dans le fait de décider si une trace symbolique T est accessible à partir de T_0 dans $(\mathcal{I}, \mathsf{E})$. L'algorithme 4.3 permet de construire le système de contraintes associé à la trace T et à la connaissance initiale T_0 pour lequel l'existence d'une solution dans $(\mathcal{I}, \mathsf{E})$ est équivalente à l'accessibilité de la trace T à partir de T_0 dans $(\mathcal{I}, \mathsf{E})$. Ce résultat est énoncé dans la proposition 4.14.

Remarque : Dans le cas d'une trace symbolique sans équation, le système de contraintes obtenus est un système de contraintes de déduction. Dans le cas d'une trace symbolique

```
Entrées: instr_1;...;instr_ℓ et T_0
Sortie: 𝓑 = 𝓒 ∪ 𝓔

Algorithme:
  𝓒 := ∅; 𝓔 := ∅; T := T_0

  pour i := 1 à ℓ faire
    si instr_i est de la forme recv(u) alors ajouter T ⊩ u dans 𝓒
    si instr_i est de la forme send(u) alors T := T ∪ {u}
    si instr_i est une équation u = v alors ajouter u = v dans 𝓔
  fin pour

  retourner 𝓒 ∪ 𝓔
```

Algorithme 4.3 - Génération d'un système de contraintes.

simple, le système de contraintes obtenus est un système de contraintes de déduction simple avec équations.

Proposition 4.14
Soient T une trace symbolique, T_0 un ensemble de messages et $(\mathcal{I}, \mathsf{E})$ un système d'inférence. Soit $\mathcal{C}$ le système de contraintes associé à T et à T_0 par l'algorithme 4.3. La trace T est accessible à partir de T_0 dans $(\mathcal{I}, \mathsf{E})$ si, et seulement si, le système $\mathcal{C}$ a une solution dans $(\mathcal{I}, \mathsf{E})$.

4.2.2.2 Exemples

Nous allons illustrer la construction d'un système de contraintes dans chacun des deux formalismes, sur l'exemple du protocole de Needham-Schroeder à clefs publiques [NS78]. Ce protocole a pour but de permettre une authentification mutuelle entre les agents A et B par l'intermédiaire des nonces N_a et N_b censés rester secrets.

$$\begin{array}{rcll} A & \rightarrow & B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

À la première étape du protocole, l'agent Alice envoie son nom A et un nonce N_a chiffré avec la clef publique de Bob. À la deuxième étape du protocole, Bob reçoit le message $\{A, N_a\}_{\mathsf{pub}(B)}$ envoyé par Alice. Il le déchiffre, et renvoie le nonce d'Alice ainsi qu'un autre nonce N_b qu'il vient de générer, le tout chiffré avec la clef publique d'Alice. À la troisième étape du protocole, Alice reçoit le message $\{N_a, N_b\}_{\mathsf{pub}(A)}$ et reconnaît son nonce N_a. Elle en déduit que Bob lui a répondu et elle lui renvoie son nonce N_b chiffré avec sa clef publique pour lui signifier qu'elle connaît maintenant le message N_b. Lorsque Bob reçoit ce message, Alice et Bob pensent qu'ils sont seuls à connaître les nonces N_a et N_b et que ces nonces permettent de les authentifier.

G. Lowe a découvert, dix-sept ans après la publication de ce protocole, que ce dernier avait une faille [Low96]. Cette faille, appelée *man-in-the-middle attack*, est réalisable par un intrus actif. L'intrus initie une session avec un agent honnête b jouant le rôle B en se faisant passer pour un agent a. L'agent b génère son nonce n_b. Ce dernier est censé rester secret entre b et l'agent avec lequel b pense communiquer, c'est-à-dire a. En fait, il n'en n'est rien : l'intrus récupère ce nonce n_b, et l'agent b termine sa session en pensant avoir communiqué avec a alors que ce dernier n'a jamais initié une communication avec b. Cette attaque est schématisée à la figure 4.4.

$$a \xrightarrow{\{a, n_a\}_{\mathsf{pub}(I)}} I$$

$$I(a) \xrightarrow{\{a, n_a\}_{\mathsf{pub}(b)}} b$$

$$a \xleftarrow{\{n_a, n_b\}_{\mathsf{pub}(a)}} b$$

$$a \xrightarrow{\{n_b\}_{\mathsf{pub}(I)}} I$$

$$I(a) \xrightarrow{\{n_b\}_{\mathsf{pub}(n)}} b$$

Figure 4.4 - Attaque sur le Protocole de Needham-Schroeder, due à G. Lowe.

L'agent a commence spontanément une conversation avec l'intrus I. L'agent I se sert de ce premier message pour se faire passer pour a auprès de b. Celui-ci répond donc à a. L'agent a, reconnaissant son nonce n_a pense que I vient de lui répondre. L'agent a, renvoie donc à I le nonce n_b que l'agent I n'aurait pas dû connaître. L'agent I termine alors le protocole avec b qui croit avoir parlé à a.

Nous allons modéliser les deux rôles du protocole de Needham-Schroeder dans chacun des deux formalismes.

Dans le modèle avec filtrage. Les deux rôles sont décrits à la figure 4.5. Le rôle A correspond à l'initiateur du protocole, ce rôle prend donc en paramètre le nom de l'agent exécutant ce rôle et le nom de celui avec lequel il souhaite établir une communication. En revanche, le rôle B apprendra le nom de son interlocuteur lors de la réception du premier message. Ce rôle a donc un unique paramètre : le nom de l'agent exécutant ce rôle.

$$\begin{array}{lll}
\text{Rôle } R_A(z_a, z_b) : & \nu n_a. & \mathsf{recv}(0); \mathsf{send}(\{z_a, n_a\}_{\mathsf{pub}(z_b)}) \\
 & & \mathsf{recv}(\{n_a, x_{n_b}\}_{\mathsf{pub}(z_a)}); \mathsf{send}(\{x_{n_b}\}_{\mathsf{pub}(z_b)}) \\
 & & \\
\text{Rôle } B(z_b) : & \nu n_b. & \mathsf{recv}(\{y_a, y_{n_a}\}_{\mathsf{pub}(z_b)}); \mathsf{send}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)}) \\
 & & \mathsf{recv}(\{n_b\}_{\mathsf{pub}(z_b)}); \mathsf{send}(0)
\end{array}$$

Figure 4.5 - Rôles du protocole de Needham-Schroeder dans le modèle avec filtrage.

Remarque : Les instructions $\mathsf{recv}(0)$ and $\mathsf{send}(0)$ permettent d'obtenir une séquence d'instruction homogène, constituée uniquement de couple $\mathsf{recv}(u); \mathsf{send}(v)$. En supposant que 0 est une constante publique, c'est-à-dire connue de l'intrus, l'ajout de ces instructions ne perturbent en rien l'exécution du rôle.

Considérons les instances $R_A(a, I)$ et $R_B(b)$ décrites ci-dessous.

$$\begin{array}{lll} \text{Rôle } R_A(a,I) : & \nu n_a. & \mathsf{recv}(0); \mathsf{send}(\{a, n_a\}_{\mathsf{pub}(I)}) \\ & & \mathsf{recv}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)}); \mathsf{send}(\{x_{n_b}\}_{\mathsf{pub}(I)}) \\ \\ \text{Rôle } B(b) : & \nu n_b. & \mathsf{recv}(\{y_a, y_{n_a}\}_{\mathsf{pub}(b)}); \mathsf{send}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)}) \\ & & \mathsf{recv}(\{n_b\}_{\mathsf{pub}(b)}); \mathsf{send}(0) \end{array}$$

Secret.
La trace symbolique ci-dessous est obtenue en considérant un ordonnancement de longueur 3 des deux instances de rôles décrites ci-dessus. Nous avons ajouté l'instruction $\mathsf{recv}(n_b)$ puisque nous nous intéressons ici au secret du nonce n_b.

$$T_s := \left\{ \begin{array}{l} \mathsf{recv}(0);\ \mathsf{send}(\{a, n_a\}_{\mathsf{pub}(I)});\ \mathsf{recv}(\{y_a, y_{n_a}\}_{\mathsf{pub}(b)});\ \mathsf{send}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)}); \\ \mathsf{recv}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)});\ \mathsf{send}(\{x_{n_b}\}_{\mathsf{pub}(I)});\ \mathsf{recv}(n_b) \end{array} \right.$$

Authentification.
On souhaite mettre en évidence le problème d'authentification suivant : l'agent b peut exécuter entièrement son rôle en pensant jouer avec a alors que ce dernier n'a jamais ouvert de sessions avec b. Pour cela, il faut instancier la variable y_a apparaissant dans le rôle $R_B(b)$ avec le nom d'agent a. Ensuite, il suffit de montrer que la trace suivante est accessible :

$$T_a := \left\{ \begin{array}{l} \mathsf{recv}(0);\ \mathsf{send}(\{a, n_a\}_{\mathsf{pub}(I)});\ \mathsf{recv}(\{a, y_{n_a}\}_{\mathsf{pub}(b)});\ \mathsf{send}(\{y_{n_a}, n_b\}_{\mathsf{pub}(a)}); \\ \mathsf{recv}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)});\ \mathsf{send}(\{x_{n_b}\}_{\mathsf{pub}(I)});\ \mathsf{recv}(\{n_b\}_{\mathsf{pub}(b)}) \end{array} \right.$$

Soit T_0 un ensemble de termes représentant la connaissance de l'intrus. Par exemple, on peut raisonnablement supposer que $T_0 = \{0, a, b, I, \mathsf{pub}(a), \mathsf{pub}(b), \mathsf{pub}(I), \mathsf{priv}(I)\}$. Ces choix ne sont pas anodins, ils ont pour but de retrouver la fameuse attaque de G. Lowe que nous avons décrite de façon informelle au début de cette partie. Le système de contraintes retourné par l'algorithme 4.3 sur l'entrée T_a et T_0, est le suivant :

$$\mathcal{C} := \left\{ \begin{array}{rcl} T_0 & \Vdash & 0 \\ T_0,\ \{a, n_a\}_{\mathsf{pub}(I)} & \Vdash & \{a, y_{n_a}\}_{\mathsf{pub}(b)} \\ T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(a)} & \Vdash & \{n_a, x_{nb}\}_{\mathsf{pub}(a)} \\ T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(a)},\ \{x_{n_b}\}_{\mathsf{pub}(I)} & \Vdash & \{n_b\}_{\mathsf{pub}(b)} \end{array} \right.$$

Considérons le système d'inférence de Dolev-Yao (dans sa version « chiffrement asymétrique »). Le système $\mathcal{C}$ admet pour solution $\sigma = \{y_{n_a} \mapsto n_a;\ \ x_{n_b} \mapsto n_b\}$. Cette solution correspond à l'attaque de G. Lowe. L'intrus construit le message $\{a, n_a\}_{\mathsf{pub}(b)}$. À la deuxième étape, il se contente de transmettre (sans le modifier) le message émis par b, c'est-à-dire $\{n_a, n_b\}_{\mathsf{pub}(a)}$. Il récupère alors $\{n_b\}_{\mathsf{pub}(I)}$ et construit le message $\{n_b\}_{\mathsf{pub}(b)}$ qu'il envoie à b.

Dans le modèle avec tests d'égalités. Les deux rôles sont représentés à la figure 4.6.

$$\begin{array}{lll}
\text{Rôle } A(z_a, z_b): & \nu n_a. & \mathsf{recv}(y_0); y_0 = 0; \mathsf{send}(\{z_a, n_a\}_{\mathsf{pub}(z_b)}) \\
& & \mathsf{recv}(y_1); \\
& & \mathsf{proj}_1(\mathsf{dec}(y_1, \mathsf{priv}(z_a))) = n_a; \\
& & \mathsf{send}(\{\mathsf{proj}_2(\mathsf{dec}(y_1, \mathsf{priv}(z_a)))\}_{\mathsf{pub}(z_b)}) \\
\\
\text{Rôle } B(z_b): & \nu n_b. & \mathsf{recv}(y_2); \\
& & \mathsf{send}(\{\mathsf{proj}_2(\mathsf{dec}(y_2, \mathsf{priv}(z_b))), n_b\}_{\mathsf{pub}(\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(z_b))))}) \\
& & \mathsf{recv}(y_3); \mathsf{dec}(y_3, \mathsf{priv}(z_b)) = n_b; \mathsf{send}(0)
\end{array}$$

Figure 4.6 - Rôles du protocole de Needham-Schroeder dans le modèle avec tests d'égalités.

Secret.
La trace symbolique obtenue en considérant les mêmes instances et le même ordonnancement que précédemment est la suivante :

$$T_s := \left\{ \begin{array}{l}
\mathsf{recv}(0);\ y_0 = 0;\ \mathsf{send}(\{a, n_a\}_{\mathsf{pub}(I)}); \\
\mathsf{recv}(y_2);\ \mathsf{send}(\{\mathsf{proj}_2(\mathsf{dec}(y_2, \mathsf{priv}(b))), n_b\}_{\mathsf{pub}(\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(b))))}); \\
\mathsf{recv}(y_1);\ \mathsf{proj}_1(\mathsf{dec}(y_1, \mathsf{priv}(z_a))) = n_a;\ \mathsf{send}(\{\mathsf{proj}_2(\mathsf{dec}(y_1, \mathsf{priv}(a)))\}_{\mathsf{pub}(I)}); \\
\mathsf{recv}(y_3);\ y_3 = n_b
\end{array} \right.$$

Authentification.
Si l'on s'intéresse à la propriété d'authentification, il faut alors ajouter le test d'égalité « $\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(b))) = a$ » pour assurer que l'agent b communique apparemment avec a. On obtient alors la trace symbolique suivante :

$$T_a := \left\{ \begin{array}{l}
\mathsf{recv}(0);\ y_0 = 0;\ ; \mathsf{send}(\{a, n_a\}_{\mathsf{pub}(I)}); \\
\mathsf{recv}(y_2);\ \mathsf{send}(\{\mathsf{proj}_2(\mathsf{dec}(y_2, \mathsf{priv}(b))), n_b\}_{\mathsf{pub}(\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(b))))}); \\
\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(b))) = a; \\
\mathsf{recv}(y_1);\ \mathsf{proj}_1(\mathsf{dec}(y_1, \mathsf{priv}(z_a))) = n_a;\ \mathsf{send}(\{\mathsf{proj}_2(\mathsf{dec}(y_1, \mathsf{priv}(a)))\}_{\mathsf{pub}(I)}); \\
\mathsf{recv}(y_3);\ \mathsf{dec}(y_3, \mathsf{priv}(b)) = n_b;
\end{array} \right.$$

Le système de contraintes $\mathcal{C} \cup \mathcal{E}$ associé à la trace T_s est décrit ci-dessous.

$$\mathcal{C} := \left\{ \begin{array}{lcl}
T_0 & \Vdash & y_0 \\
T_0,\ \{a, n_a\}_{\mathsf{pub}(I)} & \Vdash & y_2 \\
T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{\mathsf{proj}_2(\mathsf{dec}(y_2, \mathsf{priv}(b))), n_b\}_{\mathsf{pub}(\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(b))))} & \Vdash & y_1 \\
\\
T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{\mathsf{proj}_2(\mathsf{dec}(y_2, \mathsf{priv}(b))), n_b\}_{\mathsf{pub}(\mathsf{proj}_1(\mathsf{dec}(y_2, \mathsf{priv}(b))))}, & & \\
\qquad\qquad\qquad\qquad \{\mathsf{proj}_2(\mathsf{dec}(y_1, \mathsf{priv}(a)))\}_{\mathsf{pub}(I)} & \Vdash & y_3
\end{array} \right.$$

$$\mathcal{E} := \left\{ \begin{array}{l}
y_0 = 0; \\
\mathsf{proj}_1(\mathsf{dec}(y_1, \mathsf{priv}(a))) = n_a; \\
y_3 = n_b
\end{array} \right.$$

Dans le modèle de Dolev-Yao (dans sa version avec chiffrement asymétrique) avec destructeurs explicites, le système $\mathcal{B} = \mathcal{C} \cup \mathcal{E}$ admet pour solution :

$$\sigma = \{y_1 \mapsto \{n_a, n_b\}_{\mathsf{pub}(a)};\ y_2 \mapsto \{a, n_a\}_{\mathsf{pub}(b)};\ \ y_3 \mapsto n_b\}.$$

En effet, les équations de $\mathcal{E}$ sont satisfaites modulo la théorie équationnelle composée des trois axiomes : $\mathsf{dec}(\{x\}_{\mathsf{pub}(y)}, \mathsf{priv}(y)) = x$, $\mathsf{proj}_1(\langle x, y\rangle) = x$ et $\mathsf{proj}_2(\langle x, y\rangle) = y$. Il en est de même pour les contraintes de déduction closes du système $\mathcal{C}\sigma$ en supposant que les symboles $\mathsf{dec}(.,.)$ et $\{.\}_.$ sont publics. Ces contraintes sont écrites ci-dessous :

$$\mathcal{C}\sigma := \left\{ \begin{array}{ll} T_0,\ \{a, n_a\}_{\mathsf{pub}(I)} & \Vdash\ \ \{a, n_a\}_{\mathsf{pub}(b)} \\ T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{n_a, n_b\}_{\mathsf{pub}(b)} & \Vdash\ \ \{n_a, n_b\}_{\mathsf{pub}(b)} \\ T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{n_a, n_b\}_{\mathsf{pub}(b)},\ \{n_b\}_{\mathsf{pub}(I)} & \Vdash\ \ n_b \end{array} \right.$$

4.2.3 Systèmes de contraintes bien formés

Dans cette partie, nous définissons différentes propriétés (propriété de monotonie, propriété d'initialisation, ...) et nous montrons que ces propriétés sont systématiquement satisfaites par les systèmes de contraintes que nous générons à partir d'instances de rôles bien formés.

Définition 4.15 (propriété de monotonie)
Un système de contraintes de déduction $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_n \Vdash u_n\}$ est dit monotone *s'il existe une permutation σ de $\{1, \ldots, n\}$ telle que :*
pour tout i tel que $1 \leq i < n$, l'inclusion $T_{\sigma(i)} \subseteq T_{\sigma(i+1)}$ soit satisfaite.

Cette propriété est satisfaite par construction. Elle correspond au fait que la connaissance de l'intrus ne fait que croître au cours de l'exécution du protocole.

Définition 4.16 (propriété d'initialisation)
Un système de contraintes de déduction $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_n \Vdash u_n\}$ satisfait la propriété d'initialisation *s'il existe une permutation σ de $\{1, \ldots, n\}$ telle que :*
pour tout i tel que $1 \leq i \leq n$, $\forall x \in vars(T_{\sigma(i)})$, $\exists j$ tel que $x \in vars(u_{\sigma(j)})$ et $\sigma(j) < \sigma(i)$.

Remarque : Dans la suite, les systèmes de contraintes de déduction monotones et satisfaisant la propriété d'initialisation seront notés $\{T_1 \Vdash u_1, \ldots, T_n \Vdash u_n\}$ et l'ordre dans lequel les contraintes du système seront écrites correspondra à l'ordre induit par l'inclusion des membres gauches des contraintes. Il est important de noter que si un système de contraintes $\mathcal{C}$ est monotone et satisfait la propriété d'initialisation alors la permutation σ permettant d'établir la propriété de monotonie satisfait également la condition suivante :
pour tout i tel que $1 \leq i \leq n$, $\forall x \in vars(T_{\sigma(i)})$, $\exists j$ tel que $x \in vars(u_{\sigma(j)})$ et $\sigma(j) < \sigma(i)$.

Nous pouvons maintenant définir la notion de systèmes de contraintes bien formés.

Définition 4.17 (système de contraintes bien formé)
Un système $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_n \Vdash u_n\}$ est dit bien formé *si :*

1. *$\mathcal{C}$ est monotone, et*

2. *pour toute substitution θ, le système de contraintes $\mathcal{C}\theta$ satisfait la propriété d'initialisation.*

Remarque : Toutes ces notions sont définies de manière similaires dans le cas d'un système de contraintes en un pas et dans le cas d'un système de contraintes de déduction simple avec équations.

Nous montrons, dans la proposition 4.18, que les systèmes de contraintes générés à partir d'instances de rôles bien formés sont des systèmes de contraintes bien formés.

Proposition 4.18
Soient $R_1, \ldots, R_m$ des instances de rôles bien formés, T_0 un ensemble fini de messages et s un terme. Soit π un ordonnancement de $R_1, \ldots, R_m$. Notons T la trace associée à cet ordonnancement par l'algorithme 4.2 et $\mathcal{C}$ le système de contraintes obtenu en appliquant l'algorithme 4.3 sur l'entrée T, T_0. Le système $\mathcal{C}$ est un système de contraintes bien formé.

Démonstration.
Plaçons nous dans le modèle avec filtrage (la preuve est similaire dans le cas du modèle avec tests d'égalités). Soient $R_1, \ldots, R_m$ des instances de rôles bien formés. Posons :

$$R_i = \mathsf{recv}(u_1^i); \mathsf{send}(v_1^i); \ldots; \mathsf{recv}(u_{k_i}^i); \mathsf{send}(v_{k_i}^i).$$

La trace T obtenue en appliquant l'algorithme 4.2 sur $R_1 \ldots, R_m$, π et s est donc de la forme :

$$\mathsf{recv}(u_1); \mathsf{send}(v_1); \ldots; \mathsf{recv}(u_\ell); \mathsf{send}(v_\ell); \mathsf{recv}(s).$$

Nous allons montrer que T satisfait la propriété suivante : pour toute substitution θ, pour tout $i \leq \ell$, pour tout $x \in vars(v_i\theta)$, il existe $j \leq i$ tel que $x \in vars(u_j\theta)$.

Soit θ une substitution, soit $p \leq \ell$ et $x \in vars(v_p\theta)$. Tout d'abord, il existe $i \leq m$ et $j \leq k_i$ tel que $v_p = v_j^i$. Puisque R_i est une instance d'un rôle bien formé, nous savons qu'il existe $j' \leq j$ tel que $x \in vars(u_{j'}^i\theta)$. Par construction d'une trace, il existe $p' \leq p$ tel que $u_{j'}^i = u_{p'}$. On en déduit que $x \in vars(u_{p'}\theta)$, et que T satisfait la propriété énoncée.

Par construction (*cf.* algorithme 4.3), le système de contraintes de déduction $\mathcal{C}$ est de la forme :

$$\mathcal{C} := \left\{ \begin{array}{rcl} T_0 & \Vdash & u_1 \\ T_0, v_1 & \Vdash & u_2 \\ & \vdots & \\ T_0, v_1, \ldots, v_{\ell-1} & \Vdash & u_\ell \\ T_0, v_1, \ldots, v_{\ell-1}, v_\ell & \Vdash & s \end{array} \right.$$

Tout d'abord, le système $\mathcal{C}$ est monotone. Montrons maintenant que la propriété d'initialisation est stable par substitution pour ce système. Soit θ une substitution, soit $i \leq \ell$ et $x \in vars(v_i\theta)$. Nous avons montré qu'il existe $j' < j$ tel que $x \in vars(u_{j'}\theta)$. Nous en déduisons donc que le système de contraintes $\mathcal{C}$ est un système bien formé. □

Dans le cadre d'un nombre borné de sessions, le problème de la sécurité consiste à générer un nombre fini de traces symboliques et à regarder si l'une d'entre elles est accessible. Le problème de l'accessibilité d'une trace est équivalent au problème de la résolution d'un système de contraintes (*cf.* proposition 4.14). Les systèmes de contraintes générés sont monotones et satisfont la propriété d'initialisation dès lors que l'on considère des instances de rôles bien formés (*cf.* proposition 4.18). Autrement dit, pour la suite, nous pouvons nous concentrer sur la résolution de systèmes de contraintes symboliques bien formés.

Deuxième partie

Vérification des protocoles cryptographiques

Chapitre 5

Vérification dans le modèle par rôles avec tests d'égalités

Sommaire

5.1 Cadre **76**
5.1.1 Théories équationnelles considérées 76
5.1.2 Exemples 78
5.2 Problème de déduction de l'intrus **80**
5.2.1 Système d'inférence 80
5.2.2 Résultat de localité 81
5.2.3 Déductibilité en un pas 83
5.3 Résolution de systèmes de contraintes bien formés **83**
5.3.1 Procédure 84
5.3.2 Terminaison et Complexité 84
5.3.3 Correction 87
5.3.4 Complétude 88
5.3.5 NP-difficulté 95
5.3.6 Illustration de la procédure 96
5.4 Comparaisons **99**
5.4.1 Travaux de M. Abadi et V. Cortier 99
5.4.2 Travaux de M. Baudet 100
5.4.3 Travaux de Y. Chevalier et M. Rusinowitch 100

Dans le cadre d'un nombre borné de sessions, de nombreux résultats ont été obtenus au cours de ces dernières années (*e.g.* [CKRT03b, CLS03, CKRT03a]). Tous ces résultats considèrent une théorie équationnelle particulière. Compte tenu de la multitude des théories équationnelles pertinentes du point de vue de la vérification des protocoles cryptographiques, il serait intéressant d'obtenir un résultat générique. Dans ce chapitre, nous ne considérons pas des théories équationnelles complexes, impliquant des opérateurs AC. Notre but est d'obtenir une procédure générique permettant de réaliser la vérification de protocoles pour différentes variantes du modèle de Dolev-Yao (chiffrement symétrique/asymétrique, déterministe/probabiliste, ...). À notre connaissance, il s'agissait, en 2004, du premier résultat de décidabilité du problème de la sécurité pour une classe de théorie équationnelle définie de manière syntaxique [DJ04a].

Pour obtenir ce résultat générique, nous avons fait le choix d'utiliser l'approche « destructeurs explicites » et de modéliser les rôles sans avoir recours à l'opération de filtrage. Nous mettons ainsi en évidence que le problème de la vérification n'est, du point de vue théorique, pas plus difficile dans le modèle classique de Dolev-Yao que dans ce nouveau modèle, plus réaliste, avec destructeurs. À l'heure actuelle, d'autres résultats génériques existent. La plupart de ces travaux utilisent le modèle à base de destructeurs explicites. Ce dernier semble plus adapté que le modèle à base de filtrage pour l'obtention de résultats génériques.

Dans la première partie de ce chapitre, nous définissons la classe des théories équationnelles publique-effondrantes, et nous donnons des exemples de théories équationnelles publique-effondrantes pertinentes du point de vue de la vérification des protocoles cryptographiques. Nous nous intéressons ensuite au problème de déduction de l'intrus (*cf.* partie 5.2), puis au problème de la vérification en présence d'un intrus actif (*cf.* partie 5.3), en donnant une procédure permettant de décider la satisfaisabilité de systèmes de contraintes simples et bien formés avec équations. Enfin, nous comparons nos résultats à d'autres résultats génériques obtenus récemment dans le domaine (*cf.* partie 5.4.1).

5.1 Cadre

Le but de cette partie est d'introduire la classe des théories équationnelles publique-effondrantes dont nous faisons l'étude dans ce chapitre, et de donner des exemples de théories équationnelles appartenant à cette classe.

5.1.1 Théories équationnelles considérées

L'ensemble des symboles de fonctions est noté $\mathcal{F}$. Cet ensemble est composé de deux sous-ensembles disjoints :

- $\mathcal{PF}$, l'ensemble des *symboles privés* : par exemple l'opération $\mathsf{inv}(\cdot)$, permettant d'obtenir la clef inverse d'une clef, est représentée, en général, par un symbole privé. L'intrus ne doit pas être capable de récupérer la clef inverse associée à n'importe quelle clef publique.
- $\mathcal{VF}$, l'ensemble des *symboles publics* ou *visibles* : ces symboles représentent les opérations accessibles à l'intrus, comme par exemple le chiffrement, la concaténation, ...

Définition 5.1 (système de réécriture public-effondrant)
Un système de réécriture $\mathcal{R}$ est dit public-effondrant *si chacune des règles $\ell \to r$ de $\mathcal{R}$ vérifie les deux conditions suivantes :*

1. *$r \in vars(\ell)$ ou $r \in \mathcal{T}(\mathcal{VF})\!\downarrow$*
2. *si $\ell = f(l_1, \ldots, l_n)$ avec $f \in \mathcal{VF}$, alors pour tout $i \leq n$, pour toute position $p \in \mathcal{O}(l_i)$ telle que $l_i|_p = g(t_1, \ldots, t_m)$ avec $g \in \mathcal{VF}$*
   - *soit $g(t_1, \ldots, t_m) \in \mathcal{T}(\mathcal{VF})\!\downarrow$,*
   - *soit il existe $j \leq m$ tel que $t_j = r$.*

La condition 1. impose que le membre droit de chacune des règles de réécriture soit une variable ou un terme clos déductible par l'intrus. L'application d'une telle règle de réécriture en tête d'un terme permet soit d'extraire un sous-terme syntaxique, soit d'obtenir un terme public (c'est-à-dire un terme construit uniquement à partir de symboles de fonctions visibles).

La condition 2. est plus restrictive. Tout d'abord, lorsque le symbole de tête est un symbole privé, aucune autre condition supplémentaire n'est demandée. En revanche, si le symbole de tête f est un symbole public, des conditions supplémentaires sont requises. L'idée est d'assurer que lorsque l'intrus appliquera le symbole f sur des termes $m_1, \ldots, m_n$, dans le but d'obtenir un sous-terme m de l'un des m_i (par l'intermédiaire de la normalisation), il n'ait pas besoin de construire les termes $m_1, \ldots, m_n$: ces termes apparaissent déjà en tant que sous-termes dans sa connaissance, et il a simplement dû décomposer un terme dans sa connaissance pour récupérer le sous-terme en question. Pour assurer cette propriété, on impose que pour chacun des m_i, une des deux conditions suivantes soit satisfaite :

- Tout d'abord, si m_i est en-tête avec un symbole privé, dans ce cas, nous sommes sûr que l'intrus n'a pas pu construire ce terme.
- Maintenant, si m_i est en-tête avec un symbole public, l'idée est d'assurer que l'intrus n'a pas construit m_i dans le but de faire une preuve de m, puisque m est un sous-terme direct de m_i.

L'objectif, à travers ces restrictions, est d'assurer un résultat de localité pour la notion de sous-termes syntaxiques ; il est donc important que l'intrus n'ait pas besoin, pour obtenir un terme u, de construire des termes en dehors de l'ensemble $St(T, u)$. L'exemple 5.2 illustre (au moins en partie) l'intérêt de la condition 2. de la définition 5.1.

Exemple 5.2
Considérons le système de réécriture convergent suivant :

$$\mathcal{R} := \quad f_3(f_2(f_1(x))) \ \rightarrow \ x.$$

Nous supposons que les symboles f_2 et f_3 sont publics. Ce système n'est pas public-effondrant : la condition 2. n'est pas satisfaite par la règle $f_3(f_2(f_1(x))) \ \rightarrow \ x$.

*Considérons le problème de déduction de l'intrus suivant : Est-ce que le terme s est déductible à partir de $f_1(s)$? La réponse est oui (*cf.* arbre de preuve ci-dessous), mais pour faire cette preuve, il est nécessaire d'appliquer le symbole f_2 puis le symbole f_3 et donc de passer par le terme intermédiaire $f_2(f_1(s))$. Ce terme ne fait pas partie des sous-termes syntaxiques de $f_1(s)$, ni de s.*

$$\dfrac{\dfrac{\dfrac{f_1(s) \vdash f_1(s)}{f_1(s) \vdash f_2(f_1(s))}\,(\mathsf{C})}{f_1(s) \vdash f_3(f_2(f_1(s)))}\,(\mathsf{C})}{f_1(s) \vdash s}\,(\mathsf{Eq})$$

Afin d'obtenir un résultat de localité pour la notion de sous-termes syntaxiques, il est donc nécessaire d'imposer des conditions sur les sous-termes des membres gauches de chacune des règles de réécriture : c'est l'objectif des restrictions exprimées par la condition 2. de la définition 5.1.

Définition 5.3 (théorie équationnelle publique-effondrante)
Une théorie équationnelle E *est dite* publique-effondrante *s'il existe un système de réécriture $\mathcal{R}$ convergent et public-effondrant représentant* E, *c'est-à-dire tel que les relations $=_{\mathsf{E}}$ et $\xleftrightarrow{*}_{\mathcal{R}}$ soient égales.*

Le système d'inférence représentant le pouvoir de déduction de l'intrus que nous considérons ici est décrit à la figure 5.1. Il s'agit d'un système d'inférence simple, noté $\mathcal{I}_{\mathcal{VF}}$, accompagné de la règle (Eq). Cette dernière permet de prendre en compte les propriétés algébriques modélisées par la théorie équationnelle E, elle-même représentée par le système de réécriture $\mathcal{R}$.

$$\mathcal{I}_{\mathcal{VF}} := \left\{ \frac{T \vdash u_1 \quad \ldots \quad T \vdash u_n}{T \vdash f(u_1, \ldots, u_n)} \text{ (C) avec } f \in \mathcal{VF} \right. \qquad \frac{T \vdash u}{T \vdash v} \text{ (Eq) avec } u \overset{*}{\leftrightarrow}_{\mathcal{R}} v$$

Figure 5.1 - Système d'inférence ($\mathcal{I}_{\mathcal{VF}}$, E).

Remarque : Dans toute la suite, nous supposons que les termes $St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})$, représentant l'ensemble des sous-termes de $\mathcal{R}$ formés uniquement de symboles publics, sont dans la connaissance initiale de l'intrus. Cette condition ne doit pas être vue comme une restriction puisque l'intrus est, de toutes les façons, capable de construire ces termes. Elle permet simplement d'assurer que l'intrus n'aura pas besoin de construire un de ces « gros » termes pour déclencher une règle de réécriture : ces termes publics sont déjà dans sa connaissance. De plus, nous supposons l'existence d'une constante publique spéciale, que nous notons 0.

Lemme 5.4
Soit $\mathcal{R}$ un système de réécriture convergent et public-effondrant. Soient $s, s_1, \ldots, s_n \in \mathcal{T}(\mathcal{F})\downarrow$. Nous avons :

$$s = f(s_1, \ldots, s_n)\downarrow \textit{ si, et seulement si, } s = f(s_1, \ldots, s_n) \textit{ ou } f(s_1, \ldots, s_n) \xrightarrow{\Lambda} s.$$

Démonstration.
($\Rightarrow$) Par hypothèse, les termes $s_1, \ldots, s_n$ sont en formes normales. Soit $f(s_1, \ldots, s_n)$ est également en forme normale, et dans ce cas nous avons $s = f(s_1, \ldots, s_n)$, ou alors nous avons $f(s_1, \ldots, s_n) \to^* s$. La première étape de réécriture a nécessairement lieu à la position Λ (c'est-à-dire en-tête) puisque $s_1, \ldots, s_n$ sont en formes normales. De plus, le terme obtenu à l'issue de cette première étape est en forme normale compte tenu du fait que $\mathcal{R}$ est un système public-effondrant (condition 1. de la définition 5.1). D'où $f(s_1, \ldots, s_n) \xrightarrow{\Lambda} s'$ avec s' en forme normale. Puisque $\mathcal{R}$ est un système convergent, nous en déduisons que $s = s'$ et nous concluons.
($\Leftarrow$) Nous avons $f(s_1, \ldots, s_n) \to^* s$ et s en forme normale, donc par définition de $\downarrow$, nous avons $s = f(s_1, \ldots, s_n)\downarrow$. □

5.1.2 Exemples

Dans cette partie, nous illustrons la définition 5.1 en donnant plusieurs exemples de théories équationnelles pertinentes dans le cadre de la vérification des protocoles cryptographiques, et pouvant être représentées par un système de réécriture convergent et public-effondrant. Parmi les symboles de fonctions publiques, nous avons :

– des symboles permettant de représenter le chiffrement symétrique $\{\cdot\}^s_\cdot$, asymétrique $\{\cdot\}^a_\cdot$ et probabiliste $\mathsf{penc}(\cdot,\cdot,\cdot)$;
– des symboles pour représenter les algorithmes de déchiffrement correspondants, notés respectivement $\mathsf{sdec}(\cdot,\cdot)$, $\mathsf{adec}(\cdot,\cdot)$ et $\mathsf{pdec}(\cdot,\cdot)$;
– un symbole binaire pour représenter la concaténation de deux messages, $\langle\cdot,\cdot\rangle$;
– deux symboles de projections, permettant de récupérer les composantes d'une paire, que nous notons $\mathsf{proj}_1(\cdot)$ et $\mathsf{proj}_2(\cdot)$;
– un constructeur de clefs publiques, noté $\mathsf{pub}(\cdot)$.

Enfin, nous utilisons un symbole unaire privé, noté $\mathsf{inv}(\cdot)$. Il permet d'obtenir la clef inverse associée à une clef donnée. Cet opérateur est particulièrement important dans le cas du chiffrement asymétrique. Il permet de parler de la clef de déchiffrement associée à une clef de chiffrement donnée.

Variantes de la théorie de Dolev-Yao. Plusieurs variantes de la théorie standard, dite de Dolev-Yao, sont représentables par des théories équationnelles publique-effondrantes.

- La version axiomatisée du modèle standard de Dolev-Yao, notée E_{DY}, est essentiellement composée de trois axiomes.

$$\mathsf{sdec}(\{x\}_y, y) = x, \qquad \mathsf{proj}_1(\langle x_1, x_2\rangle) = x_1, \quad \text{et} \quad \mathsf{proj}_2(\langle x_1, x_2\rangle) = x_2$$

  L'orientation de ces trois axiomes de gauche à droite permet d'obtenir un système de réécriture convergent et public-effondrant représentant E_{DY}.

- Nous pouvons également représenter la version « chiffrement asymétrique » du modèle de Dolev-Yao. Pour cela, nous ajoutons les deux équations $\mathsf{adec}(\{x\}^a_y, \mathsf{inv}(y)) = x$ et $\mathsf{inv}(\mathsf{inv}(x)) = x$. Le système convergent et public-effondrant associé à cette théorie est obtenu en orientant ces équations de gauche à droite et en ajoutant la conséquence $\mathsf{adec}(\{x\}^a_{\mathsf{inv}(y)}, y) \to x$.

- Une autre variante est de considérer que les opérations de chiffrement et de déchiffrement commutent. C'est le cas, par exemple, du chiffrement RSA [RSA78]. Il faut alors prendre en compte deux équations supplémentaires :

$$\{\mathsf{sdec}(x,y)\}^s_y = x \quad \text{et} \quad \{\mathsf{adec}(x,y)\}^a_{\mathsf{inv}(y)} = x.$$

  On obtient un système convergent en orientant ces équations de gauche à droite et en ajoutant la conséquence $\{\mathsf{adec}(x, \mathsf{inv}(y))\}^a_y \to x$.

Chiffrement involutif. Ce type de chiffrement, mentionné dans [RT03], se modélise par une théorie équationnelle publique-effondrante. Il suffit d'ajouter l' axiome $\{\{x\}^a_y\}^a_{\mathsf{inv}(y)} = x$ à la théorie standard de Dolev-Yao. Pour obtenir le système convergent associé, il suffit d'ajouter les deux règles de réécriture suivantes au système de réécriture représentant E_{DY} :

$$\{\{x\}^a_y\}^a_{\mathsf{inv}(y)} \to x \quad \text{et} \quad \{\{x\}^a_{\mathsf{inv}(y)}\}^a_y \to x.$$

Chiffrement probabiliste. Ce type de chiffrement a été introduit par S. Goldwasser et S. Micali [GM84]. L'objectif d'un tel mode de chiffrement est de faire en sorte que le chiffrement de deux messages identiques par la même clef ne permettent pas d'obtenir deux fois le même résultat. On modélise ce type de chiffrement en utilisant un symbole de fonction ternaire dont l'argument supplémentaire représente en fait un nombre aléatoire. L'équation que nous avons à prendre en compte est $\mathsf{pdec}(\mathsf{penc}(m, k, r), k) = m$. En l'orientant de gauche à droite, on obtient un système convergent et public-effondrant.

Bien sûr, les théories mentionnées ci-dessus, ne sont que des exemples. Notre résultat est générique et s'applique à n'importe quelle théorie publique-effondrante. Dans la suite, la théorie équationnelle E est une théorie équationnelle publique-effondrante quelconque et $\mathcal{R}$ est un système de réécriture convergent et public-effondrant représentant E.

5.2 Problème de déduction de l'intrus

Dans cette partie, nous nous intéressons au problème de déduction de l'intrus. Nous allons appliquer le schéma de preuve présenté au chapitre 4. Il nous faut donc trouver un système d'inférence équivalent au système $(\mathcal{I}_{\mathcal{VF}}, \mathsf{E})$ ayant la propriété de localité pour la notion de sous-termes syntaxiques que nous avons définie au chapitre 2 (*cf.* Définition 2.1).

5.2.1 Système d'inférence

Nous allons considérer le système d'inférence $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$ composé de la règle suivante :

$$\frac{T \vdash u_1 \quad \dots \quad T \vdash u_n}{T \vdash f(u_1, \dots, u_n)\downarrow}\,(\mathsf{C}) \quad \text{avec } f \in \mathcal{VF}.$$

Autrement dit, conformément à ce que nous avions annoncé au chapitre 4, nous choisissons de normaliser systématiquement les termes et de ne plus utiliser la règle (Eq).

Lemme 5.5
Soient $T \subseteq \mathcal{T}(\mathcal{F})\downarrow$ et $u \in \mathcal{T}(\mathcal{F})\downarrow$. Nous avons :

$$T \vdash u \text{ dans } (\mathcal{I}_{\mathcal{VF}}, \mathcal{R}) \Leftrightarrow T \vdash u \text{ dans } (\mathcal{I}_{\mathcal{VF}}, \mathsf{E}).$$

Démonstration.
$(\Rightarrow)$ Soit P une preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. La preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathsf{E})$ s'obtient à partir de P en insérant des étapes d'égalités, c'est-à-dire des instances de la règle (Eq) pour remplacer chacune des étapes de normalisation.
$(\Leftarrow)$ Soit P une preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathsf{E})$. Nous allons montrer, par induction sur la structure de P, qu'il existe une preuve P' de $T \vdash u\downarrow$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$.

Cas de base : Si P se réduit à une feuille, alors P est l'arbre de preuve cherché.

Étape d'induction : On distingue deux cas, suivant la dernière règle d'inférence utilisée dans P :
- Si P se termine par une instance de (C), alors P est de la forme :

$$\frac{P_1\left\{\ \dfrac{\cdots}{T \vdash u_1}\right. \quad \dots \quad P_n\left\{\ \dfrac{\cdots}{T \vdash u_n}\right.}{T \vdash f(u_1, \dots, u_n)}\,(\mathsf{C})$$

Par hypothèse d'induction, il existe des preuves $P'_1, \dots, P'_n$ de $T \vdash u_1\downarrow, \dots, T \vdash u_n\downarrow$. Soit P', la preuve obtenue à partir de $P'_1, \dots, P'_n$ par application du symbole f suivi d'une normalisation. L'arbre de preuve P' est une preuve de $T \vdash f(u_1\downarrow, \dots, u_n\downarrow)\downarrow$, c'est-à-dire de $T \vdash f(u_1, \dots, u_n)\downarrow$, dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$.

– Si P se termine par une instance de (Eq), alors P est de la forme :

$$\frac{P_1 \left\{ \begin{array}{c} \dots \\ \hline T \vdash u_1 \end{array} \right.}{T \vdash u_2} \;(\mathsf{Eq})$$

Par hypothèse d'induction, il existe une preuve P'_1 de $T \vdash u_1\downarrow$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. De plus, nous savons que $u_1\downarrow = u_2\downarrow$. Nous en déduisons donc que P'_1 est une preuve de $T \vdash u_2\downarrow$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$.

Par hypothèse, nous savons qu'il existe une preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathsf{E})$ et que u est un terme en forme normale. Nous en déduisons donc qu'il existe une preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. □

5.2.2 Résultat de localité

Pour établir le résultat de localité, nous allons être amenés à distinguer deux types d'arbre de preuve : les preuves dites par *composition* et celles dites par *décomposition*. Ces notions sont définies en regardant comment le terme étiquetant la racine de l'arbre de preuve a été obtenu. Si celui-ci a été obtenu après une étape de normalisation, nous parlons de *preuve par décomposition*, sinon nous parlons de *preuve par composition*.

Définition 5.6 (preuve par composition, décomposition)
Une preuve P de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$ est une preuve par composition *si elle est de la forme :*

$$\frac{P_1 \left\{ \begin{array}{c} \dots \\ \hline T \vdash u_1 \end{array} \right. \quad \dots \quad P_n \left\{ \begin{array}{c} \dots \\ \hline T \vdash u_n \end{array} \right.}{T \vdash f(u_1, \dots, u_n)} \;(\mathsf{C}) \quad \textit{avec } f \in \mathcal{VF}.$$

Sinon, nous disons que P est une preuve par décomposition.

Exemple 5.7
Considérons la théorie E_{DY} et posons $T = \{\{m_1\}_k, k, m_2\}$.

$$P_1 := \frac{T \vdash \{m_1\}_k \quad T \vdash k}{T \vdash m_1} \;(\mathsf{C}) \qquad P_2 := \frac{T \vdash m_2 \quad T \vdash k}{T \vdash \mathsf{dec}(m_2, k)} \;(\mathsf{C}) \qquad P_3 := \; T \vdash m_2$$

Les preuves P_1 et P_2 sont obtenues par application du symbole dec $\in \mathcal{VF}$. *Les preuves P_1 et P_3 sont des preuves par décomposition alors que P_2 est une preuve par composition.*

Notation : Étant donné un arbre de preuve P, on dénote par racine(P) le terme étiquetant la racine de P. Autrement dit, racine$(P) = u$ si P est un arbre de preuve de $T \vdash u$.

Lemme 5.8
Si P est une preuve par décomposition non réduite à une feuille, alors il existe $f \in \mathcal{VF}$ tel que $f(\mathsf{racine}(P_1), \dots, \mathsf{racine}(P_n)) \xrightarrow{\Lambda} \mathsf{racine}(P)$.

Démonstration.
Ce lemme est une conséquence directe du lemme 5.4 et de la définition de preuve par décomposition. □

Nous considérons la notion de sous-termes classique, définie au chapitre 2 (*cf.* Définition 2.1). Nous notons $Sts(t)$ les sous-termes stricts de t définis par $Sts(t) = St(t) \smallsetminus \{t\}$, et nous écrivons $St(T, u)$ au lieu de $St(T \cup \{u\})$. Nous montrons que le système $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$ est locale pour la notion de sous-termes syntaxiques.

Lemme 5.9 (localité)
Soient $T \subseteq \mathcal{T}(\mathcal{F})\!\downarrow$ et $u \in \mathcal{T}(\mathcal{F})\!\downarrow$. Soit P une preuve minimale de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Les nœuds de P sont étiquetés par des termes dans $St(T, u)$. Si de plus, P est une preuve par décomposition, alors les nœuds de P sont étiquetés par des termes dans $St(T)$.

Démonstration.
Soit P un arbre de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Nous montrons ces deux résultats par induction sur la structure de P.

Cas de base : Si P est réduit à une feuille, alors l'unique nœud de P est étiqueté par $T \vdash v$ avec $v \in T$.

Étape d'induction :

- Si P est une preuve par composition de $T \vdash u$ alors P est de la forme :

$$\frac{P_1 \left\{ \frac{\cdots}{T \vdash u_1} \right. \quad \cdots \quad P_n \left\{ \frac{\cdots}{T \vdash u_n} \right.}{T \vdash u} \ (\mathsf{C})$$

avec $u = f(u_1, \ldots, u_n)$ et $f \in \mathcal{VF}$. Par hypothèse d'induction, chacun des P_i $(1 \leq i \leq n)$ est un arbre dont les nœuds sont étiquetés par des termes dans $St(T, u_i)$. Nous en déduisons donc que les nœuds de P sont étiquetés par des termes dans $St(T, u)$.

- Si P est une preuve par décomposition non réduite à une feuille alors P est de la forme :

$$\frac{P_1 \left\{ \frac{\cdots}{T \vdash u_1} \right. \quad \cdots \quad P_n \left\{ \frac{\cdots}{T \vdash u_n} \right.}{T \vdash u} \ (\mathsf{C})$$

avec $f(u_1, \ldots, u_n) \xrightarrow{\Lambda} u$ et $f \in \mathcal{VF}$ (d'après le lemme 5.4). Nous raisonnons par cas, suivant la condition satisfaite par la règle $f(l_1, \ldots, l_n) \to r$ utilisée lors de la réécriture :

1. Si r est un terme clos alors $u = r \in T$, ce qui contredit la minimalité de P.
2. Sinon, r est une variable et il existe $i \leq n$ tel que $r \in vars(l_i)$. Dans ce cas, on a $u \in St(u_i)$. Soit $j \leq n$. Montrons que les nœuds de P_j sont étiquetés par des termes dans $St(T)$.

   (a) Si $\mathsf{head}(l_j) \in \mathcal{VF}$ alors, soit $l_j \in St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})\!\downarrow \subseteq T$, soit r est un sous-terme direct de l_i. Le premier cas est trivial. Pour le second, il suffit de remarquer que P_j ne peut pas être une preuve par composition par minimalité de P. Nous en déduisons alors que P_j est une preuve par décomposition et nous concluons en appliquant l'hypothèse d'induction.

(b) Si $\mathsf{head}(l_j) \in \mathcal{PF}$, alors P_j est nécessairement une preuve par décomposition. Nous concluons en appliquant l'hypothèse d'induction.

(c) Il nous reste à traiter le cas où l_j est une variable. Si $l_j = r$ alors $u = u_j$ et on obtient une contradiction avec le fait que P est minimale. Nous avons donc $l_j \neq r$, et nous distinguons deux cas : si $l_j \in vars(l_k)$ et l_k satisfait une des deux conditions (2a) ou (2b), alors $u_j \in St(u_k) \subseteq St(T)$. Sinon, l_j n'apparaît dans aucun des l_k non variables ($0 \leq k \leq n$), et P_j est nécessairement réduit à une feuille : autrement on obtiendrait une preuve P' de $T \vdash u$ plus petite que P en remplaçant chacun des sous-arbres $P_{j'}$ tel que $l_{j'} = l_j$ par une feuille étiquetée $T \vdash 0$. Nous en déduisons donc que $u_j \in St(T)$.

Nous venons de montrer que les nœuds de chacune des preuves P_j ($1 \leq j \leq n$) sont étiquetés par des termes dans $St(T)$, nous en déduisons que $u \in St(T)$, et que les nœuds de P sont étiquetés par des termes dans $St(T)$. $\square$

5.2.3 Déductibilité en un pas

Le problème de la déductibilité en un pas ne se pose pas puisque toutes les règles d'inférence du système $\mathcal{I}_{\mathcal{VF}}$ ont un nombre fixé de prémisses. Le problème de déductibilité en un pas d'un terme u à partir d'un ensemble de messages T est donc décidable en temps polynomial. Le degré du polynôme correspond en fait à l'arité maximale des symboles dans $\mathcal{VF}$.

Grâce au lemme de localité (lemme 5.9), nous en déduisons que le problème de déduction de l'intrus est décidable en temps polynomial par l'algorithme 4.1 (chapitre 4).

Théorème 5.10
Soit E *une théorie équationnelle publique-effondrante. Le problème de déduction de l'intrus est décidable en temps polynomial pour le système d'inférence* ($\mathcal{I}_{\mathcal{VF}}$, E).

5.3 Résolution de systèmes de contraintes bien formés

Le but de cette partie est de proposer une procédure générique pour décider la satisfaisabilité des systèmes de contraintes de déductions simples avec équations. Nous nous limiterons à l'étude des systèmes bien formés (notion introduite au chapitre 4).

Cette partie est dédiée à la preuve du théorème suivant :

Théorème 5.11
Soit E *une théorie équationnelle publique-effondrante. Le problème de la satisfaisabilité d'un système de contraintes de déduction simple et bien formé avec équations dans* ($\mathcal{I}_{\mathcal{VF}}$, E) *est NP-complet.*

Nous décrivons notre procédure dans la partie 5.3.1. Nous étudions ensuite la terminaison (*cf.* partie 5.3.2), la correction (*cf.* partie 5.3.3) et la complétude (*cf.* partie 5.3.4) de notre algorithme ainsi que sa complexité (*cf* parties 5.3.2 et 5.3.5). Enfin, nous illustrons notre procédure sur une variante du protocole de Denning-Sacco (*cf.* partie 5.3.6).

5.3.1 Procédure

Nous présentons à la figure 5.2 un ensemble de règles de transformation travaillant sur des triplets $(\mathcal{P};\mathcal{C};\mathcal{S})$ où :

- $\mathcal{P}$ est un système de contraintes simple avec équations,
- $\mathcal{C}$ est un ensemble de contraintes de déduction,
- $\mathcal{S}$ est un ensemble d'équations en forme résolue : $\mathcal{S}$ est de la forme $\{x_1 = t_1;\ \ldots;x_n = t_n\}$ et chacune des variable x_i $(1 \leq i \leq n)$ a une seule occurrence dans $\mathcal{S}$.

Nous pouvons associer une substitution $\{x_1 \mapsto t_1;\ \ldots;x_n \mapsto t_n\}$ à la troisième composante $\mathcal{S}$ du triplet. Par la suite, nous ne ferons pas de distinction entre $\mathcal{S}$ et la substitution qui lui est associée.

Définition 5.12 (solution d'un système $(\mathcal{P};\mathcal{C};\mathcal{S})$ dans $(\mathcal{I}_{\mathcal{VF}},\mathcal{R})$)
Une solution d'un triplet $(\mathcal{P};\mathcal{C};\mathcal{S})$ est une substitution σ telle que :

- *pour chaque contrainte $T \Vdash u \in \mathcal{P} \cup \mathcal{C}$, on a $T\sigma \vdash u\sigma$ dans $(\mathcal{I}_{\mathcal{VF}},\mathcal{R})$,*
- *pour chaque équation $s = t \in \mathcal{P}$, on a $s\sigma =_\mathsf{E} t\sigma$,*
- *pour chaque équation $s = t \in \mathcal{S}$, on a $s\sigma = t\sigma$.*

Nous montrons qu'étant donné un système de la forme $(\mathcal{B},\varnothing,\varnothing)$, où $\mathcal{B}$ est un système de contraintes simple et bien formé avec équations, la répétition non-déterministe des règles de transformation décrites à la figure 5.2 termine (*cf.* partie 5.3.2) et produit (sur au moins une branche de dérivation) un système en forme résolue, c'est-à-dire de la forme $(\varnothing,\varnothing,\mathcal{S})$ (de tels systèmes ont toujours une solution) si, et seulement si, $(\mathcal{B},\varnothing,\varnothing)$ a une solution (*cf.* parties 5.3.3 et 5.3.4). Nous obtenons ainsi une procédure non-déterministe, s'exécutant en temps polynomial, permettant de décider la satisfaisabilité de systèmes de contraintes simples et bien formés avec équations. Cette procédure est optimale puisque le problème est en fait NP-difficile (*cf.* partie 5.3.5).

Notation : Nous notons $\Rightarrow_\mathsf{S}, \ldots$ la relation binaire définie par l'application des règles (S), ... décrites à la figure 5.2. La relation $\Rightarrow$ dénote l'union de toutes ces relations et $\Rightarrow^*$ la fermeture réflexive et transitive de $\Rightarrow$.

5.3.2 Terminaison et Complexité

Dans cette partie, nous montrons que l'application répétée des règles de transformations décrites à la figure 5.2 fait décroître une mesure associée au système $(\mathcal{P};\mathcal{C};\mathcal{S})$. Cette mesure, notée $\|(\mathcal{P};\mathcal{C};\mathcal{S})\|_{DAG}$, est définie par :

$$\|(\mathcal{P};\mathcal{C};\mathcal{S})\|_{DAG} = \|\mathcal{P}\| + \|\mathcal{C} \cup \mathcal{S}\|_{DAG}$$

où $\|\mathcal{P}\|$ représente la taille de l'ensemble $\mathcal{P}$, et $\|\mathcal{C} \cup \mathcal{S}\|_{DAG}$ représente la taille DAG de l'ensemble $\mathcal{C} \cup \mathcal{S}$.

$$\frac{\mathcal{P} \cup \{e[u]\};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P} \cup \{e[r]\};\ \mathcal{C}\eta;\ \mathcal{S}\eta \cup \eta}\ (\mathsf{S})$$

Surréduction
e est une équation
ou une contrainte de déduction,
$u \notin \mathcal{X}$,
$l \rightarrow r \in \mathcal{R}$,
$\eta = \mathsf{mgu}(l\mathcal{S}, u\mathcal{S})$,
$\mathsf{head}(l) = \mathsf{head}(u)$.

$$\frac{\mathcal{P} \cup \{t_1 = t_2\};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C}\eta;\ \mathcal{S}\eta \cup \eta}\ (\mathsf{U})$$

Unification Syntaxique
$\eta = \mathsf{mgu}(t_1\mathcal{S}, t_2\mathcal{S})$.

$$\frac{\mathcal{P} \cup \{c\};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C} \cup \{c\mathcal{S}\};\ \mathcal{S}}\ (\mathsf{B})$$

Blocage
c est une contrainte de déduction.

$$\frac{\mathcal{P};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C}\{x \mapsto t\};\ \mathcal{S}\{x \mapsto t\} \cup \{x \mapsto t\}}\ (\mathsf{EV})$$

Élimination d'une Variable
$x \in vars(\mathcal{C})$,
$t \in St(\mathcal{C}) \smallsetminus vars(\mathcal{C})$,
il n'y a pas d'occurrence de x dans t.

$$\frac{\mathcal{P};\ \mathcal{C} \cup \{T \Vdash u\};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C};\ \mathcal{S}}\ (\mathsf{Cl})$$

Contrainte Close
$T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$,
$T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$

Figure 5.2 - Règles de transformation.

Proposition 5.13 (Terminaison)
Nous avons :

1. *La longueur d'une suite de transformations issue de $(\mathcal{P}_0; \varnothing; \varnothing)$ est bornée par un polynôme en $\|\mathcal{P}_0\|$ et $\|\mathcal{R}\|$.*
2. *Soit $(\mathcal{P}; \mathcal{C}; \mathcal{S})$ un système. L'ensemble $\{(\mathcal{P}'; \mathcal{C}'; \mathcal{S}') \mid (\mathcal{P}; \mathcal{C}; \mathcal{S}) \Rightarrow (\mathcal{P}'; \mathcal{C}'; \mathcal{S}')\}$ est fini.*
3. *Soit $(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)$ un système tel que $(\mathcal{P}_0; \varnothing; \varnothing) \Rightarrow^* (\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)$. La taille DAG du système $(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)$, c'est-à-dire la valeur $\|\mathcal{P}_i\| + \|\mathcal{C}_i \cup \mathcal{S}_i\|_{DAG}$, est bornée par un polynôme en $\|\mathcal{P}_0\|$ et $\|\mathcal{R}\|$.*

Démonstration.
La mesure de complexité que nous associons à un système $(\mathcal{P}; \mathcal{C}; \mathcal{S})$ est un quadruplet dont les composantes sont les suivantes :

1. $|\mathcal{P}|$, le nombre de contraintes de déduction et d'équations dans $\mathcal{P}$,
2. $nb(\mathcal{P})$, le nombre de termes dans $St(\mathcal{P})$ unifiables avec un membre gauche de $\mathcal{R}$,
3. $|vars(\mathcal{C})|$, le nombre de variables dans $\mathcal{C}$,
4. $|\mathcal{C}|$, le nombre de contraintes de déduction dans $\mathcal{C}$.

Nous considérons l'ordre lexicographique sur ces quadruplets. Nous devons montrer que l'application d'une des règles de transformation réduit la mesure de complexité. Regardons les différentes règles :

- (B) et (U) réduisent $|\mathcal{P}|$,
- (S) réduit $nb(\mathcal{P})$ et laisse la mesure $|\mathcal{P}|$ inchangée,
- (EV) élimine une variable et réduit donc la mesure $|vars(\mathcal{C})|$ en laissant les mesures concernant $\mathcal{P}$ inchangées,
- (Cl) décroît $|\mathcal{C}|$ en laissant les autres composantes inchangées.

Toutes ces observations sont résumées dans le tableau suivant :

	(B)	(U)	(S)	(EV)	(Cl)
$\|\mathcal{P}\|$	$<$	$<$	$=$	$=$	$=$
$nb(\mathcal{P})$			$<$	$=$	$=$
$\|vars(\mathcal{C})\|$				$<$	$=$
$\|\mathcal{C}\|$					$<$

Nous pouvons maintenant montrer les trois résultats énoncés dans la proposition 5.13.

1. Considérons une suite $(\mathcal{P}_0; \varnothing; \varnothing) \Rightarrow \ldots \Rightarrow (\mathcal{P}_n; \mathcal{C}_n; \mathcal{S}_n)$. La valeur de chacune des quatre composantes de la mesure de complexité associée à ces systèmes peut être bornée par une valeur M polynomiale en $\|\mathcal{P}_0\|$ et $\|\mathcal{R}\|$. Ce résultat est trivial pour les deux premières composantes de la mesure puisqu'aucune règle ne peut augmenter la valeur de ces composantes. De même, la valeur de $|\mathcal{C}_i|$ ne peut pas être plus grande que $|\mathcal{P}_0|$. Intéressons-nous à la composante $|vars(\mathcal{C})|$. De nouvelles variables peuvent être introduites par application de la règle (S), mais pas plus de $\|\mathcal{R}\|$ à chaque application de la règle. De plus, il ne peut pas y avoir plus de $\|\mathcal{P}_0\|$ applications de la règle (S) dans la séquence considérée. Ainsi, la valeur de $|vars(\mathcal{C}_i)|$ est bornée par un polynôme en $\|\mathcal{P}_0\|$ et $\|\mathcal{R}\|$.

Notons k_i l'entier en base M donné par les valeurs des quatre composantes de la mesure associée au système $(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)$ (c'est-à-dire $k_i = |\mathcal{P}_i|.M^3 + nb(\mathcal{P}_i).M^2 + |vars(\mathcal{C}_i)|.M + |\mathcal{C}_i|$). Nous avons $k_{i+1} < k_i$ pour tout i tel que $0 \leq i < n$. La longueur n d'une telle suite est donc bornée par un polynôme en $\|\mathcal{P}_0\|$ et $\|\mathcal{R}\|$.

2. Soit $(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)$ un système. Le nombre de successeurs de $(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)$ par $\Rightarrow$ est inférieur à $\|\mathcal{R}\|.\|(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)\|_{DAG}$, et est donc fini.

3. Enfin, il existe une valeur N polynomiale en $\|\mathcal{R}\|$ telle que si $(\mathcal{P}; \mathcal{C}; \mathcal{S}) \Rightarrow (\mathcal{P}'; \mathcal{C}'; \mathcal{S}')$ alors $\|(\mathcal{P}; \mathcal{C}; \mathcal{S})\|_{DAG} + N \leq \|(\mathcal{P}'; \mathcal{C}'; \mathcal{S}')\|_{DAG}$. Ainsi, $\|(\mathcal{P}_i; \mathcal{C}_i; \mathcal{S}_i)\|_{DAG}$ est borné par un polynôme en $\|\mathcal{P}_0\|$ et $\|\mathcal{R}\|$. □

5.3.3 Correction

La proposition 5.14 a pour but d'établir la correction des règles de transformation décrites à la figure 5.2

Proposition 5.14 (Correction)
Considérons un système de la forme $(\mathcal{P}; \varnothing; \varnothing)$ où $\mathcal{P}$ est un ensemble de contraintes de déduction simple avec équations.

Si $(\mathcal{P}; \varnothing; \varnothing) \Rightarrow^* (\varnothing; \varnothing; \mathcal{S})$ alors $(\mathcal{P}; \varnothing; \varnothing)$ a une solution.

Démonstration.
Ce résultat se montre par induction sur la longueur de la dérivation. En fait, il est suffisant de montrer (pour chacune des règles de transformation (R)) que si $(\mathcal{P}_1; \mathcal{C}_1; \mathcal{S}_1) \Rightarrow_{\mathsf{R}} (\mathcal{P}_2; \mathcal{C}_2; \mathcal{S}_2)$ et si σ est une solution de $(\mathcal{P}_2; \mathcal{C}_2; \mathcal{S}_2)$, alors σ est aussi une solution de $(\mathcal{P}_1; \mathcal{C}_1; \mathcal{S}_1)$.

Règle (B) :
: Supposons que $(\mathcal{P} \cup \{c\}; \mathcal{C}; \mathcal{S}) \Rightarrow_{\mathsf{B}} (\mathcal{P}; \mathcal{C} \cup \{c\mathcal{S}\}; \mathcal{S})$ et que σ soit une solution de $(\mathcal{P}; \mathcal{C} \cup \{c\mathcal{S}\}; \mathcal{S})$. Nous allons montrer que σ est aussi une solution de $(\mathcal{P} \cup \{c\}; \mathcal{C}; \mathcal{S})$. Posons $c = T \Vdash x$. La seule difficulté consiste à montrer que $x\sigma$ est déductible de $T\sigma$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Par hypothèse, σ est une solution de $c\mathcal{S}$, il existe donc une preuve P de $(T\mathcal{S})\sigma \vdash (x\mathcal{S})\sigma$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Or, $(T\mathcal{S})\sigma = T\sigma$ et $(x\mathcal{S})\sigma = x\sigma$. Nous en déduisons donc que P est une preuve de $T\sigma \vdash x\sigma$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$ et que σ est une solution de $(\mathcal{P} \cup \{c\}; \mathcal{C}; \mathcal{S})$.

Règle (U) :
: Supposons que $(\mathcal{P} \cup \{t_1 = t_2\}; \mathcal{C}; \mathcal{S}) \Rightarrow_{\mathsf{U}} (\mathcal{P}; \mathcal{C}\eta; \mathcal{S}\eta \cup \eta)$, avec $\eta = \mathsf{mgu}(t_1\mathcal{S}, t_2\mathcal{S})$, et que σ soit une solution de $(\mathcal{P}; \mathcal{C}\eta; \mathcal{S}\eta \cup \eta)$. Nous allons montrer que σ est une solution de $(\mathcal{P} \cup \{t_1 = t_2\}; \mathcal{C}; \mathcal{S})$. Par hypothèse, σ est solution de toutes les équations et contraintes de déduction dans $\mathcal{P}$. De plus, σ est un unificateur de $\mathcal{S}\eta \cup \eta$. Nous en déduisons que σ est solution de toutes les équations et contraintes de déduction dans $\mathcal{C} \cup \mathcal{S}$. Pour finir, nous avons $t_1\eta = t_2\eta$, c'est à dire que $(t_1\eta)\sigma = (t_2\eta)\sigma$, et donc $t_1\sigma = t_2\sigma$, ce qui nous permet de conclure pour ce cas.

Règle (S) :
: Supposons que $(\mathcal{P} \cup \{e[u]\}; \mathcal{C}; \mathcal{S}) \Rightarrow_{\mathsf{S}} (\mathcal{P} \cup \{e[r]\}; \mathcal{C}\eta; \mathcal{S}\eta \cup \eta)$, où $\eta = \mathsf{mgu}(l\mathcal{S}, u\mathcal{S})$, et que σ soit une solution de $(\mathcal{P} \cup \{e[r]\}; \mathcal{C}\eta; \mathcal{S}\eta \cup \eta)$. Montrons que σ est une solution de $(\mathcal{P} \cup \{e[u]\}; \mathcal{C}; \mathcal{S})$. Comme dans le cas précédent, nous pouvons montrer que σ est une

solution de $(\mathcal{P};\mathcal{C};\mathcal{S})$. Considérons maintenant l'équation ou la contrainte de déduction e, et plus particulièrement le terme $t[u]$ sur lequel l'étape de surréduction a eu lieu. La substitution σ est telle que $t[u] =_{\mathsf{E}} t[r]$, et puisque σ est une solution de $e[r]$, nous en déduisons que σ est aussi une solution de $e[u]$.

Règle (EV) :

Supposons que $(\mathcal{P};\mathcal{C};\mathcal{S}) \Rightarrow_{\mathsf{EV}} (\mathcal{P};\mathcal{C}\{x \mapsto t\};\mathcal{S}\{x \mapsto t\} \cup \{x \mapsto t\})$ et que σ soit une solution de $(\mathcal{P};\mathcal{C}\{x \mapsto t\};\mathcal{S}\{x \mapsto t\} \cup \{x \mapsto t\})$. Nous allons montrer que σ est une solution de $(\mathcal{P};\mathcal{C};\mathcal{S})$. Comme dans le cas de la règle (U), nous avons $u\sigma = u\{x \mapsto t\}\sigma$ pour chacun des termes u. Nous en déduisons donc que σ est une solution de $(\mathcal{P};\mathcal{C};\mathcal{S})$.

Règle(Cl) :

Supposons que $(\mathcal{P};\mathcal{C} \cup \{T \Vdash u\};\mathcal{S}) \Rightarrow_{\mathsf{Cl}} (\mathcal{P};\mathcal{C};\mathcal{S})$, que les termes dans $T \Vdash u$ soient clos et que u soit déductible de T. Par hypothèse, σ est solution de $T \Vdash u$ et de $(\mathcal{P};\mathcal{C};\mathcal{S})$. D'où le résultat. □

5.3.4 Complétude

Le plus difficile est de montrer la complétude de notre procédure. Cette partie est consacrée à la preuve de ce résultat (proposition 5.21). Nous commençons par établir deux lemmes techniques (lemmes 5.15 et 5.16) utilisés dans la preuve de la proposition 5.18. Cette proposition est fondamentale pour montrer la complétude de la règle (EV) et établir la complétude de notre procédure.

Lemme 5.15
Soient $T \subseteq \mathcal{T}(\mathcal{F})\!\downarrow$ et $u \in \mathcal{T}(\mathcal{F})\!\downarrow$ tels que $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Soit $v \in St(u)$ tel que $v \notin St(T)$. Il existe une preuve par composition de $T \vdash v$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$.

Démonstration.
Soit P une preuve minimale de $T \vdash u$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Si P contient un nœud étiqueté par $T \vdash v$, alors il s'agit de la racine de la preuve de $T \vdash v$ que nous recherchons. Cette preuve est nécessairement une preuve par composition : sinon nous aurions $v \in St(T)$ d'après le lemme de localité (lemme 5.9), ce qui contredirait une des hypothèses.

Nous allons montrer que P contient nécessairement un nœud étiqueté par $T \vdash v$. Supposons que P ne contient pas de tel nœud. Nous construisons récursivement un chemin dans P, de la racine jusqu'à une feuille, tel que chacun des nœuds de ce chemin soit étiqueté par un terme t satisfaisant la condition $v \in Sts(t)$, et nous montrons que l'existence d'un tel chemin aboutit à une contradiction.

Par hypothèse, v est un sous-terme strict de u. Supposons que cette condition soit vraie pour tous les nœuds d'un chemin $(nd_0, \ldots, nd_k)$ dans P. Chaque nœud nd_i est étiqueté par $T \vdash t_i$ et nous avons $t_0 = u$. Ainsi, par hypothèse de récurrence, nous savons que $v \in Sts(t_0), \ldots, v \in Sts(t_k)$. Considérons les fils du dernier nœud nd_k de ce chemin :

- si nd_k est une feuille, alors $t_k \in T$ par définition d'un arbre de preuve. De plus, nous savons que $v \in Sts(t_k)$, ce qui contredit le fait que $v \notin St(T)$.
- si nd_k a n fils $P_1, \ldots, P_n$, l'inférence que nous considérons est de la forme :

$$\frac{P_1\left\{\ \frac{\cdots}{T \vdash u_1}\right. \quad \cdots \quad P_n\left\{\ \frac{\cdots}{T \vdash u_n}\right.}{T \vdash t_k}\ (\mathsf{C})$$

avec $t_k = f(u_1, \ldots, u_n)\downarrow$ et $f \in \mathcal{VF}$. Nous distinguons deux cas (*cf.* lemme 5.4) :

- si $t_k = f(u_1, \ldots, u_n)$, alors puisque nous savons que $v \in Sts(t_k)$, nous en déduisons qu'il existe i tel que $1 \leq i \leq n$ et $v \in St(u_i)$. Notons i_0 un entier satisfaisant cette condition. Puisque v n'étiquette aucun nœud de P, nous savons que $v \in Sts(u_{i_0})$ La racine de P_{i_0} est le nœud suivant de notre chemin.
- si $f(u_1, \ldots, u_n) \xrightarrow{\Lambda} t_k$, alors soit il existe i tel que $1 \leq i \leq n$ et $t_k \in St(u_i)$, soit $t_k \in St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})\downarrow$ (car $\mathcal{R}$ est un système de réécriture public-effondrant). Dans le premier cas, notons i_0 un des entiers satisfaisant la condition $t_k \in St(u_{i_0})$, nous en déduisons que $v \in Sts(u_{i_0})$ et que la racine de P_{i_0} est le nœud suivant de notre chemin. Dans le deuxième cas, nous avons $v \in St(T)$ (d'après la remarque écrite à la page 78) et nous obtenons une contradiction. □

Notre but est d'assurer l'existence de preuves pas trop « grosses ». Pour cela, nous serons amenés à appliquer des transformations, en particulier le remplacement de tous les occurrences d'un terme trop « gros » par la constante spéciale 0. Le lemme énoncé ci-dessous établit des conditions suffisantes permettant l'application d'un tel remplacement sans « détruire » la règle d'inférence.

Lemme 5.16
Soit $g(v_1, .., v_k) \in \mathcal{T}(\mathcal{F})\downarrow$ tel que $g(v_1, .., v_k) \notin St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})\downarrow$ et $g \in \mathcal{VF}$. Soient $u_1, \ldots, u_n \in \mathcal{T}(\mathcal{F})\downarrow$, $f \in \mathcal{VF}$ et δ le remplacement $[g(v_1, \ldots, v_k) \mapsto 0]$.
Si $f(u_1, \ldots, u_n)\downarrow \neq g(v_1, .., v_k)$ et pour tout $i \leq k$, on a $f(u_1, \ldots, u_n)\downarrow \neq v_i$, alors :
$$(f(u_1, \ldots, u_n)\downarrow)\delta = f(u_1\delta, \ldots, u_n\delta)\downarrow.$$

Exemple 5.17
On considère la théorie équationnelle $\mathsf{E} = \{\mathsf{dec}(\{x\}_y, y) = x\}$, *les symboles* $\mathsf{dec}(\cdot,\cdot)$ *et* $\{\cdot\}.$ *sont publics. Considérons l'inférence suivante, obtenue par application du symbole* dec.

$$\frac{T \vdash \{a\}_{\langle b,c\rangle} \quad T \vdash \langle b, c\rangle}{T \vdash a}\,(\mathsf{C})$$

Nous pouvons appliquer le remplacement $\delta_1 = [\langle b, c\rangle \mapsto 0]$ sur cette inférence, nous obtenons encore une instance de la règle d'inférence (C) *avec utilisation du symbole* dec. *En fait, les conditions du lemme 5.16 sont satisfaites. On a $g(v_1, \ldots, v_k) = \langle b, c\rangle$, $f = \mathsf{dec}(\cdot,\cdot)$ et* $\mathsf{dec}(\{a\}_{\langle b,c\rangle}, \langle b, c\rangle)\downarrow = a$ *n'est égal à aucun des des termes $\langle b, c\rangle, b$ et c.*

En revanche, l'application du remplacement $\delta_2 = [\{a\}_{\langle b,c\rangle} \mapsto 0]$ ne permet pas de garder une instance de la règle d'inférence. Ce remplacement ne satisfait pas les conditions du lemme 5.16.

Démonstration. (du lemme 5.16)
Grâce au lemme 5.4, nous avons seulement deux cas à considérer :

1. $f(u_1, \ldots, u_n)\downarrow = f(u_1, \ldots, u_n)$.
   Dans ce cas, nous avons $(f(u_1, \ldots, u_n)\downarrow)\delta = f(u_1, \ldots, u_n)\delta = f(u_1\delta, \ldots, u_n\delta)$ puisque nous savons que $v \neq f(u_1, \ldots, u_n)$.
2. $f(u_1, \ldots, u_n) \xrightarrow{\Lambda} u$ avec une règle de réécriture $f(l_1, ..., l_n) \to r \in \mathcal{R}$.
   Posons $v = g(v_1, \ldots, v_k)$ et notons $\mathcal{O}_v$ l'ensemble de toutes les occurrences de v dans le terme $f(u_1, \ldots, u_n)$.

(a) Si $\Lambda \in \mathcal{O}_v$, alors v n'est pas en forme normale, ce qui contredit une des hypothèses.

(b) Si, pour tout $p \in \mathcal{O}_v$, il existe un préfixe p' de p tel que $f(l_1, ..., l_n)|p' \in \mathcal{X}$ alors nous avons $f(u_1\delta, \ldots, u_n\delta) \xrightarrow{\Lambda} u\delta$ en utilisant la même règle que précédemment.

(c) Autrement, il existe $p \in \mathcal{O}_v$ tel que $f(l_1, ..., l_n)|p \notin \mathcal{X}$, et nous avons $p \neq \Lambda$. Posons $p = i.p'$. Le cas $l_i|p' \in \mathcal{X}$ est impossible par le choix de p, et le cas $l_i|p' \in \mathcal{PF}$ est impossible puisque head$(v) \in \mathcal{VF}$. Finalement, si $l_i|p' \in \mathcal{VF}$, alors soit il existe j tel que $l_i|p'.j = r$ et u est égal à v_j, ce qui contredit les hypothèses, soit $l_i|p' \in St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})\downarrow$, ce qui signifie que $v \in St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})\downarrow$, et aboutit également à une contradiction. □

L'objectif de la proposition 5.18 est d'établir qu'un système de contraintes satisfaisable, obtenu par applications répétées des règles (S), (U) et (B), admet une solution n'introduisant pas de « nouvelles structures ». Formellement, un système $\mathcal{C}$ satisfaisable admet une solution σ ayant la propriété suivante :

$$\text{pour tout } x \in vars(\mathcal{C}), \text{ il existe } t \in St(\mathcal{C}) \smallsetminus vars(\mathcal{C}) \text{ tel que } t\sigma = x\sigma.$$

Tout d'abord, il est important de remarquer que le système $\mathcal{C}$ que l'on considère ne contient plus d'équations : ces dernières ont été éliminées par applications répétées de la règle (U). D'autre part, $\mathcal{C}$ est un système de contraintes de déduction, mais ces contraintes ne sont pas nécessairement simples à cause de la substitution appliquée sur ces contraintes lors de l'utilisation de la règle (B). Cependant, les systèmes obtenus par applications répétées des règles (S), (U) et (B) sont monotones (*cf.* Définition 4.15) et satisfont la propriété d'initialisation (*cf.* Définition 4.16). Nous utilisons donc cette hypothèse pour établir la preuve de la proposition 5.18.

Soit $\prec$ un ordre bien fondé sur les termes et tel que la constante 0 soit minimale pour cet ordre. Nous notons $\ll$ l'extension multi-ensemble de $\prec$ sur les termes clos et nous étendons cette notation aux substitutions : $\sigma_1 \ll \sigma_2$ si $\{x\sigma_1 | \ x \in dom(\sigma_1)\} \ll \{x\sigma_2 | \ x \in dom(\sigma_2)\}$.

Proposition 5.18
Soit $\mathcal{C}$ un système de contraintes de déduction monotone et satisfaisant la propriété d'initialisation. Soit σ une solution minimale (pour $\ll$) de $\mathcal{C}$ telle que $\mathcal{C}\sigma = \mathcal{C}\sigma\downarrow$. Pour tout $x \in vars(\mathcal{C})$, il existe $t \in St(\mathcal{C}) \smallsetminus vars(\mathcal{C})$ tel que $t\sigma = x\sigma$.

Démonstration.
Soit $\mathcal{C} = (C_1, \ldots, C_\ell)$ un système de contraintes monotone et satisfaisant la propriété d'initialisation. Posons $C_i = T_i \Vdash u_i$ pour chaque $i \leq \ell$.

Nous raisonnons par contradiction. Supposons qu'il existe $x \in vars(\mathcal{C})$ tel que, pour tout $t \in St(\mathcal{C}) \smallsetminus vars(\mathcal{C})$, on ait $t\sigma \neq x\sigma$. Soit δ le remplacement $[x\sigma \mapsto 0]$, et $\sigma' = \sigma\delta$. Nous allons montrer que σ' est aussi une solution de $\mathcal{C}$, ce qui contredit la minimalité de σ puisque $x\sigma \neq 0$. Remarquons tout d'abord que :

Fait 1 *Pour tout $t \in \mathcal{C}$, on a $t(\sigma\delta) = (t\sigma)\delta$.*

Ce résultat est une conséquence directe du fait que pour tout $t \in St(C) \smallsetminus \mathcal{X}$, on a $t\sigma \neq x\sigma$.

Nous allons maintenant établir le fait suivant :

Fait 2 *Soit $i \leq \ell$ et $t \in T_i$. Si $x\sigma \in St(t\sigma)$, alors il existe $j < i$ tel que $x\sigma \in St(u_j\sigma)$.*

Soit $i \leq \ell$ et $t \in T_i$. Supposons que $x\sigma \in St(t\sigma)$. Puisque $x\sigma \neq t\sigma$ pour tout $t \in St(\mathcal{C}) \setminus \mathcal{X}$, il existe $y \in vars(T_i)$ tel que $x\sigma \in St(y\sigma)$. Puisque $\mathcal{C}$ satisfait la propriété d'initialisation, nous savons qu'il existe $j < i$ tel que $y \in vars(u_j)$, et donc que $x\sigma \in St(u_j\sigma)$.

Posons $m = \min\{j \mid x\sigma \in St(u_j\sigma)\}$.

Fait 3 *Il existe une preuve par composition de $T_m\sigma \vdash x\sigma$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$.*

Nous savons qu'il existe une preuve P_m de $T_m\sigma \vdash u_m\sigma$ et, par définition de m, que $x\sigma \in St(u_m\sigma)$ et que $x\sigma \notin St(T_m\sigma)$. Nous pouvons donc appliquer le lemme 5.15. Nous en déduisons qu'il existe une preuve par composition P_x de $T_m\sigma \vdash x\sigma$.

Maintenant, nous allons montrer que $\sigma' = \sigma\delta$ est une solution de $\mathcal{C}$, aboutissant ainsi à une contradiction puisque $\sigma' \ll \sigma$. Pour montrer cela, nous devons construire pour tout $i \leq \ell$, une preuve de $C_i\sigma'$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Nous distinguons deux cas :

1. Cas $i < m$: Par définition de m et par le fait 2, nous savons que $x\sigma \notin St(C_i\sigma)$. Ainsi, grâce au fait 1, nous avons $C_i\sigma' = (C_i\sigma)\delta = C_i\sigma$: σ' est une solution de C_i.
2. Cas $i \geq m$: Le reste de la preuve est consacrée à montrer que σ' est aussi une solution de $T_i \Vdash u_i$.

Nous savons que σ est une solution de C_i, cela signifie qu'il existe une preuve P_i de $T_i\sigma \vdash u_i\sigma$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. D'autre part, le fait 3 et la monotonie du système de contraintes $\mathcal{C}$, nous assure l'existence d'une preuve P_x^i de $T_i\sigma \vdash x\sigma$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$. Posons $x\sigma = f(u_1, ..., u_n)$ (avec $f \in \mathcal{VF}$). Les sous-preuves directes de P_x^i, que nous noterons $P_{x,1}^i, \ldots, P_{x,n}^i$ sont des preuves de $T_i\sigma \vdash v_1, \ldots, T_i\sigma \vdash v_n$.

Nous remplaçons dans P_i chacun des sous-arbres dont la racine est étiquetée par v_j $(j \leq n)$ par le sous-arbre $P_{x,j}^i$. Nous remplaçons ensuite chacun des sous-arbres dont la racine est étiquetée par $T_i\sigma \vdash x\sigma$ par une feuille étiquetée par le terme $T_i\sigma \vdash 0$. Ensuite, nous appliquons le remplacement δ sur chacune des étiquettes de l'arbre ainsi obtenu et nous notons P_i' l'arbre ainsi obtenu.

Fait 4 *P_i' est une preuve de $(T_i\sigma)\delta \vdash (u_i\sigma)\delta$ (c'est-à-dire de $T_i\sigma' \vdash u_i\sigma'$) dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$.*

Remarquons que $x\sigma \notin St(\mathcal{R}) \cap \mathcal{T}(\mathcal{VF})\downarrow$ et que, grâce au lemme 5.16, les nœuds internes de P_i' satisfont les conditions de la définition d'un arbre de preuve. De plus, la racine de P_i' est étiquetée par $T_i\sigma' \vdash u_i\sigma'$. Finalement, les feuilles sont étiquetées

- soit par $T_i\sigma\delta \vdash 0$, c'est-à-dire $T_i\sigma' \vdash 0$, et dans ce cas nous avons $0 \in T_i\sigma\delta$,
- soit par $T_i\sigma\delta \vdash t'$ avec t' distinct de chacun des v_j $(i \leq n)$, et dans ce cas, par construction de P_i', nous avons $t' = t\delta$ pour une feuille étiquetée $T_i\sigma \vdash t$ de P_i ou de $P_{x,j}^i$ $(1 \leq j \leq n)$. Nous avons alors $t\delta \in (T_i\sigma)\delta = T_i(\sigma\delta) = T_i\sigma'$

Nous en déduisons que P_i' est une preuve de $T_i\sigma' \vdash u_i\sigma'$ dans $(\mathcal{I}_{\mathcal{VF}}, \mathcal{R})$, c'est-à-dire que σ' est une solution de C_i.

Finalement σ' est bien une solution du système de contraintes $\mathcal{C}$ plus petite que σ. $\square$

Nous allons maintenant pouvoir établir la complétude de notre procédure. Pour cela, nous allons montrer que la stratégie consistant :

1. à effectuer toutes les étapes de surréduction (règle (S)),
2. à éliminer les équations par applications répétées de la règle (U), puis
3. à appliquer la règle (B) sur chacune des contraintes,

nous permet d'obtenir un système $(\emptyset; \mathcal{C}; \mathcal{S})$ tel que $\mathcal{CS}$ est un système de contraintes de déduction monotone satisfaisant la propriété d'initialisation.

Pour cela, nous somme amenés à appliquer des étapes de surréduction (c'est-à-dire application d'une substitution suivi d'une étape de normalisation) sur le système de contraintes de départ. Or, la propriété d'inialisation n'est pas stable par application de substitutions entraînant des étapes de normalisation.

Exemple 5.19
Considérons la théorie équationnelle E_{DY} *et le système de contraintes de déduction* $\mathcal{C}$ *suivant :*

$$\mathcal{C} = \left\{ \begin{array}{rcl} 0 & \Vdash & \mathsf{dec}(x_1, x_2) \\ 0, x_2 & \Vdash & s \end{array} \right.$$

Le système $\mathcal{C}$ *satisfait la propriété d'initialisation. En effet, la seule variable apparaissant à gauche d'une contrainte est la variable* x_2 *et cette variable apparaît dans la première contrainte à droite. Soit* $\sigma = \{x_1 \mapsto \{a\}_{x_2}\}$*. Nous avons :*

$$\mathcal{C}\sigma\!\downarrow = \left\{ \begin{array}{rcl} 0 & \Vdash & a \\ 0, x_2 & \Vdash & s \end{array} \right.$$

Le système $\mathcal{C}\sigma\!\downarrow$ *ne satisfait par la propriété d'initialisation.*

Nous allons établir une condition suffisante sur σ permettant d'assurer la stabilité de la propriété d'initialisation lors de l'application d'une substitution. Dans le lemme suivant, seule la convergence du système de réécriture $\mathcal{R}$ est nécessaire.

Lemme 5.20
Soit $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_k \Vdash u_k\}$ *un système de contraintes de déduction tel que :*

- $\mathcal{C}$ *est monotone*
- $\mathcal{C}$ *satisfait la propriété d'initialisation*

Soit σ *une substitution telle que pour tout* $i \leq k$*, le terme* $u_i\sigma$ *est un terme en forme normale. Le système* $\mathcal{C}\sigma\!\downarrow$ *est un système monotone satisfaisant la propriété d'initialisation.*

Démonstration.
Tout d'abord, la propriété de monotonie est stable par application d'une substitution. Regardons ce qu'il en est de la propriété d'initialisation. Soit $i \leq k$ et $x \in vars(T_i\sigma)$ (si $x \in vars(T_i\sigma\!\downarrow)$ alors $x \in vars(T_i\sigma)$). Nous distinguons deux cas :

- soit $x \in vars(T_i)$ et $x \notin dom(\sigma)$. Dans ce cas, par hypothèse, il existe $j \leq i$ tel que $x \in vars(u_j)$. On a donc $x \in vars(u_j\sigma)$ puisque $x \notin dom(\sigma)$,
- ou alors il existe $y \in vars(T_i)$ tel que $x \in vars(y\sigma)$. Par hypothèse, il existe $j < i$ tel que $y \in vars(u_j)$. On a donc $y\sigma \in St(u_j\sigma)$. Nous en déduisons donc que $x \in vars(u_j\sigma)$.

Dans les deux cas, nous avons montré qu'il existe j tel que $x \in vars(u_j\sigma)$. Puisque, par hypothèse, $u_j\sigma\!\downarrow = u_j\sigma$, nous en déduisons que $x \in vars(u_j\sigma\!\downarrow)$. □

Nous établissons maintenant le résultat principal de cette partie :

Proposition 5.21 (Complétude)
Soit $\mathcal{B}$ un système de contraintes de déduction simple et bien formé avec équations. Si $(\mathcal{B};\emptyset;\emptyset)$ a une solution, alors il existe une suite de réductions de la forme $(\mathcal{B};\emptyset;\emptyset) \Rightarrow^* (\emptyset;\emptyset;\mathcal{S})$.

Démonstration.
Nous montrons par induction que s'il existe une solution σ normalisée de $(\mathcal{P};\mathcal{C};\mathcal{S}')$ telle que :

(a) le système $\mathcal{P}\mathcal{S}'\!\downarrow \cup\, \mathcal{C}$ est monotone et satisfait la propriété d'initialisation,

(b) les termes dans $\mathcal{C}\sigma$ sont en formes normales,

alors il existe une suite de réductions de la forme $(\mathcal{P};\mathcal{C};\mathcal{S}') \Rightarrow^* (\emptyset;\emptyset;\mathcal{S})$.

Le cas de base $(\emptyset;\emptyset;\mathcal{S}')$ est trivial. Pour l'étape d'induction, nous supposons que σ est une solution minimale (pour $\ll$) vérifiant les conditions (a) et (b) ci-dessus et nous montrons qu'une règle de transformation de la figure 5.2 s'applique, et que le système ainsi obtenu a encore une solution σ' normalisée vérifiant les conditions (a) et (b).

Cas 1 : $\mathcal{P}$ contient (au moins) une équation. Posons $\mathcal{P} = \{t_1 = t_2\} \cup \mathcal{P}'$.

Cas 1.1 : $t_1\sigma, t_2\sigma$ sont en formes normales.
Le système de réécriture $\mathcal{R}$ est convergent, nous avons donc $t_1\sigma = t_2\sigma$: les termes t_1 et t_2 sont unifiables. Nous pouvons donc appliquer la règle (U). Nous obtenons un système plus petit ayant le même ensemble de variables. Posons $\eta = \mathsf{mgu}(t_1\mathcal{S}', t_2\mathcal{S}')$; nous avons :

$$(\mathcal{P}' \cup \{t_1 = t_2\};\mathcal{C};\mathcal{S}') \Rightarrow_{\mathsf{U}} (\mathcal{P}';\mathcal{C}\eta;\mathcal{S}'\eta \cup \eta).$$

Nous pouvons montrer (par induction structurelle) que pour tout terme $u \in \mathcal{C}$, on a $u\sigma = (u\eta)\sigma$. Nous en déduisons donc que $(\mathcal{C}\eta)\sigma = \mathcal{C}\sigma$ et il s'ensuit que σ est solution du système obtenu et que la condition (b) est satisfaite. Nous devons vérifier que la condition (a) est satisfaite.

Concernant la condition (a), il suffit de remarquer que les contraintes de déduction dans $\mathcal{P}'(\mathcal{S}'\eta \cup \eta)\!\downarrow \cup\, \mathcal{C}\eta$ sont les mêmes que celles dans $(\mathcal{P}\mathcal{S}')\eta\!\downarrow \cup\, \mathcal{C}\eta$ et que celles dans $((\mathcal{P}\mathcal{S}')\!\downarrow \cup\, \mathcal{C})\eta\!\downarrow$, puisque les termes de $\mathcal{C}\eta$ sont en formes normales d'après (b). Par hypothèse, nous savons que $(\mathcal{P}\mathcal{S}')\!\downarrow \cup \mathcal{C}$ est un système monotone et satisfaisant la propriété d'initialisation. Par le lemme 5.20, nous en déduisons que la condition (a) est satisfaite.

Cas 1.2 : $t_1\sigma$ ou $t_2\sigma$ ne sont pas en formes normales.
Sans perte de généralité, nous pouvons supposer que $t_1\sigma$ se réécrit avec $l \to r \in \mathcal{R}$ et que ce radical est profond. Puisque σ est en forme normale, nous en déduisons que ce radical se trouve à une position p non-variable dans t_1. Nous pouvons donc appliquer une étape de surréduction sur le sous-terme $u = t_1|_p$. Posons $\eta = \mathsf{mgu}(l, u\mathcal{S}')$, nous avons :

$$(\mathcal{P}' \cup \{t_1[u]_p = t_2\};\mathcal{C};\mathcal{S}') \Rightarrow_{\mathsf{S}} (\mathcal{P}' \cup \{t_1[r]_p = t_2\};\mathcal{C}\eta;\mathcal{S}'\eta \cup \eta).$$

Soit $\rho = \mathsf{mgu}(l, t_1|_p\sigma)$. Posons $\sigma' = \sigma \cup \rho$. Tout d'abord, σ' est en forme normale puisque nous avons choisi un radical profond. Nous pouvons montrer par induction structurelle que $u\sigma = (u\eta)\sigma'$ pour tout terme $u \in \mathcal{P} \cup \mathcal{C} \cup \mathcal{S}'$. La substitution σ' est donc une solution du système obtenu. Puisque $(\mathcal{C}\eta)\sigma' = \mathcal{C}\sigma$, la condition (b) est satisfaite. La condition (a) se montre comme dans le cas précédent.

Cas 2 : $\mathcal{P}$ contient (au moins) une contrainte de déduction c mais aucune équation. Posons $c = t_1, ..., t_n \Vdash x$ et $\mathcal{P} = \{c\} \cup \mathcal{P}'$.

Cas 2.1 : pour tout $i \leq n$, $t_i\sigma$ est en forme normale.
Nous appliquons (B) et nous obtenons un système plus petit ayant le même ensemble de variables, et ayant la même solution σ.

$$(\mathcal{P}' \cup \{c\}; \mathcal{C}; \mathcal{S}') \Rightarrow_{\mathsf{B}} (\mathcal{P}'; \mathcal{C} \cup \{c\mathcal{S}'\}; \mathcal{S}')$$

Nous avons $(\mathcal{C} \cup \{c\mathcal{S}'\})\sigma = \mathcal{C}\sigma \cup \{c\sigma\}$. La condition (b) est donc satisfaite.
Par hypothèse, $c\mathcal{S}'\!\downarrow = c\mathcal{S}'$, et la condition (a) est également satisfaite.

Cas 2.2 : il existe $i \leq n$ tel que $t_i\sigma$ n'est pas en forme normale.
Ce cas est similaire au cas 1.2 : nous appliquons une étape de surréduction sur le sous-terme $u = t_1|_p$ où p est la position d'un radical profond dans $t_1\sigma$. Posons $\eta = \mathsf{mgu}(l, u\mathcal{S}')$. Alors :

$$(\mathcal{P}' \cup \{t_1[u]_p, \ldots, t_n \Vdash x\}; \mathcal{C}; \mathcal{S}') \Rightarrow_{\mathsf{S}} (\mathcal{P}' \cup \{t_1[r]_p, \ldots, t_n \Vdash x\}; \mathcal{C}\eta; \mathcal{S}'\eta \cup \eta).$$

La seule difficulté consiste à établir la condition (a). Par hypothèse d'induction, nous savons que $(\mathcal{P}' \cup \{c\})\mathcal{S}'\!\downarrow \cup \mathcal{C}$ est un système monotone satisfaisant la propriété d'initialisation. Grâce au lemme 5.20, nous en déduisons que $((\mathcal{P}' \cup \{c\})\mathcal{S}'\!\downarrow \cup \mathcal{C})\eta\!\downarrow$ est un système monotone satisfaisant la propriété d'initialisation. De plus, il est égal à $((\mathcal{P}' \cup \{t_1[r]_p, \ldots, t_n \Vdash x\})\mathcal{S}')\eta\!\downarrow \cup \mathcal{C}\eta$. La condition (a) est donc satisfaite.

Cas 3 : $\mathcal{P} = \varnothing$ et $\mathcal{C}$ contient (au moins) une contrainte de déduction. Posons $\mathcal{C} = \{c\} \cup \mathcal{C}'$.

Cas 3.1 : $vars(\mathcal{C}) \neq \varnothing$.
Soit $x \in vars(\mathcal{C})$. Par hypothèse, la restriction de σ au variable de $vars(\mathcal{C})$ est une solution minimale de $\mathcal{C}$. Grâce à la proposition 5.18 (et aux conditions (a) et (b)), nous savons qu'il existe $t \in St(\mathcal{C}) \smallsetminus vars(C)$ tel que $t\sigma = x\sigma$. Nous pouvons donc appliquer (EV). Nous obtenons :

$$(\varnothing; \mathcal{C}; \mathcal{S}') \Rightarrow_{\mathsf{EV}} (\varnothing; \mathcal{C}\{x \mapsto t\}; \mathcal{S}'\{x \mapsto t\} \cup \{x \mapsto t\}).$$

La substitution σ est une solution du système obtenu. Puisque $x\sigma = t\sigma$, nous en déduisons que $\mathcal{C}\sigma = (\mathcal{C}\{x \mapsto t\})\sigma$. Ainsi, la condition (b) est satisfaite. Puisque $\mathcal{C}$ est un système monotone satisfaisant la propriété d'initialisation, le lemme 5.20 nous assure que $\mathcal{C}\{x \mapsto t\}$ est également bien formé. La condition (a) est satisfaite.

Cas 3.2 : $vars(\mathcal{C}) = \varnothing$.
Nous pouvons appliquer (CI) et nous obtenons :

$$(\varnothing; \mathcal{C}' \cup \{c\}; \mathcal{S}') \Rightarrow_{\mathsf{CI}} (\varnothing; \mathcal{C}'; \mathcal{S}')$$

Bien entendu, σ est une solution normalisée de $(\varnothing; \mathcal{C}'; \mathcal{S}')$. La condition (a) est satisfaite puisque $vars(\mathcal{C}) = \varnothing$. D'autre part, les termes de $\mathcal{C}'\sigma$ sont en formes normales, la condition (b) est donc satisfaite.

Ce dernier cas termine l'induction. Nous pouvons appliquer ce résultat sur le système initial $(\mathcal{B}; \varnothing; \varnothing)$. Les conditions (a) et (b) sont satisfaites, et nous pouvons supposer que la substitution σ est normalisée. Nous en déduisons l'existence d'une dérivation de la forme : $(\mathcal{B}; \varnothing; \varnothing) \Rightarrow^* (\varnothing; \varnothing; \mathcal{S})$. □

5.3.5 NP-difficulté

Dans cette partie, nous montrons que le problème auquel nous nous intéressons est NP-difficile. Nous établissons le résultat suivant :

Proposition 5.22
Soit E *une théorie équationnelle publique-effondrante. Le problème de la satisfaisabilité d'un système de contraintes de déduction simple et bien formé avec équations est NP-difficile.*

Démonstration.
Nous montrons ce résultat par une réduction du problème 3-SAT.

Problème 3-SAT

Entrées : Un ensemble fini $X_1, \dots, X_n$ de variables booléennes. Un ensemble fini S de clauses de la forme $X_{\alpha_{i,1}}^{\epsilon_{i,1}} \vee X_{\alpha_{i,2}}^{\epsilon_{i,2}} \vee X_{\alpha_{i,3}}^{\epsilon_{i,3}}$, où :
- $\alpha_{i,j} \in \{1, \dots, n\}$,
- $\epsilon_{i,j} \in \{0, 1\}$ (X^1 signifie X et X^0 signifie $\neg X$).

Sortie : Est-ce que S est satisfaisable ?

Le codage présenté ci-dessous est inspiré de celui proposé par M. Rusinowitch et M. Turuani [RT03]. Grâce à la flexibilité de notre formalisme (choix du système de réécriture), nous allons construire un protocole réduit à une unique instruction de la forme $\mathsf{recv}(x); \mathcal{E}; \mathsf{send}(secret)$. Ainsi, ce codage permet d'établir que, dans le cas des théories publique-effondrantes :

1. le problème de la recherche d'attaques est NP-difficile,
2. le problème de la satisfaisabilité des systèmes de contraintes simples et bien formés avec équations est également NP-difficile.

Soit S une instance du problème 3-SAT. Notons $X_1, ..., X_n$ les variables propositionnelles de S. L'instance S est de la forme :

$$\bigwedge_{i=1}^{m} (X_{\alpha_{i,1}}^{\epsilon_{i,1}} \vee X_{\alpha_{i,2}}^{\epsilon_{i,2}} \vee X_{\alpha_{i,3}}^{\epsilon_{i,3}})$$

avec $\alpha_{i,j} \in \{1, \dots, n\}$ et $\epsilon_{i,j} \in \{0, 1\}$.

La signature que nous considérons est la suivante :
- un ensemble de symboles publics : $\mathcal{VF} = \{0, 1, \mathsf{proj}_1(\cdot), \dots, \mathsf{proj}_n(\cdot), \langle \cdot, \dots, \cdot \rangle\}$, où $\langle \cdot, \dots, \cdot \rangle$ représente un symbole d'arité n.
- un ensemble des symboles privés : $\mathcal{PF} = \{\cdot \wedge \cdot, \cdot \vee \cdot, \neg \cdot, secret\}$.

$$
\begin{array}{rclcrclcrcl}
0 \wedge 0 & \to & 0 & & 0 \vee 0 & \to & 0 & & & & \\
0 \wedge 1 & \to & 0 & & 0 \vee 1 & \to & 1 & & \neg 0 & \to & 1 \\
1 \wedge 0 & \to & 0 & & 1 \vee 0 & \to & 1 & & \neg 1 & \to & 0 \\
1 \wedge 1 & \to & 1 & & 1 \vee 1 & \to & 1 & & & &
\end{array}
$$

$$\mathsf{proj}_i\big(\langle x_1, \ldots, x_n\rangle\big) \quad \to \quad x_i \qquad\qquad \text{pour } 1 \leq i \leq n$$

Figure 5.3 - Système de réécriture $\mathcal{R}_{\mathsf{Boole}}$.

Nous considérons la théorie équationnelle $\mathsf{E}_{\mathsf{Boole}}$ représentée par le système de réécriture convergent et public-effondrant $\mathcal{R}_{\mathsf{Boole}}$ décrit à la figure 5.3.
Le protocole $\mathcal{P}(S)$ est composé d'un unique rôle, lui-même composé d'une seule instruction :

$$\mathsf{recv}(x);\ f_1(x) \wedge \ldots \wedge f_m(x) = 1;\ \mathsf{send}(secret)$$

où $f_i(x) = \mathsf{proj}_{\alpha_{i,1}}(x)^{\epsilon_{i,1}} \vee \mathsf{proj}_{\alpha_{i,2}}(x)^{\epsilon_{i,2}} \vee \mathsf{proj}_{\alpha_{i,3}}(x)^{\epsilon_{i,3}}$ pour tout $i \leq n$. Nous avons volontairement omis les parenthèses pour des raisons de lisibilité.

Par construction du protocole $\mathcal{P}(S)$ associé à l'ensemble S, nous avons : l'ensemble S est satisfaisable si, et seulement si, il existe une attaque sur le protocole $\mathcal{P}(S)$ permettant à l'intrus d'obtenir le message $secret$. $\square$

5.3.6 Illustration de la procédure

Nous illustrons notre procédure sur une variante du protocole de Denning-Sacco [DS81].

$$
\begin{array}{rcl}
A & \to & B: \quad A, \{\{K_{ab}\}_{\mathsf{priv}(A)}\}_{\mathsf{pub}(B)} \\
B & \to & A: \quad \{secret\}_{K_{ab}}
\end{array}
$$

Ce protocole a pour but d'établir une clef de session K_{ab} (clef symétrique) entre les agents jouant les rôles A et B. À la première étape du protocole, l'agent jouant le rôle A envoie d'une part son nom A, et d'autre part la clef de session K_{ab} qu'il vient de générer. Cette clef est chiffrée par un algorithme de chiffrement asymétrique avec la clef privée de l'agent jouant le rôle A, puis avec la clef publique de l'agent jouant le rôle B. À la deuxième étape du protocole, l'agent jouant le rôle B reçoit le message $A, \{\{K_{ab}\}_{\mathsf{inv}(\mathsf{priv}(A))}\}_{\mathsf{pub}(B)}$. Il déchiffre la deuxième partie du message à l'aide de sa clef privée. Il effectue ensuite un déchiffrement avec la clef publique de l'agent dont le nom correspond à la première composante du message reçu, ici $\mathsf{pub}(A)$, ce qui lui permet de récupérer la clef de session K_{ab}. L'obtention de cette clef va lui permettre d'envoyer son secret, $secret$, chiffré par un algorithme de chiffrement symétrique avec la clef K_{ab}. Ces deux rôles sont formellement décrits à la figure 5.4 (nous notons $\mathsf{inv}(\mathsf{pub}(x))$ la clef privée de l'agent x).

Nous considérons les deux instances $R_A(a, b)$ et $R_B(b)$. Le système de contraintes décrit ci-dessous a pour but d'établir l'existence d'une attaque sur ce protocole : l'intrus est en mesure de récupérer le secret généré par l'agent b. Nous considérons la théorie équationnelle permettant de représenter la version asymétrique du modèle de Dolev-Yao (*cf.* partie 5.1.2), et nous supposons que l'intrus a pour connaissance initiale $T_0 = \{0, a, b, I, \mathsf{inv}(\mathsf{pub}(I))\}$.

Rôle $R_A(z_a, z_b)$: $\nu k.$ $\mathsf{send}(\langle z_a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(z_a))}\}^a_{\mathsf{pub}(z_b)}\rangle)$

Rôle $R_B(z_b)$: $\nu s.$ $\mathsf{recv}(y); \mathsf{send}(\{s\}^s_{\mathsf{adec}(\mathsf{adec}(\mathsf{proj}_2(y), \mathsf{inv}(\mathsf{pub}(z_b))), \mathsf{pub}(\mathsf{proj}_1(y)))})$

Figure 5.4 - Rôles du protocole de Denning-Sacco dans le modèle avec tests d'égalités.

Système initial. Nous considérons le système suivant :

$$\begin{array}{rl}
\mathcal{P} := & \left\{ \begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash y \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(\mathsf{adec}(\mathsf{proj}_2(y), \mathsf{inv}(\mathsf{pub}(b))), \mathsf{pub}(\mathsf{proj}_1(y)))} \Vdash z \\ z = s \end{array} \right. \\
\mathcal{C} := & \varnothing \\
\mathcal{S} := & \varnothing
\end{array}$$

Étape 1. Unification syntaxique (U) sur « $z = s$ »

$$\begin{array}{rl}
\mathcal{P} := & \left\{ \begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash y \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(\mathsf{adec}(\mathsf{proj}_2(y), \mathsf{inv}(\mathsf{pub}(b))), \mathsf{pub}(\mathsf{proj}_1(y)))} \Vdash z \end{array} \right. \\
\mathcal{C} := & \varnothing \\
\mathcal{S} := & z \mapsto s
\end{array}$$

Étape 2. Surréduction (S) avec la règle « $\mathsf{proj}_2(\langle x_1, x_2\rangle) \to x_2$ »

$$\begin{array}{rl}
\mathcal{P} := & \left\{ \begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash y \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(\mathsf{adec}(x_2, \mathsf{inv}(\mathsf{pub}(b))), \mathsf{pub}(\mathsf{proj}_1(y)))} \Vdash z \end{array} \right. \\
\mathcal{C} := & \varnothing \\
\mathcal{S} := & z \mapsto s; y \mapsto \langle x_1, x_2\rangle
\end{array}$$

Étape 3. Surréduction (S) avec la règle « $\mathsf{proj}_1(\langle x_3, x_4\rangle) \to x_3$ »

$$\begin{array}{rl}
\mathcal{P} := & \left\{ \begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash y \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(\mathsf{adec}(x_2, \mathsf{inv}(\mathsf{pub}(b))), \mathsf{pub}(x_3))} \Vdash z \end{array} \right. \\
\mathcal{C} := & \varnothing \\
\mathcal{S} := & z \mapsto s;\ y \mapsto \langle x_1, x_2\rangle;\ x_3 \mapsto x_1;\ x_4 \mapsto x_2
\end{array}$$

Étape 4. Surréduction (S) avec la règle « $\mathsf{adec}(\{x_5\}_{x_6}, \mathsf{inv}(x_6)) \to x_5$ »

$$\begin{array}{rl}
\mathcal{P} := & \left\{ \begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash y \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(x_5, \mathsf{pub}(x_3))} \Vdash z \end{array} \right. \\
\mathcal{C} := & \varnothing \\
\mathcal{S} := & \left\{ \begin{array}{l} z \mapsto s;\ y \mapsto \langle x_1, \{x_5\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto x_1;\ x_4 \mapsto \{x_5\}^a_{\mathsf{pub}(b)}; \\ x_2 \mapsto \{x_5\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b) \end{array} \right.
\end{array}$$

Étape 5. Blocage (B)

$$\begin{array}{rl}\mathcal{P} := & T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(x_5,\mathsf{pub}(x_3))} \Vdash z \\ \mathcal{C} := & T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash \langle x_1, \{x_5\}^a_{\mathsf{pub}(b)}\rangle \\ \mathcal{S} := & \left\{\begin{array}{l} z \mapsto s;\ y \mapsto \langle x_1, \{x_5\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto x_1;\ x_4 \mapsto \{x_5\}^a_{\mathsf{pub}(b)}; \\ x_2 \mapsto \{x_5\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b)\end{array}\right.\end{array}$$

Étape 6. Blocage (B)

$$\begin{array}{rl}\mathcal{P} := & \varnothing \\ \mathcal{C} := & \left\{\begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash \langle x_1, \{x_5\}^a_{\mathsf{pub}(b)}\rangle \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(x_5,\mathsf{pub}(x_1))} \Vdash s\end{array}\right. \\ \mathcal{S} := & \left\{\begin{array}{l} z \mapsto s;\ y \mapsto \langle x_1, \{x_5\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto x_1;\ x_4 \mapsto \{x_5\}^a_{\mathsf{pub}(b)}; \\ x_2 \mapsto \{x_5\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b)\end{array}\right.\end{array}$$

Étape 7. Élimination d'une variable (EV) : « $x_1 \mapsto a$ »

$$\begin{array}{rl}\mathcal{P} := & \varnothing \\ \mathcal{C} := & \left\{\begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash \langle a, \{x_5\}^a_{\mathsf{pub}(b)}\rangle \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(x_3,\mathsf{pub}(a))} \Vdash s\end{array}\right. \\ \mathcal{S} := & \left\{\begin{array}{l} z \mapsto s;\ y \mapsto \langle a, \{x_5\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto a;\ x_4 \mapsto \{x_5\}^a_{\mathsf{pub}(b)}; \\ x_2 \mapsto \{x_5\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b)\end{array}\right.\end{array}$$

Étape 8. Élimination d'une variable (EV) : « $x_5 \mapsto 0$ »

$$\begin{array}{rl}\mathcal{P} := & \varnothing \\ \mathcal{C} := & \left\{\begin{array}{l} T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle \Vdash \langle a, \{0\}^a_{\mathsf{pub}(b)}\rangle \\ T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(0,\mathsf{pub}(a))} \Vdash s\end{array}\right. \\ \mathcal{S} := & z \mapsto s;\ y \mapsto \langle a, \{0\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto a;\ x_4 \mapsto \{0\}^a_{\mathsf{pub}(b)};\ x_2 \mapsto \{0\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b)\end{array}$$

Étape 9. Contrainte Close (Cl) : $\langle a, \{0\}^a_{\mathsf{pub}(b)}\rangle$ est déductible de T_0

$$\begin{array}{rl}\mathcal{P} := & \varnothing \\ \mathcal{C} := & T_0; \langle a, \{\{k\}^a_{\mathsf{inv}(\mathsf{pub}(a))}\}^a_{\mathsf{pub}(b)}\rangle; \{s\}^s_{\mathsf{adec}(0,\mathsf{pub}(a))} \Vdash s \\ \mathcal{S} := & z \mapsto s;\ y \mapsto \langle a, \{0\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto a;\ x_4 \mapsto \{0\}^a_{\mathsf{pub}(b)};\ x_2 \mapsto \{0\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b)\end{array}$$

Étape 10. Contrainte Close (Cl) : s est déductible de T_0 et de $\{s\}^s_{\mathsf{adec}(0,\mathsf{pub}(a))}$

$$\begin{array}{rl}\mathcal{P} := & \varnothing \\ \mathcal{C} := & \varnothing \\ \mathcal{S} := & z \mapsto s;\ y \mapsto \langle a, \{0\}^a_{\mathsf{pub}(b)}\rangle;\ x_3 \mapsto a;\ x_4 \mapsto \{0\}^a_{\mathsf{pub}(b)};\ x_2 \mapsto \{0\}^a_{\mathsf{pub}(b)};\ x_6 \mapsto \mathsf{pub}(b)\end{array}$$

Le système de contraintes initial est donc satisfaisable. L'attaque que nous venons de trouver est très simple à mettre en place. L'intrus a juste besoin d'envoyer à b le message $\langle a, \{0\}_{\mathsf{pub}(b)}\rangle$. Ce dernier n'est pas exactement de la forme souhaitée, mais b ne s'en aperçoit pas et émet sur le réseau le message $\{s\}^s_{\mathsf{adec}(0,\mathsf{pub}(a))}$. Cette « clef » utilisée par b pour chiffrer son secret est connue de l'intrus. Celui-ci peut alors facilement récupérer le secret s.

5.4 Comparaisons

Dans cette partie, nous allons comparer nos résultats à quelques résultats assez proches obtenus récemment.

5.4.1 Travaux de M. Abadi et V. Cortier

M. Abadi et V. Cortier ont montré [AC04] que le problème de déduction de l'intrus était décidable en temps polynomial pour la classe des théories « sous-termes convergentes ». Cette classe correspond à l'ensemble des théories équationnelles représentable par un système de réécriture $\mathcal{R}$ convergent satisfaisant la *condition sous-termes* suivante :

$$\text{pour toute règle } l \to r \in \mathcal{R},\ r \text{ est un sous-terme strict de } l.$$

Cette classe est plus large que la classe des théories publique-effondrantes étudiée dans ce chapitre. La procédure proposée par M. Abadi et V. Cortier repose sur un résultat de localité pour la notion de sous-termes syntaxiques. Ils procèdent par applications de « petits contextes » afin d'éviter l'utilisation de certains termes dans l'arbre de preuve.

Exemple 5.23
Reprenons, l'exemple 5.2.

$$\mathcal{R} := \quad f_3(f_2(f_1(x))) \ \to\ x.$$

Ce système est convergent et nous supposons que les symboles f_2 et f_3 sont publics. Avec notre approche, une preuve de s à partir de $f_1(s)$ nécessite de passer par le terme $f_2(f_1(s))$. Avec l'approche développée par M. Abadi et V. Cortier, le contexte $f_3(f_2(\cdot))$ est un « petit contexte », l'arbre

$$\frac{f_1(s) \vdash f_1(s)}{f_1(s) \vdash s}$$

est donc un arbre de preuve de $f_1(s) \vdash s$. Cet arbre ne passe pas par le terme $f_2(f_1(s))$.

Leur procédure de décision est une procédure de saturation consistant à ajouter à l'ensemble des termes déductibles par l'intrus tous ceux (parmi l'ensemble des sous-termes syntaxiques) déductibles par application de « petits contextes ».

Leur résultat s'applique aussi au problème de l'indistinguabilité de deux séquences de messages (appelées *frames*). Cette relation d'indistinguabilité est plus difficile à décider que la simple relation de déduction. M. Abadi et V. Cortier montrent que, dans le cas des théories sous-termes convergentes, le problème de l'indistinguabilité de deux frames est également décidable en temps polynomial. En revanche, le problème de la vérification d'un protocole en présence d'un intrus actif n'est pas du tout abordé. Il est simplement mentionné comme étant un problème ouvert et difficile. C'est ce problème que nous avons résolu dans le cas des théories publique-effondrantes.

5.4.2 Travaux de M. Baudet

Les résultats obtenus par M. Baudet [Bau05] généralisent les travaux présentés dans ce chapitre :

- d'une part, il considère une classe de théories équationnelles plus générale que la nôtre : la classe des théories sous-termes convergentes introduite par M. Abadi et V. Cortier [AC04],
- d'autre part, sa procédure permet de traiter une propriété de secret particulière : elle permet de tester la résistance des protocoles aux attaques dites « par dictionnaire ».

M. Baudet utilise une approche du type résolution de contraintes. Il donne un ensemble de règles de transformations et une stratégie permettant d'assurer la terminaison. Aucune analyse concernant la complexité de cette procédure n'est donnée dans [Bau05].

5.4.3 Travaux de Y. Chevalier et M. Rusinowitch

L'originalité du travail présenté dans ce chapitre est également dans le choix du modèle. Ce modèle par rôles avec tests d'égalités, introduit dans [DJ04a] et présenté au chapitre 3, a par la suite été repris. Un certain nombre de résultats génériques ont pu être établis dans ce modèle (*e.g.* [Bau05, CR05]).

Dans [CR05], Y. Chevalier et M. Rusinowitch choisissent ce modèle pour établir un résultat de combinaisons dans le cas de théories disjointes. Ils montrent que si le problème de la satisfaisabilité des systèmes de contraintes est décidable dans $(\mathcal{I}_1, \mathsf{E}_1)$ et $(\mathcal{I}_2, \mathsf{E}_2)$, alors ce problème est également décidable dans $(\mathcal{I}_1 \cup \mathcal{I}_2, \mathsf{E}_1 \cup \mathsf{E}_2)$. Les principales conditions pour obtenir ce résultat sont les suivantes :

- les systèmes de contraintes considérés sont des systèmes de contraintes de déduction simples et bien formés avec équations,
- les modèles d'intrus considérés, $(\mathcal{I}_1, \mathsf{E}_1)$ et $(\mathcal{I}_2, \mathsf{E}_2)$, doivent êtres disjoints, c'est-à-dire ne partager aucun symbole de fonction,
- les systèmes d'inférences $\mathcal{I}_1$ et $\mathcal{I}_2$ doivent être des systèmes d'inférence simples (aucun filtrage n'est autorisé sur les prémisses).

Ce résultat est intéressant, mais il ne permet pas de combiner les procédures développées jusqu'à présent pour des théories particulières, puisque la majorité des travaux existant utilise le modèle avec filtrage. Y. Chevalier et M. Rusinowitch établissent, dans ce modèle avec tests d'égalitées, des procédures permettant de traiter les théories équationnelles « classiques » telles que les théories ACUN, AG et DH. Ils montrent que le problème de la satisfaisabilité de systèmes de contraintes bien formés est décidable en temps polynomial dans le cas de la théorie du « ou » exclusif et que ce dernier est dans NP dans le cas des théories AG et DH.

Il est important de remarquer que, pour certaines théories équationnelles (c'est le cas par exemple de AG), il est plus facile d'établir un résultat de décidabilité dans le cas du modèle avec test d'égalités que dans le modèle avec filtrage. Ceci est essentiellement dû au fait que la classe des systèmes bien formés dans le modèle avec filtrage est « plus grande » que la classe des systèmes bien formés dans le modèle avec tests d'égalités. Nous reviendrons sur ce problème à la fin du chapitre 6 pour expliquer le fait que l'algorithme proposé par J. Millen et V. Shmatikov pour traiter la théorie AG — et dont nous nous sommes inspiré pour obtenir une procédure dans le cas de la théorie ACUNh — est beaucoup plus complexe que l'algorithme proposé par Y. Chevalier et M. Rusinowitch dans [CR05] pour traiter cette même théorie.

Chapitre 6

Vérification dans le modèle par rôles avec filtrage

Sommaire

6.1 Cadre **104**
6.1.1 Théories équationnelles considérées : ACh, ACUNh et AGh 104
6.1.2 Applications 105
6.2 Problème de déduction de l'intrus **108**
6.2.1 Système d'inférence 108
6.2.2 Résultat de localité 110
6.2.3 Déductibilité en un pas 112
6.2.3.1 Procédure 112
6.2.3.2 Complexité 114
6.2.4 Comparaison avec les travaux de P. Lafourcade *et al.* 115
6.3 Résolution de systèmes de contraintes bien formés **116**
6.3.1 Existence d'une solution conservatrice 116
6.3.2 De la résolution dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ à la résolution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$ 121
6.3.3 Réduction de la signature 125
6.3.4 À propos des systèmes bien formés 126
6.3.4.1 Une nouvelle caractérisation des systèmes bien formés . . . 126
6.3.4.2 Bonne formation des systèmes obtenus après abstraction . 130
6.3.5 Résolution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$ sur signature réduite 135
6.4 Résultats d'indécidabilité **142**
6.4.1 Théorie ACh 142
6.4.2 Théorie AGh 143
6.4.2.1 Indécidabilité dans $(\mathcal{I}_{\mathsf{M_E}}, \mathsf{AGh})$ 144
6.4.2.2 Indécidabilité dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$ 147
6.5 Discussion **148**
6.6 Comparaisons **150**
6.6.1 Travaux de J. Millen et V. Shmatikov 150
6.6.2 Travaux de Y. Chevalier et M. Rusinowitch 150

DANS ce chapitre, nous menons l'étude du problème de la sécurité d'un protocole en présence de théories équationnelles mettant en jeu un opérateur AC. Cette étude est réalisée dans le modèle avec filtrage, mais une comparaison avec des résultats « similaires », obtenus par Y. Chevalier et M. Rusinowitch dans le modèle avec tests d'égalités, est effectuée dans la partie 6.6.

Dans la première partie, nous décrivons les théories équationnelles étudiées dans ce chapitre et leur intérêt du point de vue de la vérification des protocoles cryptographiques : il s'agit de théories équationnelles faisant intervenir l'axiome d'homomorphisme sur un opérateur AC (*e.g.* ACUNh et AGh). Nous commençons par revisiter le cas d'un intrus passif, déjà étudié par P. Lafourcade *et al.* [LLT05a]. Nous considérons une nouvelle approche pour traiter le symbole d'homomorphisme. Celle-ci nous permet d'améliorer les résultats de complexité existants en ce qui concerne le problème de déduction de l'intrus (*cf.* partie 6.2). D'autre part, cette nouvelle approche nous permet de résoudre le problème de la vérification (pour un nombre borné de sessions) en présence d'un intrus actif (*cf.* partie 6.3). Ce problème n'avait jamais été résolu pour la théorie ACUNh. La technique utilisée s'inspire et corrige la procédure développée par J. Millen et V. Shmatikov dans le cas de la théorie AG. Nous discuterons ce point dans la partie 6.6.

6.1 Cadre

Le but de cette partie est de définir les trois théories équationnelles ACh, ACUNh et AGh dont nous allons faire l'étude dans ce chapitre, et d'illustrer l'intérêt de ces théories du point de vue de la vérification des protocoles cryptographiques. Des exemples de protocoles utilisant des primitives satisfaisant des « propriétés d'homomorphismes » sont décrits dans la partie 6.1.2.

6.1.1 Théories équationnelles considérées : ACh, ACUNh et AGh

Nous nous intéressons aux théories équationnelles ACh, ACUNh et AGh. Ces trois théories ont comme point commun de mettre en jeu un opérateur associatif et commutatif (noté $+$), et un symbole h satisfaisant la propriété d'homomorphisme suivante :

$$\mathsf{h}(x+y) = \mathsf{h}(x) + \mathsf{h}(y).$$

Suivant la nature de l'opérateur $+$, nous obtenons les trois théories équationnelles suivantes :

- théorie équationnelle ACh :

$$\begin{array}{rcl} x+(y+z) & = & (x+y)+z \\ x+y & = & y+x \end{array} \qquad \begin{array}{rcl} \mathsf{h}(x+y) & = & \mathsf{h}(x)+\mathsf{h}(y) \end{array}$$

- théorie équationnelle ACUNh :

$$\begin{array}{rcl} x+(y+z) & = & (x+y)+z \\ x+y & = & y+x \end{array} \qquad \begin{array}{rcl} \mathsf{h}(x+y) & = & \mathsf{h}(x)+\mathsf{h}(y) \\ x+0 & = & x \\ x+x & = & 0 \end{array}$$

- théorie équationnelle AGh :

$$\begin{array}{rcl} x+(y+z) & = & (x+y)+z \\ x+y & = & y+x \end{array} \qquad \begin{array}{rcl} \mathsf{h}(x+y) & = & \mathsf{h}(x)+\mathsf{h}(y) \\ x+0 & = & x \\ x+-(x) & = & 0 \end{array}$$

L'ensemble des symboles de fonctions est noté $\mathcal{F}$. Cet ensemble est composé de deux sous-ensembles disjoints : l'ensemble des *symboles privés* $\mathcal{PF}$ et l'ensemble des *symboles publics* ou *visibles* $\mathcal{VF}$. Ce dernier contient entre autres les symboles de la théorie équationnelle considérée, c'est-à-dire les symboles de $sig(\mathsf{E})$. Le modèle d'intrus que nous considérons est représenté par le système d'inférence décrit à la figure 6.1. Il s'agit du modèle d'inférence classique (avec filtrage) introduit au chapitre 3. Les règles (Proj_1), (Proj_2), (D) et (C) sont celles du modèle noté $\mathcal{I}_{\mathsf{DY}}$. La théorie équationnelle est prise en compte par l'intermédiaire de la règle (Eq).

$$\text{Projection}\ (\mathsf{Proj}_1)\ \frac{T \vdash \langle u, v\rangle}{T \vdash u} \qquad \text{Déchiffrement}\ (\mathsf{D})\ \frac{T \vdash \{u\}_v \quad T \vdash v}{T \vdash u}$$

$$\text{Projection}\ (\mathsf{Proj}_2)\ \frac{T \vdash \langle u, v\rangle}{T \vdash v} \qquad \text{Composition}\ (\mathsf{C})\ \frac{T \vdash u_1\ \ldots\ T \vdash u_n}{T \vdash f(u_1, \ldots, u_n)}\ \text{avec } f \in \mathcal{VF}$$

$$\text{Égalité}\ (\mathsf{Eq})\ \frac{T \vdash u}{T \vdash v}\ \text{avec } u =_{\mathsf{E}} v$$

Figure 6.1 - Système d'inférence ($\mathcal{I}_{\mathsf{DY}}$, E).

6.1.2 Applications

L'ajout d'une théorie équationnelle a pour but de modéliser les propriétés algébriques des primitives cryptographiques. Nous montrons à travers deux exemples l'intérêt de la propriété d'homomorphisme sur un symbole AC, et plus particulièrement l'intérêt des théories ACUNh et AGh.

Protocole WEP (Wired Equivalent Privacy)

Le protocole WEP, décrit dans [CDL06], est utilisé pour protéger des données au cours d'une communication sans fil.

$$A \;\rightarrow\; B : \;\; v, \langle m, \mathsf{C}(m)\rangle \oplus \mathsf{RC4}(v, K_{ab})$$

Pour chiffrer un message m, l'agent jouant le rôle A applique l'opérateur $\oplus$ (« ou » exclusif) à $\mathsf{RC4}(v, K_{ab})$ et $\langle m, \mathsf{C}(m)\rangle$. Le symbole de fonction RC4 modélise l'algorithme RC4 utilisé pour générer des séquences de bits aléatoires à partir du vecteur initial v et de la clef K_{ab} partagée entre les agents jouant les rôles A et B. Pour effectuer le déchiffrement, l'agent jouant le rôle B calcule $\mathsf{RC4}(v, K_{ab})$. Après application de l'opérateur $\oplus$, il obtient $\langle m, \mathsf{C}(m)\rangle$. Il vérifie que la deuxième composante de la paire obtenue est en fait égale à la la première composante de cette même paire après application de la fonction C.

La première attaque (décrite ci-après) repose uniquement les propriétés algébriques du « ou » exclusif, alors que la seconde exploite les deux propriétés d'homomorphismes suivantes :

$$\begin{array}{rcl} \mathsf{C}(x \oplus y) & = & \mathsf{C}(x) \oplus \mathsf{C}(y) \\ \langle x_1, y_1\rangle \oplus \langle x_2, y_2\rangle & = & \langle x_1 \oplus x_2, y_1 \oplus y_2\rangle \end{array}$$

Attaque 1. Soit c_1 (resp. c_2) le chiffré du message m_1 (resp. m_2) avec la clef k. Si ces deux messages sont chiffrés en utilisant le même vecteur d'initialisation, nous avons les égalités suivantes :

$$\begin{array}{rcl} c_1 \oplus c_2 & = & (\langle m_1, c_1\rangle \oplus \mathsf{RC4}(v,k)) \oplus (\langle m_2, c_2\rangle \oplus \mathsf{RC4}(v,k)) \\ & = & \langle(m_1, c_1\rangle \oplus \langle m_2, c_2\rangle) \oplus (\mathsf{RC4}(v,k) \oplus \mathsf{RC4}(v,k)) \\ & = & \langle m_1, c_1\rangle \oplus \langle m_2, c_2\rangle \end{array}$$

Un intrus, connaissant le message m_1 et son chiffré c_1, peut alors déchiffrer n'importe quel message c_2.

Attaque 2. La deuxième attaque permet à un intrus de modifier le message transmis. Supposons que l'intrus ait intercepté $\langle m, C(m)\rangle \oplus RC4(v, Kab)$ et qu'il connaisse d. Il peut obtenir le chiffré associé au message $m \oplus d$ en calculant :

$$\begin{array}{rcl} \langle m, \mathsf{C}(m)\rangle \oplus \mathsf{RC4}(v, K_{ab}) \oplus \langle d, \mathsf{C}(d)\rangle & = & \mathsf{RC4}(v, K_{ab}) \oplus \langle m, \mathsf{C}(m)\rangle \oplus \langle d, \mathsf{C}(d)\rangle \\ & = & \mathsf{RC4}(v, K_{ab}) \oplus \langle m \oplus d, \mathsf{C}(m) \oplus \mathsf{C}(d)\rangle \\ & = & \mathsf{RC4}(v, K_{ab}) \oplus \langle m \oplus d, \mathsf{C}(m \oplus d)\rangle \end{array}$$

L'intrus peut monter cette attaque sans connaître le message m. Par exemple, s'il souhaite changer le premier bit du message m, il lui suffit de choisir $d = 1000....0$. Si l'intrus connaît le message m, il est alors en mesure de produire le chiffré du message de son choix.

Protocole TMN

Ce protocole, du à T. Tatebayashi, N. Matsuzaki et D. Newman [TMN89], a pour but l'établissement d'une clef de session (noté K_b). Il implique trois participants dont un serveur, noté S, et nécessite quatre échanges de messages.

$$\begin{array}{lcll} A & \rightarrow & S : & A, B, \{K_a\}_{\mathsf{pub}(S)} \\ S & \rightarrow & B : & A \\ B & \rightarrow & S : & B, A, \{K_b\}_{\mathsf{pub}(S)} \\ S & \rightarrow & A : & B, K_b \oplus K_a \end{array}$$

Lorsque l'agent A veut communiquer avec un agent B, il génère un nombre aléatoire K_a, le chiffre avec la clef publique du serveur S et envoie son intention de communiquer avec B au serveur S (premier message). Le serveur S reçoit ce message, et informe B que A souhaite communiquer avec lui (deuxième message). L'agent B génère un nombre aléatoire, le chiffre avec la clef publique de S et envoie ce message à S (troisième message). Pour éviter d'éventuelles attaques par rejeu, le serveur S vérifie que les nombres aléatoires K_a et K_b n'ont pas été utilisés auparavant. Dans ce cas, le serveur applique l'opération $\oplus$ sur des deux données ($\oplus$ est un opérateur ACUN) et envoie ce message à l'agent A. Maintenant, l'agent A connaît K_a et $K_a \oplus K_b$, il peut donc en déduire K_b.

Ce protocole utilisent deux types de chiffrement :

1. La première primitive de chiffrement, noté $\{.\}_.$, est un chiffrement asymétrique. Autrement dit, un agent, connaissant m, est en mesure de produire $\{m\}_{\mathsf{pub}(S)}$. Cependant, seul le serveur est capable de déchiffrer un tel message. Pour ce protocole, cette primitive est implémentée par l'algorithme de chiffrement RSA.
2. La primitive, noté $\oplus$, peut être vue comme un algorithme de chiffrement symétrique. Pour chiffrer un message m avec une clef k, il suffit d'appliquer l'opération $\oplus$ entre le message et la clef afin d'obtenir $m \oplus k$. La connaissance de la clef de chiffrement, c'est-à-dire k, permet de récupérer le message m à partir de $m \oplus k$. Il suffit d'appliquer l'opération $\oplus$ entre $m \oplus k$ et k, on obtient $(m \oplus k) \oplus k =_{\mathsf{ACUN}} m$.

Ce protocole est sujet à de nombreuses attaques [Sim94, LR97]. L'attaque décrite ci-dessous (cf. Figure 6.2), due à G. Simmons [Sim94], fait intervenir un intrus actif exploitant la propriété d'homomorphisme (satisfaite par l'algorithme de chiffrement RSA) suivante :

$$\{x \times y\}_{\mathsf{pub}(S)} = \{x\}_{\mathsf{pub}(S)} \times \{y\}_{\mathsf{pub}(S)}, \text{ où } \times \text{ représente la multiplication.}$$

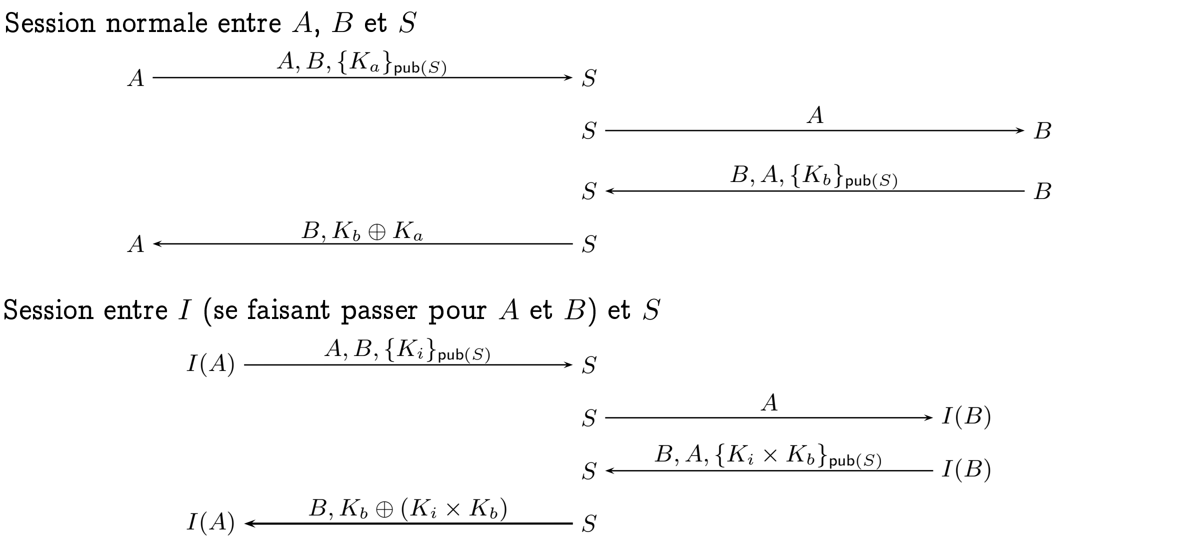

Figure 6.2 - Attaque sur le protocole TMN, due à G. Simmons.

Cette attaque permet à l'intrus de récupérer la clef K_b échangée entre des agents honnêtes jouant les rôles A et B au cours d'une session normale du protocole. L'intrus connaît le message $\{K_b\}_{\mathsf{pub}(S)}$ puisque ce message a circulé sur le réseau lorsque la session normale a été

$$\mathcal{R}_{\mathsf{ACh}} : \qquad \mathsf{h}(x+y) \to \mathsf{h}(x)+\mathsf{h}(y)$$

$$\mathcal{R}_{\mathsf{ACUNh}} : \quad \begin{array}{rcl} \mathsf{h}(x+y) & \to & \mathsf{h}(x)+\mathsf{h}(y) \\ x+0 & \to & x \end{array} \qquad \begin{array}{rcl} \mathsf{h}(0) & \to & 0 \\ x+x & \to & 0 \end{array}$$

$$\mathcal{R}_{\mathsf{AGh}} : \quad \begin{array}{rcl} \mathsf{h}(x+y) & \to & \mathsf{h}(x)+\mathsf{h}(y) \\ x+0 & \to & x \\ -(x+y) & \to & -(x)+-(y) \\ \mathsf{h}(-(x)) & \to & -(\mathsf{h}(x)) \end{array} \qquad \begin{array}{rcl} \mathsf{h}(0) & \to & 0 \\ x+-(x) & \to & 0 \\ -0 & \to & 0 \\ -(-x) & \to & x \end{array}$$

Figure 6.3 - Systèmes de réécriture AC-convergent associés aux théories ACh, ACUNh et AGh.

exécutée. Il lui suffit d'envoyer au serveur $\{K_i\}_{\mathsf{pub}(S)}$, puis $\{K_i\}_{\mathsf{pub}(S)} \times \{K_b\}_{\mathsf{pub}(S)}$. Ce dernier message est égal à $\{K_i \times K_b\}_{\mathsf{pub}(S)}$ par la propriété d'homomorphisme. Le serveur pense alors avoir reçu une donnée fraîche ($K_i \times K_b$ ne ressemble à aucun des nonces précédemment reçus) et joue le reste du protocole. En particulier, il émet $(K_i \times K_b) \oplus K_i$ sur le réseau. L'intrus, puisqu'il connaît K_i, peut facilement en déduire K_b.

Nous pouvons modéliser la propriété d'homomorphisme du chiffrement RSA, par l'axiome :

$$\mathsf{h}(x \times y) = \mathsf{h}(x) \times \mathsf{h}(y).$$

Le symbole h représente le chiffrement par la clef publique de S. Le fait que S soit un serveur digne de confiance est pris en compte par le fait que l'intrus n'est pas en mesure d'obtenir m à partir de $\mathsf{h}(m)$. La multiplication peut être modélisée par un opérateur AG.

6.2 Problème de déduction de l'intrus

Dans cette partie, nous nous intéressons au problème de déduction de l'intrus et nous montrons que ce dernier est décidable en temps polynomial pour les théories ACUNh et AGh [Del06a]. Ce problème est NP-complet dans le cas de la théorie ACh [LLT05a]. Conformément à la technique de preuve décrite au chapitre 4, nous commençons par proposer un système d'inférence équivalent au système d'inférence de départ (*cf.* partie 6.2.1). Nous montrons ensuite que ce nouveau système a la propriété de localité pour une « bonne » notion de sous-termes (*cf.* partie 6.2.2), ainsi que la propriété de déductibilité en un pas (*cf.* partie 6.2.3).

6.2.1 Système d'inférence

Nous choisissons tout d'abord de supprimer la règle (Eq), et de travailler sur des termes normalisés. Les systèmes de réécriture AC-convergents associés à chacune des théories équationnelles ACh, ACUNh et AGh sont décrits Figure 6.3.

Notation : Soit $n \in \mathbb{N}$. La notation $\mathsf{h}^n(t)$ représente le terme t si $n=0$ et $\mathsf{h}(\mathsf{h}^{n-1}(t))$ sinon. De même nt représente le terme t si $n=1$ et $t+(n-1)t$ sinon. Enfin, $(-n)t$ représente le terme $-(nt)$. Cette dernière notation n'a de sens que dans le cas de la théorie AGh. Ainsi, le terme $-2\mathsf{h}^3(a) + \mathsf{h}(b)$ représente $-(\mathsf{h}(\mathsf{h}(\mathsf{h}(a)))) + -(\mathsf{h}(\mathsf{h}(\mathsf{h}(a)))) + \mathsf{h}(b)$.

La deuxième transformation consiste à ajouter un schéma de règle afin de ne pas avoir à considérer certains « types de sous-termes » (*e.g.* les sommes partielles). Pour cela, nous introduisons le schéma ($\mathsf{M_E}$) suivant :

$$\frac{T \vdash u_1 \ \dots \ T \vdash u_n}{T \vdash C[u_1, \dots, u_n]\downarrow} \text{ avec } C \text{ un E-contexte}$$

Exemple 6.1
Considérons la théorie AGh. *Les deux inférences ci-dessous sont des instances du schéma* ($\mathsf{M_E}$) *obtenues à partir des contextes* $C[x_1, x_2] = x_1 + \mathsf{h}(x_1) + \mathsf{h}^2(x_1) + -2\mathsf{h}(x_2)$ *et* $C = 0$.

$$\frac{T \vdash a + \mathsf{h}(a) \qquad T \vdash b}{T \vdash a + \mathsf{h}^3(a) + -2\mathsf{h}(b)}\ (\mathsf{M_E}) \qquad\qquad \frac{}{T \vdash 0}\ (\mathsf{M_E})$$

Le système de déduction que nous allons considérer, noté ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$), est décrit à la figure 6.4. Dans la mesure où nous travaillons avec un système de réécriture AC-convergent, nous conservons implicitement la règle (Eq) pour la théorie AC. Nous montrons, dans la proposition 6.2, que le système ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$) est équivalent au système ($\mathcal{I}_{\mathsf{DY}}$, E).

Projection (Proj_1) $\dfrac{T \vdash \langle u, v\rangle}{T \vdash u}$ Déchiffrement (D) $\dfrac{T \vdash \{u\}_v \quad T \vdash v}{T \vdash u}$

Projection (Proj_2) $\dfrac{T \vdash \langle u, v\rangle}{T \vdash v}$ Composition (C^-) $\dfrac{T \vdash u_1 \ \dots \ T \vdash u_n}{T \vdash f(u_1, \dots, u_n)}$ avec $f \in \mathcal{VF} \smallsetminus sig(\mathsf{E})$

Contexte ($\mathsf{M_E}$) $\dfrac{T \vdash u_1 \ \dots \ T \vdash u_n}{T \vdash C[u_1, \dots, u_n]\downarrow}$ avec C un E-contexte

Figure 6.4 - Système d'inférence ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$).

Proposition 6.2
Soient $T \subseteq \mathcal{T}(\mathcal{F})\downarrow$ *et* $u \in \mathcal{T}(\mathcal{F})\downarrow$. *Nous avons :*

$$T \vdash u \textit{ dans } (\mathcal{I}_{\mathsf{DY}}, \mathsf{E}) \Leftrightarrow T \vdash u \textit{ dans } (\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}}).$$

Démonstration.
($\Leftarrow$) Soit P un arbre de preuve de $T \vdash u$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$). Pour obtenir, un arbre de preuve de $T \vdash u$ dans ($\mathcal{I}_{\mathsf{DY}}$, E), il suffit d'insérer des instances de la règle (Eq) pour remplacer les étapes de normalisation.
($\Rightarrow$) Soit P un arbre de preuve de $T \vdash u$ dans ($\mathcal{I}_{\mathsf{DY}}$, E). Soit P' la preuve obtenue en normalisant tous les termes et en supprimant les applications de la règle (Eq). Nous allons montrer par induction sur P que cette transformation nous permet d'obtenir une preuve P' de $T \vdash u\downarrow$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$) (c'est-à-dire de $T \vdash u$, puisque $u\downarrow =_{\mathsf{AC}} u$).

Cas de base : Si P est réduit à une feuille, alors on a $u \in T$ ou $u = 0$. Dans les deux cas P est également une preuve de $T \vdash u$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$).

Étape d'induction : Nous distinguons différents cas suivant le type de la dernière règle d'inférence utilisée pour obtenir P.

– Si P se termine par une instance de la règle (Eq), alors P est de la forme :

$$\frac{T \vdash v}{T \vdash u} \ (\mathsf{Eq})$$

avec $u =_{\mathsf{E}} v$.
Par hypothèse d'induction, il existe une preuve P_1' de $T \vdash v\!\downarrow$ dans ($\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}}$) et donc P_1' convient puisque $u\!\downarrow =_{\mathsf{AC}} v\!\downarrow$.

– Si P se termine par une instance de (C), alors P est de la forme :

$$\frac{T \vdash v_1 \ \ \ldots \ \ T \vdash v_n}{T \vdash f(v_1, \ldots, v_n)} \ (\mathsf{C}).$$

Par hypothèse d'induction, il existe des preuves $P_1, \ldots,\ P_n$ de $T \vdash v_1\!\downarrow, \ldots, T \vdash v_n\!\downarrow$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$). Si $f \notin sig(\mathsf{E})$, alors on obtient P à partir de $P_1, \ldots,\ P_n$ en appliquant une instance de (C^-). L'arbre P ainsi obtenu est une preuve de $T \vdash f(v_1\!\downarrow, \ldots, v_n\!\downarrow)$, c'est-à-dire de $T \vdash f(v_1, \ldots, v_n)\!\downarrow$. Si $f \in sig(\mathsf{E})$, alors on obtient l'arbre P à partir des arbres $P_1, \ldots, P_n$ en appliquant une instance de (M_{E}). L'arbre P, ainsi obtenu, est une preuve de $T \vdash f(v_1\!\downarrow, \ldots, v_n\!\downarrow)\!\downarrow$, c'est-à-dire de $T \vdash f(v_1, \ldots, v_n)\!\downarrow$.

– Si P se termine par une instance de (Proj_1) (resp. Proj_2 ou D), alors P est de la forme :

$$\frac{T \vdash \langle u, v\rangle}{T \vdash u} \ (\mathsf{Proj}_1)$$

Par hypothèse d'induction, il existe une preuve P_1' de $T \vdash \langle u, v\rangle\!\downarrow$. Or, nous avons l'égalité suivante : $\langle u, v\rangle\!\downarrow =_{\mathsf{AC}} \langle u\!\downarrow, v\!\downarrow\rangle$. Nous en déduisons donc que l'arbre P' obtenu à partir de P_1' en appliquant une instance de (Proj_1) est un arbre de preuve de $T \vdash u\!\downarrow$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$). □

Remarque : Dans toute la suite du chapitre, nous travaillons sur les termes en forme normale. Autrement dit, t signifie $t\!\downarrow$. De plus, la relation $=$ entre termes représente l'égalité modulo les propriétés associatives et commutatives du symbole $+$.

6.2.2 Résultat de localité

Avant d'établir le résultat de localité (lemme 6.9), nous devons définir la notion de sous-termes que nous allons considérer. Pour cela, nous introduisons la notion de terme standard et de facteurs.

Définition 6.3 (terme standard)
Un terme t est dit standard *si t est une variable ou si* head(t) $\notin sig(\mathsf{E})$.

Définition 6.4 (facteur)
Soient t un terme, C un E*-contexte et $t_1, \ldots, t_n$ des termes standards tels que $t = C[t_1, \ldots, t_n]$. L'ensemble des facteurs de t est défini par $Fact_{\mathsf{E}}(t) = \{t_1, \ldots, t_n\}$.*

Définition 6.5 (sous-terme)
L'ensemble $St_{\mathsf{E}}(t)$ des sous-termes *de t est défini inductivement de la façon suivante :*
- *$t \in St_{\mathsf{E}}(t)$,*
- *si $f(t_1, \ldots, t_n) \in St_{\mathsf{E}}(t)$ est standard alors $t_1, \ldots t_n \in St_{\mathsf{E}}(t)$,*
- *si $s \in St_{\mathsf{E}}(t)$ n'est pas standard alors $Fact_{\mathsf{E}}(s) \subseteq St_{\mathsf{E}}(t)$.*

Ces notions s'étendent naturellement à un ensemble de termes T. D'autre part, il est important de remarquer que, par définition, les facteurs d'un termes sont des termes standards. D'autre part, étant donné un ensemble de termes T, le nombre de sous-termes de T est linéaire en la taille de T.

Exemple 6.6
Considérons les termes $t_1 = \mathsf{h}^2(a) + b + c$ et $t_2 = \mathsf{h}(\langle a, b\rangle) + c$. On a $Fact_{\mathsf{E}}(t_1) = \{a, b, c\}$, $Fact_{\mathsf{E}}(t_2) = \{\langle a, b\rangle, c\}$, $St_{\mathsf{E}}(t_1) = \{t_1, a, b, c\}$ et $St_{\mathsf{E}}(t_2) = \{t_2, \langle a, b\rangle, a, b, c\}$.

Pour établir le lemme de localité (lemme 6.9), nous procédons par induction sur l'arbre de preuve minimale de $T \vdash u$. En fait, nous sommes obligés de renforcer l'hypothèse d'induction et de montrer un résultat légèrement plus fort pour les *preuves par décomposition.*

Définition 6.7 (preuve par décomposition)
Soit P un arbre de preuve. P est une preuve par décomposition *s'il satisfait une des deux conditions suivantes :*
- *P se termine par une instance d'une règle de décomposition :* (Proj_1), (Proj_2) *ou* (D),
- *P est arbre de preuve de $T \vdash u$ avec u standard et il se termine par une instance du schéma* $(\mathsf{M}_{\mathsf{E}})$.

Exemple 6.8
Considérons la théorie ACUNh. *Soit $T = \{b + \mathsf{h}^2(a) + \mathsf{h}^3(a), \langle \mathsf{h}^2(a), c\rangle\}$. La preuve P ci-dessous est une preuve de $T \vdash b$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$.*

$$\dfrac{T \vdash b + \mathsf{h}^2(a) + \mathsf{h}^3(a) \qquad \dfrac{T \vdash \langle \mathsf{h}^2(a), c\rangle}{T \vdash \mathsf{h}^2(a)}\,(\mathsf{Proj}_1)}{T \vdash b}\,(\mathsf{M}_{\mathsf{E}})$$

Le terme b est un terme standard, la preuve P est donc une preuve par décomposition.

Lemme 6.9 (localité)
Soit T un ensemble de termes et u un terme. Soit P une preuve minimale de $T \vdash u$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Les nœuds de P sont étiquetés par des termes dans $St_{\mathsf{E}}(T, u)$. Si de plus, la preuve P est une preuve par décomposition, alors les nœuds de P sont étiquetés par des termes dans $St_{\mathsf{E}}(T)$.

Démonstration.
Nous montrons ce résultat par induction sur P. Nous considérons tous les cas possibles en ce qui concerne la dernière règle d'inférence de P et nous concluons en appliquant l'hypothèse d'induction.
- Si P se termine par une instance de (C^-), alors P est de la forme :

$$\dfrac{P_1\left\{\ \dfrac{\cdots}{T \vdash u_1}\right. \quad \ldots \quad P_n\left\{\ \dfrac{\cdots}{T \vdash u_n}\right.}{T \vdash f(u_1, \ldots, u_n)}\,(\mathsf{C}^-)$$

avec $f \in \mathcal{F} \setminus sig(\mathsf{E})$. Par hypothèse d'induction, les nœuds de $P_1, \ldots, P_n$ sont étiquetés par des termes dans $St_{\mathsf{E}}(T, u_1, \ldots, u_n) \subseteq St_{\mathsf{E}}(T, f(u_1, \ldots, u_n))$. On en déduit donc que les nœuds de P sont étiquetés par des termes dans $St_{\mathsf{E}}(T, u)$, ce qui nous permet de conclure puisque, par définition, P n'est pas une preuve par décomposition.

– Si P se termine par une instance de (Proj_1) (ou Proj_2, D), alors P est une preuve par décomposition de la forme :

$$\dfrac{P_1 \left\{ \dfrac{\cdots}{T \vdash \langle u_1, u_2 \rangle} \right.}{T \vdash u_1}\ (\mathsf{Proj}_1)$$

Tout d'abord, P_1 est nécessairement une preuve par décomposition, par minimalité de P. Par hypothèse d'induction, les nœuds de P_1 sont étiquetés par des termes de $St_{\mathsf{E}}(T)$. Nous en déduisons que $\langle u_1, u_2 \rangle \in St_{\mathsf{E}}(T)$, et donc que $u_1 \in St_{\mathsf{E}}(T)$. Ceci nous permet de conclure.

– Le cas le plus intéressant est lorsque la dernière règle d'inférence utilisée est une instance de $(\mathsf{M}_{\mathsf{E}})$. Nous avons alors l'arbre de preuve suivant :

$$\dfrac{P_1 \left\{ \dfrac{\cdots}{T \vdash u_1} \right. \qquad \cdots \qquad P_n \left\{ \dfrac{\cdots}{T \vdash u_n} \right.}{T \vdash C[u_1, \ldots, u_n]\!\downarrow}\ (\mathsf{M}_{\mathsf{E}})$$

Par l'hypothèse d'induction, nous savons que chacune des preuves P_i ne contient que des termes dans $St_{\mathsf{E}}(T, u_i)$. De plus, si u_i n'est pas un terme standard, P_i est nécessairement une preuve par décomposition, par minimalité de P. Autrement, P_i se terminerait par une instance de la règle $(\mathsf{M}_{\mathsf{E}})$, seule façon d'obtenir un terme non-standard avec une preuve par « composition », et les deux instances de la règle $(\mathsf{M}_{\mathsf{E}})$ pourraient alors être regroupées en une seule. Nous pouvons donc appliquer l'hypothèse d'induction.

Maintenant, intéressons nous aux termes u_i standards. Soit $u_i \in St_{\mathsf{E}}(u)$, soit il existe u_j avec $(j \neq i)$ tel que $u_j = u_i + u'_j$. Autrement dit, le facteur u_i disparaît à l'aide du terme u_j. Dans ce cas, puisque la preuve P est minimale, u_j est non-standard, c'est-à-dire $u'_j \neq 0$. Sinon, le terme u_i apparaîtrait deux fois dans les prémisses, contredisant la minimalité de P. Ainsi, nous savons que $u_i \in St_{\mathsf{E}}(T)$. Tous les nœuds de P sont étiquetés par des termes dans $St_{\mathsf{E}}(T, u)$. De plus, si u est un terme standard, nous avons $u \in St_{\mathsf{E}}(u_i)$, où u_i est un terme standard. Nous avons donc que $u_i \in St_{\mathsf{E}}(T)$ et que tous les nœuds de P sont étiquetés par des termes dans $St_{\mathsf{E}}(T)$. □

6.2.3 Déductibilité en un pas

Dans cette partie, nous allons nous intéresser à la complexité de la déductibilité en un pas pour la règle $(\mathsf{M}_{\mathsf{E}})$. Notre méthode s'inspire de la technique utilisée dans [Nut90] pour résoudre les problèmes d'unification dans le cadre des théories monoïdales. Cette méthode à déjà été appliquée pour résoudre le problème de la déductibilité en un pas pour des schémas de règles plus simples (*e.g.* [CKRT03b, CKRT03a, LLT05a]).

6.2.3.1 Procédure

L'algorithme décrit ci-dessous réduit le problème de la déductibilité en un pas pour la règle $(\mathsf{M}_{\mathsf{E}})$ au problème de la satisfaisabilité d'un système d´équations linéaire dans $\mathbb{N}[\mathsf{h}]$, $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ ou $\mathbb{Z}[\mathsf{h}]$, suivant la théorie équationnelle E considérée.

Entrée : un ensemble fini $T = \{t_1, \dots t_n\}$ de termes clos et un terme s.

Sortie : Posons $\mathcal{B} = \{b \mid b \in \mathit{Fact}_{\mathsf{E}}(T \cup \{s\})\}$ et $m = |\mathcal{B}|$. L'algorithme retourne :
- une matrice A de taille $n \times m$ à coefficients dans $\mathbb{N}[\mathsf{h}]$ (resp. $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$, $\mathbb{Z}[\mathsf{h}]$), et
- un vecteur b de taille m à coefficients dans $\mathbb{N}[\mathsf{h}]$ (resp. $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$, $\mathbb{Z}[\mathsf{h}]$)

tels que s est déductible en une étape par ($\mathsf{M_E}$) avec E = ACh (resp. ACUNh, AGh) si, et seulement si, il existe $Y \in \mathbb{N}[\mathsf{h}]^n$ (resp. $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^n$, $\mathbb{Z}[\mathsf{h}]^n$) tel que $A \cdot Y = b$ (*cf.* lemme 6.11).

Algorithme : Posons $\mathcal{T}_{\mathcal{B}} = \{t \in \mathcal{T}(\mathcal{F}) \mid \mathit{Fact}_{\mathsf{E}}(t) \subseteq \mathcal{B}\}$, et écrivons $\mathcal{B} = \{b_1, \dots, b_m\}$. Considérons la fonction $\psi : \mathcal{T}_{\mathcal{B}} \to \mathbb{N}[h]^m$ (resp. $\mathbb{Z}/2\mathbb{Z}[h]^m$, $\mathbb{Z}[h]^m$) définie de la façon suivante :
- si $x = b_i$ alors $\psi(x) = (0, \dots, 0, 1, 0, \dots, 0)$, où 1 est à la $i^{\text{ème}}$ position,
- si $\mathsf{head}(x) = \mathsf{h}$, alors $x = \mathsf{h}^k(u)$ et $\psi(x) = \psi(u) \cdot \mathsf{h}^k$,
- si $\mathsf{head}(x) = +$, alors $x = u_1 + \dots + u_l$ et $\psi(x) = \sum_{1 \le i \le l} \psi(u_i)$,
- si $\mathsf{head}(x) = -$ (cas AGh), alors $x = -(u)$ et $\psi(x) = -\psi(u)$.

L'algorithme retourne $A = (\psi(t_1), \dots, \psi(t_n))$ et $b = \psi(s)$.

Exemple 6.10
Considérons les termes $t_1 = 2a_1 + 3\mathsf{h}(a_1) + 2\mathsf{h}^2(a_1)$, $t_2 = -3a_2 + 2\mathsf{h}^2(a_1)$ *et* $t_3 = \mathsf{h}(a_2) + \mathsf{h}^2(a_1)$. *Posons* $\mathcal{B} = \{a_1, a_2\}$. *Nous avons :*

$$\psi(t_1) \;=\; \begin{pmatrix} 2 + 3\mathsf{h} + 2\mathsf{h}^2 \\ 0 \end{pmatrix} \qquad \psi(t_2) \;=\; \begin{pmatrix} 2\mathsf{h}^2 \\ -3 \end{pmatrix} \qquad \psi(t_3) \;=\; \begin{pmatrix} \mathsf{h}^2 \\ \mathsf{h} \end{pmatrix}$$

Lemme 6.11
L'algorithme décrit ci-dessus est tel que $A \cdot Y = b$ a une solution dans $\mathbb{N}[\mathsf{h}]$ (resp. $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$, $\mathbb{Z}[\mathsf{h}]$) si, et seulement si, il existe un E*-contexte C tel que $C[t_1, \dots, t_n] =_{\mathsf{E}} s$ avec* E = ACh *(resp.* ACUNh, AGh*), c'est-à-dire si, et seulement si, s est déductible en un pas de $t_1, \dots, t_n$ par le schéma* ($\mathsf{M_E}$).

Démonstration.
La fonction ψ, décrite ci-dessus, associe à un terme $t \in \mathcal{T}_{\mathcal{B}}$, un élément $\psi(t)$ de $\mathbb{N}[\mathsf{h}]^m$ (resp. $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^m$, $\mathbb{Z}[\mathsf{h}]^m$), et est telle que : pour tous termes t, t', nous avons $\psi(t) = \psi(t')$ si, et seulement si, $t =_{\mathsf{E}} t'$. De même, nous définissons $\phi : \mathcal{C} \to \mathbb{N}[\mathsf{h}]^n$ (resp. $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^n$, $\mathbb{Z}[\mathsf{h}]^n$) où $\mathcal{C} = \{C \mid C \text{ est un E-contexte à } n \text{ variables}\}$. Notons que ϕ est surjective et que nous avons : $\psi(C[t_1, \dots, t_n]) = (\psi(t_1), \dots, \psi(t_n)) \cdot \phi(C)$. Nous avons donc :

$$\begin{aligned}
\exists Y \text{ tel que } A \cdot Y = b &\Leftrightarrow \exists Y \text{ tel que } (\psi(t_1), \dots, \psi(t_n)) \cdot Y = \psi(s) \\
&\Leftrightarrow \exists Y \text{ tel que } (\psi(t_1), \dots, \psi(t_n)) \cdot \phi(C) = \psi(s) \\
&\Leftrightarrow \exists C \text{ tel que } \psi(C[t_1, \dots, t_n]) = \psi(s) \\
&\Leftrightarrow \exists C \text{ tel que } C[t_1, \dots, t_n] =_{\mathsf{E}} s.
\end{aligned}$$
□

Exemple 6.12
Soient $T = \{a_1 + \mathsf{h}(a_1) + \mathsf{h}^2(a_1), a_2 + \mathsf{h}^2(a_1), \mathsf{h}(a_2) + \mathsf{h}^2(a_1)\}$ *et* $s = a_1 + \mathsf{h}^2(a_1)$ *où* a_1, a_2 *sont des termes standards. Nous avons :*

$$A \;=\; \begin{pmatrix} 1 + \mathsf{h} + \mathsf{h}^2 & \mathsf{h}^2 & \mathsf{h}^2 \\ 0 & 1 & \mathsf{h} \end{pmatrix} \qquad b \;=\; \begin{pmatrix} 1 + \mathsf{h}^2 \\ 0 \end{pmatrix}$$

L'équation $A \cdot Y = b$ a une solution dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$: $Y = (1 + \mathsf{h}, \mathsf{h}, 1)$ est une telle solution. Ainsi, le terme s est déductible en un pas par ($\mathsf{M_E}$) (E = ACUNh) *en utilisant le* *contexte* $x_1 + \mathsf{h}(x_1) + \mathsf{h}(x_2) + x_3$ *dans lequel* x_i *est utilisé pour représenter le* $i^{\text{ème}}$ *terme de* T.

6.2.3.2 Complexité

Nous avons vu, au chapitre 2, que différentes représentations des termes sont envisageables. Comme dans [LLT05a], nous pouvons considérer le cas où la taille d'un terme t est définie par le nombre de positions dans t. Dans [CKRT03a], la taille d'un terme t est définie par $|t|_{DAG} + \|t\|$ où $|t|_{DAG}$ est la taille de la représentation DAG des différents facteurs du terme t (*cf.* chapitre 2) et $\|t\|$ est le nombre de bits nécessaire pour représenter les coefficients (ici des polynômes) des facteurs apparaissant dans t. De toutes les façons, les résultats de complexité énoncés ci-dessous sont indépendants de la représentation choisie.

Résolution de système d'équations linéraires sur $\mathbb{N}[\mathsf{h}]$:

Nous pouvons voir que chacune des composantes d'un vecteur Y solution de $A \cdot Y = b$ a un degré plus petit que le degré de la composante correspondante dans b. La question de l'existence d'un Y tel que $A \cdot Y = b$ se réduit à la satisfaisabilité d'un système d'équations linéaires sur $\mathbb{N}$. Nous obtenons une procédure non-déterministe s'exécutant en temps polynomial. Ce problème est en fait NP-complet [LLT05a].

Résolution de système d'équations linéraires sur $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$

La satisfaisabilité d'un système d'équations linéaires est décidable en temps polynomial [KKS87].

Résolution de système d'équations linéraires sur $\mathbb{Z}[\mathsf{h}]$

Un résultat récent (théorème 6.5 dans [Asc04]) montre que s'il existe une solution au système d'équations $A \cdot Y = b$, alors il en existe une où le degré de chacune des composantes est borné par un polynôme dépendant du degré et de la taille des coefficients apparaissant dans A et b. Cette borne permet de réduire le problème à la satisfaisabilité d'un gros (mais polynomial) système d'équations sur $\mathbb{Z}$. Nous obtenons une procédure s'exécutant en temps polynomial pour la satisfaisabilité d'un système d'équations linéaires sur $\mathbb{Z}[\mathsf{h}]$ à partir de la procédure polynomiale connue pour la résolution de systèmes d'équations linéaires sur $\mathbb{Z}$ [Sch86].

Théorème 6.13
Le problème de déduction de l'intrus est décidable en temps polynomial pour les systèmes d'inférence $(\mathcal{I}_{\mathsf{DY}}, \mathsf{ACUNh})$ *et* $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$.

Démonstration.
Soient $T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$. Le cardinal de $St_{\mathsf{E}}(T, u)$ est polynomial en la taille de $T \cup \{u\}$. De plus, nous venons de voir que le problème de la deductibilité en un pas est décidable en temps polynomial dans chacun des deux cas (ACUNh et AGh). Nous en déduisons donc que l'algorithme proposé au chapitre 4 (Algorithme 4.1) s'exécute en temps polynomial. $\square$

6.2.4 Comparaison avec les travaux de P. Lafourcade *et al.*

Dans [LLT05a], P. Lafourcade *et al.* étudient le problème de déduction de l'intrus pour les trois théories équationnelles ACh, ACUNh et AGh. Ils montrent que le problème est NP-complet dans le cas de la théorie ACh et décidable en temps exponentiel dans le cas des théories ACUNh et AGh. Ils obtiennent des procédures polynomiales dans ces deux derniers cas en se restreignant au « cas binaire » : toutes les sommes apparaissant dans les termes (et sous-termes) sont binaires.

Pour obtenir leurs résultats, ils utilisent également l'approche décrite au chapitre 4. Mais, ils considèrent un système d'inférence différent du nôtre : ils normalisent les termes et ils introduisent le schéma classique (déjà utilisé dans [CKRT03b, CKRT03a]). Autrement dit, à la place de notre règle schéma (M_E), ils considèrent les deux règles suivantes :

$$\frac{T \vdash u_1 \quad \dots \quad T \vdash u_n}{T \vdash u_1 + \ldots + u_n\!\downarrow}\;(\mathsf{GX}) \qquad\qquad \frac{T \vdash u}{T \vdash \mathsf{h}(u)\!\downarrow}\;(\mathsf{h})$$

Toute la difficulté consiste alors à obtenir un résultat de localité pour une « bonne » notion de sous-termes. Pour ne pas avoir à considérer les sommes partielles, il est nécessaire d'utiliser le schéma (GX). Mais, comme le montre l'exemple ci-dessous, il semble difficile de contrôler la taille des « tours de h » à appliquer sur chacune des prémisses avant l'application du schéma (GX).

Exemple 6.14
Considérons la théorie équationnelle $\mathsf{E} = \mathsf{ACUNh}$. *Soit* T *l'ensemble de termes suivant :*

$$T = \{a_1 + \mathsf{h}(a_2);\ a_2 + \mathsf{h}(a_3);\ a_3 + \mathsf{h}(c);\ \mathsf{h}(c) + b_3;\ \mathsf{h}(b_3) + b_2;\ \mathsf{h}(b_2) + b_1\}.$$

Il existe une preuve de $T \vdash a_1 + b_1$ *dans le système d'inférence proposée par P. Lafourcade et al. Celle-ci est décrite ci-dessous :*

$$\frac{T \vdash a_1 + \mathsf{h}(a_2) \quad \dfrac{T \vdash a_2 + \mathsf{h}(a_3)}{T \vdash \mathsf{h}(a_2) + \mathsf{h}^2(a_3)} \quad \dfrac{\dfrac{T \vdash a_3 + \mathsf{h}(c)}{T \vdash \mathsf{h}(a_3) + \mathsf{h}^2(c)}}{T \vdash \mathsf{h}^2(a_3) + \mathsf{h}^3(c)} \quad \ldots \quad T \vdash \mathsf{h}(b_2) + b_1}{T \vdash a_1 + b_1}$$

Pour obtenir cette preuve, nous sommes obligés d'appliquer des tours de h *de hauteur 2, alors qu'aucun terme n'a une telle tour de* h *dans* T. *En fait, cet exemple se généralise très facilement, et illustre le fait qu'il est nécessaire d'appliquer des tours de* h *dont la taille semble difficilement contrôlable.*

Lorsque toutes les sommes apparaissant dans les termes sont des sommes binaires (c'est le cas de l'exemple 6.14), P. Lafourcade *et al.* montrent qu'il est possible de borner la hauteur de ces tours de h. Ils en déduisent une procédure polynomiale. La technique utilisée ne semble pas généralisable au-delà du « cas binaire ».

Nos travaux améliorent ces résultats en proposant des procédures polynomiales permettant de résoudre le problème de déduction de l'intrus dans le cas général.

6.3 Résolution de systèmes de contraintes bien formés

Cette partie est consacrée à la résolution de systèmes de contraintes de déduction bien formés dans ($\mathcal{I}_{\mathsf{DY}}$,E). La première partie de la procédure, assurant l'existence d'une solution dite *conservatrice*, est valable pour les trois théories équationnelles étudiées dans ce chapitre (ACh, ACUNh et AGh). Le reste de la procédure est spécifique à la théorie ACUNh. Le problème est en fait indécidable pour les théorie ACh et AGh (*cf.* partie 6.4).

Théorème 6.15
Le problème de la satisfaisabilité d'un système de contraintes bien formé dans ($\mathcal{I}_{\mathsf{DY}}$, ACUNh) *est décidable.*

Toute cette partie est consacrée à la preuve de ce théorème. Notre procédure est composée de plusieurs étapes, et nous montrons la correction et la complétude de chacune de ces étapes au fur et à mesure. Ainsi, nous commençons par montrer que nous pouvons nous limiter à chercher des solutions conservatrices (des substitutions qui n'introduisent pas de « nouvelles structures ») en restant complets (*cf.* partie 6.3.1). Ensuite nous montrons comment réduire le problème de la satisfaisabilité d'un système de contraintes dans ($\mathcal{I}_{\mathsf{DY}}$, ACUNh) au problème de la satisfaisabilité d'un ensemble fini de système de contraintes à résoudre dans ($\mathcal{I}_{\mathsf{M_E}}$, ACUNh). Autrement dit, cette étape de la procédure permet de traiter la « partie Dolev-Yao » du système d'inférence. Cette étape est décrite dans la partie 6.3.2. Il nous reste alors à résoudre le problème de la satisfaisabilité d'un système de contraintes pour le système d'inférence composé de l'unique schéma de règle ($\mathsf{M_E}$). Nous verrons comment résoudre ce problème dans les parties 6.3.3, 6.3.4 et 6.3.5.

6.3.1 Existence d'une solution conservatrice

La complétude de notre procédure est assurée par l'existence d'une solution conservatrice, c'est-à-dire n'introduisant pas de « nouvelles structures ».

Définition 6.16 (substitution conservatrice)
Soit $\mathcal{C}$ un système de contraintes, et σ une substitution. La substitution σ est conservatrice *par rapport à $\mathcal{C}$ si pour tout $x \in vars(\mathcal{C})$, on a $Fact_{\mathsf{E}}(x\sigma) \subseteq (St_{\mathsf{E}}(\mathcal{C}) \smallsetminus vars(\mathcal{C}))\sigma$.*

Exemple 6.17
Considérons le système de contraintes $\mathcal{C}$ suivant :

$$\begin{array}{lcl} a, \mathsf{h}(b) & \Vdash & \mathsf{h}(x) \\ a, \mathsf{h}(b), x & \Vdash & \langle a, b\rangle \end{array}$$

Une solution de $\mathcal{C}$ est $\sigma = \{x \mapsto \langle a,a\rangle + b\}$. Cette solution n'est pas conservatrice. En effet, nous avons $Fact_{\mathsf{E}}(\langle a,a\rangle + b) = \{\langle a,a\rangle, b\}$, et $\langle a,a\rangle \notin (St_{\mathsf{E}}(\mathcal{C}) \smallsetminus \{x\})\sigma$ puisque :

$$(St_{\mathsf{E}}(\mathcal{C}) \smallsetminus \{x\})\sigma = \{\mathsf{h}(b), b, \mathsf{h}(\langle a,a\rangle + b), \langle a,b\rangle, a\}.$$

Le lemme 6.21 montre que l'on peut toujours considérer que le système de contraintes que l'on souhaite résoudre admet une solution conservatrice. Avant d'établir ce lemme, nous avons besoin de montrer un résultat intermédiaire. Le lemme 6.20 a pour but d'assurer l'existence d'une preuve dans laquelle un terme donné n'est jamais décomposé. Le but est en fait de garantir que toutes les occurrences de ce « gros » terme vont pouvoir être remplacées par la constante 0 en préservant la structure des arbres de preuve. Ce résultat est utilisé dans le lemme 6.21.

Définition 6.18 (terme décomposé)
Soit P une preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ et v un terme standard. Le terme v est décomposé *dans P si :*
- *$v = \langle u_1, u_2\rangle$ et P contient une instance de* (Proj$_1$) *ou* (Proj$_2$) *dont la prémisse est étiquetée par $T \vdash \langle u_1, u_2\rangle$, ou*
- *$v = \{u_1\}_{u_2}$ et P contient une instance de* (D) *dont les prémisses sont étiquetées par $T \vdash \{u_1\}_{u_2}$ et $T \vdash u_2$.*

Exemple 6.19
Soit $T = \{a + h(a), \langle b, c\rangle\}$, et P la preuve de $T \vdash a + h^3(a) + -2h(b)$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ décrite ci-dessous ($\mathsf{E} = \mathsf{AGh}$). *Le terme standard $\langle b, c\rangle$ est décomposé dans P.*

$$\dfrac{T \vdash a + h(a) \qquad \dfrac{T \vdash \langle b, c\rangle}{T \vdash b}\,(\mathsf{Proj_1})}{T \vdash a + h^3(a) + -2h(b)}\,(\mathsf{M_E})$$

Le lemme ci-dessous est prouvé dans [RT03] dans le cas du modèle $\mathcal{I}_{\mathsf{DY}}$ (sans théorie équationnelle). Cette preuve s'étend sans difficulté à notre modèle d'intrus qui comprend, en plus des règles standards du modèle de Dolev-Yao, la règle ($\mathsf{M_E}$).

Lemme 6.20
Soit P un arbre de preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Soit P' une preuve minimale de $T \vdash \gamma$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ se terminant par une instance de (C^-). *Il existe un arbre de preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ dans lequel γ n'est jamais décomposé.*

Démonstration.
Nous montrons ce résultat par induction sur le nombre d'instances de règles d'inférence dans la preuve P qui décompose γ.

Cas de base : s'il n'y a aucune instance décomposant γ, P est la preuve recherchée.

Étape d'induction : supposons qu'il existe une telle instance décomposant γ dans P. Nous distinguons deux cas suivant que γ soit une paire (c'est-à-dire $\langle \gamma_1, \gamma_2\rangle$), ou un chiffré (c'est-à-dire $\{\gamma_1\}_{\gamma_2}$). Dans le premier cas, cela signifie qu'il existe une instance de (Proj$_1$) (ou Proj$_2$) ayant pour prémisse $\langle \gamma_1, \gamma_2\rangle$ et pour conclusion γ_1 (ou γ_2). À partir de P', nous pouvons extraire une preuve P_1 de $T \vdash \gamma_1$ (resp. P_2 de $T \vdash \gamma_2$). La preuve P_1 (resp. P_2) ne décompose pas γ par minimalité de P'. Une telle preuve « se branche » pour remplacer la sous-preuve de $T \vdash \gamma_1$ (resp. $T \vdash \gamma_2$) décomposant γ dans P. Le deuxième cas où $\gamma = \{\gamma_1\}_{\gamma_2}$ se traite de façon similaire. Nous obtenons une preuve de $T \vdash u$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ qui contient moins d'instances de règles d'inférence décomposant γ que n'en contenait la preuve P. □

Bien que le principal objectif de cette partie soit d'établir une procédure de décision dans le cas de la théorie ACUNh, le lemme 6.21 est vrai pour les trois théories ACh, ACUNh et AGh. Nous utiliserons ce lemme pour établir le résultat d'indécidabilité dans le cas de la théorie AGh (*cf.* partie 6.4.2).

D'autre part, nous rappelons que les termes sont systématiquement normalisés, et nous notons $u\sigma$ à la place de $u\sigma\!\downarrow$.

Lemme 6.21 (existence d'une solution conservatrice)
Soit $\mathcal{C}$ un système de contraintes bien formé. Si $\mathcal{C}$ admet une solution dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$, alors $\mathcal{C}$ admet une solution conservatrice.

Nous supposons donné un ordre $\prec$ sur les termes standards de $\mathcal{T}(\mathcal{F}, \mathcal{X})$ tel que 0 soit minimal pour cet ordre. Nous notons $\ll$ l'extension multi-ensembles de $\prec$, et nous écrivons $\sigma_1 \ll \sigma_2$ si $\mathit{Fact}_{\mathsf{E}}(\{x\sigma_1 \mid x \in \mathit{dom}(\sigma_1)\}) \ll \mathit{Fact}_{\mathsf{E}}(\{x\sigma_1 \mid x \in \mathit{dom}(\sigma_1)\})$.

Démonstration. (du lemme 6.21)
Soit σ une solution minimale (par rapport à $\ll$) de $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Nous montrons par contradiction que σ est une solution conservatrice. Supposons qu'il existe $x \in \mathit{vars}(\mathcal{C})$ et $v_x \in \mathit{Fact}_{\mathsf{E}}(x\sigma)$ tels que $v_x \notin (\mathit{St}_{\mathsf{E}}(\mathcal{C}) \smallsetminus \mathit{vars}(\mathcal{C}))\sigma$, c'est-à-dire pour tout $t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \smallsetminus \mathcal{X}$ avec $t\sigma =_E v_x$, nous avons $t \notin \mathit{St}_{\mathsf{E}}(\mathcal{C})$. Nous allons montrer que $\mathcal{C}$ admet une solution σ' plus petite que σ (par rapport à $\ll$). Posons $\mathcal{C} = \{C_1, \ldots, C_k\}$ et pour chaque $i \leq k$, notons $T_i \Vdash u_i$ la contrainte C_i.

Fait 1 *Soit $i \leq k$ et $s \in T_i$. Si $v_x \in \mathit{St}_{\mathsf{E}}(s\sigma)$, alors il existe $j < i$ tel que $v_x \in \mathit{St}_{\mathsf{E}}(u_j\sigma)$.*

Nous montrons ce premier résultat par contradiction. Supposons qu'il existe $i \leq k$ et $s \in T_i$ tels que $v_x \in \mathit{St}_{\mathsf{E}}(s\sigma)$ et pour tout $j < i$, nous avons $v_x \notin \mathit{St}_{\mathsf{E}}(u_j\sigma)$. Soit z une nouvelle variable, et ρ le remplacement $\{v_x \mapsto z\}$. Soit $\theta = \sigma\rho$. Nous allons montrer que $\mathcal{C}\theta$ ne satisfait pas la propriété d'initialisation en montrant que $z \in \mathit{vars}(T_i\theta)$ et que pour tout $j < i$, on a $z \notin \mathit{vars}(u_j\theta)$.

Tout d'abord, puisque $v_x \notin (\mathit{St}_{\mathsf{E}}(\mathcal{C}) \smallsetminus \mathit{vars}(\mathcal{C}))\sigma$, nous avons $(\mathcal{C}\sigma)\rho = \mathcal{C}(\sigma\rho)$ $(= \mathcal{C}\theta)$. Par hypothèse, $v_x \in \mathit{St}_{\mathsf{E}}(T_i\sigma)$, ainsi $z \in \mathit{vars}(T_i\theta)$. Cependant, pour tout $j < i$, nous avons $z \notin \mathit{vars}(u_j\theta)$ puisque $v_x \notin \mathit{St}_{\mathsf{E}}(u_j\sigma)$.

Posons $m = \min\{j \mid v_x \in \mathit{St}_{\mathsf{E}}(u_j\sigma)\}$. Cet entier représente l'indice de la première contrainte (obtenue après instanciation et normalisation) dans laquelle le facteur v_x apparaît.

Fait 2 *Il existe une preuve P' de $T_m\sigma \vdash v_x$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ se terminant par une instance de la règle d'inférence* (C^-).

Puisque σ est une solution de $\mathcal{C}$, il existe une preuve minimale P de $T_m\sigma \vdash u_m\sigma$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Nous montrons tout d'abord qu'il existe dans P un nœud étiqueté $T_m\sigma \vdash v_x$. Si P contient un tel nœud, alors c'est le nœud que l'on cherchait. Sinon, nous pouvons construire un chemin dans P, depuis la racine jusqu'à une feuille tel que chaque nœud de ce chemin étiqueté par $T_m\sigma \vdash u$ satisfasse la condition $v_x \in \mathit{St}_{\mathsf{E}}(u)$. L'existence d'un tel chemin contredit la minimalité de m. De plus, le lemme de localité (lemme 6.9) nous assure que la sous-preuve P' de $T_m\sigma \vdash v_x$ ne peut pas être une preuve par décomposition : dans le cas contraire on aurait $v_x \in \mathit{St}_{\mathsf{E}}(T_m\sigma)$. Puisque v_x est un terme standard, P' est une preuve de $T_m\sigma \vdash v_x$ se terminant par une instance de (C^-).

Maintenant, soit δ le remplacement $[v_x \mapsto 0]$ et $\sigma' = \sigma\delta$. Nous allons montrer que σ' est une solution de $\mathcal{C}$, aboutissant à une contradiction puisque $\sigma' \ll \sigma$. Pour montrer cela, nous devons construire pour tout $i \leq k$, une preuve de $C_i\sigma'$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Nous distinguons deux cas :

1. Cas $i < m$: Par définition de m, $v_x \notin St_{\mathsf{E}}(C_i\sigma)$. Nous avons $(C_i\sigma)\delta = C_i\sigma = C_i\sigma'$, et donc σ' est une solution de C_i.
2. Cas $i \geq m$: Le reste de la preuve est consacrée à montrer que σ' est aussi une solution de $C_i = T_i \Vdash u_i$.

Remarquons que $C_i(\sigma\delta) = (C_i\sigma)\delta$ puisque par hypothèse $v_x \notin (St_{\mathsf{E}}(\mathcal{C}) \smallsetminus vars(C))\sigma$. Nous savons que σ est une solution de C_i, cela signifie qu'il existe une preuve P de $T_i\sigma \vdash u_i\sigma$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. D'autre part, le fait 2 et la monotonie du système de contraintes $\mathcal{C}$ nous assurent l'existence d'une preuve de $T_i\sigma \vdash v_x$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ pour tout $i \geq m$. Maintenant, nous pouvons appliquer le lemme 6.20 pour obtenir une preuve P_i de $T_i\sigma \vdash u_i\sigma$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ dans laquelle v_x n'est jamais décomposé. Nous pouvons construire à partir de P_i une preuve P_i' de $(T_i\sigma)\delta \vdash (u_i\sigma)\delta$ en remplaçant chacun des sous-arbres se terminant par

$$\frac{T_i\sigma \vdash v_1 \quad \ldots \quad T_i\sigma \vdash v_n}{T_i\sigma \vdash v_x}\,(\mathsf{C}^-)$$

par une feuille étiquetée $T_i\sigma \vdash v_x$, et en appliquant δ à chacun des termes de l'arbre ainsi obtenu.

Fait 3 *P_i' est une preuve de $(T_i\sigma)\delta \vdash (u_i\sigma)\delta$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$.*

Pour prouver ce résultat, nous devons montrer que, pour chaque nœud dans P_i' étiqueté par $T_i\sigma\delta \vdash v_0$ et ayant n fils étiquetés respectivement par $T_i\sigma\delta \vdash v_1, \ldots T_i\sigma\delta \vdash v_n$, l'inférence

$$\frac{T_i\sigma\delta \vdash v_1 \quad \ldots \quad T_i\sigma\delta \vdash v_n}{T_i\sigma\delta \vdash v_0}$$

est une instance d'une règle de $(\mathcal{I}_{\mathsf{DY}},\mathcal{R}_{\mathsf{E}})$.

Nous distinguons deux cas :

- Si le nœud est une feuille ajoutée lors du remplacement d'une instance de (C^-) lors de la construction P_i' donnée ci-dessus, alors nous avons $v_0 = 0$. Cette inférence correspond donc à l'application d'une instance de $(\mathsf{M}_{\mathsf{E}})$.
- Dans le cas contraire, il existe une instance correspondante dans P_i. Cela signifie qu'il existe
$$\frac{T_i\sigma \vdash u_1 \quad \ldots \quad T_i\sigma \vdash u_n}{T_i\sigma \vdash u_0}$$
une inférence dans P_i telle que $v_i = u_i\delta$ pour tout i tel que $0 \leq i \leq n$. Puisque, par construction de P_i', nous savons que v_x n'est jamais décomposé dans P_i, et que la conclusion d'une instance de (C^-) ne peut pas être v_x, nous pouvons montrer, par cas sur la règle d'inférence utilisée, que lorsque nous appliquons δ sur l'instance ci-dessus, nous obtenons une autre instance de la même règle d'inférence.

Nous en concluons que σ' est une solution de $\mathcal{C}$ plus petite que σ. □

Exemple 6.22
Reprenons l'exemple 6.17

$$\mathcal{C} := \left\{ \begin{array}{lcl} a, \mathsf{h}(b) & \Vdash & \mathsf{h}(x) \\ a, \mathsf{h}(b), x & \Vdash & \langle a, b\rangle \end{array} \right.$$

Nous avons vu que $\sigma = \{x \mapsto \langle a, a\rangle + b\}$ est une solution de $\mathcal{C}$, mais que celle-ci n'est pas conservatrice à cause du facteur $\langle a, a\rangle$. La substitution $\sigma' = \sigma\{\langle a, a\rangle \mapsto 0\} = \{x \mapsto b\}$ est une solution conservatrice de $\mathcal{C}$.

Notre but est désormais de « relever » le lemme de localité, obtenu dans la partie 6.2.2, aux termes avec variables et d'établir que, dans le cas des solutions conservatrices, les nœuds d'une preuve minimale de $(T \Vdash u)\sigma$ sont étiquetés par des termes dans $St_{\mathsf{E}}(T, u)\sigma$. Nous commençons par montrer le lemme suivant :

Lemme 6.23
Soient t un terme et σ une substitution. Nous avons :

$$St_{\mathsf{E}}(t\sigma) \subseteq St_{\mathsf{E}}(t)\sigma \cup \bigcup_{x \in vars(t)} St_{\mathsf{E}}(x\sigma)$$

Démonstration.
Ce résultat se montre par induction structurelle sur t. Si t est une constante ou une variable, c'est immédiat. Maintenant, supposons que t soit un terme standard $f(t_1, \ldots, t_n)$ avec $f \in \mathcal{F} \smallsetminus sig(\mathsf{E})$. Nous avons :

$$\begin{array}{rcll} St_{\mathsf{E}}(t\sigma) & = & \{t\sigma\} \cup \bigcup_{i=1}^{n} St_{\mathsf{E}}(t_i\sigma) & \\ & \subseteq & \{t\sigma\} \cup \bigcup_{i=1}^{n} (St_{\mathsf{E}}(t_i)\sigma \cup \bigcup_{x \in vars(t_i)} St_{\mathsf{E}}(x\sigma)) & \text{par hypothèse d'induction} \\ & \subseteq & St_{\mathsf{E}}(f(t_1, \ldots, t_n)\sigma) \cup \bigcup_{x \in vars(\{t_1, \ldots, t_n\})} St_{\mathsf{E}}(x\sigma) & \\ & \subseteq & St_{\mathsf{E}}(t)\sigma \cup \bigcup_{x \in vars(t)} St_{\mathsf{E}}(x\sigma) & \end{array}$$

Finalement, si t n'est pas un terme standard, alors nous avons $t = C[t_1, \ldots, t_n]$ où les termes $t_1, \ldots, t_n$ sont standards et C est un E-contexte, et nous pouvons faire le même raisonnement que précédemment. □

Ce résultat s'étend à n'importe quel ensemble de termes. Comme illustré par l'exemple ci-dessous, compte tenu de la normalisation des termes, cette inclusion peut être stricte.

Exemple 6.24
Soient $t = x + y$ et $\sigma = \{x \mapsto a; y \mapsto a\}$. Nous avons $St_{\mathsf{E}}(t)\sigma \cup St_{\mathsf{E}}(\{x\sigma, y\sigma\}) = \{0, a\}$ alors que $St_{\mathsf{E}}(t\sigma) = \{0\}$.

Proposition 6.25
Soit σ une solution conservatrice de $\mathcal{C} = \{C_1, \ldots, C_k\}$. Pour tout $i \leq k$, il existe une preuve P_i de $C_i\sigma$ dont tous les nœuds sont étiquetés par des termes dans $St_{\mathsf{E}}(\mathcal{C})\sigma$.

Démonstration.
Grâce au lemme de localité (lemme 6.9), nous savons que pour tout $i \leq k$, il existe une preuve P_i de $T_i\sigma \vdash u_i\sigma$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ dont tous les nœuds sont étiquetés par des termes dans $St_{\mathsf{E}}(\mathcal{C}\sigma)$. Le lemme 6.23 nous assure que $St_{\mathsf{E}}(\mathcal{C}\sigma) \subseteq St_{\mathsf{E}}(\mathcal{C})\sigma \cup \bigcup_{x \in vars(\mathcal{C})} St_{\mathsf{E}}(x\sigma)$. Ainsi, nous avons :

$$\begin{array}{rcll} St_{\mathsf{E}}(\mathcal{C}\sigma) & \subseteq & St_{\mathsf{E}}(\mathcal{C})\sigma \cup \displaystyle\bigcup_{x \in vars(\mathcal{C})} St_{\mathsf{E}}(x\sigma) & \\ & \subseteq & (St_{\mathsf{E}}(\mathcal{C}) \smallsetminus vars(\mathcal{C}))\sigma \cup \displaystyle\bigcup_{x \in vars(\mathcal{C})} St_{\mathsf{E}}(x\sigma) & \\ & \subseteq & S_{\mathsf{E}}(\mathcal{C})\sigma & \text{puisque } \sigma \text{ est conservatrice} \end{array}$$

où $S_{\mathsf{E}}(\mathcal{C}) = \{C[t_1, \ldots, t_n] \mid \forall i.\ t_i \in St_E(\mathcal{C}) \smallsetminus vars(\mathcal{C})$ et C est un E-contexte$\}$.

Soit $\dfrac{T_i\sigma \vdash u_1 \ldots T_i\sigma \vdash u_n}{T_i\sigma \vdash u_0}$ une inférence dans P_i autre qu'une instance de la règle ($\mathsf{M_E}$).
Pour tout j tel que $1 \leq j \leq n$, nous avons $u_j \in St_{\mathsf{E}}(\mathcal{C})\sigma$. Maintenant, nous devons regarder les instances de la règle ($\mathsf{M_E}$). Par minimalité de P_i, une instance de la règle ($\mathsf{M_E}$) ne peut pas être suivie d'une autre instance de la règle ($\mathsf{M_E}$) (nous pourrions fusionner ces deux instances). Ainsi, pour chaque prémisse $T_i\sigma \vdash u$ d'une instance de ($\mathsf{M_E}$),
- soit $T_i\sigma \vdash u$ est la conclusion d'une instance d'une règle d'inférence autre que ($\mathsf{M_E}$),
- soit $u \in T_i\sigma$.

De même, nous avons que la conclusion $T_i\sigma \vdash u$ d'une instance de ($\mathsf{M_E}$) est :
- soit la prémisse d'une instance d'une règle d'inférence autre que ($\mathsf{M_E}$),
- soit $u = u_i\sigma$.

Ainsi, nous en concluons qu'il existe une preuve P_i de $T_i\sigma \vdash u_i\sigma$ dont tous les nœuds sont étiquetés par des termes dans $St_{\mathsf{E}}(\mathcal{C})\sigma$. □

6.3.2 De la résolution dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ à la résolution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$

Dans cette partie, nous allons réduire le problème de la satisfaisabilité d'un système de contraintes dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$) à la satisfaisabilité d'un système de contraintes dans ($\mathcal{I}_{\mathsf{M_E}}$,$\mathcal{R}_{\mathsf{E}}$). Nous procédons en deux étapes :

1. Nous réduisons notre problème à la satisfaisabilité d'un système de contraintes de déduction en un pas (noté $\Vdash_1$) (lemme 6.26).
2. Ensuite, nous réduisons le problème ainsi obtenu à la satisfaisabilité d'un système de contraintes (en un pas) dans ($\mathcal{I}_{\mathsf{M_E}}$,$\mathcal{R}_{\mathsf{E}}$). Autrement dit, toutes les contraintes se résolvent par application du schéma ($\mathsf{M_E}$) (lemme 6.30).

L'algorithme non-déterministe 6.5 consiste à deviner parmi les sous-termes de $\mathcal{C}$ ceux qui sont déductibles par l'intrus. Ensuite, chacun de ces sous-termes est inséré dans le système de contraintes. La complétude de cette étape est essentiellement due à l'existence d'une solution conservatrice (lemme 6.21) et à la proposition 6.25. Le système de contraintes résultant est un système de contraintes en un pas : chacune des contraintes est satisfaisable par l'application d'une seule règle d'inférence.

Lemme 6.26
Sot $\mathcal{C}$ un système de contraintes bien formé. Soit $\mathscr{C}'$ l'ensemble des systèmes de contraintes de déduction en un pas obtenu en appliquant l'algorithme 6.5 sur $\mathcal{C}$.

1. *$\mathscr{C}'$ est un ensemble fini de systèmes de contraintes de déduction en un pas. Ces systèmes sont bien formés.*

```
Entrée: C = {T_1 ⊩ u_1, ..., T_k ⊩ u_k}
Sortie: C'

Algorithme:
  deviner S ⊆ St_E(C).
  pour chaque s ∈ S, deviner j(s) ∈ {1, ..., k}.
  C' := ∅
  pour i = 1 à k faire
     S_i := {s | j(s) = i}.
     choisir un ordre total sur S_i (S_i = {s_i^1, ..., s_i^{ki}})
     pour j = 1 à ki faire
        T := T_i ∪ S_1 ... ∪ S_{i-1} ∪ {s_i^1,..., s_i^{j-1}}
        C' := C' ∪ {T ⊩_1 s_i^j}
     fin
     C' := C' ∪ {T ⊩_1 u_i}
  fin
  retourner C'.
```

Algorithme 6.5 - Procédure pour obtenir les systèmes de contraintes de déduction en un pas.

2. *Soit $\mathcal{C}' \in \mathscr{C}'$. Si σ est une solution de $\mathcal{C}'$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ alors σ est une solution de $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$.*
3. *Si σ est une solution conservatrice de $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ alors il existe $\mathcal{C}' \in \mathscr{C}'$ tel que σ est une solution de $\mathcal{C}'$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$.*
4. *Soit $\mathcal{C}' \in \mathscr{C}'$, σ est une substitution conservatrice par rapport à $\mathcal{C}$ si, et seulement si, σ est conservatrice par rapport à $\mathcal{C}'$.*

Démonstration.

1. L'algorithme 6.5 est non-déterministe. À chaque étape, il y a un nombre fini de choix à considérer. L'ensemble $\mathscr{C}'$ obtenu en considérant tous les choix possibles est donc fini. Par construction, chacun des systèmes dans $\mathscr{C}'$ ne contient que des contraintes de déduction en un pas. Soit $\mathcal{C}'$ un système de contraintes dans $\mathscr{C}'$. Tout d'abord $\mathcal{C}'$ est monotone par construction et par monotonie de $\mathcal{C}$. Pour montrer que $\mathcal{C}'$ est bien formé, il nous suffit d'observer que chaque terme apparaissant en hypothèse d'une contrainte (c'est-à-dire à gauche d'une contrainte) est soit un terme introduit par l'algorithme (c'est-à-dire un terme dans S) ou un terme issu de la partie gauche d'une contrainte de $\mathcal{C}$. Dans le premier cas, cela signifie que le terme apparaît avant à droite d'une contrainte. Dans le deuxième cas, nous concluons grâce au fait que $\mathcal{C}$ est bien formé.
2. Soit $T_i \Vdash u_i \in \mathcal{C}$, on peut lui associer une contrainte $T_i \cup S_1 \cup \ldots \cup S_i \Vdash_1 u_i \in \mathcal{C}'$. Par hypothèse, σ est une solution de $\mathcal{C}'$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Cela signifie que $u_i\sigma$ est déductible en un pas à partir de $T_i\sigma \cup S_1\sigma \cup \ldots \cup S_i\sigma$. Par construction de $\mathcal{C}'$, nous pouvons montrer que chaque terme dans $S_j\sigma$ est déductible à partir des termes dans $T_j\sigma$. Intuitivement, chaque preuve est obtenue en empilant les « preuves de déductibilité en un pas » dans

le bon ordre. À partir de là, nous en déduisons que $u_i\sigma$ est déductible de $T_1\sigma \cup \ldots \cup T_i\sigma$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$). Par monotonie, il existe une preuve de $T_i\sigma \vdash u_i\sigma$ dans ($\mathcal{I}_{\mathsf{DY}}$, $\mathcal{R}_{\mathsf{E}}$).

3. Par hypothèse, pour chaque contrainte $T_i \Vdash u_i \in \mathcal{C}$, il existe une preuve P_i de $T_i\sigma \vdash u_i\sigma$. Puisque σ est une solution conservatrice de $\mathcal{C}$, nous pouvons supposer, grâce à la proposition 6.25, que tous les nœuds de P_i sont étiquetés par des termes dans $St_{\mathsf{E}}(\mathcal{C})\sigma$. Posons $S'_i = \{s \in St_{\mathsf{E}}(\mathcal{C}) \mid T_i\sigma \vdash s\sigma\}$. Intuitivement, S'_i contient tous les sous-termes de $\mathcal{C}$ qui, après instanciation par σ, sont déductibles à l'étape i, c'est-à-dire en utilisant des termes dans T_i. Il est important de remarquer que, grâce à la monotonie de $\mathcal{C}$, nous avons $S'_i \subseteq S'_{i+1}$ pour tout i compris entre 1 et ℓ.

   Maintenant, posons $S_1 = S'_1$ et $S_i = S'_i \smallsetminus (S'_1 \cup \ldots \cup S'_{i-1})$. L'ensemble S_i contient tous les sous-termes de $\mathcal{C}$ dont les instances par σ sont déductibles à l'étape i mais pas à l'étape $i-1$. Pour chaque i, nous ordonnons les éléments de S_i afin de satisfaire la propriété suivante : pour tout $s, s' \in S_i$ tels que $T_i\sigma \vdash s\sigma$ est la racine d'une sous-preuve minimale de $T_i\sigma \vdash s'\sigma$, alors $s \prec_i s'$. Ainsi, par construction, pour chaque $s \in S_i$, nous avons que $s\sigma$ est déductible en un pas de $S_1\sigma \cup \ldots \cup S_{i-1}\sigma \cup \{s'\sigma \mid s' \prec_i s \text{ et } s' \in S_i\}$. Il reste à montrer que $u_i\sigma$ est déductible en un pas de $T_i\sigma \cup S_1\sigma \cup \ldots \cup S_i\sigma$. Par définition de S_j et par le fait que $u_i\sigma$ est déductible au moins à l'étape i, nous savons que $u_i \in S_1 \cup \ldots \cup S_i$. Nous en déduisons que $u_i\sigma \in T_i\sigma \cup S_1\sigma \cup \ldots \cup S_i\sigma$.

4. Soit $\mathcal{C}' \in \mathscr{C}'$, nous avons $St_{\mathsf{E}}(\mathcal{C}') = St_{\mathsf{E}}(\mathcal{C})$, d'où le résultat. □

Maintenant, nous allons éliminer les contraintes de déduction en un pas satisfaisables par application d'une inférence autre que le schéma (M_{E}) (*cf.* lemme 6.30). Pour cela, nous avons besoin de quelques résultats sur l'unification modulo ACUNh [DLLT05].

Théorème 6.27 ([DLLT05])
Le problème de l'unification modulo la théorie équationnelle ACUNh *est décidable et finitaire.*

Nous avons besoin d'un résultat plus fin pour contrôler les sous-termes introduits lors de l'application d'une substitution solution d'un problème d'unification particulier.

Lemme 6.28 ([DLLT05])
Soit P un problème d'unification dans ACUNh *et* E-*mgu*(P) *un ensemble complet et minimale d'* E-*unificateurs de P. Pour tout $\theta \in$* E-*mgu*(P)*, pour tout $x \in dom(\theta)$ et pour tout $v \in St_{\mathsf{E}}(x\theta) \smallsetminus vars(x\theta)$, il existe $t \in St_{\mathsf{E}}(P)$ tel que $v =_{\mathsf{E}} t\theta$.*

Nous introduisons la notion de substitution non-effondrante.

Définition 6.29 (substitution non-effondrante)
Soit $\mathcal{C}$ un système de contraintes de déduction. Une substitution σ est dite non-effondrante par rapport à $\mathcal{C}$ si pour tout $u, v \in St_{\mathsf{E}}(\mathcal{C}) \smallsetminus \mathcal{X}$ tels que $u\sigma =_{\mathsf{E}} v\sigma$, on a $u =_{\mathsf{E}} v$.

Le lemme 6.30 nous permet de réduire la satisfaisabilité d'un système de contraintes de déduction en un pas à la satisfaisabilité d'un système de contraintes dans ($\mathcal{I}_{\mathsf{M}_{\mathsf{E}}}$,$\mathcal{R}_{\mathsf{E}}$). Premièrement, nous devinons un ensemble d'égalités entre sous-termes. Nous obtenons un problème d'unification que nous résolvons. Ce dernier admet un ensemble complet et fini de solutions (théorème 6.27). Nous appliquons un des unificateur ainsi obtenu. Ensuite, les contraintes sastisfaisables par application d'une règle d'inférence classique (autre que M_{E}) sont reconnaissables syntaxiquement et nous pouvons les éliminer.

Lemme 6.30
Soit $\mathcal{C}$ un système bien formé de contraintes de déduction en un pas. Posons :

$$\mathcal{P} = \{ \bigwedge_{(s_1,s_2)\in S'} s_1 = s_2 \mid S' \subseteq St_E(\mathcal{C})^2\}.$$

Soit $\mathscr{C}' = \{\mathcal{C}_\theta \mid R \in \mathcal{P}$ et $\theta \in$ E-mgu$(R)\}$ où

$$\mathcal{C}_\theta = \{T\theta \Vdash u\theta \mid T \Vdash_1 u \in \mathcal{C} \text{ et } T\theta \not\vdash_1 u\theta \text{ par } (\mathsf{Proj_1}), (\mathsf{Proj_2}), (\mathsf{D}) \text{ ou } (\mathsf{C}^-)\}.$$

1. *$\mathscr{C}'$ est un ensemble fini de systèmes de contraintes bien formés.*
2. *Soit $\mathcal{C}' \in \mathscr{C}'$. Si σ est une solution de $\mathcal{C}'$ dans $(\mathcal{I}_{\mathsf{M_E}},\mathcal{R}_\mathsf{E})$ alors σ est une solution de $\mathcal{C}$ dans $(\mathcal{I}_\mathsf{DY},\mathcal{R}_\mathsf{E})$.*
3. *Si $\mathcal{C}$ a une solution conservatrice dans $(\mathcal{I}_\mathsf{DY},\mathcal{R}_\mathsf{E})$ alors il existe $\mathcal{C}' \in \mathscr{C}'$ ayant une solution non-effondrante dans $(\mathcal{I}_{\mathsf{M_E}},\mathcal{R}_\mathsf{E})$.*

Démonstration.

1. L'ensemble $\mathcal{P}$ est fini puisque $|St_\mathsf{E}(\mathcal{C})|$ est fini. Soit $R \in \mathcal{P}$, l'ensemble E-mgu(R) est également fini (théorème 6.27). Soit $\theta \in \mathsf{E} - \mathsf{mgu}(R)$ et $\mathcal{C}_\theta$ le système de contraintes obtenu en utilisant la substitution θ. Nous devons montrer que $\mathcal{C}_\theta$ est un système de contraintes bien formé. La propriété de monotonie est satisfaite. Nous devons montrer que la propriété d'initialisation est stable par substitution. Soit σ une substitution, montrons que $\mathcal{C}_\theta\sigma$ satisfait la propriété d'initialisation. Posons $\mathcal{C}' = \mathcal{C}\theta\sigma$. Puisque $\mathcal{C}$ est bien formé, nous en déduisons que $\mathcal{C}'$ est un système bien formé. Il nous reste à montrer que les contraintes que nous sommes amenés à supprimer pour obtenir $\mathcal{C}_\theta$ à partir de $\mathcal{C}'$ ne change rien en ce qui concerne la bonne formation du système. Autrement dit, nous devons montrer qu'une contrainte $T\theta \Vdash_1 u\theta$ satisfaisant la « condition de suppression » ne peut pas introduire une variable pour la première fois. Autrement dit, s'il existe $x \in vars(u\theta)$ alors $x \in vars(T\theta)$. Par hypothèse, nous savons que $u\theta$ est déductible en un pas de $T\theta$ avec $(\mathsf{Proj_1})$, $(\mathsf{Proj_2})$, (D) ou (C^-). On en déduit donc que $vars(u\theta) \subseteq vars(T\theta)$.
2. Soit $\mathcal{C}' \in \mathscr{C}'$. Il existe $R \in \mathcal{P}$ et $\theta \in$ E-mgu(R) tel que

$$\mathcal{C}' = \{T\theta \Vdash u\theta \mid T \Vdash_1 u \in \mathcal{C} \text{ et } T\theta \not\vdash_1 u\theta \text{ par } (\mathsf{Proj_1}), (\mathsf{Proj_2}), (\mathsf{D}) \text{ ou } (\mathsf{C}^-)\}.$$

   Soit θ' une solution de $\mathcal{C}'$ dans $(\mathcal{I}_{\mathsf{M_E}},\mathcal{R}_\mathsf{E})$. Nous allons montrer que $\theta\theta'$ est une solution de $\mathcal{C}$. Soit $T \Vdash_1 u \in \mathcal{C}$. Nous considérons deux cas : soit $T\theta \vdash_1 u\theta$ par $(\mathsf{Proj_1})$, $(\mathsf{Proj_2})$, (D) ou (C^-), soit $T\theta \Vdash u\theta \in \mathcal{C}'$. Dans les deux cas, cela signifie que $T\theta\theta' \vdash u\theta\theta'$ dans $(\mathcal{I}_\mathsf{DY},\mathcal{R}_\mathsf{E})$. Ainsi, $\theta\theta'$ est une solution de $\mathcal{C}$ dans $(\mathcal{I}_\mathsf{DY},\mathcal{R}_\mathsf{E})$.
3. Soit σ une solution conservatrice de $\mathcal{C}$ dans $(\mathcal{I}_\mathsf{DY},\mathcal{R}_\mathsf{E})$. Posons

$$R = \{(s_1, s_2) \mid s_1, s_2 \in St_E(\mathcal{C}) \text{ et } s_1\sigma =_\mathsf{E} s_2\sigma\}$$

   Soit $\theta \in$ E-mgu(R) tel que θ soit plus général que σ. Soit θ' la substitution telle que $\theta \circ \theta' =_\mathsf{E} \sigma$. Posons

$$\mathcal{C}_\theta = \{T\theta \Vdash u\theta \mid T \Vdash_1 u \in \mathcal{C} \text{ et } T\theta \not\vdash_1 u\theta \text{ par } (\mathsf{Proj_1}), (\mathsf{Proj_2}), (\mathsf{D}) \text{ ou } (\mathsf{C}^-)\}.$$

Nous allons montrer que θ' est une solution de $\mathcal{C}_\theta$ dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}},\mathcal{R}_\mathsf{E})$.

Soit $T \Vdash_1 u \in \mathcal{C}$ tel que $T\sigma \vdash_1 u\sigma$ par (Proj$_1$), (Proj$_2$), (D) ou (C$^-$). Nous allons montrer que $T\theta \vdash_1 u\theta$ par (Proj$_1$), (Proj$_2$), (D) ou (C$^-$). Ainsi, les contraintes restantes dans $\mathcal{C}_\theta$ sont celles satisfaisables par la règle (M_E). Si $u\sigma \in T\sigma$, cela signifie qu'il existe $t \in T$ tel que $t\sigma = u\sigma$. Nous avons $t\theta = u\theta$, puisque $t, u \in St_\mathsf{E}(\mathcal{C})$, et donc $u\theta \in T\theta$: $u\theta$ est déductible en un pas de $T\theta$. Sinon, $T\sigma \vdash_1 u\sigma$ par (Proj$_1$), (Proj$_2$), (D) ou (C$^-$).

Considérons la règle (C$^-$). Dans ce cas, il existe $v_1, \ldots, v_n \in T\sigma$ et $f \in \mathcal{F} \smallsetminus sig(\mathsf{E})$ tels que $u\sigma = f(v_1, \ldots, v_n)$. Pour tout $i \leq n$ il existe $v'_i \in T$ tel que $v_i = v'_i\sigma$. Nous devons distinguer deux cas :

- Si u n'est pas une variable, alors $u = f(u'_1, \ldots, u'_n)$ et nous avons $u'_i, v'_i \in St_\mathsf{E}(\mathcal{C})$ et $u'_i\sigma = v'_i\sigma$ pour tout $i \leq n$. Nous en déduisons que $u'_i\theta = v'_i\theta$. Nous avons donc $T\theta \vdash_1 u\theta$ par (C$^-$).
- Si u est une variable, cela signifie (puisque σ est une solution conservatrice de $\mathcal{C}$) qu'il existe $t \in St_\mathsf{E}(\mathcal{C}) \smallsetminus vars(\mathcal{C})$ tel que $u\sigma =_\mathsf{E} t\sigma$. Il existe $t_1, \ldots, t_n \in St_\mathsf{E}(\mathcal{C})$ tels que $t = f(t_1, \ldots, t_n)$. Nous en déduisons que $t_i = v'_i$. Ainsi $T\theta \vdash_1 u\theta$ par (C$^-$).

Les autres cas (Proj$_1$), (Proj$_2$) et (D) sont similaires.

Pour finir, nous devons montrer que θ' est non-effondrante par rapport au système $\mathcal{C}_\theta$. Soient $u, v \in St_E(\mathcal{C}_\theta) \smallsetminus \mathcal{X}$. Nous avons $u, v \in St_E(\mathcal{C})\theta \cup \bigcup_{x \in vars(\mathcal{C})} St_E(x\theta)$ (lemme 6.23), et donc $u, v \in St_E(\mathcal{C})\theta$. Par le lemme 6.28, il existe $u_1, v_1 \in St_E(\mathcal{C})$ tels que $u = u_1\theta$ et $v = v_1\theta$. Supposons que $u\theta' = v\theta'$; alors nous avons $u_1\theta\theta' = v_1\theta\theta'$. Nous en déduisons donc que $u_1\sigma = v_1\sigma$. Par définition de R, nous avons $(u_1, v_1) \in R$ et par construction de θ, nous en déduisons que $u_1\theta = v_1\theta$, et que donc $u = v$. Nous en déduisons donc que θ' est une solution non-effondrante de $\mathcal{C}_\theta$ dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}},\mathcal{R}_\mathsf{E})$. □

6.3.3 Réduction de la signature

Maintenant, nous devons résoudre des systèmes de contraintes bien formés dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$. Nous réduisons la satisfaisabilité de systèmes de contraintes dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$ à la satisfaisabilité de systèmes de contraintes sur une signature réduite comprenant simplement les symboles de $sig(\mathsf{E})$ (c'est-à-dire 0, $+$ et h), et des constantes (*cf.* lemme 6.31).

Notation : Si $\rho : M \to N$ est un remplacement, c'est-à-dire une bijection entre deux ensembles finis de termes M et N, alors nous notons t^ρ, le terme obtenu à partir de t en remplaçant toutes les occurrences d'un sous-terme $s \in M$ par $s\rho$. Nous effectuons ces remplacements en commençant par les positions les « plus hautes » dans le terme t considéré. Nous étendons cette notation à un système de contraintes.

Lemme 6.31
Soit $\mathcal{C}$ un système de contraintes et $F = Fact_\mathsf{E}(\mathcal{C}) \smallsetminus \mathcal{X}$. Soit $\mathcal{F}_0$ un ensemble de constantes « fraîches » de même cardinalité que F et $\rho : F \to \mathcal{F}_0$ une bijection.

1. *Si $\mathcal{C}$ a une solution non-effondrante dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$ alors $\mathcal{C}^\rho$ a une solution dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$.*
2. *Si $\mathcal{C}^\rho$ a une solution dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$ alors $\mathcal{C}$ a une solution dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$.*

Démonstration.

1. Soit σ une solution non-effondrante de $\mathcal{C}$. Pour tout $v_1, v_2 \in Fact_{\mathsf{E}}(\mathcal{C}) \smallsetminus \mathcal{X}$, nous avons :
$$v_1\sigma = v_2\sigma \Rightarrow v_1 = v_2.$$
Pour tout $v_1, v_2 \in Fact_{\mathsf{E}}(\mathcal{C}) \smallsetminus \mathcal{X}$ tels que $v_1\sigma = v_2\sigma$, nous avons $v_1^\rho = v_2^\rho$. Nous en déduisons donc que σ^ρ est une solution de $\mathcal{C}^\rho$ dans $(\mathcal{I}_{\mathsf{M}_{\mathsf{E}}}, \mathcal{R}_{\mathsf{E}})$.
2. Soit σ une solution de $\mathcal{C}^\rho$ dans $(\mathcal{I}_{\mathsf{M}_{\mathsf{E}}}, \mathcal{R}_{\mathsf{E}})$. Alors, $\sigma^{(\rho^{-1})}$ est une solution de $\mathcal{C}$. $\square$

Exemple 6.32
Considérons le système $\mathcal{C}$ décrit ci-dessous. Nous avons $Fact_{\mathsf{E}}(\mathcal{C}) \smallsetminus \mathcal{X} = \{a, b, \langle x_1, x_2\rangle\}$. Posons $\rho = [a \mapsto a_1;\ b \mapsto a_2;\ \langle x_1, x_2\rangle \mapsto a_3]$. Nous obtenons le système $\mathcal{C}^\rho$ suivant :

$$\mathcal{C} = \left\{ \begin{array}{rcl} a & \Vdash & \langle x_1, x_2\rangle \\ a, x_1, x_2 & \Vdash & b \end{array} \right. \qquad \mathcal{C}^\rho = \left\{ \begin{array}{rcl} a_1 & \Vdash & a_3 \\ a_1, x_1, x_2 & \Vdash & a_2 \end{array} \right.$$

6.3.4 À propos des systèmes bien formés

Nous devons assurer que les systèmes obtenus après abstraction sont des systèmes bien formés (*cf.* partie 6.3.4.2). Il est important de maintenir cette propriété de bonne formation puisque la procédure de décision développée dans la partie 6.3.5 exploite cette hypothèse. Pour obtenir ce résultat, nous avons été amenés à considérer une nouvelle caractérisation des systèmes bien formés sur signature réduite (*cf.* partie 6.3.4.1).

6.3.4.1 Une nouvelle caractérisation des systèmes bien formés

Notre nouvelle caractérisation des systèmes bien formés est une caractérisation algébrique. Elle s'exprime en terme de dépendance entre vecteurs. Ces derniers sont calculés à partir de certains termes du système de contraintes considéré. Cette nouvelle caractérisation est énoncée dans la proposition 6.40. Nous commençons par introduire quelques notations, puis nous décrivons comment construire l'ensemble de vecteurs $\mathcal{B}(\mathcal{C})$. Enfin, nous montrons que la nouvelle caractérisation proposée est équivalente à la notion de bonne formation que nous avions introduite auparavant.

Définition 6.33 (opération $\odot$)
Soit $p = b_1\mathsf{h} + b_2\mathsf{h}^2 + \ldots + b_n\mathsf{h}^n$ un polynôme dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ et t un terme. Le produit $\odot$ de p par t est le terme $b_1\mathsf{h}(t) + b_2\mathsf{h}^2(t) + \ldots + b_n\mathsf{h}^n(t)$.

Exemple 6.34
Soient $t_1 = x_1 + a$ et $t_2 = \{x_1\}_{x_2} + x_1$. Soit $p = (\mathsf{h}^2 + 1)$. Nous avons :

$$\begin{array}{rcl} p \odot t_1 & = & (\mathsf{h}^2+1) \odot (x_1 + a) \\ & = & \mathsf{h}^2(x_1) + x_1 + \mathsf{h}^2(a) + a, \end{array} \quad \textit{et} \quad \begin{array}{rcl} p \odot t_2 & = & (\mathsf{h}^2+1) \odot \{x_1\}_{x_2} + x_1 \\ & = & \mathsf{h}^2(\{x_1\}_{x_2}) + \{x_1\}_{x_2} + \mathsf{h}^2(x_1) + x_1 \end{array}$$

Définition 6.35 (vecteur $\vec{t}$)
Soit t un terme tel que $vars(t) = \{x_1, \ldots, x_p\}$, le terme t s'écrit $t^{x_1} \odot x_1 + \ldots t^{x_p} \odot x_p + t^0$ avec $t^{x_1}, \ldots, t^{x_p} \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ et $t^0 \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ tel que $Fact_{\mathsf{E}}(t^0) \cap \mathcal{X} = \varnothing$. Nous notons $\vec{t}$ le vecteur $(t^{x_1}, \ldots, t^{x_p})$.

Exemple 6.36
Considérons les termes $t_1 = x_1 + \mathsf{h}(x_2) + x_2 + a$ *et* $t_2 = \mathsf{h}^2(\{x_1\}_{x_2}) + \{x_1\}_{x_2} + \mathsf{h}^2(x_1) + x_1$. *Nous avons* $t_1 = 1 \odot x_1 + (1+\mathsf{h}) \odot x_2 + a$ *et* $t_2 = (1+\mathsf{h}^2) \odot x_1 + \mathsf{h}^2(\{x_1\}_{x_2}) + \{x_1\}_{x_2}$. *Nous en déduisons donc que* $\vec{t_1} = (1, 1+\mathsf{h})$ *et* $t_1^0 = a$, *et que* $\vec{t_2} = (1+\mathsf{h}^2, 0)$ *et* $t_2^0 = \mathsf{h}^2(\{x_1\}_{x_2}) + \{x_1\}_{x_2}$.

Remarque : Lorsque le terme t est un terme sur la signature réduite, le terme t^0 qui lui est associé est nécessairement un terme clos.

Définition 6.37 (indépendant)
Soit $\mathcal{V} = \{v_1, \ldots, v_m\}$ *un sous-ensemble fini de* $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^n$. *L'ensemble* $\mathcal{V}$ *est* indépendant *si pour tout* $\alpha_1, \ldots, \alpha_m \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ *tels que* $\alpha_1 v_1 + \ldots + \alpha_m v_m = 0$, *on a* $(\alpha_1, \ldots, \alpha_m) = (0, \ldots, 0)$. *Sinon* $\mathcal{V}$ *est dit* dépendant.

Si l'ensemble $\mathcal{V}$ est indépendant et l'ensemble $\mathcal{V} \cup \{\vec{v}\}$ est dépendant, alors nous disons que le *vecteur* $\vec{v}$ *est dépendant de* $\mathcal{V}$.

Exemple 6.38
Soient $t_1 = a + \mathsf{h}(a) + b + x_1 + \mathsf{h}^3(x_1) + \mathsf{h}^2(x_2)$ *et* $t_2 = \mathsf{h}(a) + \mathsf{h}(x_1) + x_2$. *Les vecteurs* $\vec{t_1}$ *et* $\vec{t_2}$ *associés aux termes* t_1 *et* t_2 *sont :*

$$\vec{t_1} = \begin{pmatrix} 1+\mathsf{h}^3 \\ \mathsf{h}^2 \end{pmatrix} \qquad \vec{t_2} = \begin{pmatrix} \mathsf{h} \\ 1 \end{pmatrix}$$

Les vecteurs $\vec{t_1}$ *et* $\vec{t_2}$ *sont indépendants. Soit* $t_3 = \mathsf{h}(a) + b + \mathsf{h}(x_1)$, *le vecteur* $\vec{t_3}$ *est dépendant de* $\{\vec{t_1}, \vec{t_2}\}$. *En effet, nous avons* $\mathsf{h}.\vec{t_3} = \vec{t_1} + \mathsf{h}^2.\vec{t_2}$.

Considérons $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_k \Vdash u_k\}$. Notons $L_i(\mathcal{C})$ l'ensemble des indices obtenus en appliquant l'algorithme 6.6 sur l'entrée $\mathcal{C}, k$. L'ensemble $L(\mathcal{C})$ est égal à $L_k(\mathcal{C})$ et correspond aux indices des contraintes dites *définissantes.* Notons $\mathcal{B}_i(\mathcal{C}) = \{\vec{u_j} \mid j \in L_i(\mathcal{C})\}$, et $\mathcal{B}(\mathcal{C}) = \mathcal{B}_k(\mathcal{C})$. Par construction des $L_i(\mathcal{C})$, les ensembles $\mathcal{B}_i(\mathcal{C})$ sont indépendants.

```
Entrées: C = {T_1 ⊩ u_1, ..., T_k ⊩ u_k} et i ≤ k
Sortie: L

Algorithme:
 L := ∅;
 pour l = 1 à i faire
   si {u_l} ∪ {u_j | j ∈ L } est indépendant alors L := L ∪ {l};
fin
retourner L.
```

Algorithme 6.6 - Construction de $L_i(\mathcal{C})$ (indices des contraintes définissantes).

Exemple 6.39
Considérons le système de contraintes suivant :

$$\mathcal{C} := \begin{cases} \mathsf{h}(a)+a, b+\mathsf{h}^2(a) & \Vdash \quad \mathsf{h}(x_1)+\mathsf{h}^2(x_2) \\ \mathsf{h}(a)+a, b+\mathsf{h}^2(a), x_1+\mathsf{h}(x_2) & \Vdash \quad x_1+a \\ \mathsf{h}(a)+a, b+\mathsf{h}^2(a), x_1+\mathsf{h}(x_2), \mathsf{h}(x_1)+\mathsf{h}(a) & \Vdash \quad \mathsf{h}(x_1)+\mathsf{h}^2(x_2)+x_1+a \end{cases}$$

Posons $u_1 = \mathsf{h}(x_1)+\mathsf{h}^2(x_2)$, $u_2 = x_1+a$ *et* $u_3 = \mathsf{h}(x_1)+\mathsf{h}^2(x_2)+x_1+a$. *Nous avons* $\vec{u_1} = (\mathsf{h}, \mathsf{h}^2)$, $\vec{u_2} = (1, 0)$ *et* $\vec{u_3} = (1+\mathsf{h}, \mathsf{h}^2)$, $L(\mathcal{C}) = \{1, 2\}$ *et* $\mathcal{B}(\mathcal{C}) = \{\vec{u_1}, \vec{u_2}\}$.

Proposition 6.40 (nouvelle caractérisation)
Soit $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_k \Vdash u_k\}$ *un système de contraintes monotone sur signature réduite. Le système* $\mathcal{C}$ *est bien formé si, et seulement si, pour tout* $i \leq k$, *pour tout* $t \in T_i$, *le vecteur* $\vec{t}$ *est dépendant de* $\mathcal{B}_{i-1}(\mathcal{C})$.

Pour montrer ce lemme, nous exploitons le fait suivant :

Fait 4 *Soit* A *une matrice* $n \times m$ *sur* $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ *dont les* n *vecteurs lignes forment un ensemble indépendants* ($n \leq m$). *Il existe un polynôme* $Q \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ *calculable tel que :*

$$\forall b \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^n, \exists X \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^m \text{ tel que } A \cdot X = Q \cdot b$$

Le polynôme Q *est en fait le déterminant de la matrice obtenue après complétion de* A *par* $m-n$ *vecteurs lignes indépendants.*

Notation : Soit $\mathcal{C}$ un système de contrainte, nous notons $Q_{max}(\mathcal{C})$ le polynôme Q associé à la matrice $\mathcal{B}(\mathcal{C})$. Nous notons $d^\circ(\mathcal{C})$ le degré du polynôme $Q_{max}(\mathcal{C})$.

Exemple 6.41
Reprenons le système de contraintes décrit dans l'exemple 6.39. On obtient $Q_{max}(\mathcal{C}) = \mathsf{h}^2$ *et* $d^\circ(\mathcal{C}) = 2$.

Démonstration. (de la proposition 6.40)
($\Leftarrow$) Nous devons montrer que la propriété d'initialisation est stable par substitution. Soit θ une substitution, soit $i \leq k$ et $t \in T_i$. Soit Z une variable telle que $Z \in vars(t)$. Nous devons montrer qu'il existe $i' < i$ tel que $Z \in vars(u_{i'}\theta)$. Posons $L_{i-1}(\mathcal{C}) = \{i_1, \ldots, i_n\}$. Par hypothèse, nous savons qu'il existe $\alpha \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ ($\alpha \neq 0$) et $\alpha_{i_1}, \ldots, \alpha_{i_n} \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ tels que :

$$\alpha\vec{t} = \sum_{j \,\in\, L_{i-1}(\mathcal{C})} \alpha_j \vec{u_j}.$$

Notons $X_1, \ldots, X_p$, les variables de $\mathcal{C}$. Ainsi, nous avons :

$$\begin{array}{rrcl}
 & \alpha\vec{t} & = & \sum_{j \in L_{i-1}(\mathcal{C})} \alpha_j \vec{u_j} \\
\Rightarrow & \alpha.(\sum_{l=1}^{l=p} t^{X_l} \odot X_l + t^0 - t^0) & = & \sum_{j \in L_{i-1}(\mathcal{C})} \alpha_j.(\sum_{l=1}^{l=p} u_j^{X_l} \odot X_l + u_j^0 - u_j^0) \\
\Rightarrow & \alpha.(t - t^0) & = & \sum_{j \in L_{i-1}(\mathcal{C})} \alpha_j.(u_j - u_j^0) \\
\Rightarrow & \alpha.(t\theta - t^0) & = & \sum_{j \in L_{i-1}(\mathcal{C})} \alpha_j.(u_j\theta - u_j^0) \\
\Rightarrow & \alpha.t\theta & = & \sum_{j \in L_{i-1}(\mathcal{C})} \alpha_j.(u_j\theta - u_j^0) + \alpha t^0
\end{array}$$

Par hypothèse, nous savons que $Z \in vars(t\theta)$. Puisque nous sommes sur signature réduite, les termes t^0, $u_{i_1}^0, \ldots, u_{i_n}^0$ sont clos. Nous en déduisons donc qu'il existe $i' < i$ tel que $Z \in vars(u_{i'}\theta)$.

($\Rightarrow$) Supposons qu'il existe i ($1 \leq i \leq k$) et $t \in T_i$ tels que $\vec{t}$ ne soit pas dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$. Posons $L_{i-1}(\mathcal{C}) = \{i_1, \ldots, i_n\}$, et notons $X_1, \ldots, X_p$ les variables de $\mathcal{C}$. Le fait 4 nous assure l'existence d'un polynôme $Q \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ tel que le système d'équations suivant a une solution dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]^p$:

$$\begin{pmatrix} u_{i_1}^{X_1} & u_{i_1}^{X_2} & \ldots & u_{i_1}^{X_p} \\ \vdots & & & \\ u_{i_n}^{X_1} & u_{i_n}^{X_2} & \ldots & u_{i_n}^{X_p} \\ & & & \\ t^{X_1} & t^{X_2} & \ldots & t^{X_p} \end{pmatrix} \cdot \begin{pmatrix} Y_1 \\ \vdots \\ \\ \vdots \\ Y_p \end{pmatrix} = Q \cdot \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \\ 1 \end{pmatrix}$$

Soit $(c_1, \ldots, c_p)$ une solution de ce système. Soit Z une variable « fraîche » ($Z \neq X_1, \ldots, X_p$) et θ la substitution définie de la façon suivante :

$$\theta := \{X_1 \mapsto c_1 \odot Z;\ \ldots;\ X_p \mapsto c_p \odot Z\}.$$

Par construction de θ, nous avons $u_j\theta = u_j^0$ pour tout $j \in \{i_1, \ldots, i_n\}$. De plus, par définition de $L_{i-1}(\mathcal{C})$, nous savons que pour tout $j \notin L_{i-1}(\mathcal{C})$, $u_j\theta = u_j^0$ (puisque $\vec{u_j}$ est dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$). Or nous avons $t\theta = Q \odot Z + t^0$, et donc $Z \in vars(t\theta)$. Nous en déduisons donc que le système $\mathcal{C}$ n'est pas un système bien formé. □

Remarque : Grâce à cette nouvelle caractérisation des systèmes bien formés, nous obtenons un algorithme permettant de décider si un système de contraintes est bien formé.

Cette caractérisation n'est valable que sur la signature réduite (*cf.* exemple 6.42), et ne semble malheureusement pas généralisable sur la signature complète (*cf.* exemple 6.45).

Exemple 6.42
Considérons le système de contraintes $\mathcal{C}$ suivant :

$$\begin{array}{rcl} a & \Vdash & \langle x, y\rangle \\ a, x & \Vdash & a \end{array}$$

Posons $u_1 = \langle x, y \rangle$ *et* $t = x$. *Nous obtenons* $\vec{u_1} = (0,0)$ *et* $\vec{t} = (1,0)$ *: le vecteur* $\vec{t}$ *n'est pas dépendant de* $\vec{u_1}$ *alors que le système* $\mathcal{C}$ *est bien formé.*

À la vue de cet exemple, on peut se demander si une généralisation de la caractérisation proposée dans le proposition 6.40 ne permettrait pas d'obtenir une caractérisation algébrique de la propriété de bonne formation sur la signature complète. L'idée, dans cet exemple, serait de travailler sur les *sous-termes non-standards* et pas simplement sur les termes du système de contraintes. Dans l'exemple précédent, cela reviendrait à considérer les sous-termes x et y du terme $\langle x, y \rangle$. Dans ce cas, on a bien que $\vec{t}$ est dépendant de $(1,0)$ et $(0,1)$.

Pour être plus précis, nous introduisons donc la notion de *sous-termes non-standards*.

Définition 6.43 (sous-termes non-standards)
Soit $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. *L'ensemble* $NSt_{\mathsf{E}}(t)$ *des sous-termes non-standards de* t *est défini de la façon suivante :*
- $NSt_{\mathsf{E}}(f(t_1, \dots, t_n)) = NSt_{\mathsf{E}}(t_1) \cup \ldots \cup NSt_{\mathsf{E}}(t_n)$ *si* $f \notin sig(\mathsf{E})$,
- $NSt_{\mathsf{E}}(t) = \{t\} \cup \bigcup_{f \in Fact_{\mathsf{E}}(t) \smallsetminus \mathcal{X}} NSt_{\mathsf{E}}(f)$ *sinon.*

Exemple 6.44
Soit $t = \mathsf{h}(x_1) + x_2 + \langle x_3, x_4 + x_5 \rangle$. *Nous avons* $NSt_{\mathsf{E}}(t) = \{t, x_3, x_4 + x_5\}$.

Soit $\mathcal{C} = \{T_1 \Vdash u_1, \dots, T_k \Vdash u_k\}$. L'idée serait alors d'obtenir un résultat (sur signature complète) disant que $\mathcal{C}$ est bien formé si, et seulement si, pour tout $i \leq k$, pour tout $t \in NSt_{\mathsf{E}}(T_i)$, le vecteur $\vec{t}$ est dépendant de $\{\vec{u} \mid u \in NSt_{\mathsf{E}}(\{u_1, \dots, u_{i-1}\})\}$. Malheureusement, ce résultat est faux :

Exemple 6.45
Considérons le système de contraintes $\mathcal{C}$ *suivant :*

$$\begin{array}{rcl} a & \Vdash & x \oplus y \oplus \{y\}_k \\ a, x & \Vdash & a \end{array}$$

Ce système n'est pas bien formé ($\theta = \{x \mapsto y \oplus \{y\}_k\}$). *Or, le vecteur* $\vec{x} = (1,0)$ *est dépendant de* $\{\vec{u} \mid u \in \{x \oplus y \oplus \{y\}_k,\ y\}\} = \{(1,1),(0,1)\}$

Nous verrons (*cf.* lemme 6.50) que nous pouvons obtenir (en ajoutant une condition supplémentaire sur $\mathcal{C}$), une des deux directions du résultat que nous venons d'énoncer.

6.3.4.2 Bonne formation des systèmes obtenus après abstraction

Malheureusement, comme illustré par l'exemple 6.32, la propriété de bonne formation n'est pas stable par l'abstraction des facteurs par des constantes. Pour obtenir la stabilité de la propriété de bonne formation par abstraction, nous allons nous restreindre aux systèmes *facteurs-préservant.*

Définition 6.46 (facteurs-préservant)
Soit $\mathcal{C} = \{T_1 \Vdash u_1, \dots, T_k \Vdash u_k\}$ *un système de contraintes. Le système* $\mathcal{C}$ *est* facteurs-préservant *si pour tout* i *tel que* $1 \leq i \leq k$, *on a* $Fact_{\mathsf{E}}(u_i) \smallsetminus \mathcal{X} \subseteq Fact_{\mathsf{E}}(T_i)$.

Exemple 6.47
Le système de contraintes $\mathcal{C}$ de l'exemple 6.32 ne préserve pas les facteurs : le facteur $\langle x_1, x_2\rangle$ ne satisfait pas la propriété demandée.

Cette notion est importante. Elle permet d'assurer la bonne formation du système abstrait (système obtenu après application de l'abstraction ρ). D'autre part, cette propriété est décidable et ne met pas en danger la complétude de notre procédure. En effet, nous avons vu que l'on pouvait se restreindre (lemme 6.30) à chercher des solutions non-effondrantes. Or, il s'avère qu'un système bien formé ne préservant pas les facteurs, ne peut pas avoir une telle solution. En effet, nous avons :

Lemme 6.48
Soit $\mathcal{C}$ un système de contraintes bien formé. Si le système $\mathcal{C}$ a une solution non-effondrante dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$ alors $\mathcal{C}$ est facteurs-préservant.

Démonstration.
Soit $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_k \Vdash u_k\}$ un système de contraintes bien formé et σ une solution non-effondrante de $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$. Nous montrons que pour tout $i \leq k$:

1. $\mathit{Fact}_\mathsf{E}(u_i\sigma) \subseteq (\mathit{Fact}_\mathsf{E}(T_i) \smallsetminus \mathcal{X})\sigma$, et
2. pour tout $x \in \mathit{vars}(u_i)$ tel que pour tout $j < i$ on a $x \notin \mathit{vars}(u_j)$, la propriété suivante est satisfaite :
$$\mathit{Fact}_\mathsf{E}(x\sigma) \subseteq (\mathit{Fact}_\mathsf{E}(T_i) \smallsetminus \mathcal{X})\sigma.$$

Cas de base : Les termes dans T_1 sont clos. Nous avons $\mathit{Fact}_\mathsf{E}(u_1\sigma) \subseteq \mathit{Fact}_\mathsf{E}(T_1)$. Nous avons $\mathit{Fact}_\mathsf{E}(T_1) = (\mathit{Fact}_\mathsf{E}(T_1) \smallsetminus \mathcal{X})\sigma$ et nous concluons en ce qui concerne le point 1. Soit $x \in \mathit{vars}(u_1)$. Si $x \in \mathit{Fact}_\mathsf{E}(u_1)$ alors $\mathit{Fact}_\mathsf{E}(x\sigma) \subseteq \mathit{Fact}_\mathsf{E}(u_1\sigma)$ et nous concluons grâce au point 1. Sinon il existe $f \in \mathit{Fact}_\mathsf{E}(u_1)$ tel que $x \in \mathit{vars}(f)$. On a alors $f\sigma = t_g$ avec $t_g \in \mathit{St}_\mathsf{E}(T_1)$, ce qui est impossible car σ est non-effondrante.

Étape d'induction : Soit $i > 1$. Nous avons $\mathit{Fact}_\mathsf{E}(u_i\sigma) \subseteq \mathit{Fact}_\mathsf{E}(T_i)\sigma$. Supposons qu'il existe $f \in \mathit{Fact}_\mathsf{E}(u_i\sigma)$ et $x \in \mathit{Fact}_\mathsf{E}(T_i)$ tel que $f = x\sigma$. Par hypothèse d'induction, nous savons que : $\mathit{Fact}_\mathsf{E}(x\sigma) \subseteq (\mathit{Fact}_\mathsf{E}(T_i) \smallsetminus \mathcal{X})\sigma$. Nous concluons pour le point 1. Maintenant, supposons qu'il existe $x \in \mathit{vars}(u_i\sigma)$ tel que $x \notin \mathit{vars}(u_j\sigma)$ pour tout $j < i$. Puisque $\mathcal{C}$ est un système bien formé, $x \notin \mathit{vars}(T_i)$. Nous en déduisons donc que $x \in \mathit{Fact}_\mathsf{E}(u_i)$ pour ne pas contredire le fait que σ est non-effondrante. Nous avons donc $\mathit{Fact}_\mathsf{E}(x\sigma) \subseteq \mathit{Fact}_\mathsf{E}(u_i\sigma)$ et nous concluons grâce au point 1. □

Proposition 6.49
Soit $\mathcal{C}$ un système de contraintes bien formé et facteurs-préservant. Posons $F = \mathit{Fact}_\mathsf{E}(\mathcal{C}) \smallsetminus \mathcal{X}$. Soit $\mathcal{F}_0$ un ensemble de constantes fraîches de même cardinalité que F et $\rho : F \to \mathcal{F}_0$ une bijection. Le système C^ρ est un système de contraintes bien formé.

Avant de faire cette preuve nous avons besoin d'introduire quelques notations et d'établir un lemme intermédiaire (lemme 6.50).

Le lemme, énoncé ci-dessous, peut être vu comme une généralisation de la proposition 6.40 à la signature complète. Il permet d'assurer une condition de dépendance entre certains vecteurs dès lors que le système $\mathcal{C}$ considéré est un système bien formé et facteurs-préservant.

Lemme 6.50
Soit $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_k \Vdash u_k\}$ un système de contraintes bien formé et facteurs-préservant (sur signature complète). Pour tout $i \leq k$, pour tout $s \in NSt_{\mathsf{E}}(T_i)$, le vecteur $\vec{s}$ est dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$.

Démonstration.
Nous montrons ce résultat par induction sur i.

Cas de base : $i = 1$. Soit $s \in NSt_{\mathsf{E}}(T_1)$. Nous savons que s est un terme clos, on a donc $\vec{s} = (0, \ldots, 0)$ et nous concluons.

Étape d'induction : Soit $1 < i \leq k$. Supposons qu'il existe $s \in NSt_{\mathsf{E}}(T_i)$ tel que $\vec{s}$ soit dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$. Nous allons montrer que ceci est impossible. Pour cela, nous allons construire une substitution θ témoin du fait que $\mathcal{C}$ n'est pas un système bien formé. Posons $L_{i-1}(\mathcal{C}) = \{i_1, \ldots, i_n\}$ et notons $X_1, \ldots, X_p$ les variables de $\mathcal{C}$. Le fait 4 (énoncé à la page 128), nous assure l'existence d'un polynôme $Q \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}], (Q \neq 0)$, et d'un vecteur $(c^{X_1}, \ldots, c^{X_p}) \in (\mathbb{Z}/2\mathbb{Z}[\mathsf{h}])^p$ tels que :

$$\begin{pmatrix} u_{i_1}^{X_1} & \ldots & u_{i_1}^{X_p} \\ \vdots & & \vdots \\ u_{i_n}^{X_1} & \ldots & u_{i_n}^{X_p} \\ & & \\ s^{X_1} & \ldots & s^{X_p} \end{pmatrix} \cdot \begin{pmatrix} c^{X_1} \\ \vdots \\ \vdots \\ \\ c^{X_p} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \\ Q \end{pmatrix}$$

Nous définissons la substitution θ de la façon suivante :

$$X_i \mapsto X_i + c^{X_i} \odot Z \qquad \text{pour tout } 1 \leq i \leq p.$$

Nous allons montrer que :
- pour tout $j < i$, on a $Z \notin vars(u_j\theta)$ (*cf.* **fait 6**),
- $Z \in vars(T_i\theta)$ (*cf.* **fait 7**).

Nous commençons par établir le fait suivant :

Fait 5 *Pour tout $j < i$, pour tout $t \in T_j$, on a $Z \notin vars(t\theta)$.*

Soit $j < i$. Nous montrons que pour tout $t \in NSt_{\mathsf{E}}(T_j)$, on a $Z \notin vars(t\theta)$.

Cas de base : Si $Fact_{\mathsf{E}}(t) \subseteq \mathcal{X} \cup \mathcal{T}(\mathcal{F})$ (autrement dit, si les facteurs de t sont soit des variables, soit des termes clos), il existe $t^0 \in \mathcal{T}(\mathcal{F})$, $t^{X_q} \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ $(1 \leq q \leq p)$ tel que $t = t^{X_1} \odot X_1 + \ldots + t^{X_p} \odot X_p + t^0$. Par hypothèse d'induction (du lemme 6.50), nous savons que $\vec{t}$ est dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$. Nous en déduisons donc que $t\theta = t^0$. Cela nous permet de conclure que $Z \notin vars(t\theta)$.

Étape d'induction : Nous distinguons deux cas.

1. Le terme t est un terme standard, il existe un $\{\mathcal{F} \smallsetminus sig(\mathsf{E})\}$-contexte C et des termes non-standards (ou des variables) $t_1, \ldots, t_n \in NSt_{\mathsf{E}}(t)$ tels que $t = C[t_1, \ldots, t_n]$. Dans ce cas, nous avons $t\theta = C[t_1\theta, \ldots, t_n\theta]$ et nous concluons en appliquant l'hypothèse d'induction sur $t_1, \ldots, t_n$.

2. Le terme t est non-standard, il est donc de la forme :

$$t = \sum_{i=1}^{p} (t^{X_i} \odot X_i) + \sum_{f \in Fact_{\mathsf{E}}(t) \smallsetminus \mathcal{X}} (t^f \odot f)$$

Par définition de θ, nous avons :

$$\begin{aligned} t\theta &= \sum_{i=1}^{p} (t^{X_i} \odot X_i\theta) + \sum_{f \in Fact_{\mathsf{E}}(t) \smallsetminus \mathcal{X}} (t^f \odot f\theta) \\ &= \underbrace{\sum_{i=1}^{p} (t^{X_i} \odot X_i)}_{t_1} + \underbrace{\sum_{i=1}^{p} ((t^{X_i} \cdot c^{X_i}) \odot Z)}_{t_2} + \underbrace{\sum_{f \in Fact_{\mathsf{E}}(t) \smallsetminus \mathcal{X}} (t^f \odot f\theta)}_{t_3} \end{aligned}$$

Tout d'abord, nous avons $Z \notin vars(t_1)$. Par hypothèse d'induction (du fait 5), nous savons que $Z \notin vars(t_3)$. Par hypothèse d'induction (du lemme 6.50), $\vec{t}$ est dépendant de $\mathcal{B}_{j-1}(\mathcal{C})$, et donc de $\mathcal{B}_{i-1}(\mathcal{C})$: il existe $\alpha, \alpha_{i_1}, \ldots, \alpha_{i_n} \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ tel que $\alpha \neq 0$ et :

$$\begin{array}{rrcl} & \alpha \cdot \vec{t} &=& \alpha_{i_1} \cdot \vec{u_{i_1}} + \ldots + \alpha_{i_n} \cdot \vec{u_{i_n}} \\ \Rightarrow & \alpha \cdot \sum_{i=1}^{p} (t^{X_i} \cdot c^{X_i}) &=& \alpha_{i_1} \cdot \sum_{i=1}^{p} (u_{i_1}^{X_i} \cdot c^{X_i}) + \ldots + \alpha_{i_n} \cdot \sum_{i=1}^{p} (u_{i_n}^{X_i} \cdot c^{X_i}) \\ \Rightarrow & &=& 0 \qquad \text{par définition de } \theta \end{array}$$

Nous en déduisons donc que $\sum_{i=1}^{p} (t^{X_i} \cdot c^{X_1}) = 0$, et que $Z \notin vars(t_2)$.

Nous obtenons alors que pour tout $j < i$, pour tout $t \in NSt_{\mathsf{E}}(T_j)$, on a $Z \notin vars(t\theta)$.

Soit $t \in T_j$. Si t est un terme non-standard, nous avons $t \in NSt_{\mathsf{E}}(t)$ et nous concluons. Sinon il existe un $\{\mathcal{F} \smallsetminus sig(\mathsf{E})\}$-contexte C et des termes non-standards $t_1, \ldots, t_n \in NSt_{\mathsf{E}}(t)$ tels que $t = C[t_1, \ldots, t_n]$. Nous appliquons le fait 5 sur $t_1, \ldots, t_n$ et nous concluons.

Fait 6 *Pour tout $j < i$, on a $Z \notin vars(u_j\theta)$.*

Supposons qu'il existe $j < i$ tel que $Z \in vars(u_j\theta)$. Si $j \in L_{i-1}(\mathcal{C})$ alors $\sum_{i=1}^{p} (u_j^{X_i} \cdot c^{X_i}) = 0$ par construction de θ, et si $j \notin L_{i-1}(\mathcal{C})$, nous avons aussi que $\sum_{i=1}^{p} (u_j^{X_i} \cdot c^{X_i}) = 0$ puisque par construction de $L_{i-1}(\mathcal{C})$, $\vec{u_j}$ est dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$. Nous en déduisons donc qu'il existe $f \in Fact_{\mathsf{E}}(u_j\theta) \smallsetminus \mathcal{X}$ tel que $Z \in vars(f)$, et qu'il existe $f' \in Fact_{\mathsf{E}}(u_j)$ tel que $Z \in vars(f'\theta)$. Puisque $\mathcal{C}$ est facteurs-préservant, il existe $j' \leq j$ tel que $f' \in Fact_{\mathsf{E}}(T_{j'})$. Le lemme 6.51, que nous prouvons ci-dessous, nous assure que pour tout $f'' \in Fact_{\mathsf{E}}(T_{j'})$ tel que $f' \neq f''$, on a $f'\theta \neq f''\theta$. Ainsi, $Z \in vars(T_{j'}\theta)$, ce qui contredit le fait 5.

Fait 7 *On a $Z \in vars(T_i\theta)$.*

Le terme s est de la forme :

$$s = \sum_{i=1}^{p} (s^{X_i} \odot X_i) + \sum_{f \in Fact_{\mathsf{E}}(s) \smallsetminus \mathcal{X}} (s^f \odot f).$$

Par définition de θ, nous savons que :

$$s\theta = \sum_{i=1}^{p}(s^{X_i} \odot X_i) + \underbrace{\sum_{i=1}^{p}((s^{X_i} \cdot c^{X_i}) \odot Z)}_{=Q\odot Z} + \sum_{f \in Fact_{\mathsf{E}}(s) \smallsetminus \mathcal{X}} (s^f \odot f\theta).$$

Nous en déduisons donc que $Z \in vars(s\theta)$.

Si $s \in T_i$, nous concluons que $Z \in vars(T_i\theta)$. Sinon, il existe $f \in Fact_{\mathsf{E}}(T_i) \smallsetminus \mathcal{X}$ tel que $s \in NSt_{\mathsf{E}}(f)$. Le lemme 6.51, nous assure que le facteur $f\theta$ ne peut pas être éliminé. Nous en déduisons donc que $Z \in vars(f\theta)$, et que $Z \in vars(T_i\theta)$.

Ainsi, nous avons $Z \in vars(T_i\theta)$ et $Z \notin vars(u_j\theta)$ pour tout $j < i$. Ceci contredit le fait que $\mathcal{C}$ est un système bien formé. □

Dans la preuve du lemme ci-dessus, nous avons utilisé le lemme suivant pour assurer que des « facteurs différents » ne peuvent pas être rendus égaux par application de la substitution θ que nous avions choisie. Nous montrons maintenant ce lemme.

Lemme 6.51
Soit Z une variable « fraîche » et θ une substitution de la forme $X \mapsto X + c^X \odot Z$ pour tout $X \in \mathcal{X}$, où $c^X \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ pour tout $X \in \mathcal{X}$. Si $t_1 \neq t_2$ alors $t_1\theta \neq t_2\theta$.

Démonstration.
Nous montrons ce résultat par induction sur la taille des termes t_1 et t_2. Le cas de base est trivial. Nous distinguons différents cas :

- Si t_1 et t_2 sont tous les deux des termes standards, nous avons $t_1 = f_1(t_1^1, \dots, t_1^n)$ et $t_2 = f_2(t_2^1, \dots, t_2^m)$. Si $f_1 \neq f_2$ alors nous concluons que $t_1\theta \neq t_2\theta$. Sinon, nous avons $n = m$ et il existe $i < n$ tel que $t_1^i \neq t_2^i$. Par hypothèse d'induction, nous savons que $t_1^i\theta \neq t_2^i\theta$, et nous en déduisons que $t_1\theta = f_1(t_1^1\theta, \dots, t_1^n\theta) \neq f_2(t_2^1\theta, \dots, t_2^n\theta) = t_2\theta$.
- Si t_1 est un terme standard et t_2 est un terme non-standard, nous avons $t_1 = f_1(t_1^1, \dots, t_1^n)$ et $t_2 = \sum_{s \in Fact_{\mathsf{E}}(t_2)}(p^s \odot s)$. L'ensemble $Fact_{\mathsf{E}}(t_2)$ contient au moins deux éléments. Par hypothèse d'induction, nous savons que pour tout $s_1, s_2 \in Fact_{\mathsf{E}}(t_2)$ tels que $s_1 \neq s_2$, on a $s_1\theta \neq s_2\theta$. Ainsi, $t_2\theta$ n'est pas standard alors que $t_1\theta$ est un terme standard. Ceci nous permet de conclure.
- Si t_1 et t_2 sont tous les deux non-standards. Posons $F = Fact_{\mathsf{E}}(t_1) \cup Fact_{\mathsf{E}}(t_2)$ et notons $X_1, \dots, X_p$ les variables de t_1 et de t_2. Nous pouvons décomposer t_1 et t_2 de la façon suivante :

$$\begin{array}{rcl} t_1 & = & \sum_{i=1}^{p}(p_1^i \odot X_i) + \sum_{f \in F \smallsetminus \mathcal{X}} (p_1^f \odot f) \\ t_2 & = & \sum_{i+1}^{p}(p_2^i \odot X_i) + \sum_{f \in F \smallsetminus \mathcal{X}} (p_2^f \odot f) \end{array}$$

  Par définition de θ, nous obtenons que :

$$\begin{array}{rcl} t_1\theta & = & \sum_{i=1}^{p}(p_1^i \odot X_i) + \sum_{i=1}^{p}((p_1^i \cdot c^{X_i}) \odot Z) + \sum_{f \in F \smallsetminus \mathcal{X}} (p_1^f \odot f\theta) \\ t_2\theta & = & \sum_{i=1}^{p}(p_2^i \odot X_i) + \sum_{i=1}^{p}((p_2^i \cdot c^{X_i}) \odot Z) + \sum_{f \in F \smallsetminus \mathcal{X}} (p_2^f \odot f\theta) \end{array}$$

Par hypothèse, nous savons que $t_1 \neq t_2$. Nous distinguons deux cas. Soit il existe i ($1 \leq i \leq p$) tel que $p_1^i \neq p_2^i$. Dans ce cas, nous obtenons que X_i apparaît dans $t_1\theta$ avec un coefficient p_1^i et dans $t_2\theta$ avec un coefficient p_2^i. Nous avons donc $t_1\theta \neq t_2\theta$. Soit il existe $f \in F \setminus \mathcal{X}$ tel que $p_1^f \neq p_2^f$. Par hypothèse d'induction, pour tout $f, f' \in F \setminus \mathcal{X}$ tel que $f \neq f'$, nous avons $f\theta \neq f'\theta$. Le facteur $f\theta$ apparaît dans $t_1\theta$ avec un coefficient p_1^s et dans $t_2\theta$ avec un coefficient p_2^s. Ainsi, $t_1\theta \neq t_2\theta$. □

Nous pouvons maintenant faire la preuve de la proposition 6.49.

Démonstration. (de la proposition 6.49)
Posons $\mathcal{C} = \{T_1 \Vdash u_1, \ldots, T_k \Vdash u_k\}$. Par hypothèse, $\mathcal{C}$ est un système bien formé et facteurs-préservant. Par le lemme 6.50, nous savons que pour tout $i \leq k$ et pour tout $s \in NSt_{\mathsf{E}}(T_i)$, le vecteur $\vec{s}$ est dépendant de $\mathcal{B}_{i-1}(\mathcal{C})$. En remarquant que, pour tout terme t, nous avons $\vec{t} = \vec{t\rho}$, nous concluons en appliquant la proposition 6.40. □

6.3.5 Résolution dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$ sur signature réduite

Dans cette partie, nous proposons une procédure permettant de décider la satisfaisabilité, dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathcal{R}_\mathsf{E})$, de systèmes de contraintes bien formés sur signature réduite. Nous supposons donné un système $\mathcal{C}$ bien formé de la forme suivante :

$$\mathcal{C} = \left\{ \begin{array}{rcl} t_1, \ldots, t_n & \Vdash & u_1 \\ t_1, \ldots, t_n, t_{n+1} & \Vdash & u_2 \\ & \ldots & \\ t_1, \ldots, t_n, t_{n+1}, \ldots, t_{n+k-1} & \Vdash & u_k \end{array} \right.$$

avec $u_1, \ldots, u_k, t_1, \ldots, t_{n+k-1} \in \mathcal{T}(\mathcal{F}_0 \cup sig(\mathsf{E}), \mathcal{X})$. Le fait qu'à chaque étape, un unique terme soit ajouté à l'ensemble des hypothèses, n'est pas une restriction. On peut toujours obtenir un système satisfaisant cette propriété en insérant des contraintes de déduction. À partir de ce système de contraintes, nous construisons le système d'équations $\mathcal{S}(\mathcal{C})$ suivant :

$$\mathcal{S}(\mathcal{C}) := \left\{ \begin{array}{l} z[1,1] \odot t_1 + \ldots + z[1,n] \odot t_n = u_1 \\ z[2,1] \odot t_1 + \ldots + z[2,n] \odot t_n + z[2,n+1] \odot t_{n+1} = u_2 \\ \vdots \\ z[p,1] \odot t_1 + \ldots + z[p,n] \odot t_n + \ldots + z[p,n+p-1] \odot t_{n+k-1} = u_k \end{array} \right.$$

Il s'agit d'un système d'équations entre termes. Ce système contient deux types de variables. Les variables de termes, c'est-à-dire les variables de $vars(\mathcal{C})$, et les *variables de contexte.*

Définition 6.52 (variables de contexte)
L'ensemble des variables de contexte *de $\mathcal{C}$, noté $\mathcal{Z}(\mathcal{C})$, est défini par :*

$$\mathcal{Z}(\mathcal{C}) = \{z[i,j] \mid 1 \leq i \leq k, 1 \leq j \leq n+i-1\}.$$

Ces variables prennent leurs valeurs dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$.

Un tel système est satisfaisable s'il existe une substitution $\theta : vars(\mathcal{C}) \mapsto \mathcal{T}(\mathcal{F}_0 \cup sig(\mathsf{E}))$ telle que $\mathcal{S}(\mathcal{C})$ admette une solution dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$.

Exemple 6.53
Pour illustrer notre procédure, considérons le système de contraintes suivant, que nous avons introduit dans l'exemple 6.39 :

$$\begin{array}{lcl}
\mathsf{h}(a)+a, b+\mathsf{h}^2(a) & \Vdash & \mathsf{h}(x_1)+\mathsf{h}^2(x_2) \\
\mathsf{h}(a)+a, b+\mathsf{h}^2(a), x_1+\mathsf{h}(x_2) & \Vdash & x_1+a \\
\mathsf{h}(a)+a, b+\mathsf{h}^2(a), x_1+\mathsf{h}(x_2), \mathsf{h}(x_1)+\mathsf{h}(a) & \Vdash & \mathsf{h}(x_1)+\mathsf{h}^2(x_2)+x_1+a
\end{array}$$

Nous obtenons le système $\mathcal{S}(\mathcal{C})$ suivant :

$$\mathcal{S}(\mathcal{C}) := \begin{cases}
z[1,1]\odot t_1 + z[1,2]\odot t_2 = \mathsf{h}(x_1)+\mathsf{h}^2(x_2) \\
z[2,1]\odot t_1 + z[2,2]\odot t_2 + z[2,3]\odot(x_1+\mathsf{h}(x_2)) = x_1+a \\
z[3,1]\odot t_1 + z[3,2]\odot t_2 + z[3,3]\odot(x_1+\mathsf{h}(x_2)) + z[3,4]\odot(\mathsf{h}(x_1)+\mathsf{h}(a)) = \\
\hfill \mathsf{h}(x_1)+\mathsf{h}^2(x_2)+x_1+a
\end{cases}$$

La satisfaisabilité de $\mathcal{C}$ est équivalente à l'existence d'une solution pour le système d'équations $\mathcal{S}(\mathcal{C})$ correspondant. Autrement dit, nous avons le lemme suivant :

Lemme 6.54
Soit $\mathcal{C}$ un système de contraintes sur signature réduite et $\mathcal{S}(\mathcal{C})$ le système d'équations qui lui est associé. Le système de contraintes $\mathcal{C}$ a une solution dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathsf{E})$ sur signature réduite si, et seulement si, le système d'équations $\mathcal{S}(\mathcal{C})$ a une solution.

Définition 6.55 (variables de contexte définissantes)
L'ensemble des variables de contexte définissantes *de $\mathcal{C}$, noté $\mathcal{Z}_L(\mathcal{C})$, est défini par :*

$$\{z[i,j] \mid i \in L \text{ et } 1 \leq j < n+i\}.$$

Définition 6.56 (ordre $\prec$ sur $\mathcal{Z}(\mathcal{C})$)
Nous définissons l'ordre $\prec$ sur les variables de $\mathcal{Z}(\mathcal{C})$ de la façon suivante :

$$z[i,j] \prec z[i',j'] \quad \text{si} \quad i < i \quad \text{ou} \quad \text{si } i = i' \text{ et } j < j'$$

L'ordre $\prec$ est un ordre total sur les variables de $\mathcal{Z}(\mathcal{C})$.

La proposition énoncée ci-dessous est cruciale. Elle permet de garantir qu'un système de contraintes $\mathcal{C}$ satisfaisable admet une solution où les variables de contexte définissantes prennent leurs valeurs dans l'ensemble $\{p \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}] \mid d°(p) < d°(\mathcal{C})\}$. Une fois la valeur de ces variables fixée, nous verrons comment déterminer la valeur des autres variables du système $\mathcal{S}(\mathcal{C})$ dans la proposition 6.58.

Proposition 6.57
Si $\mathcal{S}(\mathcal{C})$ est satisfaisable, il existe une solution σ telle que pour tout $z \in \mathcal{Z}_L(\mathcal{C})$, $d°(z\sigma) < d°(\mathcal{C})$.

Démonstration.
Nous montrons que pour tout ensemble $\mathcal{Z}' \subseteq \mathcal{Z}_L(\mathcal{C})$ clos par le bas (pour tout $z_2 \in \mathcal{Z}'$ tel que $z_1 \prec z_2$ alors $z_1 \in \mathcal{Z}'$), nous avons :

Si $\mathcal{S}(\mathcal{C})$ a une solution alors il existe une solution σ telle que $d^\circ(z\sigma) < d^\circ(\mathcal{C})$ pour tout $z \in \mathcal{Z}'$.

Nous montrons ce résultat par induction sur la cardinalité de l'ensemble $\mathcal{Z}'$. Le cas de base ($|\mathcal{Z}'| = 0$) est trivial. Supposons que $|\mathcal{Z}'| = n + 1$, et que le résultat est vrai pour tout $|\mathcal{Z}'| \leq n$. Soit z l'élément maximal de $\mathcal{Z}'$ pour l'ordre $\prec$. Puisque $\mathcal{Z}' \subseteq \mathcal{Z}_L(\mathcal{C})$, il existe deux entiers N et M tels que $N \in L(\mathcal{C})$, $1 \leq M < n + M$ et $z = z[N, M]$. Par hypothèse d'induction, nous savons qu'il existe une solution σ telle que $d^\circ(z'\sigma) < d^\circ(\mathcal{C})$ pour tout $z' \in \mathcal{Z}'$ tel que $z' \neq z$. Nous allons construire une solution σ' de $\mathcal{S}(\mathcal{C})$ telle que $d^\circ(z\sigma') < d^\circ(\mathcal{C})$ pour tout $z \in \mathcal{Z}'$. Cette construction se fait en quatre étapes :

1. Pour tout $z' \prec z$, nous posons $z'\sigma' = z\sigma$.
2. Soit $K, r \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$ tel que $d^\circ(r) < d^\circ(\mathcal{C})$ et $z\sigma = r + K \cdot Q_{max}$. Nous posons $z\sigma' = r$.
3. Définition de $x\sigma'$ pour $x \in vars(\mathcal{C})$.
   Posons $L(\mathcal{C}) = \{i_1, \ldots, i_\ell\}$. Notre objectif est de trouver une substitution σ' satisfaisant les conditions suivantes :
   – pour chaque $i \in L(\mathcal{C}) \setminus \{N\}$, $u_i\sigma - u_i\sigma' = 0$
   – $u_N\sigma - u_N\sigma' = K \cdot Q_{max}t_M\sigma$
   Pour cela, nous devons résoudre le système d'équations suivant (il s'agit d'égalités entre termes) où chacune des variables X_i correspond à $X_i\sigma - X_i\sigma'$:

$$\begin{pmatrix} u_{i_1}^{X_1} & u_{i_1}^{X_2} & \ldots & u_{i_1}^{X_p} \\ \vdots & & & \\ u_N^{X_1} & u_N^{X_2} & \ldots & u_N^{X_p} \\ \vdots & & & \\ u_{i_\ell}^{X_1} & u_{i_\ell}^{X_2} & \ldots & u_{i_\ell}^{X_p} \end{pmatrix} \odot \begin{pmatrix} X_1' \\ \vdots \\ \vdots \\ X_p' \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ K \cdot Q_{max} \odot t_M\sigma \\ \vdots \\ 0 \end{pmatrix} \tag{6.1}$$

   Pour cela, il suffit de résoudre le système d'équations suivant, où les inconnues Y_i prennent leurs valeurs dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$:

$$\begin{pmatrix} u_{i_1}^{X_1} & u_{i_1}^{X_2} & \ldots & u_{i_1}^{X_p} \\ \vdots & & & \\ u_N^{X_1} & u_N^{X_2} & \ldots & u_N^{X_p} \\ \vdots & & & \\ u_{i_\ell}^{X_1} & u_{i_\ell}^{X_2} & \ldots & u_{i_\ell}^{X_p} \end{pmatrix} \cdot \begin{pmatrix} Y_1 \\ \vdots \\ \vdots \\ Y_p \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ K \cdot Q_{max} \\ \vdots \\ 0 \end{pmatrix} \tag{6.2}$$

   Grâce au fait 4, nous savons que l'équation (6.2) a une solution $(c_1, \ldots, c_p)$. Nous en déduisons que $(c_1 \odot t_M\sigma, \ldots, c_p \odot t_M\sigma)$ est une solution de (6.1).
   Nous définissons la substitution σ' sur les variables de $vars(\mathcal{C})$ de la façon suivante :

$$X_i\sigma' = X_i\sigma - c_i \odot t_M\sigma \qquad \text{pour tout } i \text{ tel que } 1 \leq i \leq p.$$

4. Définition de $z'\sigma'$ pour $z \prec z'$.
   Soit z' tel que $z \prec z'$. Il existe deux entiers i, q tels que $z' = z[i,q]$. Par définition de $\prec$, soit $i = N$ et $q > M$ soit $i > N$. Nous définissons $z[i,q]\sigma'$ de la façon suivante :

$$z[i,q]\sigma' = \begin{cases} z[i,q]\sigma + \displaystyle\sum_{j=n+N}^{n+i-1} \left(\sum_{l=1}^{p} t_j^{X_l} \cdot c_l\right) \cdot z[i,j]\sigma & \text{si } q = M, i > N \\ z[i,q] & \text{si } q \neq M \end{cases}$$

Nous devons maintenant vérifier que σ' est une solution de $\mathcal{S}(\mathcal{C})$. Tout d'abord, notons que nous avons l'égalité suivante :

$$t_j\sigma = t_j\sigma' \qquad \text{pour tout } j \text{ tel que } 1 \leq j < n+N \tag{6.3}$$

C'est une conséquence directe du fait que $u_i\sigma = u_i\sigma'$ pour $1 \leq i < N$.

Nous procédons à la vérification contrainte par contrainte en distinguant trois cas :

1. cas $i < N$: dans ce cas, le résultat est une conséquence immédiate de (6.3) et du fait que $u_i\sigma = u_i\sigma'$ pour tout i tel que $1 \leq i < N$.
2. cas $i = N$: remarquons que

$$r = z[N,M]\sigma - K \cdot Q_{max}(\mathcal{C}) \tag{6.4}$$

Ainsi,

$$\begin{array}{ll}
 & \displaystyle\sum_{j=1}^{n+N-1} z[N,j]\sigma' \odot t_j\sigma' \\
= & \displaystyle\sum_{j=1}^{M-1} z[N,j]\sigma' \odot t_j\sigma' + z[N,M]\sigma' \odot t_M\sigma' + \sum_{j=M+1}^{n+N-1} z[N,j]\sigma' \odot t_j\sigma' \\
= & \displaystyle\sum_{j=1}^{M-1} z[N,j]\sigma \odot t_j\sigma + r \odot t_M\sigma + \sum_{j=M+1}^{n+N-1} z[N,j]\sigma \odot t_j\sigma \\
 & \hfill \text{(6.3) et } z[N,j]\sigma = z[N,j]\sigma' \text{ pour } j \neq M \\
= & \displaystyle\sum_{j=1}^{M-1} z[N,j]\sigma \odot t_j\sigma + (z[N,M]\sigma - K \cdot Q_{max}(\mathcal{C})) \odot t_M\sigma + \sum_{j=M+1}^{n+N-1} z[N,j]\sigma \odot t_j\sigma \\
= & \displaystyle\sum_{j=1}^{n+N-1} z[N,j]\sigma \odot t_j\sigma - K.Q_{max}(\mathcal{C}) \odot t_M\sigma \\
= & u_N\sigma - K.Q_{max}(\mathcal{C}) \odot t_M\sigma \qquad \text{puisque } \sigma \text{ est une solution de } \mathcal{S} \\
= & u_N\sigma' \qquad \text{par définition de } \sigma'
\end{array}$$

3. cas $i > N$: nous considérons la $i^{\text{ème}}$ équation du système $\mathcal{S}(\mathcal{C})$, c'est-à-dire :

$$\sum_{1 \leq j < n+i} z[i,j] \odot t_j = u_i.$$

En utilisant $X_i\sigma' = X_i\sigma - c_i \odot t_M\sigma$, nous obtenons :

$$\begin{aligned} t_j\sigma' &= \sum_{v \in Fact_{\mathsf{E}}(\mathcal{C}) \smallsetminus vars(\mathcal{C})} (t_j^v \odot v)\sigma' + \sum_{v \in vars(\mathcal{C})} (t_j^v \odot v)\sigma' \\ &= \sum_{v \in Fact_{\mathsf{E}}(\mathcal{C}) \smallsetminus vars(\mathcal{C})} (t_j^v \odot v) + \sum_{l=1}^{p} (t_j^{X_l} \odot X_l\sigma) - \sum_{l=1}^{p} (c_l t_j^{X_l} \odot t_M\sigma). \end{aligned} \tag{6.5}$$

Ainsi, nous avons :

$$\begin{aligned}
&\sum_{j=1}^{n+i-1} z[i,j]\sigma' \odot t_j\sigma' \\
&= \sum_{j=1}^{M-1} z[i,j]\sigma' \odot t_j\sigma' + z[i,M]\sigma' \odot t_M\sigma' + \sum_{j=M+1}^{n+N-1} z[i,j]\sigma' \odot t_j\sigma' \\
&\qquad + \sum_{j=n+N}^{n+i-1} z[i,j]\sigma' \odot t_j\sigma' \\
&= \sum_{j=1}^{M-1} z[i,j]\sigma \odot t_j\sigma + z[i,M]\sigma' \odot t_M\sigma' + \sum_{j=M+1}^{n+N-1} z[i,j]\sigma \odot t_j\sigma \\
&\qquad + \sum_{j=n+N}^{n+i-1} z[i,j]\sigma \odot t_j\sigma' \qquad \text{(6.3) et } z[N,j]\sigma = z[N,j]\sigma' \text{ pour } j \neq M \\
&= \sum_{j=1}^{M-1} z[i,j]\sigma \odot t_j\sigma + \left(z[i,M]\sigma + \sum_{j=n+N}^{n+i-1} (\sum_{l=1}^{p} t_j^{X_l} \cdot c_l) \cdot z[i,j]\sigma \right) \odot t_M\sigma \\
&\quad + \sum_{j=M+1}^{n+N-1} z[i,j]\sigma \odot t_j\sigma \\
&\quad + \sum_{j=n+N}^{n+i-1} z[i,j]\sigma \cdot \Bigg(\sum_{\substack{v \in Fact_{\mathsf{E}}(\mathcal{C}) \\ v \notin vars(\mathcal{C})}} (t_j^v \odot v) + \sum_{l=1}^{p} (t_j^{X_l} \odot X_l\sigma) - \sum_{l=1}^{p} (t_j^{X_l} \cdot c_l \odot t_M\sigma) \Bigg) \\
&\qquad \text{par définition de } \sigma' \text{ et (6.5))} \\
&= \sum_{j=1}^{n+i-1} z[i,j]\sigma \odot t_j\sigma \\
&= u_i\sigma \qquad \text{puisque } \sigma \text{ est une solution de } \mathcal{S} \\
&= u_i\sigma' \qquad \text{puisque } u_i\sigma = u_i\sigma' \text{ pour } i > N.
\end{aligned}$$

Nous concluons que σ' est une solution du système $\mathcal{S}(\mathcal{C})$. Par définition de σ', nous avons $d^\circ(z\sigma') < d^\circ(\mathcal{C})$ pour tout $z \in \mathcal{Z}'$. Cela termine l'induction. Nous montrons le lemme en posant $\mathcal{Z}' = \mathcal{Z}_L(\mathcal{C})$. $\square$

Entrée: $\mathcal{C} = \{t_1, \ldots, t_n \Vdash u_1; \ldots ; t_1, \ldots, t_n, \ldots, t_{n+k-1} \Vdash u_k\}$
Sortie: oui/non

Algorithme:
  calculer $\mathcal{Z}_L(\mathcal{C})$
  pour chaque $z \in \mathcal{Z}_L(\mathcal{C})$, deviner $z\sigma \in \{p \in \mathbb{Z}/2\mathbb{Z}[\mathsf{h}] \mid d^\circ(p) < d^\circ(\mathcal{C})\}$.

  $\mathcal{M} := \varnothing$
  pour chaque $i \in L(\mathcal{C})$ faire
  $\mathcal{M} := \mathcal{M} \cup \{z[i,1]\sigma \odot t_1 + \ldots + z[i, n+i-1]\sigma \odot t_{n+i-1} = u_i\}$

  $\Theta := \{\theta \mid \theta$ solution de $\mathcal{M}\}$
  si $\Theta = \varnothing$ retourner non
  sinon choisir $\theta \in \Theta$
      si $\mathcal{C}\theta$ est satisfaisable retourner oui
      sinon retourner non

Algorithme 6.7 - Satisfaisabilité d'un système bien formé dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$.

Proposition 6.58
L'algorithme 6.7 permet de décider la satisfaisabilité d'un système $\mathcal{C}$ de contraintes bien formé (sur signature réduite) dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$.

Démonstration.
La seule difficulté consiste à montrer la complétude de la procédure. Soit $\mathcal{C}$ un système de contraintes bien formé (sur signature réduite). Supposons que ce système soit satisfaisable dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$. Par le lemme 6.54, nous obtenons que le système $\mathcal{S}(C)$ qui lui est associé est satisfaisable. Soit σ une solution du système $\mathcal{S}(\mathcal{C})$ (on a $dom(\sigma) = \mathcal{Z}(\mathcal{C}) \cup vars(\mathcal{C})$). Grâce à la proposition 6.57, nous pouvons, sans perte de généralité, supposer que pour tout $z \in \mathcal{Z}_L(\mathcal{C})$, nous avons $d^\circ(z\sigma) < d^\circ(\mathcal{C})$. Posons $\sigma_1 = \sigma|_{\mathcal{Z}_L(\mathcal{C})}$. Puisque σ est une solution de $\mathcal{S}(\mathcal{C})$, cela signifie que le système d'équations suivant admet une solution :

$$\mathcal{M} = \bigcup_{i \in L(\mathcal{C})} \{z[i,1]\sigma_1 \odot t_1 + \ldots + z[i, n+i-1]\sigma_1 \odot t_{n+i-1} = u_i\}$$

Ce problème est en fait un problème d'unification modulo ACUNh (toutes les variables de contextes sont instanciées). On sait décider si un tel problème a une solution (théorème 6.27). Soit θ une solution de $\mathcal{M}$. Nous allons montrer que $\mathcal{C}\theta = \mathcal{C}\sigma$. Nous en déduirons que $\mathcal{C}\theta$ (système de contraintes closes) est satisfaisable.

Tout d'abord, puisque les termes $t_1, \ldots, t_n$ sont clos, nous avons $t_j\sigma = t_j\theta$ pour tout j ($1 \leq j \leq n$). Nous avons également $u_1\sigma = u_1\theta$. En effet, si u_1 est clos, ce résultat est trivial. Sinon, on a :

$$\begin{array}{rcll} u_1\sigma & = & z[1,1]\sigma \odot t_1\sigma + \ldots + z[1,n]\sigma \odot t_n\sigma & \\ & = & z[1,1]\sigma \odot t_1 + \ldots + z[1,n]\sigma \odot t_n & \text{car les termes } t_1, \ldots, t_n \text{ sont clos} \\ & = & z[1,1]\sigma_1 \odot t_1 + \ldots + z[1,n]\sigma_1 \odot t_n & \text{car } \sigma_1 = \sigma|_{\mathcal{Z}_L(\mathcal{C})} \\ & = & u_1\theta & \text{par définition de } \theta \end{array}$$

Nous allons montrer par induction que pour tout i ($1 < i \leq k$), on a :

1. $t_{n+i-1}\sigma = t_{n+i-1}\theta$, et
2. $u_i\sigma = u_i\theta$.

1. Par hypothèse ($\mathcal{C}$ est un système bien formé), nous savons que le vecteur associé au terme t_{n+i-1} est dépendant de $\mathcal{B}_{i-1}(\mathcal{C}) = \{\vec{u_j} \mid j \in L_{i-1}(\mathcal{C})\}$. En appliquant l'hypothèse d'induction, nous obtenons $t_{n+i-1}\sigma = t_{n+i-1}\theta$.
2. Nous distinguons deux cas :
   - Si $i \notin L(\mathcal{C})$, alors le vecteur $\vec{u_i}$ est dépendant de $\mathcal{B}_{i-1}(\mathcal{C}) = \{\vec{u_j} \mid j \in L_{i-1}(\mathcal{C})\}$. En appliquant l'hypothèse d'induction, nous en déduisons que $u_i\sigma = u_i\theta$.
   - Si $i \in L(\mathcal{C})$, alors nous avons :

$$\begin{array}{rcll} u_i\sigma & = & z[i,1]\sigma \odot t_1\sigma + \ldots + z[i,n+i-1]\sigma \odot t_{n+i-1}\sigma & \\ & = & z[i,1]\sigma \odot t_1\theta + \ldots + z[i,n+i-1]\sigma \odot t_{n+i-1}\theta & \text{par hypothèse d'induction} \\ & = & z[i,1]\sigma_1 \odot t_1\theta + \ldots + z[i,n+i-1]\sigma_1 \odot t_{n+i-1}\theta & \text{car } \sigma_1 = \sigma|_{\mathcal{Z}_L(\mathcal{C})} \\ & = & u_i\theta & \text{par définition de } \theta \end{array}$$

Nous en déduisons donc que $\mathcal{C}\theta = \mathcal{C}\sigma$, et donc que $\mathcal{C}\theta$ est satisfaisable. □

Exemple 6.59
Reprenons l'exemple 6.53. Par la proposition 6.57, nous savons que les variables de contextes correspondant aux contraintes définissantes, c'est-à-dire $z[1,1], z[1,2], z[2,1], z[2,2]$ *et* $z[2,3]$ *peuvent être instanciées par un polynôme ayant un degré inférieur à* $d°(\mathcal{C}) = 2$. *Nous choisissons :*

$$\sigma : \{z[1,1] \mapsto 0; z[1,2] \mapsto h; z[2,1] \mapsto h+1; z[2,2] \mapsto 1; z[2,3] \mapsto 0\}.$$

Nous effectuons le remplacement dans les deux premières équations :

$$\begin{array}{l} h \odot (b + h^2(a)) = h(x_1) + h^2(x_2) \\ (h+1) \odot (h(a) + a) + 1 \odot (b + h^2(a)) = x_1 + a \end{array}$$

Nous construisons $\mathcal{M} = \{h(x_1) + h^2(x_2) = h(b) + h^3(a); a + b = x_1 + a\}$. *Ce problème d'unification a (au moins) une solution :* $\theta = \{x_1 \mapsto b, x_2 \mapsto h(a)\}$. *Nous appliquons* θ *sur le système de contraintes* $\mathcal{C}$ *de départ. Nous obtenons :*

$$\mathcal{C}\theta := \left\{ \begin{array}{lcl} \mathsf{h}(a) + a, b + \mathsf{h}^2(a) & \Vdash & \mathsf{h}(b) + \mathsf{h}^3(a) \\ \mathsf{h}(a) + a, b + \mathsf{h}^2(a), b + \mathsf{h}^2(a) & \Vdash & b + a \\ \mathsf{h}(a) + a, b + \mathsf{h}^2(a), b + \mathsf{h}^2(a), \mathsf{h}(b) + \mathsf{h}(a) & \Vdash & \mathsf{h}(b) + \mathsf{h}^3(a) + b + a \end{array} \right.$$

Nous obtenons un système de contraintes clos. Les deux premières contraintes sont satisfaisables par construction. Il reste à vérifier que la dernière contrainte est satisfaisable. Il suffit de prendre : $\{z[3,1] \mapsto h+1; z[3,2] \mapsto h+1; z[3,3] \mapsto 0; z[3,4] \mapsto 0\}$.

Nous pouvons maintenant montrer le théorème suivant, énoncé au début de la partie 6.3.

Théorème 6.15
Le problème de la satisfaisabilité d'un système de contraintes bien formé dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{ACUNh})$ *est décidable.*

Démonstration.
La procédure décrite tout le long de la partie 6.3 est correcte et complète.

Correction : Soit $\mathcal{C}_1$ un système de contraintes facteurs-préservant obtenu en appliquant la première partie de notre procédure sur un système $\mathcal{C}$ de contraintes bien formé. Les lemmes 6.26 et 6.30 nous assurent que le système $\mathcal{C}_1$ est bien formé. Soit $\mathcal{C}_2$ le système de contraintes obtenu à partir de $\mathcal{C}_1$ en remplaçant les différents facteurs par des constantes (application de l'abstraction ρ). Le système $\mathcal{C}_2$ obtenu est bien formé (*cf.* proposition 6.49). Notons $\mathcal{S}(\mathcal{C}_2)$ le système d'équations associé à $\mathcal{C}_2$ et supposons que ce système ait une solution. Nous en déduisons que le système $\mathcal{C}_2$ a une solution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$ (lemme 6.54). Par le lemme 6.31, nous en déduisons que $\mathcal{C}_1$ a une solution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$, et les lemmes 6.26 et 6.30 nous assurent que $\mathcal{C}$ a une solution dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Les systèmes $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$ et $(\mathcal{I}_{\mathsf{DY}}, \mathsf{E})$ sont équivalents (proposition 6.2), ce qui nous permet de conclure.

Complétude : Soit $\mathcal{C}$ un système de contraintes bien formé et σ une solution du système $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{E})$. Par la proposition 6.2, nous savons que σ est une solution de $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Grâce au lemme 6.21, nous pouvons supposer que σ est une solution conservatrice de $\mathcal{C}$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathcal{R}_{\mathsf{E}})$. Soit $\mathscr{C}'$ l'ensemble fini de systèmes de contraintes bien formés obtenu en appliquant l'algorithme 6.5 sur $\mathcal{C}$. Par le lemme 6.26, nous savons qu'il existe $\mathcal{C}' \in \mathscr{C}$ tel que σ est une solution conservatrice de $\mathcal{C}'$. Par le lemme 6.30, nous savons qu'il existe un système de contraintes bien formé $\mathcal{C}_\theta$ ayant une solution non-effondrante dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$. Nous en déduisons que $\mathcal{C}_\theta$ est un système facteurs-préservant (lemme 6.48). Par le lemme 6.31, $\mathcal{C}_\theta^\rho$ a une solution (sur signature réduite) dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$, et la proposition 6.58, nous permet de conclure. $\square$

6.4 Résultats d'indécidabilité

Dans cette partie, nous établissons l'indécidabilité du problème de la satisfaisabilité de systèmes de contraintes bien formés dans le cas des théories ACh et AGh.

6.4.1 Théorie ACh

Dans cette partie, nous montrons que le problème de la satisfaisabilité de systèmes de contraintes bien formés est indécidable dans le cas de la théorie ACh. Ce résultat s'obtient :

1. en réduisant le problème de l'unification modulo une théorie équationnelle au problème de la satisfaisabilité d'un système de contraintes bien formé, et
2. en montrant que le problème d'unification modulo ACh est indécidable. Ce résultat est dû à P. Narendran [Nar96].

Problème d'unification modulo une théorie équationnelle E

Entrée : Deux termes $u, v \in \mathcal{T}(\mathcal{F}, \mathcal{X})$.

Sortie : Est-ce qu'il existe une substitution σ telle que $u\sigma =_{\mathsf{E}} v\sigma$?

Codage. Soit $u, v \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ une instance du problème d'unification. Tous les symboles de fonctions sont supposés publics à l'exception d'une constante que nous notons k. Nous supposons que la constante k n'apparaît pas dans les termes u et v. Nous notons $\mathcal{F}_0$ le sous-ensemble de $\mathcal{F}$ composé des symboles de fonctions d'arité 0 (c'est-à-dire des constantes), à l'exception de la constante k. Notons $x_1, \ldots, x_n$ les variables de u et de v. Nous considérons le système de contraintes bien formé suivant :

$$\mathcal{C} = \left\{ \begin{array}{rcl} \mathcal{F}_0 & \Vdash & x_1 \\ & \vdots & \\ \mathcal{F}_0 & \Vdash & x_n \\ \mathcal{F}_0,\ \{u\}_k & \Vdash & \{v\}_k \end{array} \right.$$

Proposition 6.60
Soient $u, v \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ et E une théorie équationnelle. Le système $\mathcal{C}$, dont la construction est décrite ci-dessus, admet une solution dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{E})$ si, et seulement si, les termes u et v sont unifiables modulo E.

Démonstration.
($\Rightarrow$) Si le système $\mathcal{C}$ admet une solution, cela signifie qu'il existe une substitution σ telle que $\{u\sigma\}_k =_{\mathsf{E}} \{v\sigma\}_k$. Le symbole de chiffrement est un symbole libre, nous en déduisons donc que $u\sigma =_{\mathsf{E}} v\sigma$.
($\Leftarrow$) Réciproquement, supposons qu'il existe une substitution σ telle que $u\sigma =_{\mathsf{E}} v\sigma$. Pour tout $i \leq n$, on a $x_i\sigma \in \mathcal{T}(\mathcal{F})$ et l'on peut supposer que la constante k n'apparaît dans aucun des $x_i\sigma$. Nous en déduisons que $x_i\sigma$ est déductible de $\mathcal{F}_0$ par l'intermédiaire de la règle (C). La substitution σ est solution des n premières contraintes. De plus, nous avons $\{u\sigma\}_k =_{\mathsf{E}} \{v\sigma\}_k$; la substitution σ est donc également solution de la dernière contrainte. D'où le résultat. $\square$

L'unification modulo ACh est indécidable [Nar96]. Nous en déduisons donc le résultat suivant :

Théorème 6.61
Le problème de la satisfaisabilité d'un système de contraintes bien formé dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{ACh})$ est indécidable.

6.4.2 Théorie AGh

Dans cette partie, nous montrons que le problème de la satisfaisabilité d'un système de contraintes bien formé est indécidable dans le cas de la théorie équationnelle AGh [Del06b]. Tout d'abord, et contrairement au cas de la théorie ACh, le problème de l'unification est connu comme étant décidable pour la théorie AGh [Baa93]. Le codage générique décrit dans la partie précédente ne peut donc pas s'appliquer. En fait, la « partie Dolev-Yao » du modèle ne joue aucun rôle dans ce résultat d'indécidabilité. Le codage que nous proposons permet d'établir que le problème de la satisfaisabilité de systèmes de contraintes bien formés est indécidable dans le système d'inférence réduit à la règle d'inférence (M_{E}). Dans la partie 6.4.2.1, nous établissons le résultat suivant en réduisant le $10^{\text{ème}}$ problème d'Hilbert, connu pour être indécidable [DMR76].

Théorème 6.62
Le problème de la satisfaisabilité d'un système de contraintes bien formé dans $(\mathcal{I}_{\mathsf{M_E}}, \mathsf{AGh})$ *est indécidable.*

Nous montrons ensuite (*cf.* partie 6.4.2.2) comment obtenir le résultat d'indécidabilité pour le problème de la satisfaisabilité de système de contraintes bien formé dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$.

6.4.2.1 Indécidabilité dans $(\mathcal{I}_{\mathsf{M_E}}, \mathsf{AGh})$

Étant donné une instance S du $10^{\text{ème}}$ problème d'Hilbert, nous construisons $\mathcal{C}(S)$, un système de contraintes bien formé, tel que S a une solution $(v_1, \ldots, v_n)$ dans $\mathbb{Z}$ si, et seulement si, $\mathcal{C}(S)$ a une solution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{AGh}})$.

$10^{\text{ème}}$ Problème d'Hilbert

Entrée : un ensemble fini S d'équations diophantiennes où chacune des équations est de la forme : $x_i = m,\ x_i + x_{i'} = x_j$, ou $x_i^2 = x_j$.

Sortie : Est-ce que S a une solution dans $\mathbb{Z}$?

Pour réaliser ce codage, nous devons trouver un moyen de coder un entier dans un terme. Dans la théorie AGh, un terme $t \in \mathcal{T}(\mathcal{F})$ tel que $Fact_{\mathsf{E}}(t) = \{f_1, \ldots, f_n\}$ peut s'écrire :

$$p_{f_1} \odot f_1 + \ldots + p_{f_n} \odot f_n \text{ où } p_{f_1}, \ldots, p_{f_n} \in \mathbb{Z}[\mathsf{h}].$$

Nous choisissons de coder l'entier v dans le terme t en comptant le nombre d'occurrences d'un facteur donné dans t. Ce nombre d'occurrence est défini formellement de la façon suivante :

Définition 6.63 (nombre d'occurrences)
Soit $t \in \mathcal{T}(\mathcal{F})$ *et* f *un facteur. Le nombre d'occurrences de* f *dans* t*, noté* $\mathcal{N}(f,t)$*, est égal à* 0 *si* $f \notin Fact_{\mathsf{E}}(t)$ *et* $p_f(0)$ *sinon.*

Exemple 6.64
Soit $p = (3\mathsf{h}^2 + -2)$ *et* $t = a + 2b$*. Nous avons :*

$$p \odot t = 3\mathsf{h}^2(a + 2b) + -2(a + 2b) = (3\mathsf{h}^2 + -2) \odot a + (6\mathsf{h}^2 + -4) \odot b.$$

Nous avons $\mathcal{N}(a, p \odot t) = -2$ *et* $\mathcal{N}(b, p \odot t) = -4$.

Notre codage est composé de deux parties. La première est indépendante des équations de S. Elle est utilisée pour introduire les différentes variables de notre système de contraintes et assurer quelques égalités après instanciation de ces variables par σ, une solution de $\mathcal{C}(S)$ (*cf.* lemme 6.66). Dans la deuxième partie de notre codage, nous prenons en compte les équations de S, chaque équation étant codé dans une contrainte de déduction.

Partie 1 : Codage du produit

Soient $x_1, \ldots, x_n$ les variables de S. Nous décrivons ici la première partie $\mathcal{A}(n)$ de notre système de contraintes. Pour chaque i tel que $1 \leq i \leq n$, le système de contraintes $\mathcal{A}(n)$ contient les cinq contraintes de déductions suivantes (X_i, X_i' et Y_i sont des variables) :

$$\begin{array}{rcll}
\mathsf{h}^{p+n+2}(a) & \Vdash & \mathsf{h}^{p+n+2}(X_i) & (\tau_1)\\
\mathsf{h}^{p+n+2}(a) & \Vdash & \mathsf{h}^{p+n+2}(Y_i) & (\tau_1)\\
\mathsf{h}^{p+n+1}(b),\ \mathsf{h}^{p+n+2}(a) & \Vdash & \mathsf{h}^{p+n+1}(X'_i) & (\tau_1)\\
\mathsf{h}^{p+n}(a+b),\ \mathsf{h}^{p+n+1}(b),\ \mathsf{h}^{p+n+2}(a) & \Vdash & \mathsf{h}^{p+n}(X_i+X'_i) & (\tau_2)
\end{array}$$

$$\begin{array}{rcll}
\mathsf{h}^{p+n-1}(X_1+b),\ \mathsf{h}^{p+n-2}(X_2+b),\ \ldots,\ \mathsf{h}^{p+n-i}(X_i+b), & & & \\
\mathsf{h}^{p+n}(a+b),\ \mathsf{h}^{p+n+1}(b),\ \mathsf{h}^{p+n+2}(a) & \Vdash & \mathsf{h}^{p+n-i}(Y_i+X'_i) & (\tau_3)
\end{array}$$

On note $\mathcal{A}_1(n)$ (resp. $\mathcal{A}_2(n)$, $\mathcal{A}_3(n)$) le système de contraintes composé des contraintes de déduction de type τ_1 (resp. τ_2, τ_3).

Exemple 6.65
Nous illustrons la première partie de notre codage en choisissant $n = 3$, et nous regroupons les contraintes du même type.

$$\mathcal{A}_1(3) := \left\{ \begin{array}{lllllllll}
\mathsf{h}^8(a) & \Vdash & \mathsf{h}^8(X_1) & \mathsf{h}^8(a) & \Vdash & \mathsf{h}^8(Y_1) & \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^7(X'_1)\\
\mathsf{h}^8(a) & \Vdash & \mathsf{h}^8(X_2) & \mathsf{h}^8(a) & \Vdash & \mathsf{h}^8(Y_2) & \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^7(X'_2)\\
\mathsf{h}^8(a) & \Vdash & \mathsf{h}^8(X_3) & \mathsf{h}^8(a) & \Vdash & \mathsf{h}^8(Y_3) & \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^7(X'_3)
\end{array}\right.$$

$$\mathcal{A}_2(3) := \left\{ \begin{array}{lll}
\mathsf{h}^6(a+b),\ \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^6(X_1+X'_1)\\
\mathsf{h}^6(a+b),\ \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^6(X_2+X'_2)\\
\mathsf{h}^6(a+b),\ \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^6(X_3+X'_3)
\end{array}\right.$$

$$\mathcal{A}_3(3) := \left\{ \begin{array}{rll}
\mathsf{h}^5(X_1+b),\ \mathsf{h}^6(a+b),\ \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^5(Y_1+X'_1)\\
\mathsf{h}^4(X_2+b),\ \mathsf{h}^5(X_1+b),\ \mathsf{h}^6(a+b),\ \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^4(Y_2+X'_2)\\
\mathsf{h}^3(X_3+b),\ \mathsf{h}^4(X_2+b),\ \mathsf{h}^5(X_1+b),\ \mathsf{h}^6(a+b),\ \mathsf{h}^7(b),\ \mathsf{h}^8(a) & \Vdash & \mathsf{h}^3(Y_3+X'_3)
\end{array}\right.$$

Lemme 6.66
Soit $n \in \mathbb{N}$ et σ une solution de $\mathcal{A}(n)$ dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$. Nous avons :

1. *Pour $1 \leq i \leq n$, $\mathcal{N}(a, X_i\sigma) = \mathcal{N}(b, X'_i\sigma)$,*
2. *Pour $1 \leq i \leq n$, $\mathcal{N}(a, Y_i\sigma) = \mathcal{N}(a, X_i\sigma)^2$.*

Démonstration.
Soit σ une solution de $\mathcal{A}(n)$. Les contraintes de type τ_1 nous assurent que :

$$\mathcal{N}(b, X_i\sigma) \ = \ \mathcal{N}(b, Y_i\sigma) \ = \ \mathcal{N}(a, X'_i\sigma) \ = \ 0 \tag{6.6}$$

Grâce aux contraintes de type τ_2, nous avons :

$$\mathcal{N}(a, X_i\sigma) + \mathcal{N}(a, X'_i\sigma) \ = \ \mathcal{N}(b, X_i\sigma) + \mathcal{N}(b, X'_i\sigma)$$

On en déduit que : $\mathcal{N}(a, X_i\sigma) = \mathcal{N}(b, X'_i\sigma)$.

Maintenant, considérons la $i^{\text{ème}}$ contrainte de type τ_3, c'est-à-dire :

$$\begin{array}{rcl}
\mathsf{h}^{p+n-1}(X_1+b),\ \mathsf{h}^{p+n-2}(X_2+b),\ \ldots,\ \mathsf{h}^{p+n-i}(X_i+b), & & \\
\mathsf{h}^{p+n}(a+b),\ \mathsf{h}^{p+n+1}(b),\ \mathsf{h}^{p+n+2}(a) & \Vdash & \mathsf{h}^{p+n-i}(Y_i+X'_i).
\end{array}$$

Cette contrainte nous assure qu'il existe $z \in \mathbb{Z}$ tel que :

$$\begin{array}{rcl} z + z \times \mathcal{N}(b, X_i\sigma) & = & \mathcal{N}(b, X_i'\sigma) + \mathcal{N}(b, Y_i\sigma) \\ z \times \mathcal{N}(a, X_i\sigma) & = & \mathcal{N}(a, X_i'\sigma) + \mathcal{N}(a, Y_i\sigma) \end{array}$$

Grâce à (6.6) et au fait que $\mathcal{N}(a, X_i\sigma) = \mathcal{N}(b, X_i'\sigma)$, on en déduit que $\mathcal{N}(a, Y_i\sigma) = \mathcal{N}(a, X_i\sigma)^2$. $\square$

Partie 2 : Codage des équations de S

Dans cette partie, nous décrivons la deuxième partie de notre codage. Nous construisons un système de contraintes $\mathcal{B}(S)$ à partir des équations de $S = \{e_1, \ldots, e_p\}$. Le système $\mathcal{B}(S)$ contient autant de contraintes que S contient d'équations. Nous les notons $\mathsf{d}_1, \ldots, \mathsf{d}_p$.

Posons $T_0 = \{\mathsf{h}^{p+n-j}(X_j + b) \mid 1 \leq j \leq n\} \cup \{\mathsf{h}^{p+n}(a + b),\ \mathsf{h}^{p+n+1}(b),\ \mathsf{h}^{p+n+2}(a)\}$. L'ensemble T_0 est l'ensemble des termes obtenus à la fin de la première partie du codage. Nous devons les conserver dans l'ensemble des hypothèses de chacune de nos contraintes pour obtenir un système bien formé. Pour k allant de 1 à p, nous construisons la contrainte de déduction d_k de la façon suivante :

– si $e_k =$ « $x_i = m$ » alors

$$T_k = T_{k-1} \cup \{\mathsf{h}^{p-k}(X_i) + c_k\} \text{ et } \mathsf{d}_k = T_k \Vdash \mathsf{h}^{p-k}(ma) + c_k,$$

– si $e_k =$ « $x_i + x_{i'} = x_j$ » alors :

$$T_k = T_{k-1} \cup \{\mathsf{h}^{p-k}(X_i + X_{i'}) + c_k\} \text{ et } \mathsf{d}_k = T_k \Vdash \mathsf{h}^{p-k}(X_j) + c_k,$$

– si $e_k =$ « $x_i = x_j^2$ » alors :

$$T_k = T_{k-1} \cup \{\mathsf{h}^{p-k}(X_i) + c_k\} \text{ et } \mathsf{d}_k = T_k \Vdash \mathsf{h}^{p-k}(Y_j) + c_k.$$

Exemple 6.67
Soit $S_e = \{x_1 = 2,\ \ x_2^2 = x_3,\ \ 3x_2 + x_3 = x_1\}$. Nous obtenons :

$$\mathcal{B}(S_e) = \left\{ \begin{array}{rcl} \mathsf{h}^2(X_1) + c_1,\ T_0 & \Vdash & \mathsf{h}^2(2a) + c_1 \\ \mathsf{h}(X_3) + c_2,\ \mathsf{h}^2(X_1) + c_1,\ T_0 & \Vdash & \mathsf{h}(Y_2) + c_2 \\ 3X_2 + X_3 + c_3,\ \mathsf{h}(X_3) + c_2,\ \mathsf{h}^2(X_1) + c_1,\ T_0 & \Vdash & \mathsf{h}(X_1) + c_3 \end{array} \right.$$

Proposition 6.68
Soit S un système d'équations (à n variables) et $\mathcal{C}(S)$ le système de contraintes $\mathcal{A}(n) \cup \mathcal{B}(S)$ obtenu en appliquant la procédure décrite ci-dessus. Nous avons :

1. *$\mathcal{C}(S)$ est un système de contraintes de déduction bien formé,*
2. *S a une solution dans $\mathbb{Z}$ si, et seulement si, $\mathcal{C}(S)$ a une solution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$.*

Démonstration.
1. Par construction, le système $\mathcal{C}(S)$ est monotone. Le fait que les variables aient été introduites au début et une à une nous assure que le système de contraintes $\mathcal{C}(S)$ est bien formé.
2. Soit $v_1, \ldots, v_n$ une solution de S. Posons :

$$\sigma = \{X_1 \mapsto v_1 \odot a; \ldots; X_n \mapsto v_n \odot a; \;\; X'_1 \mapsto v_1 \odot b; \ldots, X'_n \mapsto v_n \odot b; \\ Y_1 \mapsto v_1^2 \odot a; \ldots; Y_n \mapsto v_n^2 \odot a\}.$$

Nous allons montrer que σ est une solution de $\mathcal{C}(S)$. Pour cela, nous devons montrer que pour chaque contrainte $T \Vdash u \in \mathcal{C}(S)$, il existe une preuve de $T\sigma \vdash u\sigma$ dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$. De telles preuves s'obtiennent en utilisant le dernier terme introduit dans l'ensemble des hypothèses.

Réciproquement, soit σ une solution de $\mathcal{C}(S)$. Posons $v_i = \mathcal{N}(a, X_i\sigma)$ pour tout i tel que $1 \leq i \leq n$. Nous allons montrer que $(v_1, \ldots, v_n)$ est une solution de S. Par le lemme 6.66, nous savons que $\mathcal{N}(a, Y_i\sigma) = \mathcal{N}(a, X_i\sigma)^2$. Il nous reste à montrer que $(v_1, \ldots, v_n)$ est une solution de chacune des équations de S. Soit e_k la $k^{\text{ème}}$ équation de S. Considérons la contrainte de $\mathcal{B}(S)$ correspondant à cette équation. Supposons que cette équation soit de la forme « $x_i = x_j^2$ » (les autres cas sont similaires). La contrainte correspondante à cette équation (après instanciation des variables par σ) est de la forme :

$$T_{k-1}\sigma, \; \mathsf{h}^{p-k}(X_i\sigma) + c_k \Vdash \mathsf{h}^{p-k}(Y_j\sigma) + c_k$$

La satisfaisabilité de cette contrainte nous assure que $\mathcal{N}(a, X_i\sigma) = \mathcal{N}(a, Y_j\sigma)$ et nous concluons grâce au lemme 6.66. □

6.4.2.2 Indécidabilité dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$

Nous venons d'établir que le problème de la satisfaisabilité d'un système de contraintes bien formé est indécidable dans $(\mathcal{I}_{\mathsf{M_E}}, \mathcal{R}_{\mathsf{E}})$. Le système que nous avons construit ne fait intervenir que des constantes et des symboles de $sig(\mathsf{AGh})$. Nous allons pouvoir montrer que ce même codage permet d'établir le résultat d'indécidabilité dans le cas du système d'inférence $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$. Ce résultat s'obtient par l'intermédiaire du lemme 6.21 assurant l'existence d'une solution conservatrice.

Théorème 6.69
Le problème de la satisfaisabilité d'un système de contraintes bien formé dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$ *est indécidable.*

Démonstration.
Soit S une instance du $10^{\text{ème}}$ problème d'Hilbert et $\mathcal{C}(S)$ le système de contraintes de déduction bien formé obtenu en appliquant la procédure décrite dans la partie 6.4.2.1. Tout d'abord, il est évident que si $\mathcal{C}(S)$ a une solution dans $(\mathcal{I}_{\mathsf{M_E}}, \mathsf{AGh})$ alors $\mathcal{C}(S)$ a aussi une solution dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$. Réciproquement, soit σ une solution de $\mathcal{C}(S)$ dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$. Par le lemme 6.21, nous pouvons supposer sans perte de généralité que σ est une solution conservatrice. Ainsi, tous les termes de $\mathcal{C}(S)\sigma$ sont construits à partir des symboles de $sig(\mathsf{AGh})$ et des constantes. Soit $T \vdash u \in \mathcal{C}(S)\sigma$. Grâce au lemme de localité (lemme 6.9), nous savons que tous les nœuds d'une preuve minimale de $T \vdash u$ sont étiquetés par des termes dans $St_{\mathsf{E}}(T, u)$, c'est-à-dire des termes dans $T \cup \{u\}$ ou des constantes. Autrement dit, les

règles d'inférence telles que $\mathsf{Proj}_1, \mathsf{Proj}_2, \mathsf{D}, \mathsf{C}^-$ ne peuvent pas être utilisées dans un tel arbre de preuve. Cela nous permet de conclure que σ est une solution de $\mathcal{C}(S)$ dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathsf{AGh})$. Ainsi, nous savons que $\mathcal{C}(S)$ a une solution dans $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$ si, et seulement si, $\mathcal{C}(S)$ a une solution dans $(\mathcal{I}_{\mathsf{M}_\mathsf{E}}, \mathsf{AGh})$. Nous en déduisons donc que S a une solution dans $\mathbb{Z}$ si, et seulement si, $\mathcal{C}(S)$ a une solution dans $\mathcal{I}_{\mathsf{M}_\mathsf{E}}$. Ceci nous permet de conclure. □

6.5 Discussion

La procédure que nous avons développée dans la partie 6.3 est en fait assez générale et permet de traiter le problème de la résolution de systèmes de contraintes bien formés pour différentes théories équationnelles (*e.g.* ACUNh, ACUN, voire même AG). En revanche, notre procédure ne s'applique pas à certaines théories équationnelles semblant pourtant « assez proches » (*e.g.* AGh). Nous passons en revue différentes théories équationnelles dans le but de mettre en évidence la ou les étapes de la procédure qui ne sont pas correctes et/ou complètes pour la théorie considérée.

Théorie standard de Dolev-Yao : $\mathcal{I}_{\mathsf{DY}}$

Dans le cas de théorie standard de Dolev-Yao, notre procédure est correcte et complète. Il s'avère en fait qu'après avoir appliqué les deux premières étapes décrites dans les parties 6.3.1 et 6.3.2, le système de contraintes obtenu est vide (et donc trivialement satisfaisable) si, et seulement si, le système de départ est satisfaisable.

Théorie du « ou » exclusif : $(\mathcal{I}_{\mathsf{DY}}, \mathsf{ACUN})$

Pour traiter cette théorie, nous n'avons pas besoin de toute la « puissance » de notre procédure. En effet, à l'issue des deux premières étapes, nous obtenons un système de contraintes que nous devons résoudre à l'aide du schéma de règle suivant :

$$\frac{T \vdash u_1 \quad \dots \quad T \vdash u_n}{T \vdash u_1 \oplus \dots \oplus u_n}$$

Résoudre un tel système de contraintes est alors équivalent à résoudre un système d'équations quadratiques dans $\mathbb{Z}/2\mathbb{Z}$. Puisque $\mathbb{Z}/2\mathbb{Z}$ est un corps fini, il est possible d'énumérer toutes les possibilités et de vérifier si l'une de ces possibilités correspond à une solution du système.

Théorie groupe abélien : $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AG})$

Dans le cas de la théorie AG, toutes les étapes de notre procédure sont utiles. Le seul résultat, non établi à l'heure actuelle, est le résultat technique concernant l'unification et énoncé dans le lemme 6.28 dans le cas de la théorie ACUNh. Si ce lemme reste vrai dans le cas de la théorie équationnelle AG, alors la procédure proposée dans ce chapitre permet également de traiter cette théorie. Après abstraction des différents facteurs par des constantes, nous devons rétablir la propriété de bonne formation des systèmes de contraintes obtenus afin de pouvoir les résoudre. Ensuite, nous pouvons appliquer les différents résultats établis dans le cas de la théorie ACUNh. En particulier, la proposition 6.57 nous permet de borner les contextes des contraintes définissantes. Dans le cas de la théorie AG, la constante $Q_{max}(\mathcal{C})$ est un élément de $\mathbb{Z}$, ce qui nous permet d'assurer (lorsque le système est satisfaisable) l'existence d'une solution où les valeurs des contextes des contraintes définissantes sont comprises entre 0 et $Q_{max}(\mathcal{C})$. La connaissance de cette borne nous permet d'énumérer les différents cas et de trouver la valeur des autres variables. Nous obtenons alors un système clos dont nous pouvons décider la satisfaisabilité.

Théorie associative et commutative avec homomorphisme : $(\mathcal{I}_{\mathsf{DY}}, \mathsf{ACh})$

Nous avons vu que le problème de la résolution de systèmes de contraintes bien formés est indécidable pour cette théorie. Bien entendu, la procédure que nous avons proposée, dans le cas de la théorie ACUNh, ne s'applique pas. Le premier obstacle que nous rencontrons est que le problème d'unification est indécidable pour cette théorie (Le lemme 6.30 est donc faux dans le cas de la théorie ACh).

Théorie groupe abélien avec homomorphisme : $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AGh})$

Dans le cas de la théorie AGh, l'unification est décidable et finitaire [Baa93]. Nous avons vu (*cf.* partie 6.4.2) que le problème de la résolution des systèmes de contraintes bien formés est indécidable pour cette théorie : la procédure que nous avons proposée pour ACUNh ne s'applique donc pas pour traiter la théorie AGh. Les principaux problèmes viennent en fait de la dernière partie de la procédure, c'est-à-dire de la résolution des systèmes d'équations quadratiques dans $\mathbb{Z}[\mathsf{h}]$. En supposant que l'on arrive à calculer le polynôme $Q_{max}(\mathcal{C})$ et à borner le degré des polynômes utilisés comme valeurs pour les contextes des contraintes définissantes, cela ne nous permet pas de conclure puisque, dans $\mathbb{Z}[h]$, il existe un nombre infini de polynôme ayant un degré inférieur à un entier donné. L'existence de cette borne, ne nous permet donc pas de conclure.

Théorie associative et commutative : $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AC})$

Dans le cas $(\mathcal{I}_{\mathsf{DY}}, \mathsf{AC})$, c'est également cette dernière partie de la procédure qui ne marche pas, mais le problème est un peu différent. La proposition 6.57 nous assurant l'existence de petits contextes, pour les contextes des contraintes définissantes, ne peut s'établir que si l'on peut compenser la « perte subie » en choisissant de prendre plus ou moins des autres termes présents dans l'ensemble des hypothèses. Le but est en effet de rétablir l'équilibre et de montrer que les nouvelles valeurs proposées constituent encore une solution. Le problème est que dans le cas AC, il n'est pas toujours possible de diminuer la valeur d'un contexte pour rétablir l'équilibre. En effet, nous pouvons réaliser cette opération seulement si le nouveau contexte obtenu est encore un entier positif.

Le problème de la résolution du cas AC est un problème difficile. Il est particulièrement intéressant et a des retombées importantes, bien au-delà de la seule théorie AC. En effet, nous verrons au chapitre 7, que la résolution du cas AC est un problème central pour obtenir une procédure générique (valable pour un grand nombre de théories équationnelles). Nous verrons comment la plupart des théories équationnelles pertinentes du point de vue de la vérification des protocoles cryptographiques peuvent se réduire à l'associativité et commutativité d'un ou plusieurs symboles. Résoudre le cas « AC pur » permettrait de faire un pas supplémentaire faire l'obtention d'une procédure générique.

6.6 Comparaisons

6.6.1 Travaux de J. Millen et V. Shmatikov

La procédure que nous avons proposée pour résoudre les systèmes de contraintes bien formés en présence de la théorie ACUNh (*cf.* partie 6.3) s'inspire du travail réalisé par J. Millen et V. Shmatikov dans le cas de la théorie AG [MS05]. Nous avons, comme eux, décomposé notre procédure en plusieurs étapes. Les deux premières, concernant l'existence d'une solution conservatrice, et la réduction à un système se résolvant avec la règle ($\mathsf{M_E}$), sont assez similaires. J. Millen et V. Shmatikov établissent ensuite un lien entre la satisfaisabilité des systèmes de contraintes et la résolution de systèmes d'équations quadratiques dans $\mathbb{Z}$. Mais, comme l'illustre l'exemple 6.70, l'étape de leur procédure consistant à effectuer un changement de variable n'est pas correcte.

Exemple 6.70
Considérons le système suivant :

$$\mathcal{C} = \left\{ \begin{array}{rcl} 3a+b & \Vdash & 2x+b \\ 3a+b,\ 2x+b & \Vdash & 3a+b \end{array} \right.$$

Ce système n'admet pas de solution puisque la première contrainte n'est pas satisfaisable. J. Millen et V. Shmatikov commencent par effectuer le changement de variable suivant : $2x+b=y$. Ils obtiennent alors un nouveau système (noté ici $\mathcal{C}'$).

$$\mathcal{C}' = \left\{ \begin{array}{rcl} 3a+b & \Vdash & y \\ 3a+b,\ y & \Vdash & 3a+b \end{array} \right.$$

Or, ce système $\mathcal{C}'$, obtenu à partir de $\mathcal{C}$ après application du changement de variable, est trivialement satisfaisable.

D'autre part, ils réalisent, sans aucune justification, l'abstraction des facteurs par des constantes. Ils se ramènent ainsi à résoudre des systèmes bien formés sur signature réduite. Or, comme nous l'avons vu, cette abstraction ne préserve pas la bonne formation des systèmes (*cf.* exemple 6.32). Cette étape nécessite donc d'être justifiée et c'est l'objet de la partie 6.3.3 de ce chapitre. La dernière étape de notre procédure, c'est-à-dire celle consistant à résoudre les systèmes d'équations quadratiques, est finalement assez éloignée de celle proposée par J. Millen et V. Shmatikov dans [MS05].

6.6.2 Travaux de Y. Chevalier et M. Rusinowitch

Dans [CR05], Y. Chevalier et M. Rusinowitch montrent un résultat de combinaison, pour le problème de la résolution de systèmes de contraintes bien formés, dans le cas des théories disjointes. Leur résultat de combinaison est établi dans le cas des systèmes de contraintes simples avec équations. De plus, ils montrent que pour un certain nombre de théories équationnelles (*e.g.* ACUN, AG), ce problème est décidable. La procédure proposée dans le cas de la théorie AG est très simple et sans communes difficultés avec la procédure proposée par J. Millen et V. Shmatikov dans [MS05], ou avec la procédure que nous avons proposée dans la partie 6.3 pour résoudre le cas ACUNh. Afin d'expliquer cette différence de complexité entre les deux procédures, nous exposons brièvement la procédure proposée par Y. Chevalier et M. Rusinowitch dans le cadre de la théorie AG.

Considérons un système de contraintes simples $\mathcal{C}$ et un ensemble d'équations $\mathcal{E}$. La procédure fonctionne de la façon suivante :

1. on commence par calculer un système de contraintes $\mathcal{C}'$, équivalent au système $\mathcal{C}$ de départ, et satisfaisant la condition suivante :
   pour toute contrainte $T \Vdash u \in \mathcal{C}'$, l'ensemble T est un ensemble de termes clos.
2. Le système de contraintes ainsi obtenu peut alors être traduit en un système d'équations linéaires, et il en est de même pour le système d'équations $\mathcal{E}$. La satisfaisabilité de tels systèmes d'équations dans $\mathbb{Z}$ est décidable [Sch86].

La transformation permettant d'obtenir $\mathcal{C}'$ à partir de $\mathcal{C}$ est très simple : elle consiste, pour chacun des termes non-clos t apparaissant dans un membre gauche d'une contrainte, à le remplacer par la partie constante du terme, c'est-à-dire par ce que nous avions appelé t^0 (*cf.* page 126). Cette transformation est illustrée dans l'exemple 6.71.

Exemple 6.71

$$\mathcal{C} = \left\{ \begin{array}{rcl} a+3b & \Vdash & x \\ a+3b,\ 2x+3b & \Vdash & 4b \end{array} \right. \qquad \mathcal{C}' = \left\{ \begin{array}{rcl} a+3b & \Vdash & x \\ a+3b,\ 3b & \Vdash & 4b \end{array} \right.$$

Le fait que le système $\mathcal{C}'$ soit équivalent au système $\mathcal{C}$, c'est-à-dire que ces deux systèmes aient même ensemble de solutions, est un résultat facile à établir. La procédure proposée par Y. Chevalier et M. Rusinowitch est très simple, mais il est clair qu'à théorie équationnelles égales, la classe des systèmes de contraintes bien formés est « plus grande » que la classe des systèmes simples bien formés avec équations.

Dans le cas de la théorie ACUNh, la technique, brièvement décrite ci-dessus dans le cas de la théorie AG, s'applique également et permet de réduire le problème de la résolution de systèmes de contraintes simples et bien formés avec équations en un problème de résolution de système d'équations linéaires dans $\mathbb{Z}/2\mathbb{Z}[\mathsf{h}]$. Ce résultat s'applique également dans le cas de la théorie AGh, et permet d'obtenir un résultat de décidabilité alors que nous avons montré (*cf.* partie 6.4.2) que pour cette même théorie, le problème est indécidable dans le modèle avec filtrage.

La contrepartie est que, à théories équationnelles égales, ce modèle avec tests d'égalités est moins riche, et ne permet pas de modéliser certains protocoles. En particulier, nous ne pouvons pas modéliser le protocole TMN (dont la description est rappelée à la figure 6.6.2) dans le modèle avec tests d'égalités, en considérant la théorie équationnelle AGh.

$$\begin{array}{rcll} A & \rightarrow & S: & A, B, \{K_a\}_{\mathsf{pub}(S)} \\ S & \rightarrow & B: & A \\ B & \rightarrow & S: & B, A, \{K_b\}_{\mathsf{pub}(S)} \\ S & \rightarrow & A: & B, K_b \oplus K_a \end{array}$$

Figure 6.8 - Description du protocole TMN, dû à T. Tatebayashi, N. Matsuzaki et D. Newman.

Le rôle de S dans le modèle avec filtrage peut se modéliser de la façon suivante (comme mentionné dans la partie 6.1.2, nous utilisons le symbole d'homomorphisme pour modéliser le chiffrement avec la clef publique du serveur) :

$$\begin{gathered}\mathsf{recv}(x_A, x_B, \mathsf{h}(x_{K_a}));\ \mathsf{send}(x_A)\\ \mathsf{recv}(x_B, x_A, \mathsf{h}(x_{K_b}));\ \mathsf{send}(x_B, x_{K_b} \oplus x_{K_a})\end{gathered}$$

Pour modéliser ce rôle dans le modèle avec tests d'égalités, il faudrait considérer un opérateur privé $\mathsf{h}^{-1}(\cdot)$ satisfaisant l'équation $\mathsf{h}^{-1}(\mathsf{h}(x)) = x$. Autrement dit, pour modéliser ce protocole dans le modèle par rôles avec tests d'égalités, il est nécessaire de considérer une théorie équationnelle plus complexe.

Chapitre 7

Réduction de la théorie équationnelle

Sommaire

7.1 Étude de cas : protocole de porte-monnaie électronique 154
- 7.1.1 Description du protocole 154
- 7.1.2 Modélisation du protocole 156

7.2 Propriété de variants finis 157
- 7.2.1 Préliminaires 158
- 7.2.2 Première caractérisation 161
- 7.2.3 Principale application 163

7.3 Comment établir la propriété de variants finis ? 166
- 7.3.1 Deuxième caractérisation 166
- 7.3.2 En présence d'axiomes orientables 168
- 7.3.3 En présence d'axiomes non-orientables 169

7.4 Application à différentes théories équationnelles 170
- 7.4.1 Variantes de Dolev-Yao 170
- 7.4.2 Théorie « signature en aveugle » 171
- 7.4.3 Théorie ACUN 171
- 7.4.4 Théorie AG 172
- 7.4.5 Théorie DH 174
- 7.4.6 Théorie équationnelle de l'étude de cas 176

7.5 Discussion 178
- 7.5.1 Travaux sur l'unification 178
- 7.5.2 Travaux de H. Comon-Lundh et V. Cortier 178
- 7.5.3 Travaux de V. Bernat et H. Comon-Lundh 179
- 7.5.4 De la difficulté à résoudre le cas AC 179

À l'heure actuelle, les procédures permettant de résoudre le problème de la vérification des protocoles cryptographiques sont nombreuses et permettent de prendre en considération différentes théories équationnelles. Au vu de la diversité de ces résultats, on pourrait croire qu'ils permettent de couvrir la majorité des protocoles dont nous aimerions faire l'étude, mais il n'en est rien. En effet, les protocoles sont développés dans des contextes variés, chacun ayant ses contraintes spécifiques de ressources en temps et en espace. Ces spécificités conduisent les concepteurs à mettre au point des primitives cryptographiques satisfaisant des propriétés de plus en plus complexes. Du point de vue de la vérification, cela demande la mise au point de nouveaux résultats afin de tenir compte de ces primitives cryptographiques et de leurs propriétés. Nous illustrons ce constat dans la partie 7.1 en présentant un protocole de porte-monnaie électronique développé récemment par une équipe de France Télécom. La vérification de ce protocole, avec les outils existant aujourd'hui, n'a pu se faire qu'au prix d'un travail de modélisation important [DK04]. Mais l'étape de modélisation est une tâche délicate qui ne peut se justifier que de manière informelle. Il est donc important de disposer de modèles suffisamment expressifs permettant la réalisation d'une modélisation précise et convaincante pour avoir confiance dans le résultat fourni par la vérification formelle.

Compte tenu de la diversité des théories équationnelles pertinentes du point de vue de la vérification des protocoles cryptographiques, il semble nécessaire de mettre au point des méthodes de vérification générique. Dans [Com04], H. Comon-Lundh a établi une ligne directrice qui permettrait d'obtenir un tel résultat.

Dans ce chapitre, nous faisons un premier pas vers l'obtention d'un résultat générique en définissant une propriété permettant de se débarrasser des axiomes orientables composant une théorie équationnelle : la *propriété des variants finis*. Dans la partie 7.2, nous définissons formellement cette propriété et nous illustrons son intérêt du point de vue de la vérification des protocoles cryptographiques. Dans la partie 7.3 nous donnons des critères permettant d'assurer qu'une théorie équationnelle donnée satisfait la propriété de variants finis, et dans la partie 7.4, nous utilisons ces critères pour établir cette propriété sur de nombreuses théories (*e.g.* « ou » exclusif, groupe abélien, ...). Enfin, dans la partie 7.5, nous mettons en avant certains travaux en rapport avec la propriété de variants finis.

7.1 Étude de cas : protocole de porte-monnaie électronique

Les protocoles cryptographiques ont pris une importance considérable avec le développement du commerce électronique. Ce dernier terme recouvre pour commencer les échanges chiffrés d'information qui ont lieu lorsque l'on retire de l'argent dans un distributeur de billets à l'aide d'une carte bancaire. Le distributeur de billets s'engage alors dans un dialogue avec la carte du client et la banque pour vérifier que l'utilisateur de la carte est honnête, dispose de la somme demandée sur son compte et ne pourra pas récupérer les billets demandés sans que son compte en soit débité. Le commerce électronique recouvre également les paiements par carte à l'aide de terminaux portables où l'utilisateur doit confirmer son identité en entrant son code secret à quatre chiffres, et aussi la monnaie électronique qui a pour but d'émuler électroniquement la monnaie courante. Cette monnaie est, pour des montants peu élevés, plus intéressante qu'une transaction bancaire par carte qui a un coût élevé pour le commerçant.

L'étude de cas considérée ici est un protocole de commerce électronique, développé récemment par une équipe de France Télécom R&D. Ce protocole permet la réalisation d'une transaction entre un porte-monnaie électronique et un serveur. Son but est de garantir un bon niveau de sécurité et de gagner en ouverture par l'utilisation de méthodes de chiffrement asymétrique, et ce à faible coût. Ces besoins spécifiques ont conduit à développer une solution originale mettant en jeu des propriétés algébriques complexes [GP01].

7.1.1 Description du protocole

Le protocole est composé de deux phases. La première a pour but d'authentifier le serveur S du point de vue du porte-monnaie P, alors que la seconde permet au porte-monnaie de s'authentifier au près du serveur et de réaliser les opérations de débit et de crédit sur les différents comptes.

Phase 1. Au cours de la première phase, le porte-monnaie P contacte le serveur S afin d'informer ce dernier qu'il souhaite réaliser une transaction. Le serveur s'authentifie auprès du porte-monnaie en répondant au challenge N_p émis par le porte-monnaie. Cette première phase est composée des trois échanges suivants :

$$\begin{array}{rcll} P & \rightarrow & S: & P,\ N_p,\ \mathsf{b}^{-s},\ \{P, \mathsf{b}^{-s}\}_{\mathsf{priv}(A)} \\ S & \rightarrow & P: & S,\ N_s,\ \{P, N_p\}_{K_S(P)} \\ & \rightarrow & P: & Mt \end{array}$$

Au cours d'un premier échange, le porte-monnaie électronique envoie au serveur S son identité P, un nonce N_p, sa clef publique b^{-s} ainsi qu'un certificat $\{P, \mathsf{b}^{-s}\}_{\mathsf{priv}(A)}$. Ce dernier a pour but d'assurer que la clef b^{-s} est bien la clef publique du porte-monnaie P. Ce certificat est un message chiffré par la clef privée d'une autorité digne de confiance, notée A. À la réception de ce premier message, le serveur S vérifie le certificat en le déchiffrant avec la clef publique de A et calcule la clef symétrique $K_S(P)$. Il s'agit d'une clef symétrique partagée entre le porte-monnaie P et le serveur S. Sur le porte-monnaie, la clef en question est directement stockée alors que le serveur S connaît le fonction $K_S(\cdot)$, une clef maître lui permettant de dériver les clefs stockées sur les différents porte-monnaie à partir de la seule connaissance de l'identité du porte-monnaie. Une fois cette clef calculée, le serveur envoie son identité S, accompagné d'un nonce N_s et d'un message chiffré afin de convaincre le porte-monnaie P qu'il est bien le serveur. Le porte-monnaie, à la réception de ce triplet, déchiffre la troisième composante avec sa clef symétrique $K_S(P)$ qu'il partage avec S et vérifie qu'il retrouve le nonce N_p engendré par ses soins à la première étape. Un montant Mt est alors saisi sur le terminal de paiement.

Phase 2. La deuxième phase permet au porte-monnaie de s'authentifier auprès du serveur S et de réaliser la transaction. Cette phase est beaucoup plus complexe et fait intervenir les propriétés algébriques de l'exponentielle modulaire.

P calcule $M = \{S, N_s, N_p, Mt\}_{K_A(P)}$
P choisit un coupon $(N, \mathsf{b}^N \mathsf{mod}\ r)$

$P \rightarrow S:\ \mathsf{hash}(\mathsf{b}^N \mathsf{mod}\ r, S, N_s, N_p, M, Mt)$
$S \rightarrow P:\ N_c$
P débite son compte d'un montant Mt

$P \rightarrow S:\ N + s \times N_c, M, Mt$
S vérifie le message hash(...) qu'il a reçu à l'étape précédente.
Puis il crédite son compte d'un montant Mt et stocke les messages M, N_s, N_p et Mt.

Le porte-monnaie construit le message M : il s'agit d'un message chiffré avec une clef symétrique partagée entre le porte-monnaie P et l'autorité A, un juge habilité à résoudre les litiges qui pourraient survenir ultérieurement. Le message M sera stocké sur le serveur S à l'issue de la transaction et pourra être utilisé en cas de conflit entre le porte-monnaie P et le serveur S. Ensuite, le porte-monnaie choisit un coupon, c'est-à-dire un couple de la forme $(N, \mathsf{b}^N \mathsf{mod}\ r)$ stocké sur sa puce électronique. Il applique la fonction de hachage, notée hash, à diverses informations concernant la transaction. Le serveur reçoit alors ce message, mais il n'est pas en mesure de l'analyser. Il le stocke et engendre un nonce N_c. Ce nonce constitue un challenge que seul le porte-monnaie à l'origine du message hash(...) devrait pouvoir résoudre. En effet, pour répondre à ce challenge, il faut connaître le secret s stockée sur la carte engagée dans la transaction. Le porte-monnaie engagé dans la transaction débite son compte et répond à ce challenge en envoyant au serveur le message $N + s \times N_c$ et les informations nécessaires pour reconstituer le message que le serveur n'a pas pu analyser à l'étape précédente. Le serveur est désormais capable de vérifier le message hash$(\cdot)$ qu'il a reçu précédemment. Il peut le construire d'une autre façon et les propriétés de l'exponentielle modulaire nous assurent l'égalité des deux expressions. En effet, nous avons :

$$\begin{array}{ll} & \mathsf{hash}((\mathsf{b}^{-s})^{N_c} \times \mathsf{b}^{N+s\times N_c},\ S,\ N_s,\ N_p,\ M,\ Mt) \\ = & \mathsf{hash}(\mathsf{b}^{-s\times N_c} \times \mathsf{b}^{N+s\times N_c},\ S,\ N_s,\ N_p,\ M,\ Mt) \\ = & \mathsf{hash}(\mathsf{b}^{-s\times N_c + N + s\times N_c},\ S,\ N_s,\ N_p,\ M,\ Mt) \\ = & \mathsf{hash}(\mathsf{b}^{N},\ S,\ N_s,\ N_p,\ M,\ Mt) \end{array}$$

L'originalité de ce protocole est de proposer une approche asymétrique à faible coût : les calculs réalisés par le porte-monnaie électronique sont limités. Les coupons sont stockés sur la carte afin d'éviter à cette dernière d'avoir à effectuer un calcul très coûteux d'exponentiation modulaire. La carte est rechargée en coupons au moment de réapprovisionner le porte-monnaie.

7.1.2 Modélisation du protocole

La modélisation de ce protocole présente plusieurs difficultés. D'une part, la représentation d'une exécution normale du protocole nécessite la prise en compte de nombreuses propriétés algébriques. Elles ont pour but de modéliser la vérification réalisée par le serveur à la fin de la session. D'autre part, si l'on souhaite réaliser une modélisation complète du protocole, d'autre traits doivent être ajouter à notre modèle, comme par exemple l'ajout de variables mutables pour représenter le solde des différents comptes ou le stockage des messages sur le serveur. Nous ne nous intéresserons pas à ce deuxième aspect ici, mais une modélisation complète du protocole, tenant compte de toutes ces spécificités, a été réalisée dans le langage PROUVÉ [BDKV05].

En ce qui concerne les propriétés algébriques des primitives cryptographiques, l'idéal serait de tenir compte de la théorie équationnelle complète, c'est-à-dire des propriétés de l'addition $+$, de la multiplication $\times$ ainsi que de toutes les propriétés de l'exponentielle modulaire. Malheureusement, la théorie ainsi obtenue est complexe et le problème d'unification, et donc le problème de la sécurité pour un nombre borné de sessions (*cf.* codage décrit à la page 143), est vraisemblablement indécidable. L'idée est donc de se restreindre aux axiomes nécessaires à l'exécution du protocole. Cela nécessite déjà de considérer plusieurs axiomes concernant l'exponentielle modulaire : les deux axiomes classiquement considérés, et correspondant à la théorie connue sous le nom de Diffie-Hellman, ne suffissent pas pour traiter cette étude de cas. Nous avons besoin de tenir compte des égalités suivantes :

$$\begin{aligned}\mathsf{exp}(x,y)\times\mathsf{exp}(x,z) &= \mathsf{exp}(x,y+z)\\ \mathsf{exp}(\mathsf{exp}(x,y),z) &= \mathsf{exp}(x,y\times z)\end{aligned}$$

En fait, nous n'avons pas réellement besoin de considérer l'axiome de distributivité. Nous pouvons nous restreindre à un axiome d'homomorphisme puisque la base utilisée est une constante (notée b). La théorie équationnelle que nous allons considérer est décrite à la figure 7.1.

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

$$\begin{aligned}\mathsf{h}(x+y) &= \mathsf{h}(x)\times\mathsf{h}(y)\\ \mathsf{exp}(\mathsf{h}(x),y) &= \mathsf{h}(x\times y)\end{aligned}$$

Figure 7.1 - Théorie équationnelle $\mathsf{E_P}$ (protocole de porte-monnaie électronique).

Pour nous restreindre à l'étude de cette théorie équationnelle, déjà complexe, nous devons effectuer deux modifications mineures au moment de la modélisation du protocole :

- la clef publique du porte-monnaie, noté b^{-s}, est modélisée par $\mathsf{h}(s)$ au lieu de $\mathsf{h}(-s)$,
- la réponse du porte-monnaie au challenge envoyé par le serveur est $N+-(s\times N_c)$.

Avec ces changements nous n'avons pas à considérer d'axiomes supplémentaires, comme par exemple $(-x)\times y = -(x\times y)$. Les axiomes décrits à la figure 7.1 sont suffisants pour modéliser le test réalisé par le serveur, puisque nous pouvons exprimer les égalités suivantes :

$$\begin{aligned}\mathsf{exp}(\mathsf{h}(s),N_c)\times\mathsf{h}(N+-(s\times N_c)) &= \mathsf{h}(s\times N_c)\times\mathsf{h}(N+-(s\times N_c))\\ &= \mathsf{h}(s\times N_c+N+-(s\times N_c))\\ &= \mathsf{h}(N)\end{aligned}$$

Les deux rôles constituant le protocole peuvent ensuite être modélisés dans le modèle de notre choix (modèle avec filtrage ou avec tests d'égalités). Le modèle avec tests d'égalités est cependant plus approprié pour modéliser le test réalisé par le serveur à la dernière étape du protocole.

Cette étude de cas met en évidence l'intérêt d'un résultat générique : le protocole étudié ici est un protocole original et la théorie équationnelle permettant de le modéliser est très différente des théories étudiées jusqu'à présent (*e.g.* « ou » exclusif, groupe abélien, Diffie-Hellman).

7.2 Propriété de variants finis

Le but de cette partie est de définir la propriété de variants finis (*cf.* partie 7.2.2) et de mettre en évidence l'intérêt de cette propriété du point de vue de la vérification des protocoles cryptographiques. Nous verrons en particulier comment cette propriété permet de réduire la théorie équationnelle considérée (*cf.* partie 7.2.3), et de se ramener, dans la majorité des cas, à l'étude du problème de la sécurité d'un protocole dans la théorie vide ou dans la théorie AC.

7.2.1 Préliminaires

Nous commençons par introduire une nouvelle notion de réécriture, ainsi que la relation de surréduction, notion couramment utilisée dans le domaine de la réécriture. Ensuite, nous relions ces deux notions par un lemme de relèvement permettant d'associer à toute séquence de réécriture une séquence de surréduction (*cf.* lemme 7.7).

Cette nouvelle relation de réécriture, notée $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$, ou parfois $\rightarrow_{\mathcal{R},\mathsf{E}}$, se définit de la façon suivante :

Définition 7.1 (réécriture modulo E, $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$)
Soit $\mathcal{R}$ un ensemble fini de règles de réécriture et E un ensemble fini d'équations (typiquement AC). La réécriture modulo E, notée $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$, est la relation définie par : $s \rightarrow_{\mathsf{E}\backslash\mathcal{R}} t$ s'il existe une position $p \in \mathcal{O}(s)$, une règle $l \rightarrow r \in \mathcal{R}$ et une substitution σ telles que $s|_p =_{\mathsf{E}} l\sigma$ et $t|_p =_{\mathsf{E}} r\sigma$.

Dans toute la suite de ce chapitre, sauf mention explicite du contraire, nous faisons référence à cette nouvelle notion de réécriture. Elle présente l'avantage d'être plus effective : même si les classes d'équivalence modulo E sont calculables, elles peuvent être très grandes et rendre le calcul d'une étape de réécriture $\rightarrow_{\mathcal{R}/\mathsf{E}}$ très coûteux. Ces difficultés sont contournées par l'utilisation de la relation $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$. Cette notion est plus restrictive puisqu'un terme se réécrit si, et seulement si, il a un sous-terme équivalent à une instance d'un membre gauche d'une règle de réécriture. Avec cette nouvelle définition, les étapes modulo E sont appliquées en dessous de la position à laquelle on réécrit. Comme l'illustre l'exemple 7.2, cette relation de réécriture est plus faible (au sens de l'inclusion) que la relation $\rightarrow_{\mathcal{R}/\mathsf{E}}$, introduite au chapitre 2.

Exemple 7.2
Considérons le système de réécriture $\mathcal{R}$ composé des deux règles $x + x \rightarrow 0$ et $x + 0 \rightarrow x$. Ce système est un système AC-convergent représentant ACUN lorsque l'on considère la relation $\rightarrow_{\mathcal{R}/\mathsf{AC}}$. En particulier, nous avons $a+(a+b) \rightarrow_{\mathcal{R}/\mathsf{AC}} 0+b$. En revanche, le terme $a + (a + b)$ est irréductible pour la relation $\rightarrow_{\mathsf{AC}\backslash\mathcal{R}}$.

La notion de système E-convergent se définit comme dans le cas de la réécriture $\rightarrow_{\mathcal{R}/\mathsf{E}}$. Cependant, cette nouvelle notion de réécriture étant plus faible, un système E-convergent pour la relation $\rightarrow_{\mathcal{R}/\mathsf{E}}$ ne l'est pas nécessairement pour la relation $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$. Les systèmes de réécriture AC-convergents associés aux théories équationnelles classiques (*e.g.* ACUN, AG) s'obtiennent en ajoutant des règles dites *d'extensions* aux systèmes AC-convergents que nous avions précédemment obtenus pour la relation $\rightarrow_{\mathcal{R}/\mathsf{E}}$.

Exemple 7.3
Considérons la théorie équationnelle ACUN. Le système de réécriture AC-convergent pour la relation $\rightarrow_{\mathcal{R}/\mathsf{AC}}$ est composée des deux règles $x + x \rightarrow 0$ et $x + 0 \rightarrow x$. Ce système n'est pas AC-convergent pour la relation $\rightarrow_{\mathsf{AC}\backslash\mathcal{R}}$. En effet, on a $a + (a + b) =_{\mathsf{ACUN}} b$, et pourtant les termes $a + (a + b)$ et b sont irréductibles pour $\rightarrow_{\mathsf{AC}\backslash\mathcal{R}}$. Le système AC-convergent

représentant ACUN *(pour la relation* $\rightarrow_{\mathsf{AC}\backslash\mathcal{R}}$*) s'obtient en ajoutant la règle d'extension suivante :* $x + (x + y) \rightarrow y$*. Avec cette nouvelle règle, le terme* $a + (a + b)$ *se réduit en* b*.*

Notation : Nous notons $\xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}}$ la fermeture réflexive-transitive de $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$, et $s \xrightarrow{\leq n}_{\mathsf{E}\backslash\mathcal{R}} t$ s'il existe une réduction de s à t de longueur au plus n.

Nous avons également besoin de définir une relation classiquement utilisée en réécriture, appelée *surréduction*. Nous étendons ensuite cette définition en présence d'une théorie équationnelle E.

Définition 7.4 (surréduction, surréduction modulo E)
Soit $\mathcal{R}$ *un ensemble fini de règles de réécriture et* t *un terme. Le terme* t *se* surréduit *en* t' *à la position* $p \in \mathcal{O}(t)$ *avec la règle* $l \rightarrow r \in \mathcal{R}$ *et la substitution* σ *s'il existe un renommage* $l' \rightarrow r'$ *de* $l \rightarrow r$ *tel que* σ *soit un unificateur de* $t|_p$ *et* l'*, et* $t' = (t[r']_p)\sigma$*. Dans ce cas, nous écrivons* $t \rightsquigarrow_\sigma t'$*.*

Si E *est une théorie équationnelle pour laquelle un algorithme d'unification existe, nous étendons cette définition pour obtenir la relation de* surréduction modulo E*, la substitution* σ *est alors un* E*-unificateur de* $t|_p$ *et* l'*.*

Notation : Nous écrivons $t \overset{*}{\rightsquigarrow}_\sigma t'$ s'il existe une dérivation $t = t_1 \rightsquigarrow_{\sigma_1} t_2 \ldots \rightsquigarrow_{\sigma_{n-1}} t_n = t'$ telle que $\sigma = \sigma_1 \circ \ldots \circ \sigma_{n-1}$.

Le lemme 7.7, énoncé ci-dessous, a pour but de « relever » une séquence de réécriture modulo E (c'est-à-dire $\xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}}$) en une séquence de surréduction modulo E. Des lemmes similaires existent dans la littérature [Kir85]. Malheureusement, ce dernier est énoncé pour une unique étape de réécriture et la preuve proposée ne semble pas pouvoir s'étendre à une dérivation de longueur arbitraire. Nous proposons donc une preuve de ce résultat. Pour cela, nous nous restreignons aux cas des théories équationnelles dites *régulières*. Ces théories se définissent formellement de la façon suivante :

Définition 7.5 (théorie équationnelle régulière)
Soit E *une théorie équationnelle,* E *est dite* régulière *si pour toute équation* $t_1 = t_2 \in \mathsf{E}$*, on a* $vars(t_1) = vars(t_2)$*.*

Exemple 7.6
Les théories $\mathsf{E} = \varnothing$ *et* $\mathsf{E} = \mathsf{AC}$ *sont régulières.*

Lemme 7.7 (lemme de relèvement)
Soit E *une théorie équationnelle régulière pour laquelle un algorithme d'unification existe. Soient* $t, s' \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ *et* θ *une substitution normalisée telle que* $t\theta \xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}} s'$*. Il existe un terme* t'*, une substitution* σ *et une substitution normalisée* θ' *tels que :*

1. $t \overset{*}{\rightsquigarrow}_\sigma t'$*,*
2. $t'\theta' =_{\mathsf{E}} s'$*,*

De plus, la séquence de surréduction $t \overset{*}{\rightsquigarrow}_\sigma t'$ *et la séquence de réécriture* $t\theta \xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}} s'$ *utilisent les mêmes règles de réécriture aux mêmes positions.*

Démonstration.
Nous montrons ce résultat par induction sur la longueur de la dérivation $t\theta \xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}} s'$.
Le cas de base est trivial, il correspond au cas où la longueur de la dérivation est nulle.
Posons $V = vars(t) \cup dom(\theta) \subseteq V$ et supposons que

$$\mathcal{D}: \qquad s = t\theta \rightarrow_{\mathsf{E}\backslash\mathcal{R}} s_1 \xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}} s'$$

est une séquence de réduction de longueur $n+1$ et que la première étape de réduction a lieu à la position p avec la règle de réécriture $l \rightarrow r \in \mathcal{R}$. Nous pouvons également supposer que $vars(l) \cap V = \varnothing$. Par définition d'une étape de réécriture avec $\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$, il existe une substitution τ telle que $(t\theta)|_p =_{\mathsf{E}} l\tau$, $dom(\tau) \subseteq vars(l)$ et $s_1 = s[r\tau]_p$. Puisque θ est une substitution normalisée, la position p est nécessairement une position non-variable de t : nous avons donc $(t\theta)|_p = (t|_p)\theta$. Posons $\mu = \tau \cup \theta$. Nous avons $(t|_p)\mu = (t|_p)\theta =_{\mathsf{E}} l\tau = l\mu$, les termes $t|_p$ et l sont donc E-unifiables. Soit σ_1 un E-mgu de $t|p$ et l plus général que μ. Par définition de σ_1, il existe une substitution ρ telle que $\rho \circ \sigma_1 =_{\mathsf{E}} \mu$ pour tout $x \in V \cup vars(l)$. Nous posons $V_1 = (V \setminus dom(\sigma_1)) \cup \{vars(x\sigma_1) | x \in V\}$ et nous définissons θ_1 de la façon suivante :

$$\theta_1(x) \quad = \quad \begin{cases} \rho(x) & \text{pour tout } x \in V_1 \\ x & \text{sinon} \end{cases}$$

Puisque $\rho \circ \sigma_1 =_{\mathsf{E}} \mu$ pour tout $x \in V \cup vars(l)$, nous en déduisons que $\rho \circ \sigma_1 =_{\mathsf{E}} \theta$ pour tout $x \in V$ et finalement, nous obtenons :

$$\theta_1 \circ \sigma_1 =_{\mathsf{E}} \theta \quad \text{pour tout } x \in V \tag{7.1}$$

Posons $t_1 = (t[r]_p)\sigma_1$. Par définition, nous avons :

$$t \rightsquigarrow_{\sigma_1} t_1 \tag{7.2}$$

$$\begin{array}{llll} \text{Nous avons} & vars(t_1) & = vars((t[r]_p)\sigma_1) & \\ & & \subseteq vars((t[l]_p)\sigma_1) & \text{car } vars(r) \subseteq vars(l), \\ & & = vars(t\sigma_1) & \text{car } t\sigma_1 =_{\mathsf{E}} t[l]_p\sigma_1 \text{ et E est régulière,} \\ & & \subseteq V_1 & \text{car } vars(t) \subseteq V \text{ et par définition de } V_1. \end{array}$$

Maintenant, nous allons montrer que $t_1\theta_1 =_{\mathsf{E}} s_1$.

$$\begin{array}{llll} \text{Nous avons} & t_1\theta_1 & = t_1\rho & \text{car } x\theta_1 = x\rho \text{ pour tout } x \in V_1 \text{ et } vars(t_1) \subseteq V_1, \\ & & = (t[r]_p)\sigma_1\rho & \text{par définition de } t_1, \\ & & =_{\mathsf{E}} (t[r]_p)\mu & \text{car } (x\sigma_1)\rho =_{\mathsf{E}} x\mu \text{ pour tout } x \in V \cup vars(l) \\ & & & \text{et } vars(t[r]_p) \subseteq V \cup vars(l), \\ & & = (t\mu)[r\mu]_p & \\ & & = (t\theta)[r\tau]_p & \text{puisque nous avons } x\mu = x\theta \text{ pour tout } x \in V \\ & & & \text{et } x\mu = x\tau \text{ pour tout } x \in vars(r), \\ & & = s[r\tau]_p & \text{car } s = t\theta, \\ & & = s_1. & \end{array}$$

Ainsi, nous en déduisons que

$$t_1\theta_1 =_{\mathsf{E}} s_1. \tag{7.3}$$

Avant de pouvoir appliquer l'hypothèse d'induction, nous devons vérifier que la substitution θ_1 est une substitution normalisée. Puisque $dom(\theta_1) \subseteq V_1$, il est suffisant de montrer que $x\theta_1$ est irréductible pour tout $x \in V_1$. Nous distinguons deux cas :

- Si $x \in V \setminus dom(\sigma_1)$, nous avons $x\sigma_1 = x$. De plus, par (7.1), nous avons $x\theta =_{\mathsf{E}} (x\sigma_1)\theta_1$. Nous en déduisons que $x\theta =_{\mathsf{E}} x\theta_1$. Puisque $x\theta$ est irréductible, nous savons que $x\theta_1$ est également irréductible puisque $\mathcal{R}$ est un système E-convergent.
- Si $x \in \{vars(y\sigma_1) | y \in V\}$, alors il existe $y \in V$ tel que $x \in vars(y\sigma_1)$. Nous en déduisons que $x\theta_1$ est un sous-terme de $(y\sigma_1)\theta_1$. Nous avons $(y\sigma_1)\theta_1 =_{\mathsf{E}} y\theta$ grâce à (7.1) et ce dernier est irréductible par hypothèse. Ainsi, nous en déduisons que $(y\sigma_1)\theta_1$ est irréductible ainsi que tous ses sous-termes. En particulier $x\theta_1$ est irréductible.

Maintenant, nous pouvons appliquer l'hypothèse d'induction. Nous avons un terme t_1, une substitution normalisée θ_1. Nous considérons $\mathcal{D}'$ « la fin de la dérivation $\mathcal{D}$ » : $\mathcal{D}'$ est une dérivation de $s_1 =_{\mathsf{E}} t_1\theta_1$ (grâce à (7.3)) à sa forme normale s'. Par hypothèse d'induction, nous en déduisons qu'il existe un terme t', une substitution σ_1' et une substitution normalisée θ' tels que :

(a) $t_1 \overset{*}{\rightsquigarrow}_{\sigma_1'} t'$
(b) $t'\theta' =_{\mathsf{E}} s'$

De plus, nous pouvons supposer que les séquences $t_1 \overset{*}{\rightsquigarrow}_{\sigma'} t'$ et $s_1 \xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}} s'$ utilisent les même règles de réécriture aux mêmes positions.

Posons $\sigma = \sigma_1' \circ \sigma_1$. En concaténant (7.2) et (a), nous obtenons $t \overset{*}{\rightsquigarrow}_{\sigma} t'$. Par construction, cette séquence de surréduction utilise les même règles de réécriture aux mêmes positions que la séquence $t\theta \xrightarrow{*}_{\mathsf{E}\backslash\mathcal{R}} s'$. Finalement, nous avons $t'\theta' =_{\mathsf{E}} s'$, et θ' est une substitution normalisée par hypothèse d'induction. □

7.2.2 Première caractérisation

Nous supposons donné un ordre, noté $\geq$, bien fondé et total sur les termes clos. Étant donné une théorie équationnelle E et un terme clos t, nous notons $t\downarrow_{\mathsf{E}}$ le terme minimal (par rapport à $\geq$) parmi tous les termes dans la classe d'équivalence de t.

Définition 7.8 (E-variants)
Soient E *une théorie équationnelle, le terme t' est un* E-variant *du terme t s'il existe une substitution θ telle que $t\theta =_{\mathsf{E}} t'$.*

Définition 7.9 (ensemble complet de variants modulo E′)
Soient E *et* E′ *deux théories équationnelles. Soit S un ensemble d'*E-variants *de t. L'ensemble S est un* complet *d'*E-variants *de t modulo* E′ *si, pour toute substitution σ, il existe un terme $t' \in S$ et une substitution θ tels que $t\sigma\downarrow_{\mathsf{E}} =_{\mathsf{E}'} t'\theta$.*

Lorsque les théories E et E′ sont claires d'après le contexte, nous parlerons simplement de variants et d'ensemble complet de variants. En général, et c'est le cas de tous les exemples proposés dans ce chapitre, la théorie E′ est soit la théorie vide, soit la théorie AC.

Exemple 7.10
Considérons les théories E = ACUN *et* E′ = AC. *Soit $t = x + f(x+y)$. Un ensemble complet de variants de t est composé d'un unique terme z. En effet, nous avons*

$$(x + f(x+y))\{x \mapsto f(z) + z; y \mapsto f(z)\} =_{\mathsf{AC}} f(z) + z + f(f(z) + z + f(z)) =_{\mathsf{ACUN}} z$$

et donc z est un variant de t. De plus, cet ensemble est complet puisque, pour toute substitution σ, on a $(x + f(x+y))\sigma\!\downarrow\, =_{\mathsf{AC}} z\theta$ pour une certaine substitution θ.

Définition 7.11 (propriété de variants finis)
Soient E *et* E′ *deux théories équationnelles. Nous disons que la paire* (E, E′) *a la* propriété de variants finis *(par rapport à $\geq$) si pour tout terme t, nous pouvons calculer un ensemble fini et complet d'*E*-variants de t modulo* E′.

Notation : Soit t un terme, on dénote par $\mathcal{V}ar(t)$ un ensemble complet de variants de t. Cette notation s'étend naturellement à différentes structures composées de termes (*e.g.* système de contrainte, règle d'inférence, ...).

Exemple 7.12
Considérons la théorie équationnelle ACUN *et la règle d'inférence* (I) *suivante :*

$$\frac{x_1 \qquad x_2}{x_1 \oplus x_2}$$

Les règles d'inférence ci-dessous sont des variants de la règle d'inférence (I). *Ils sont obtenus en considérant les substitutions $\theta_1 = \{x_1 \mapsto y_1; x_2 \mapsto y_2\}$, $\theta_2 = \{x_1 \mapsto y_1 \oplus y_2; x_2 \mapsto y_2\}$ et $\theta_3 = \{x_1 \mapsto y_1 \oplus y_3; x_2 \mapsto y_2 \oplus y_3\}$.*

$$\frac{y_1 \qquad y_2}{y_1 \oplus y_2} \qquad\qquad \frac{y_1 \oplus y_2 \qquad y_2}{y_1} \qquad\qquad \frac{y_1 \oplus y_3 \qquad y_2 \oplus y_3}{y_1 \oplus y_2}$$

Nous avons besoin d'une procédure uniforme pour calculer les E-variants d'un terme donné. C'est pour cette raison que nous considérons une théorie équationnelle E représentable par un système de réécriture E′-convergent et que nous choisissons l'ordre $\geq$ de telle sorte que $\rightarrow_{\mathsf{E}'\backslash\mathcal{R}} \subseteq \geq$.

En résumé, notre but est, étant donné une théorie équationnelle E, de trouver une décomposition de E en $(\mathcal{R}, \mathsf{E}')$ et un ordre $\geq$ tels que :

1. $\mathcal{R}$ soit un système de réécriture E′-convergent représentant E (pour la relation $\rightarrow_{\mathsf{E}'\backslash\mathcal{R}}$) et que $\rightarrow_{\mathsf{E}'\backslash\mathcal{R}} \subseteq \geq$ soit une relation décidable,
2. pour tout terme t, il existe un ensemble fini $t_1, \ldots, t_n$ et effectivement calculable de variants de t tel que pour toute substitution σ, il existe un indice i et une substitution θ telle que $t\sigma\!\downarrow_{\mathsf{E}'\backslash\mathcal{R}} =_{\mathsf{E}'} t_i\theta$.

Nous dirons que $(\mathcal{R}, \mathsf{E}')$ est une *décomposition* de E satisfaisant la *propriété de variants finis* si les deux conditions précédentes sont satisfaites. Il existe plusieurs techniques bien connues pour obtenir la première condition. Nous nous concentrerons donc sur la deuxième.

Le lemme ci-dessous montre que, si $(\mathcal{R}, \mathsf{E}')$ a la propriété de variants finis, on peut non seulement calculer en avance un ensemble fini d'instances $t_1, \ldots, t_n$ de t tel que $t\sigma\!\downarrow$ soit toujours une instance d'un des t_i, mais on peut même calculer en avance les substitutions $\theta_1, \ldots, \theta_n$ telles que $t_1 = t\theta_1\!\downarrow, \ldots, t_n = t\theta_n\!\downarrow$ soit un ensemble complet de variants et que toutes les substitutions normalisées puissent se factoriser à travers une substitution θ_i.

Lemme 7.13
Une décomposition $(\mathcal{R}, \mathsf{E}')$ a la propriété de variants finis si, et seulement si, pour tout terme t, il existe un ensemble fini $\Sigma(t)$ de substitutions tel que :

$$\forall \sigma,\ \exists \theta \in \Sigma(t),\ \exists \tau \ \textit{tels que}\ \ \sigma{\downarrow} =_{\mathsf{E}'} \theta\tau \ \ \textit{et}\ \ (t\sigma){\downarrow} =_{\mathsf{E}'} (t\theta){\downarrow}\tau$$

Démonstration.
($\Rightarrow$) Soit t un terme et $x_1, \ldots, x_n$ les variables de t. Considérons le terme $T = \langle t, \langle x_0, \langle \ldots, x_n\rangle\rangle\rangle$ où $\langle\_,\_\rangle$ est un symbole binaire libre (sans propriété algébrique). Puisque $(\mathcal{R}, \mathsf{E}')$ satisfait la propriété de variants finis, il existe un ensemble fini $\Sigma_A(T)$ de substitutions tel que, pour toute substitution normalisée σ, il existe une substitution $\theta \in \Sigma_A(T)$ et une substitution τ telles que $(T\sigma){\downarrow} =_{\mathsf{E}'} (T\theta){\downarrow}\tau$. Posons $\Sigma(t) = \{\theta{\downarrow} \mid \theta \in \Sigma_A(T)\}$. Nous allons montrer que $\Sigma(t)$ satisfait la condition demandée.

Soit σ une substitution. Par hypothèse, il existe $\theta \in \Sigma_A(T)$ et une substitution τ telles que $(T\sigma){\downarrow} =_{\mathsf{E}'} (T\theta){\downarrow}\tau$. Posons $\theta' = \theta{\downarrow}$. Par définition $\theta' \in \Sigma(t)$, il nous reste donc à montrer que $\sigma{\downarrow} =_{\mathsf{E}'} \theta'\tau$ et que $(t\sigma){\downarrow} =_{\mathsf{E}'} (t\theta'){\downarrow}\tau$. Nous avons les égalités suivantes :

$$\begin{array}{rcl}
(T\sigma){\downarrow} & =_{\mathsf{E}'} & (T\theta){\downarrow}\tau \\
(T\sigma){\downarrow} & =_{\mathsf{E}'} & (T\theta'){\downarrow}\tau \\
(\langle t, \langle x_0, \langle \ldots, x_n\rangle\rangle\rangle\sigma){\downarrow} & =_{\mathsf{E}'} & (\langle t, \langle x_0, \langle \ldots, x_n\rangle\rangle\rangle\theta'){\downarrow}\tau \\
\langle (t\sigma){\downarrow}, \langle (x_0\sigma){\downarrow}, \langle \ldots, (x_n\sigma){\downarrow}\rangle\rangle\rangle & =_{\mathsf{E}'} & \langle (t\theta'){\downarrow}\tau, \langle (x_0\theta'){\downarrow}\tau, \langle \ldots, (x_n\theta'){\downarrow}\tau\rangle\rangle\rangle
\end{array}$$

Nous avons $(x_i\theta'){\downarrow} = x_i\theta'$ puisque θ' est une substitution normalisée. Nous en déduisons donc que $x_i\theta'\tau = (x_i\theta'){\downarrow}\tau =_{\mathsf{E}'} (x_i\sigma){\downarrow}$. Nous obtenons $\sigma{\downarrow} =_{\mathsf{E}'} \theta'\tau$ et $(t\sigma){\downarrow} =_{\mathsf{E}'} (t\theta'){\downarrow}\tau$.

($\Leftarrow$) C'est une conséquence directe de la définition de la propriété de variants finis. $\square$

7.2.3 Principale application

Dans la suite, nous considérons une théorie équationnelle E pour laquelle il existe une décomposition $(\mathcal{R}, \mathsf{E}')$ satisfaisant la propriété de variants finis. Nous verrons, dans la partie 7.4, que de nombreuses théories, pertinentes du point de vue de la vérification des protocoles cryptographiques, satisfont cette propriété. Dans cette partie, nous montrons comment la propriété de variants finis nous permet de réduire le problème de la sécurité d'un protocole dans la théorie de l'intrus $(\mathcal{I}, \mathsf{E})$ à la théorie $(\mathcal{I}', \mathsf{E}')$. Nous énonçons ce résultat au niveau de la satisfaisabilité des systèmes de contraintes de déduction, mais ce résultat est également valable pour les systèmes de contraintes avec équations. Il s'énonce formellement de la façon suivante :

Théorème 7.14
Soit $\mathcal{C}$ un système de contraintes et $\mathcal{I}$ un système d'inférence. Le système de contraintes $\mathcal{C}$ est satisfaisable dans $(\mathcal{I}, \mathsf{E})$ si, et seulement si, il existe $\mathcal{C}' \in \mathcal{V}ar(\mathcal{C})$ tel que $\mathcal{C}'$ soit satisfaisable dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$.

Démonstration.
($\Rightarrow$) Soit $\mathcal{C}$ un système de contraintes et σ une solution de $\mathcal{C}$ dans $(\mathcal{I}, \mathsf{E})$. Par définition de $\mathcal{V}ar(\mathcal{C})$, nous savons qu'il existe $\mathcal{C}' \in \mathcal{V}ar(\mathcal{C})$ et une substitution θ tels que $\mathcal{C}\sigma\!\downarrow =_{\mathsf{E}'} \mathcal{C}'\theta$. La proposition 7.15 (énoncée et prouvée ci-dessous) nous permet d'établir que θ est une solution de $\mathcal{C}'$ dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$.
($\Leftarrow$) Soit $\mathcal{C}$ un système de contraintes. Supposons qu'il existe $\mathcal{C}' \in \mathcal{V}ar(\mathcal{C})$ admettant une solution σ dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$. Par définition de $\mathcal{V}ar(\mathcal{C})$, nous savons qu'il existe une substitution θ telle que $\mathcal{C}' =_{\mathsf{E}} \mathcal{C}\theta$. Nous savons donc que $\mathcal{C}'\sigma =_{\mathsf{E}} \mathcal{C}\theta\sigma$. La proposition 7.15 (énoncée et prouvée ci-dessous) nous permet d'établir que $\theta\sigma$ est une solution de $\mathcal{C}$ dans $(\mathcal{I}, \mathsf{E})$. □

Proposition 7.15
Soient $T \subseteq \mathcal{T}(\mathcal{F})$ et $u \in \mathcal{T}(\mathcal{F})$. Nous avons :

$$\textit{Si } T \vdash u \textit{ dans } (\mathcal{V}ar(\mathcal{I}),\mathsf{E}') \textit{ alors } T \vdash u \textit{ dans } (\mathcal{I}, \mathsf{E}).$$

Si T et u sont en formes normales, alors la réciproque est également vraie.

Démonstration.
($\Rightarrow$) Soit P un arbre de preuve de $T \vdash u$ dans $(\mathcal{V}ar(\mathcal{I}),\mathsf{E}')$. Nous allons montrer par induction sur la structure de P, qu'il existe une preuve P' de $T \vdash u$ dans $(\mathcal{I}, \mathsf{E})$.

Cas de base : Si P se réduit à une feuille, alors P est l'arbre de preuve cherché.

Étape d'induction : On distingue deux cas, suivant la dernière règle d'inférence utilisée dans P :

- Si P se termine par une instance de la règle d'inférence (Eq), alors P est de la forme :

$$\frac{P_1 \left\{ \dfrac{\cdots}{T \vdash u_1} \right.}{T \vdash u_2} \ (\mathsf{Eq})$$

avec $u_1 =_{\mathsf{E}'} u_2$. Par hypothèse d'induction, il existe une preuve P_1' de $T \vdash u_1$ dans $(\mathcal{I}, \mathsf{E})$. Puisque $\mathsf{E}' \subseteq \mathsf{E}$, on a $u_1 =_{\mathsf{E}} u_2$ et on en déduit que $T \vdash u_2$ dans $(\mathcal{I}, \mathsf{E})$.
- Si P se termine par une instance d'une règle d'inférence $\mathsf{I} \in \mathcal{V}ar(\mathcal{I})$, alors P est de la forme :

$$\frac{P_1 \left\{ \dfrac{\cdots}{T \vdash u_1} \right. \quad \cdots \quad P_n \left\{ \dfrac{\cdots}{T \vdash u_n} \right.}{T \vdash u} \ (\mathsf{I})$$

Par définition de $\mathcal{V}ar(\mathcal{I})$, il existe $\mathsf{I}' \in \mathcal{I}$ et une substitution θ tels que $\mathsf{I}'\theta =_{\mathsf{E}} \mathsf{I}$. Par hypothèse d'induction, il existe des preuves $P_1', \ldots, P_n'$ de $T \vdash u_1, \ldots, T \vdash u_n$ dans $(\mathcal{I}, \mathsf{E})$. Soit P', la preuve obtenue à partir de $P_1', \ldots, P_n'$ par application de la règle (I'), éventuellement précédée et suivie d'instances de la règle (Eq). L'arbre de preuve P' ainsi obtenu est une preuve de $T \vdash u$ dans $(\mathcal{I}, \mathsf{E})$.

($\Leftarrow$) Soit T un ensemble de termes en formes normales et P une preuve de $T \vdash u$ dans $(\mathcal{I}, \mathsf{E})$. Nous allons montrer par induction sur la structure de P qu'il existe un arbre de preuve P' de $T \vdash u\!\downarrow$ dans $(\mathcal{V}ar(\mathcal{I}),\mathsf{E}')$.

Cas de base : Si P se réduit à une feuille, alors u est en forme normale et l'arbre P' se réduit également à une feuille.

Étape d'induction : On distingue deux cas, suivant la dernière règle d'inférence utilisée dans P :

– Si P se termine par une instance de la règle d'inférence (Eq), alors P est de la forme :

$$\frac{P_1\left\{\ \frac{\cdots}{T \vdash u_1}\right.}{T \vdash u_2}\ (\mathsf{Eq})$$

avec $u_1 =_{\mathsf{E}} u_2$. Par hypothèse d'induction, il existe une preuve P'_1 de $T \vdash u_1\!\downarrow$ dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$. De plus, puisque $u_1 =_{\mathsf{E}} u_2$, nous avons $u_1\!\downarrow =_{\mathsf{E}'} u_2\!\downarrow$. L'arbre

$$\frac{P'_1\left\{\ \frac{\cdots}{T \vdash u_1\downarrow}\right.}{T \vdash u_2\downarrow}\ (\mathsf{Eq})$$

est un arbre de preuve de $T \vdash u_2\!\downarrow$ dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$.

– Si P se termine par une instance d'une règle d'inférence $\mathsf{I} \in \mathcal{I}$, alors P est de la forme :

$$\frac{P_1\left\{\ \frac{\cdots}{T \vdash u_1}\right. \quad \cdots \quad P_n\left\{\ \frac{\cdots}{T \vdash u_n}\right.}{T \vdash u}\ (\mathsf{I})$$

Par hypothèse d'induction, il existe des preuves $P'_1, \ldots, P'_n$ de $T \vdash u_1\!\downarrow, \ldots, T \vdash u_n\!\downarrow$ dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$. D'autre part, nous savons qu'il existe une substitution θ telle que

$$\mathsf{I}\theta = \frac{T \vdash u_1 \ \ \ldots \ \ T \vdash u_n}{T \vdash u}$$

Soit $\mathsf{I}' \in \mathcal{V}ar(\mathcal{I})$ et σ une substitution tels que $\mathsf{I}'\sigma =_{\mathsf{E}'} \mathsf{I}\theta\!\downarrow$. On construit aisément un arbre de preuve de $T \vdash u\!\downarrow$ dans $(\mathcal{V}ar(\mathcal{I}), \mathsf{E}')$, en appliquant l'instance $\mathsf{I}'\sigma$ de I' aux sous-preuves $P'_1, \ldots, P'_n$. $\square$

L'intérêt principale de la propriété de variants finis est donc de se débarrasser de certains axiomes en les orientant, et de réduire ainsi la théorie équationnelle considérée dans le problème de départ. Le théorème 7.14 est énoncé au niveau de la satisfaisabilité du système de contraintes, mais on peut énoncer un résultat similaire en calculant directement les variants des différents rôles composant le protocole.

Exemple 7.16

Considérons la théorie équationnelle E *composée de l'unique équation* $\mathsf{dec}(\{x\}_y, y) = x$*, et le système d'inférence suivant :*

$$\frac{T \vdash u_1 \quad T \vdash u_2}{T \vdash \{u_1\}_{u_2}} \qquad \frac{T \vdash u_1 \quad T \vdash u_2}{T \vdash \mathsf{dec}(u_1, u_2)} \qquad \frac{T \vdash u_1}{T \vdash u_2}\ u_1 =_{\mathsf{E}} u_2$$

Considérons le protocole $\mathcal{P}$ *composé d'un unique rôle, lui-même composé d'une unique instruction :*

$$\mathit{recv}(\{x\}_{k_2})\ ;\mathit{send}(\{\mathsf{dec}(x, k1)\}_{k_2})$$

La théorie équationnelle E *considérée a la propriété de variants finis, ce qui nous permet de nous débarrasser de l'unique équation la composant, en l'orientant de gauche à droite. Le système d'inférence* $\mathcal{V}ar(\mathcal{I})$ *est composé des trois règles d'inférence suivantes :*

$$\frac{T \vdash u_1 \quad T \vdash u_2}{T \vdash \{u_1\}_{u_2}} \qquad \frac{T \vdash u_1 \quad T \vdash u_2}{T \vdash \mathsf{dec}(u_1, u_2)} \qquad \frac{T \vdash \{u_1\}_{u_2} \quad T \vdash u_2}{T \vdash u_1}$$

Nous pouvons également calculer $\mathcal{V}ar(\mathcal{P})$*, un ensemble fini et complet de variants pour le protocole* $\mathcal{P}$*, et nous obtenons les deux protocoles suivants :*

$$\begin{array}{lcl} \mathcal{P}_1 & := & \mathit{recv}(\{x\}_{k_2})\ ;\mathit{send}(\{\mathsf{dec}(x, k1)\}_{k_2}) \\ \mathcal{P}_2 & := & \mathit{recv}(\{\{y\}_{k_1}\}_{k_2})\ ;\mathit{send}(\{y\}_{k_2}) \end{array}$$

Pour savoir si l'intrus est en mesure de récupérer le secret s *à partir de sa connaissance* $T_0 = \{\{s\}_{k_1},\ k_2\}$ *en utilisant ses capacités de déduction (c'est-à-dire le système d'inférence* $(\mathcal{I}, \mathsf{E})$*), il nous suffit de regarder si cette attaque sur* s *est réalisable sur un des variants du protocole (c'est-à-dire sur* $\mathcal{P}_1$ *ou* $\mathcal{P}_2$*) dans la théorie de l'intrus* $(\mathcal{V}ar(\mathcal{I}), \emptyset)$*. N'ayant plus de théorie équationnelle à considérer, il est relativement aisé de visualiser cette attaque. Considérons le protocole* $\mathcal{P}_2$*. L'intrus peut chiffrer* $\{s\}_{k_1}$ *avec la clef* k_2 *et envoyer ce message sur le réseau. Cela lui permet d'obtenir* $\{s\}_{k_2}$ *et d'en déduire* s *puisqu'il connaît* k_2*. Cette « même » attaque existe sur le protocole* $\mathcal{P}$ *de départ, mais elle fait intervenir une étape de raisonnement équationnelle.*

7.3 Comment établir la propriété de variants finis ?

Nous avons défini une nouvelle propriété et nous venons de voir son intérêt du point de vue de la vérification des protocoles cryptographiques. Il nous faut maintenant trouver un moyen d'établir qu'une théorie équationnelle a la propriété de variants finis. Pour cela, nous proposons une deuxième caractérisation, appelée *propriété de boundedness*[1] (*cf.* partie 7.3.1) et nous donnons des conditions suffisantes permettant d'assurer que cette propriété est satisfaite par une théorie donnée (*cf.* parties 7.3.2 et 7.3.3).

7.3.1 Deuxième caractérisation

Dans la suite, nous supposons que la théorie équationnelle E considérée est donnée sous la forme d'un système de réécriture $\mathcal{R}$. De plus, nous supposons que $\mathcal{R}$ est un système E′-convergent.

Définition 7.17 (propriété de boundedness)
La décomposition $(\mathcal{R}, \mathsf{E}')$ *satisfait* la propriété de boundedness *si pour tout terme* t*, il existe un entier* n *tel que pour toute substitution normalisée* σ*, la forme normale de* $t\sigma$ *est accessible par une dérivation de longueur inférieure ou égale à* n *(la borne* n *est indépendante de* σ*) :*

$$\forall t, \exists n, \forall \sigma.\ t(\sigma\!\downarrow) \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\sigma)\!\downarrow$$

[1] Ce terme exprime le caractère borné des dérivations à considérer. N'ayant pas d'équivalent en français, il ne sera pas traduit.

La proposition suivante établit le lien entre la propriété de boundedness et la propriété de variants finis.

Proposition 7.18
Soit E' *une théorie équationnelle régulière pour laquelle un algorithme d'unification existe. La décomposition* $(\mathcal{R}, \mathsf{E}')$ *satisfait la propriété de boundedness si, et seulement si, la décomposition* $(\mathcal{R}, \mathsf{E}')$ *satisfaisait la propriété de variants finis.*

Démonstration.
($\Rightarrow$) Nous utilisons la surréduction pour borner la longueur d'une dérivation. Soit t un terme. Par hypothèse, nous avons :

$$\exists n, \forall \sigma.\ t(\sigma\!\downarrow) \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\sigma)\!\downarrow.$$

Notons S l'ensemble des termes accessibles à partir de t par une séquence de surréduction de longueur inférieure ou égale à n. L'ensemble S est un ensemble fini de variants de t. Il est effectivement calculable. Il nous reste donc à montrer que S est un ensemble complet. Soit σ une substitution. Nous devons montrer qu'il existe un terme $t' \in S$ et une substitution θ tels que $(t\sigma)\!\downarrow =_{\mathsf{E}'} t'\theta$. Par hypothèse, nous savons que $t(\sigma\!\downarrow) \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\sigma)\!\downarrow$. Grâce au lemme 7.7, nous savons qu'il existe un terme t' et une séquence de surréduction $t \overset{*}{\rightsquigarrow} t'$ de longueur au plus n et une substitution normalisée θ tels que $t(\sigma\!\downarrow)\!\downarrow =_{\mathsf{E}'} t'\theta$. Nous en déduisons donc que $t' \in S$ et que $(t\sigma)\!\downarrow =_{\mathsf{E}'} t'\theta$.
($\Leftarrow$) Pour montrer la réciproque, supposons que $(\mathcal{R}, \mathsf{E}')$ satisfasse la propriété de variants finis. Grâce au lemme 7.13, nous savons que pour tout terme t, il existe un ensemble fini de substitutions $\Sigma(t)$ tel que :

$$\forall \sigma,\ \exists \theta \in \Sigma(t),\ \exists \tau.\ \sigma\!\downarrow =_{\mathsf{E}'} \theta\tau \quad \wedge \quad (t\sigma)\!\downarrow =_{\mathsf{E}'} (t\theta)\!\downarrow\!\tau$$

Soit t un terme. Soit n l'entier tel que $t\theta \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\theta)\!\downarrow$ pour toute substitution $\theta \in \Sigma(t)$. Remarquons qu'un tel entier existe puisque $\rightarrow_{\mathsf{E}'\backslash\mathcal{R}}$ termine et que l'ensemble $\Sigma(t)$ est fini. Il nous reste à montrer que pour toute substitution normalisée σ, on a $t\sigma \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\sigma)\!\downarrow$. Soit σ une substitution normalisée. Soit $\theta \in \Sigma(t)$ et τ une substitution telles que $\sigma\!\downarrow =_{\mathsf{E}'} \theta\tau$ et $(t\sigma)\!\downarrow =_{\mathsf{E}'} (t\theta)\!\downarrow\!\tau$. Par définition de n, nous avons $t(\theta\tau) \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\theta)\!\downarrow\!\tau$. Le lemme 7.19, énoncé et prouvé ci-après, nous permet de conclure puisque nous avons $\sigma =_{\mathsf{E}'} \theta\tau$. Ainsi, nous obtenons $t\sigma \xrightarrow{\leq n}_{\mathsf{E}'\backslash\mathcal{R}} (t\sigma)\!\downarrow$. $\square$

Remarque : La preuve de ce théorème fournit un moyen effectif de calculer un ensemble complet de variants associé à un terme t donné. Il suffit en effet de regarder toutes les séquences de surréduction issues de t et de longueur au plus n, où n est l'entier satisfaisant la condition de boundedness.

Lemme 7.19
Soit t *un terme. Considérons deux substitutions normalisées* σ_1 *et* σ_2 *telles que* $\sigma_1 =_{\mathsf{E}'} \sigma_2$. *Soit* $\mathcal{D} : t\sigma_1 \xrightarrow{*}_{\mathsf{E}'\backslash\mathcal{R}} s_1$ *une séquence de réécriture (par rapport à* $\rightarrow_{\mathsf{E}'\backslash\mathcal{R}}$*). Il existe* s_2 *et une séquence de réécriture* $\mathcal{D}' : t\sigma_2 \xrightarrow{*}_{\mathsf{E}'\backslash\mathcal{R}} s_2$ *tels que* $s_1 =_{\mathsf{E}'} s_2$ *et les deux séquences* $\mathcal{D}$ *et* $\mathcal{D}'$ *utilisent les mêmes règles de réécriture aux mêmes positions.*

Démonstration.
Nous montrons ce résultat par induction sur la longueur de la dérivation $\mathcal{D}$ allant de $t\sigma_1$ à s_1. Le cas de base ($\mathcal{D}$ est une dérivation de longueur 0) est évident. Maintenant, supposons que :

$$\mathcal{D}: \quad t\sigma_1 \rightarrow_{\mathsf{E}'\backslash\mathcal{R}} s'_1 \xrightarrow{*}_{\mathsf{E}'\backslash\mathcal{R}} s_1$$

est une séquence réécriture de longueur $n+1$ et que la première étape de réécriture a lieu à la position p avec la règle de réécriture $l \rightarrow r \in \mathcal{R}$. Nous avons, par définition d'une étape de réécriture avec $\rightarrow_{\mathsf{E}'\backslash\mathcal{R}}$, $(t\sigma_1)|_p =_{\mathsf{E}'} l\tau$ pour une substitution τ avec $dom(\tau) \subseteq vars(l)$ et $s'_1 = (t\sigma_1)[r\tau]_p$. Puisque σ_1 est une substitution normalisée et que p est une position non-variable de t, nous avons $(t\sigma_1)|_p = (t|_p)\sigma_1$. Ainsi, on a $s'_1 = (t[r\tau]_p)\sigma_1$. Puisque $\sigma_1 =_{\mathsf{E}'} \sigma_2$, nous avons $(t|_p)\sigma_1 =_{\mathsf{E}'} (t|_p)\sigma_2$ et nous en déduisons que $(t|_p)\sigma_2 =_{\mathsf{E}'} l\tau$.

Nous avons $t\sigma_2 \rightarrow_{\mathsf{E}'\backslash\mathcal{R}} (t[r\tau]_p)\sigma_2$, et nous pouvons appliquer l'hypothèse d'induction. Nous avons le terme $t[r\tau]_p$ et deux substitutions normalisées σ_1, σ_2 telles que $\sigma_1 =_{\mathsf{E}'} \sigma_2$. Nous avons une dérivation $\mathcal{D}_1$ de $(t[r\tau]_p)\sigma_1$ à s_1. Par hypothèse d'induction, il existe un terme s_2 tel que $s_1 =_{\mathsf{E}'} s_2$ et une dérivation $\mathcal{D}'_1 : (t[r\tau]_p)\sigma_2 \xrightarrow{*}_{\mathsf{E}'\backslash\mathcal{R}} s_2$. De plus, les deux dérivations $\mathcal{D}_1$ et $\mathcal{D}'_1$ utilisent les mêmes règles de réécriture aux mêmes positions. Nous obtenons ainsi le résultat attendu. □

7.3.2 En présence d'axiomes orientables

Dans le cas où la théorie équationnelle E' est vide, autrement dit lorsque tous les axiomes sont orientables, nous allons montrer que nous pouvons nous restreindre a des séquences de surréduction dites *basiques*, et obtenir ainsi un critère permettant d'établir la propriété de variants finis. Nous verrons que ce critère nous permet de conclure pour un certain nombre de théories équationnelles intéressantes (*cf.* partie 7.4.1).

Définition 7.20 (position basique)
Soit $t_1 \rightsquigarrow_{\sigma_1} t_2 \rightsquigarrow_{\sigma_2} \dots \rightsquigarrow_{\sigma_{n-1}} t_n$ une séquence de surréduction, et supposons que la $i^{\text{ème}}$ étape soit effectuée à la position p_i avec la règle $l_i \rightarrow r_i$. Les ensembles de positions basiques *$B_1, \dots, B_n$ sont définis inductivement de la façon suivante :*

$$B_1 = \bar{\mathcal{O}}(t) \qquad \textit{et} \qquad B_{i+1} = \mathcal{B}(B_i, p_i, r_i) \quad \textit{pour } 1 \leq i < n$$

où $\mathcal{B}(B_i, p_i, r_i)$ désigne l'ensemble $(B_i \smallsetminus \{q \in B_i \mid p_i \leq q\}) \cup \{p_i.q \mid q \in \bar{\mathcal{O}}(r_i)\}$. Les positions dans B_i sont appelées positions basiques. *Une séquence de surréduction est dite* basique *si $p_i \in B_i$ pour tout i tel que $1 \leq i < n$.*

De la même façon, nous disons qu'une séquence de réécriture $t_1 \rightarrow_{\mathsf{E}\backslash\mathcal{R}} t_2 \rightarrow_{\mathsf{E}\backslash\mathcal{R}} \dots \rightarrow_{\mathsf{E}\backslash\mathcal{R}} t_n$ est basée sur un ensemble de position $B_1 \subseteq \bar{\mathcal{O}}(t_1)$ si $p_i \in B_i$ pour tout $1 \leq i < n$ avec les ensembles $B_2, \dots, B_{n-1}$ définis comme précédemment.

Lemme 7.21 ([Hul80a])
Soit t un terme et σ une substitution normalisée. Une séquence de réductions (par rapport à $\rightarrow_{\mathcal{R}}$) suivant une stratégie de l'intérieur vers l'extérieur issue de $t\sigma$ est basée sur $\bar{\mathcal{O}}(t)$.

Grâce au lemme 7.21, la séquence de surréduction associée par le lemme 7.7, à une dérivation suivant une stratégie de l'intérieur vers l'extérieur, est nécessairement basique. De plus, puisque $\mathcal{R}$ est un système convergent, nous pouvons toujours choisir une séquence de réécriture suivant une stratégie de l'intérieur vers l'extérieur. Ainsi, nous obtenons un critère suffisant permettant d'établir la propriété de variants finis d'une théorie équationnelle E représentable par un système de réécriture convergent. Ce critère est énoncé dans la proposition ci-dessous :

Proposition 7.22
Si la surréduction basique termine pour $\mathcal{R}$, alors $(\mathcal{R}, \varnothing)$ est une décomposition satisfaisant la propriété de boundedness, et donc la propriété de variants finis.

Démonstration.
Soit $\mathcal{R}$ un système de réécriture convergent représentant E. Soit t un terme. Par hypothèse, nous savons que la surréduction basique termine. Soit n la longueur de la dérivation la plus longue issue de t. Nous allons montrer que n est la borne cherchée, c'est-à-dire que pour toute substitution σ normalisée, on a :

$$t(\sigma\downarrow) \xrightarrow{\leq n}_{\mathcal{R}} (t\sigma)\downarrow$$

Soit σ une substitution normalisée. Soit $\mathcal{D}$ une dérivation de $t\sigma$ à $(t\sigma)\downarrow$ suivant une stratégie de l'intérieur vers l'extérieur. Par le lemme 7.21, nous savons que $\mathcal{D}$ est basée sur $\bar{\mathcal{O}}(t)$ et par le lemme de relèvement, nous on déduisons que la séquence de surréduction associée à $\mathcal{D}$ est basique. Nous en déduisons donc que la longueur de $\mathcal{D}$ est inférieure ou égale à n. □

7.3.3 En présence d'axiomes non-orientables

La situation est un peu différente en présence d'axiomes non-orientables. Les résultats que nous venons de voir ne s'étendent pas au cas de la réécriture modulo une théorie équationnelle. En particulier, le lemme 7.21 est faux en présence d'axiomes non-orientables.

Exemple 7.23
Soit $\mathcal{R}$ le système de réécriture composé des règles $x + 0 \to x$, $x + x \to 0$ et $x + (x + y) \to y$. Considérons le terme $t = x_1 + x_2$ et la substitution normalisée $\sigma = \{x_1 \mapsto a + b, x_2 \mapsto a + b\}$. Maintenant, intéressons-nous à la dérivation (par rapport à $\to_{\mathsf{AC}\backslash\mathcal{R}_+}$) suivante :

$$(a + b) + (a + b) \quad \xrightarrow[x+(x+y)\to y]{\Lambda} \quad b + b \quad \xrightarrow[x+x\to 0]{\Lambda} \quad 0$$

La première étape de réécriture a lieu à la position $\Lambda \in B_1 = \bar{\mathcal{O}}(t)$ avec la règle de réécriture $x + (x + y) \to y$. Ainsi, l'ensemble B_2 est vide. Nous en déduisons que la seconde étape de réécriture n'est pas basée sur $\bar{\mathcal{O}}(t)$ bien que la dérivation suive une stratégie de l'intérieur vers l'extérieur.

Cet exemple se généralise pour obtenir une dérivation issue de $t\sigma$ arbitrairement longue. Remarquons cependant qu'il existe une autre dérivation plus courte (ici de longueur 1).

De toutes les façons, pour les théories AG et DH, indépendemment de l'orientation choisie pour l'équation $y^{-1} \times x^{-1} = (x \times y)^{-1}$, la surréduction modulo AC (même basique) ne termine pas. Autrement dit, il est inutile de chercher à obtenir la propriété de boundedness en utilisant une approche similaire à celle utilisée lorsque les axiomes sont orientables. Nous devons donc trouver un autre critère permettant de conclure pour ces théories équationnelles. Nous proposons le critère suivant :

Proposition 7.24
Si $(\mathcal{R}, \mathsf{E}')$ *est une décomposition de* E *satisfaisant*

$$\forall f \in sig(\mathsf{E}),\ \exists c,\ \forall t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \text{ tels que } f(t_1\downarrow, \ldots, t_n\downarrow) \xrightarrow{\leq c}_{\mathsf{E}'\backslash\mathcal{R}} f(t_1, \ldots, t_n)\downarrow.$$

alors $(\mathcal{R}, \mathsf{E}')$ *satisfait la propriété de boundedness.*

Démonstration.
Soit t un terme et c_{max} l'entier satisfaisant la condition énoncée dans la proposition. Un tel c_{max} existe puisque $sig(\mathsf{E})$ est fini. Soit θ une substitution, nous allons montrer, par induction structurelle sur t que $t(\theta\downarrow)$ peut se réduire à sa forme normale par une dérivation de longueur au plus $c_{max} \times |t|$. Nous distinguons différents cas :

1. Si t est une variable, alors $t(\theta\downarrow)$ est déjà en forme normale.
2. Si $t = f(t_1, \ldots, t_n)$, alors nous avons $t_i(\theta\downarrow) \xrightarrow{\leq c_{max} \times |t_i|}_{\mathsf{E}'\backslash\mathcal{R}} (t_i\theta)\downarrow$ pour tout i tel que $1 \leq i \leq n$ (hypothèse d'induction).
   - Soit $f \in sig(\mathsf{E})$, et par hypothèse, $f((t_1\theta)\downarrow, \ldots, (t_n\theta)\downarrow) \xrightarrow{c_{max}}_{\mathsf{E}'\backslash\mathcal{R}} f(t_1\theta, \ldots, t_n\theta)\downarrow)$, ce qui nous permet de conclure.
   - Soit $f \notin sig(\mathsf{E})$, et dans ce cas, $f((t_1\theta)\downarrow, \ldots, (t_n\theta)\downarrow)$ est déjà en forme normale. □

7.4 Application à différentes théories équationnelles

Dans cette partie, nous allons passer en revue différentes théories équationnelles, et nous allons établir la propriété de variants finis pour chacune d'entre elles. Nous commençons par établir cette propriété pour des variantes de la théorie de Dolev-Yao (*cf.* partie 7.4.1) puis nous aborderons des théories plus complexes avec un opérateur associatif et commutatif (*e.g.* ACUN, AG, DH, ...).

7.4.1 Variantes de Dolev-Yao

La théorie équationnelle la plus classique, déjà introduite au chapitre 2, est E_{DY}.

$$\mathsf{E}_{\mathsf{DY}} := \left\{ \begin{array}{rcl} \mathsf{dec}(\{x\}_y, y) & = & x, \\ \mathsf{proj}_1(\langle x_1, x_2 \rangle) & = & x_1, \\ \mathsf{proj}_2(\langle x_1, x_2 \rangle) & = & x_2. \end{array} \right.$$

En orientant ces équations de gauche à droite, on obtient un système de réécriture convergent. Et nous pouvons conclure grâce au lemme suivant :

Lemme 7.25 ([Hul80a])
Soit $\mathcal{R}$ *un système de réécriture convergent. Si toute séquence de surréduction basique issue d'un membre droit d'une règle de réécriture termine, alors la surréduction basique termine.*

Une conséquence directe de ce lemme est que la surréduction basique termine pour tous les les systèmes convergents dont les membres droits sont réduits à des variables. C'est le cas du système obtenu à partir de E_{DY} et de toutes les variantes de Dolev-Yao que nous avons présentées au chapitre 5 (*cf.* partie 5.1.2). Nous en déduisons, grâce à la proposition 7.22, que ces théories satisfont la propriété de variants finis.

7.4.2 Théorie « signature en aveugle »

Un système de réécriture convergent représentant la théorie équationnelle « signature en aveugle » présentée au chapitre 2 se compose des règles de réécriture suivantes :

$$\begin{array}{rcl}\mathsf{checksign}(\mathsf{sign}(x,\mathsf{priv}(y)),\mathsf{pub}(y)) & \to & x\\ \mathsf{unblind}(\mathsf{blind}(x,y),y) & \to & x\\ \mathsf{unblind}(\mathsf{sign}(\mathsf{blind}(x,y),z),y) & \to & \mathsf{sign}(x,z)\end{array}$$

Une conséquence directe du lemme 7.25 est que la surréduction basique termine pour le système de réécriture décrit ci-dessus. Nous en déduisons donc, grâce à la proposition 7.22, que cette théorie satisfait la propriété de variants finis.

7.4.3 Théorie ACUN

Un système de réécriture AC-convergent représentant ACUN se compose des trois règles de réécriture suivantes :

$$\mathcal{R}_+ \left\{\begin{array}{rcl} x+x & \to & 0\\ x+0 & \to & x\\ x+(x+y) & \to & y\end{array}\right.$$

Nous allons montrer que ce système de réécriture satisfait la propriété de boundedness. Pour cela, nous devons établir le lemme suivant :

Lemme 7.26
Soient t_1 et t_2 deux termes irréductibles (par rapport à $\to_{\mathsf{AC}\backslash\mathcal{R}_+}$). Le terme t_1+t_2 se réduit à sa forme normale par une dérivation de longueur au plus 1.

Avant de montrer ce lemme, nous introduisons quelques notations. Soient t, u deux termes tels que $t =_{\mathsf{AC}} t_1 + \cdots + t_n$ et $u =_{\mathsf{AC}} u_1 + \cdots + u_m$ où $t_1, \ldots, t_n, u_1, \ldots, u_m$ sont des termes standards (c'est-à-dire $\mathsf{head}(t_i) \neq +$ et $\mathsf{head}(u_i) \neq +$). Nous utilisons les notations suivantes :

$$\begin{array}{rcl} t \wedge u & = & \displaystyle\sum_{f \in \{t_1,\ldots,t_n\}\cap\{u_1,\ldots,u_m\}} f\\ t \smallsetminus u & = & w_1 \text{ tel que } w_1 + (t \wedge u) =_{\mathsf{AC}} t\end{array}$$

Exemple 7.27
Soient $t = a+b+c$ et $u = a+c+d$. Nous avons $t \wedge u = a+c$, $t \smallsetminus u = c$ et $u \smallsetminus t = d$.

Nous pouvons maintenant faire la preuve du Lemme 7.26.

Démonstration.
L'ensemble $\mathcal{NF}$ des termes en formes normales est défini de la façon suivante :

$$\begin{array}{rcl} \mathcal{NF} & = & \mathcal{NF}_0 \cup \mathcal{NF}_+\\ \mathcal{NF}_0 & = & \{0\}\\ \mathcal{NF}_f & = & \bigcup_{f\in\mathcal{F}\backslash sig(\mathsf{E})} f(\mathcal{NF},\cdots,\mathcal{NF})\\ \mathcal{NF}_+ & = & \{u_1+\ldots+u_m \mid u_1,\ldots,u_m \in \mathcal{NF}_f \cup \mathcal{X} \text{ et } \forall i\neq j \text{ on a } u_i \neq u_j\}\end{array}$$

Soient t_1 et t_2 deux termes en formes normales, nous allons montrer que le terme t_1+t_2 se réduit à sa forme normale en au plus une étape. Nous distinguons deux cas :

1. Si $t_1 \in \mathcal{NF}_0$ (resp. $t_2 \in \mathcal{NF}_0$), alors $t_1 + t_2 \xrightarrow[x+0\to x]{} t_2$ (resp. t_1) qui est en forme normale par hypothèse.
2. Sinon, nous avons $t_1 = u_1 + \cdots + u_n$ et $t_2 = v_1 + \cdots + v_m$, avec $u_1, \ldots, u_n \in \mathcal{NF}_f \cup \mathcal{X}$ distincts deux à deux et $v_1, \ldots, v_m \in \mathcal{NF}_f \cup \mathcal{X}$ également distincts deux à deux. Ainsi, soit $u_1, \ldots, u_n, v_1, \ldots, v_m$ sont distincts deux à deux, et dans ce cas le terme $t_1 + t_2$ est déjà en forme normale, soit nous avons :

$$t_1 + t_2 =_{\mathsf{AC}} (t_1 \wedge t_2) + (t_1 \wedge t_2) + (t_1 \setminus t_2) + (t_2 \setminus t_1)$$

et dans ce cas, $t_1 + t_2$ est réductible en une étape en $(t_1 \setminus t_2) + (t_2 \setminus t_1)$ ou 0 (suivant si $t_1 \setminus t_2$ et $t_2 \setminus t_1$ sont vides ou non), et ces termes sont en formes normales. □

Corollaire 7.28
La décomposition $(\mathcal{R}_+, \mathsf{AC})$ *est une décomposition de la théorie équationnelle* ACUN *satisfaisant la propriété de variants finis.*

7.4.4 Théorie AG

Le système de réécriture AC-convergent classiquement utilisé pour représenter AG est le suivant :

$$\mathcal{R}_\times \left\{ \begin{array}{rclrcl} x \times 1 & \to & x & x \times x^{-1} & \to & 1 \\ (x \times y)^{-1} & \to & y^{-1} \times x^{-1} & 1^{-1} & \to & 1 \\ x \times (x^{-1} \times y) & \to & y & (x^{-1})^{-1} & \to & x \end{array} \right.$$

Malheureusement, cette décomposition de la théorie AG ne satisfait pas la propriété de boundedness (*cf.* exemple 7.29). Ce problème est essentiellement dû à la règle

$$(x \times y)^{-1} \to y^{-1} \times x^{-1}.$$

Exemple 7.29
Considérons le terme $t = x^{-1}$ *et la substitution* $\sigma = \{x \mapsto a_0 \times \ldots \times a_n\}$. *Pour réduire* $t\sigma$ *à sa forme normale nous avons besoin d'au moins* n *étapes de réduction.*

Cependant il existe une décomposition moins usuelle de la théorie AG satisfaisant la propriété de variants finis. Cette présentation a été proposée pour la première fois par D. Lankford (*cf.* [Hul80b]). Ce système de réécriture est représenté à la figure 7.2.

$$\begin{array}{rclrcl} x \times 1 & \to & x & x^{-1^{-1}} & \to & x \\ 1^{-1} & \to & 1 & (x^{-1} \times y)^{-1} & \to & y^{-1} \times x \\ x \times x^{-1} & \to & 1 & x \times (x^{-1} \times y) & \to & y \\ y^{-1} \times x^{-1} & \to & (x \times y)^{-1} & x^{-1} \times (y^{-1} \times z) & \to & (y \times x)^{-1} \times z \\ (x \times y)^{-1} \times y & \to & x^{-1} & (y \times x)^{-1} \times (y \times z) & \to & x^{-1} \times z \end{array}$$

Figure 7.2 - Système de réécriture $\mathcal{R}'_\times$.

Le système $\mathcal{R}'_\times$ est un système AC-convergent représentant AG [Hul80b]. Nous allons montrer que ce système satisfait la propriété de boundedness en établissant le lemme suivant :

Lemme 7.30
Soient t_1 et t_2 deux termes irréductibles (par rapport à $\rightarrow_{\mathsf{AC}\backslash\mathcal{R}'_\times}$), les termes t_1^{-1} et $t_1 + t_2$ se réduisent à leurs formes normales par une dérivation de longueur au plus 1 et 2 respectivement.

Les notations $u \wedge v$ et $u \setminus v$ sont définies comme précédemment (nous sommes simplement passés de la notation additive à la notation multiplicative).

Démonstration.
L'ensemble $\mathcal{NF}$ des termes en formes normales est défini de la façon suivante :

$$\begin{array}{lcl}
\mathcal{NF} & = & \mathcal{NF}_1 \cup \mathcal{NF}_{inv} \cup \mathcal{NF}_p \cup \mathcal{NF}_\times \\
\mathcal{NF}_1 & = & \{1\} \\
\mathcal{NF}_f & = & \bigcup_{f \in \mathcal{F} \setminus sig(\mathsf{E})} f(\mathcal{NF}, \ldots, \mathcal{NF}) \\
\mathcal{NF}_{inv} & = & \mathcal{NF}_p^{-1} \\
\mathcal{NF}_p & = & \mathcal{NF}_f \cup \mathcal{X} \cup \mathcal{NF}_p \times \mathcal{NF}_p \\
\mathcal{NF}_\times & = & \{u^{-1} \times v \mid u, v \in \mathcal{NF}_p \text{ et } u \wedge v = 1\}
\end{array}$$

Soit t_1 un terme irréductible, nous allons montrer que t_1^{-1} peut se réduire à sa forme normale en moins d'une étape. Nous distinguons différents cas :

1. Si $t_1 \in \mathcal{NF}_p$, alors $t_1^{-1} \in \mathcal{NF}_{inv} \subseteq \mathcal{NF}$
2. Si $t_1 \in \mathcal{NF}_1$, alors $t_1^{-1} \xrightarrow[1^{-1} \rightarrow 1]{} 1 \in \mathcal{NF}$
3. Si $t_1 \in \mathcal{NF}_{inv}$, alors $t_1 = u^{-1}$ avec $u \in \mathcal{NF}_p$ et on a $t_1^{-1} \xrightarrow[(x^{-1})^{-1} \rightarrow x]{} u \in \mathcal{NF}_p \subseteq \mathcal{NF}$
4. Si $t_1 \in \mathcal{NF}_\times$, alors $t_1 = u^{-1} \times v$, avec $u, v \in \mathcal{NF}_p$ et $u \wedge v = 1$ et dans ce cas on a la dérivation suivante :

$$t_1^{-1} \xrightarrow[(x^{-1} \times y)^{-1} \rightarrow y^{-1} \times x]{} v^{-1} \times u \in \mathcal{NF}_\times \subseteq \mathcal{NF}.$$

Soient t_1 et t_2 deux termes irréductibles, nous allons montrer que $t_1 \times t_2$ peut se réduire à sa forme normale en moins de deux étapes, c'est-à-dire qu'il existe un terme $u \in \mathcal{NF}$ tel que $t_1 \times t_2 \xrightarrow{\leq 2}_{\mathsf{AC}\backslash\mathcal{R}'_\times} u$.

1. Si $t_1 \in \mathcal{NF}_1$ (resp. $t_2 \in \mathcal{NF}_1$), alors $t_1 \times t_2 \xrightarrow[x \times 1 \rightarrow x]{} t_2$ (resp. t_1), lequel est en forme normale par hypothèse.
2. Si $t_1, t_2 \in \mathcal{NF}_p$, alors $t_1 \times t_2 \in \mathcal{NF}_p \subseteq \mathcal{NF}$.
3. Si $t_1, t_2 \in \mathcal{NF}_{inv}$, alors $t_1 = u_1^{-1}$ et $t_2 = u_2^{-1}$ avec $u_1, u_2 \in \mathcal{NF}_p$, alors nous avons

$$t_1 \times t_2 \xrightarrow[y^{-1} \times x^{-1} \rightarrow (x \times y)^{-1}]{} (u_2 \times u_1)^{-1} \in \mathcal{NF}_{inv}.$$

4. Si $t_1 \in \mathcal{NF}_p$ et $t_2 \in \mathcal{NF}_{inv}$ (ou inversement), alors $t_2 = u_2^{-1}$ avec $u_2 \in \mathcal{NF}_p$ et soit $t_1 \wedge u_2 = 1$, et dans ce cas nous avons $t_1 \times t_2 \in \mathcal{NF}_\times \subseteq \mathcal{NF}$, soit $t_1 \wedge u_2 = v$ et, dans ce cas, suivant si $t_1 \setminus u_2$ et $u_2 \setminus t_1$ sont vides ou non, nous sommes dans un des quatre cas suivants :
   (a) Si $t_1 = u_2$, alors $t_1 \times t_2 \xrightarrow[x \times x^{-1} \rightarrow 1]{} 1 \in \mathcal{NF}$.
   (b) Si $t_1 = u_2 \times u_1$, alors $t_1 \times t_2 \xrightarrow[x \times (x^{-1} \times y) \rightarrow y]{} u_2 \in \mathcal{NF}_p \subseteq \mathcal{NF}$.

(c) Si $u_2 = t_1 \times u_1$ avec $u_1 \in \mathcal{NF}_p$: $t_1 \times t_2 \xrightarrow[(y\times x)^{-1}\times y \to x^{-1}]{} u_1^{-1} \in \mathcal{NF}_{inv} \subseteq \mathcal{NF}$.

(d) sinon, $t_1 \times t_2 \xrightarrow[(y\times x)^{-1}\times(y\times z)\to x^{-1}\times z]{} (u_2 \setminus t_1)^{-1} \times (t_1 \setminus u_2) \in \mathcal{NF}$.

5. Si $t_1 \in \mathcal{NF}_\times$ et $t_2 \in \mathcal{NF}_{inv}$ (ou inversement), alors on a $t_1 = u_1^{-1} \times v_1$ et $t_2 = u_2^{-1}$ avec $u_1, v_1, u_2 \in \mathcal{NF}_p$, $t_1 \times t_2 \xrightarrow[x^{-1}\times(y^{-1}\times z)\to(y\times x)^{-1}\times z]{} (u_2 \times u_1)^{-1} \times v_1$ qui se normalise en une étape (*cf.* cas 4).
6. Si $t_1 \in \mathcal{NF}_\times$ et $t_2 \in \mathcal{NF}_p$ (ou inversement), alors on a $t_1 = u_1^{-1} \times v_1$ avec $u_1, v_1 \in \mathcal{NF}_p$ et $t_1 \times t_2 =_{\mathsf{AC}} u_1^{-1} \times (v_1 \times t_2)$, lequel se normalise en une étape (*cf.* cas 4).
7. Si $t_1 \in \mathcal{NF}_\times$ et $t_2 \in \mathcal{NF}_\times$, alors $t_1 = u_1^{-1} \times v_1$ et $t_2 = u_2^{-1} \times v_2$ avec $u_1, v_1, u_2, v_2 \in \mathcal{NF}_p$. Dans ce cas, $t_1 \times t_2 \xrightarrow[x^{-1}\times(y^{-1}\times z)\to(y\times x)^{-1}\times z]{} (u_2 \times u_1)^{-1} \times (v_1 \times v_2)$ et nous concluons encore grâce au cas 4. □

Exemple 7.31
Considérons les termes $t_1 = a \times (b \times c)^{-1}$ et $t_2 = a^{-1} \times b$. Nous avons la dérivation suivante permettant de réduire $t_1 \times t_2$ à sa forme normale c^{-1}.

$$(a \times (b \times c)^{-1}) \times (a^{-1} \times b) \to_{\mathsf{AC}\backslash\mathcal{R}'_\times} ((b \times c) \times a)^{-1} \times (a \times b) \to_{\mathsf{AC}\backslash\mathcal{R}'_\times} c^{-1}$$

Une conséquence directe du lemme 7.30 et des propositions 7.24 et 7.18 est le corollaire suivant :

Corollaire 7.32
La décomposition $(\mathcal{R}'_\times, \mathsf{AC})$ est une décomposition de la théorie équationnelle AG satisfaisant la propriété de variants finis.

7.4.5 Théorie DH

Le système de réécriture AC-convergent utilisé pour représenter DH s'obtient en ajoutant les deux règles décrites à la figure 7.3 au système de réécriture $\mathcal{R}_\times$ (ou $\mathcal{R}'_\times$) représentant AG (*cf.* page 172). Nous notons $\mathcal{R}_{\mathsf{DH}}$ le système de réécriture composé des règles de $\mathcal{R}'_\times$ et de $\mathcal{R}_{\mathsf{exp}}$.

$$\begin{array}{rcl} \mathsf{exp}(x, 1) & \to & x \\ \mathsf{exp}(\mathsf{exp}(x, y), z) & \to & \mathsf{exp}(x, y \times z) \end{array}$$

Figure 7.3 - Système de réécriture $\mathcal{R}_{\mathsf{exp}}$.

Lemme 7.33
Soient t_1 et t_2 deux termes irréductibles (par rapport à $\to_{\mathsf{AC}\backslash\mathcal{R}_{\mathsf{DH}}}$), les termes t_1^{-1}, $t_1 \times t_2$ et $\mathsf{exp}(t_1, t_2)$ se réduisent à leurs formes normales par une dérivation de longueur au plus 1, 2 et 4 respectivement.

Démonstration.
L'ensemble $\mathcal{NF}$ des termes en formes normales est défini de la façon suivante :

$$
\begin{array}{lcl}
\mathcal{NF} & = & \mathcal{NF}_1 \cup \mathcal{NF}_{inv} \cup \mathcal{NF}_p \cup \mathcal{NF}_\times \cup \mathcal{NF}_{\mathsf{exp}} \\
\mathcal{NF}_1 & = & \{1\} \\
\mathcal{NF}_f & = & \bigcup_{f \in \mathcal{F} \setminus sig(\mathsf{E})} f(\mathcal{NF}, \ldots, \mathcal{NF}) \\
\mathcal{NF}_{inv} & = & \mathcal{NF}_p^{-1} \cup \mathcal{NF}_{\mathsf{exp}}^{-1} \\
\mathcal{NF}_p & = & \mathcal{NF}_f \cup \mathcal{X} \cup \mathcal{NF}_p \times \mathcal{NF}_p \cup \mathcal{NF}_{\mathsf{exp}} \times \mathcal{NF}_p \cup \mathcal{NF}_{\mathsf{exp}} \times \mathcal{NF}_{\mathsf{exp}} \\
\mathcal{NF}_\times & = & \{u^{-1} \times v \mid u, v \in \mathcal{NF}_p \text{ et } u \wedge v = 1\} \\
\mathcal{NF}_{\mathsf{exp}} & = & \mathsf{exp}(\mathcal{NF}_{base}, \mathcal{NF}_{exposant}) \\
\mathcal{NF}_{base} & = & \mathcal{NF}_{inv} \cup \mathcal{NF}_1 \cup \mathcal{NF}_p \cup \mathcal{NF}_\times \\
\mathcal{NF}_{exposant} & = & \mathcal{NF}_{inv} \cup \mathcal{NF}_p \cup \mathcal{NF}_\times \cup \mathcal{NF}_{\mathsf{exp}}
\end{array}
$$

Soient $t_1, t_2 \in \mathcal{NF}$, on a $t_1^{-1} \xrightarrow[\mathsf{AC}\backslash\mathcal{R}_{\mathsf{DH}}]{\leq 1} t \in \mathcal{NF}$ et $t_1 \times t_2 \xrightarrow[\mathsf{AC}\backslash\mathcal{R}_{\mathsf{DH}}]{\leq 2} t \in \mathcal{NF}$. Ces résultats s'obtiennent comme dans le lemme 7.30. Il nous reste à traiter le cas du symbole de fonction exp.

Soient t_1 et t_2 deux termes irréductibles, nous allons montrer que $\mathsf{exp}(t_1, t_2)$ peut se réduire à sa forme normale en moins de 4 étapes. Autrement dit, nous allons établir le fait suivant :

$$\text{il existe } t \in \mathcal{NF} \text{ tel que } \mathsf{exp}(t_1, t_2) \xrightarrow{\leq 4}_{\mathsf{AC}\backslash\mathcal{R}_{\mathsf{DH}}} t.$$

- Si $t_1 \notin \mathcal{NF}_{\mathsf{exp}}$ et $t_2 \notin \mathcal{NF}_1$, alors $\mathsf{exp}(t_1, t_2) \in \mathcal{NF}_{\mathsf{exp}} \subseteq \mathcal{NF}$.
- Si $t_2 \in \mathcal{NF}_1$, alors $\mathsf{exp}(t_1, t_2) \xrightarrow[\mathsf{exp}(x,1) \to x]{} t_1$.
- Si $t_1 \in \mathcal{NF}_{\mathsf{exp}}$ alors $t_1 = \mathsf{exp}(u_1, v_1)$ avec $u_1 \in \mathcal{NF}_{base}$ et $v_1 \in \mathcal{NF}_{exposant}$. Dans ce cas, nous avons $\mathsf{exp}(t_1, t_2) \xrightarrow[\mathsf{exp}(\mathsf{exp}(x,y),z) \to \mathsf{exp}(x, y \times z)]{} \mathsf{exp}(u_1, v_1 \times t_2)$. Maintenant, nous avons $v_1 \times t_2 \xrightarrow{\leq 2}_{\mathsf{AC}\backslash\mathcal{R}_{\mathsf{DH}}} w \in \mathcal{NF}$ et, soit $w \notin \mathcal{NF}_1$, auquel cas le terme obtenu est en forme normale, soit $\mathsf{exp}(u_1, w) \xrightarrow[\mathsf{exp}(x,1) \to x]{} u_1$, auquel cas u_1 est en forme normale. Dans le pire des cas, nous avons donc besoin de quatre étapes de réduction. □

Nous illustrons le cas où nous avons besoin de quatre étapes de réduction pour obtenir un terme en forme normale dans l'exemple 7.34.

Exemple 7.34
Soient $t_1 = \mathsf{exp}(e, a^{-1} \times b)$ et $t_2 = b^{-1} \times a$, le terme $t = \mathsf{exp}(t_1, t_2)$ se réduit à sa forme normale (par rapport à $\to_{\mathsf{AC}\backslash\mathcal{R}_{\mathsf{DH}}}$) par une dérivation de longueur 4. En effet, nous avons :

$$
\begin{array}{lcl}
\mathsf{exp}(\mathsf{exp}(e, a^{-1} \times b), b^{-1} \times a) & \to & \mathsf{exp}(e, (a^{-1} \times b) \times (b^{-1} \times a)) \\
& \to & \mathsf{exp}(e, (a \times b)^{-1} \times (a \times b)) \\
& \to & \mathsf{exp}(e, 1) \\
& \to & e
\end{array}
$$

Une conséquence directe des propositions 7.18 et 7.24 et du lemme 7.37 est le corollaire suivant :

Corollaire 7.35
La décomposition $(\mathcal{R}_{\mathsf{DH}}, \mathsf{AC})$ est une décomposition de la théorie équationnelle DH *satisfaisant la propriété de variants finis.*

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

$$
\begin{array}{rcl}
\mathsf{h}(x)\times\mathsf{h}(y) & \to & \mathsf{h}(x+y) \\
\mathsf{exp}(\mathsf{h}(x),y) & \to & \mathsf{h}(x\times y)
\end{array}
$$

Figure 7.4 - Système de réécriture $\mathcal{R}_{\mathsf{P}}$ représentant EP.

7.4.6 Théorie équationnelle de l'étude de cas

La théorie équationnelle E_{P} (décrite à la figure 7.1) est une théorie équationnelle très particulière n'ayant fait l'objet d'aucune étude jusqu'à présent. L'outil CiME [CM96] nous a permis d'obtenir un système de réécriture AC-convergent représentant E_{P}. Ce système de réécriture est décrit à la figure 7.4.

Lemme 7.36
Le système de réécriture $\mathcal{R}_{\mathsf{P}}$ (cf. figure 7.4) est un système AC-convergent représentant E_{P}.

Lemme 7.37
Soient t_1 et t_2 des termes irréductibles (par rapport à $\to_{\mathsf{AC}\backslash\mathcal{R}_{\mathsf{P}}}$), les termes $\mathsf{h}(t_1)$, $-(t_1)$, t_1+t_2, $t_1\times t_2$ et $\mathsf{exp}(t_1,t_2)$ se réduisent à leurs formes normales par une dérivation de longueur au plus 0, 1, 2, 3 et 4 respectivement.

Démonstration.
L'ensemble $\mathcal{NF}$ des termes en formes normales est défini de la façon suivante :

$$
\begin{array}{lcl}
\mathcal{NF} & = & \mathcal{NF}_0 \cup \mathcal{NF}_{opp} \cup \mathcal{NF}_s \cup \mathcal{NF}_+ \cup \mathcal{NF}_\times \cup \mathcal{NF}_{\mathsf{h}} \cup \mathcal{NF}_{\mathsf{exp}} \\
\mathcal{NF}_0 & = & \{0\} \\
\mathcal{NF}_f & = & \bigcup_{f\in\mathcal{F}\smallsetminus sig(\mathsf{E})} f(\mathcal{NF},\ldots,\mathcal{NF}) \\
\mathcal{NF}_{opp} & = & -\mathcal{NF}_s \cup -\mathcal{NF}_{\mathsf{exp}} \cup -\mathcal{NF}_\times \cup \mathcal{NF}_{\mathsf{h}} \\
\mathcal{NF}_s & = & \mathcal{NF}_f \cup \mathcal{X} \cup \mathcal{NF}_s + \mathcal{NF}_s \cup \mathcal{NF}_\times \cup \mathcal{NF}_{\mathsf{exp}} \cup \mathcal{NF}_s + \mathcal{NF}_{\mathsf{h}} \cup \mathcal{NF}_{\mathsf{h}} + \mathcal{NF}_{\mathsf{h}} \\
\mathcal{NF}_+ & = & \{-(u)+v \mid u,v\in\mathcal{NF}_p \text{ et } u\wedge v=0\} \\
\mathcal{NF}_\times & = & \mathcal{N}\times\mathcal{N} \qquad\qquad\qquad \text{où } \mathcal{N}=\mathcal{NF}_{\mathsf{exp}}\cup\mathcal{NF}_s\cup\mathcal{NF}_{opp}\cup\mathcal{NF}_+\cup\mathcal{NF}_0 \\
\mathcal{NF}_{\mathsf{h}} & = & \mathsf{h}(\mathcal{NF}) \\
\mathcal{NF}_{\mathsf{exp}} & = & \mathsf{exp}(\mathcal{NF}_{base},\mathcal{NF}) \\
\mathcal{NF}_{base} & = & \mathcal{NF}_{\mathsf{exp}} \cup \mathcal{NF}_s \cup \mathcal{NF}_{opp} \cup \mathcal{NF}_0 \cup \mathcal{NF}_\times \cup \mathcal{NF}_+
\end{array}
$$

Pour tout terme $t\in\mathcal{NF}$, on a $\mathsf{h}(t)\in\mathcal{NF}$, d'où le résultat en ce qui concerne le symbole h. Soient $t_1,t_2\in\mathcal{NF}$. On obtient, par une preuve similaire à celle réalisée dans le lemme 7.30, les résultats suivants : $-t_1 \xrightarrow[\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}]{\leq 1} t \in \mathcal{NF}$ et $t_1 + t_2 \xrightarrow[\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}]{\leq 2} t \in \mathcal{NF}$. Il nous reste donc à traiter le cas des symboles de fonction $\times$ et exp.

Soient t_1 et t_2 deux termes irréductibles. Nous allons montrer que $t_1 \times t_2$ peut se réduire à sa forme normale en moins de trois étapes. Autrement dit, nous allons établir le résultat suivant :

$$\text{il existe } t \in \mathcal{NF} \text{ tel que } t_1 \times t_2 \xrightarrow{\leq 3}_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}} t.$$

- Si $t_1 \notin \mathcal{NF}_\mathsf{h}$ ou $t_2 \notin \mathcal{NF}_\mathsf{h}$, alors $t_1 \times t_2 \in \mathcal{NF}_\times \subseteq \mathcal{NF}$.
- Si $t_1, t_2 \in \mathcal{NF}_\mathsf{h}$, alors $t_1 = \mathsf{h}(u_1)$ et $t_2 = \mathsf{h}(u_2)$ avec $u_1, u_2 \in \mathcal{NF}$ et dans ce cas, nous avons $t_1 \times t_2 \xrightarrow[\mathsf{h}(x)\times\mathsf{h}(y)\to\mathsf{h}(x+y)]{} \mathsf{h}(u_1 + u_2)$. Nous savons que $u_1 + u_2 \xrightarrow{\leq 2}_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}} w$ avec $w \in \mathcal{NF}$. Nous en déduisons que $\mathsf{h}(u_1) \times \mathsf{h}(u_2) \xrightarrow{\leq 3}_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}} \mathsf{h}(w)$ avec $\mathsf{h}(w) \in \mathcal{NF}$.

Soient t_1 et t_2 deux termes irréductibles, nous allons montrer que $\mathsf{exp}(t_1, t_2)$ peut se réduire à sa forme normale en moins de quatre étapes. Autrement dit, nous allons montrer le résultat suivant :

$$\text{il existe } t \in \mathcal{NF} \text{ tel que } \mathsf{exp}(t_1, t_2) \xrightarrow{\leq 4}_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}} t.$$

- Si $t_1 \notin \mathcal{NF}_\mathsf{h}$ alors $\mathsf{exp}(t_1, t_2) \in \mathcal{NF}_\mathsf{exp} \subseteq \mathcal{NF}$.
- Si $t_1 \in \mathcal{NF}_\mathsf{h}$ alors $t_1 = \mathsf{h}(u_1)$ avec $u_1 \in \mathcal{NF}$ et dans ce cas, nous avons

$$\mathsf{exp}(t_1, t_2) \xrightarrow[\mathsf{exp}(\mathsf{h}(x),y)\to\mathsf{h}(x\times y)]{} \mathsf{h}(u_1 \times t_2).$$

  Nous savons que $u_1 \times t_2 \xrightarrow{\leq 3}_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}} w$ avec $w \in \mathcal{NF}$, ce qui nous permet de déduire que $\mathsf{exp}(\mathsf{h}(u_1), t_2) \xrightarrow{\leq 4}_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}} \mathsf{h}(w)$ avec $\mathsf{h}(w) \in \mathcal{NF}$. □

Nous illustrons le cas où nous avons besoin de quatre étapes de réduction pour obtenir un terme en forme normale dans l'exemple 7.38.

Exemple 7.38
Soit $t_1 = \mathsf{h}(\mathsf{h}(-a + b))$ et $t_2 = \mathsf{h}(-b + a + c)$, le terme $t = \mathsf{exp}(t_1, t_2)$ se réduit à sa forme normale (par rapport à $\to_{\mathsf{AC}\backslash\mathcal{R}_\mathsf{P}}$) par une dérivation de longueur 4. En effet, nous avons :

$$\begin{array}{rcl} \mathsf{exp}(\mathsf{h}(\mathsf{h}(-a+b), \mathsf{h}(-b+a+c)) & \to & \mathsf{h}(\mathsf{h}(-a+b) \times \mathsf{h}(-b+a+c)) \\ & \to & \mathsf{h}(\mathsf{h}((-a+b) + (-b+a+c))) \\ & \to & \mathsf{h}(\mathsf{h}(-(a+b) + (a+b+c))) \\ & \to & \mathsf{h}(\mathsf{h}(c)) \end{array}$$

Une conséquence directe des propositions 7.18 et 7.24 et du lemme 7.4.6 est le corollaire suivant :

Corollaire 7.39
La décomposition $(\mathcal{R}_\mathsf{P}, \mathsf{AC})$ est une décomposition de la théorie équationnelle E_P satisfaisant la propriété de variants finis.

7.5 Discussion

La propriété de variants finis, introduite dans ce chapitre, est une propriété nouvelle, motivée par ses applications concernant la vérification des protocoles cryptographiques. Ce concept, consistant à deviner à l'avance les réductions susceptibles d'être déclenchées après application d'une substitution sur un terme donné, a déjà été utilisé dans [CLC03] (*cf.* partie 7.5.2) dans le cas de la théorie ACUN. Plus récemment, il a permis d'établir un résultat générique dans le cadre de la vérification des protocoles cryptographiques (*cf.* partie 7.5.3). D'autre part, l'idée de borner la longueur de la dérivation issue d'un terme $t\sigma$ indépendemment de la substitution σ que l'on choisit d'appliquer (propriété de boundedness) a également été utilisée à plusieurs reprises pour obtenir des procédures génériques permettant de décider le problème d'unification (*cf.* partie 7.5.1).

7.5.1 Travaux sur l'unification

Dans [NPS97], P. Narendran *et al.* s'intéressent au problème d'unification pour différentes théories équationnelles. Ils définissent la notion de système de réécriture *optimally reducing*. Un système $\mathcal{R}$ est dit *optimally reducing* si chacune des règles $l \to r \in \mathcal{R}$ satisfait la condition suivante :

> « pour toute substitution θ, si $s\theta$ est irréductible pour chacun des sous-termes propres de l, alors $r\theta$ est également irréductible. »

Cette condition est suffisante pour établir la propriété de boundedness d'un système $\mathcal{R}$, puisque cela signifie que $\mathcal{R}$ satisfait la propriété de boundedness (avec pour borne $c = 1$). Cependant, pour de nombreuses théories équationnelles, nous avons besoin de considérer des constantes plus grande que 1 (*e.g.* groupe abélien on a $c = 2$). De plus, même si la borne permettant de satisfaire la propriété de boundedness est égale à 1, le système peut ne pas être *optimally reducing* simplement par la présence d'une règle inutile qui ne vérifie pas la condition. Enfin, en présence des axiomes associativité et commutativité, P. Narendran *et al.* sont obligés d'imposer des conditions supplémentaires très restrictives. Nos conditions sont plus faibles et sont donc vérifiées par un plus grand nombre de théories équationnelles (*cf.* partie 7.4).

Des travaux, plus récents, ont également été faits par E. Viola [Vio01]. L'idée était d'établir des conditions sur une théorie équationnelle E dans le but d'assurer la décidabilité du problème d'unification, par une procédure non-déterministe s'exécutant en temps polynomial. Pour cela, il établit l'existence d'une borne sur la longueur maximale d'une séquence de surréduction basique. Il étend ensuite son résultat au cas des théories faisant intervenir un opérateur AC.

7.5.2 Travaux de H. Comon-Lundh et V. Cortier

Cette propriété de variants finis, consistant à deviner à l'avance les réductions possibles, a été utilisé par H. Comon-Lundh et V. Cortier [CLC03] dans le cas particulier de la théorie du « ou » exclusif (*cf.* lemme 4 dans [CLC03]). Ils s'intéressent en fait au problème de la vérification d'un protocole en présence de la théorie du « ou » exclusif dans le formalisme des clauses de Horn. En particulier, ils montrent qu'ils peuvent calculer, à partir d'un ensemble de clauses de départ $\mathcal{S}$, un ensemble fini $\mathcal{S}'$ d'instances de $\mathcal{S}$, et qu'ils peuvent ne considérer que les instances de $\mathcal{S}'$ obtenues après application d'une substitution n'entraînant pas de réduction. La construction de l'ensemble $\mathcal{S}'$ se fait en devinant à l'avance tous les partages possibles entre variables. Le résultat qu'ils établissent est spécifique au cas de la théorie équationnelle du « ou » exclusif et est en réalité plus précis que la simple propriété de variants finis : ils ont besoin de contrôler la taille des clauses « variants » générées (c'est-à-dire la taille des clauses dans l'ensemble $\mathcal{S}'$).

7.5.3 Travaux de V. Bernat et H. Comon-Lundh

La propriété de variant finis, introduite pour la première fois dans [Com04], est une première étape vers l'obtention d'un résultat pour la vérification des protocoles cryptographiques. Elle permet de se débarrasser d'un certain nombre d'axiomes et de se ramener, dans la majorité des cas, à la théorie vide ou à la théorie AC.

Dans le cas de la théorie vide, des travaux récents [Ber06] ont permis d'obtenir des conditions suffisantes sur le système d'inférence $\mathcal{V}ar(\mathcal{I})$, obtenu par calcul des variants du système d'inférence $\mathcal{I}$ représentant le pouvoir de déduction de l'intrus. Ces conditions permettent d'assurer la décidabilité du problème de la sécurité d'un protocole pour un nombre borné de sessions. Tout d'abord, une propriété de localité (*cf.* partie 4.1.2) est demandée sur le système d'inférence $\mathcal{V}ar(\mathcal{I})$. D'autre part, des conditions syntaxiques supplémentaires doivent être satisfaites par le système d'inférence $\mathcal{V}ar(\mathcal{I})$.

Ce résultat générique permet de conclure pour différentes théories de l'intrus comme par exemple la théorie standard de Dolev-Yao ou la théorie permettant de prendre en compte la propriété préfixe (chiffrement par blocs en mode CBC).

7.5.4 De la difficulté à résoudre le cas AC

Pour traiter une théorie équationnelle E faisant intervenir un opérateur AC, la propriété de variants finis permet de réduire la théorie équationnelle considérée à la théorie AC. Il faut noter que cette réduction de la théorie équationnelle se fait au prix d'une complication du système d'inférence (*cf.* exemple 7.12). Le système d'inférence $\mathcal{V}ar(\mathcal{I})$, obtenu par calcul des variants des différentes règles d'inférence composant le système d'inférence de départ, est susceptible de contenir plus de règles, et ces dernières peuvent sembler, au moins à première vue, plus complexes.

Une première étape naturelle avant l'obtention de conditions suffisantes sur le système d'inférence $\mathcal{V}ar(\mathcal{I})$ est de résoudre le cas « AC pur ». Il s'agit de trouver un algorithme permettant de décider la satisfaisabilité de systèmes de contraintes bien formés dans le cas de la théorie AC et du système d'inférence $\mathcal{I}$, composé d'une unique règle correspondant à l'application de l'opérateur AC sur deux termes déductibles par l'intrus. Ce problème peut sembler, à première vue, relativement simple, mais les techniques développées dans le cas des théories ACUN, AG, ... ne s'appliquent pas (*cf.* partie 6.5). À notre connaissance, il n'existe à l'heure actuelle aucune procédure permettant de traiter ce problème.

Chapitre 8

Conclusion et perspectives

La vérification formelle d'un problème, et donc en particulier d'un protocole cryptographique, se compose de deux étapes essentielles : la modélisation et la vérification. Nous proposons, selon ces deux grands axes, une synthèse de nos contributions ainsi que des améliorations et des prolongements possibles.

Modélisation

Les résultats obtenus dans cette thèse ont été réalisés dans deux variantes du modèle par rôles. Alors que la modélisation d'un protocole par des systèmes de réécriture (modèle par rôle avec filtrage) était déjà bien établie avant le début de cette thèse, l'utilisation du modèle par rôle avec tests d'égalités était beaucoup moins répandue. L'avantage du modèle avec filtrage est qu'il permet une représentation compacte et directe de chaque pas de protocole, et évite de spécifier les actions réalisées par l'agent pour traiter un message. Cet avantage s'accompagne d'un défaut majeur : ce modèle permet la spécification de protocoles non-réalistes comme c'est le cas du rôle $\mathsf{recv}(\{x_1\}_{x_2});\ \mathsf{send}(x_2)$. Ainsi, les procédures développées dans le modèle avec filtrage sont parfois rendues complexes en raison de la classe de protocoles très large considérée. Or, les motivations justifiant l'étude d'une telle classe ne sont pas claires. En effet, bon nombre de protocoles de cette classe ne sont pas « exécutables ».

Le modèle par rôle avec tests d'égalités semble plus adapté à l'obtention de résultats génériques. Le problème de l'exécutabilité d'un protocole ne se pose plus puisque les actions réalisées par les agents sont entièrement décrites. Ainsi, le simple fait de considérer des rôles bien formés (notion triviale à décider dans ce modèle, *cf.* chapitre 3) permet de ne considérer que des protocoles réalistes. À théorie équationnelle égale, les procédures de vérification sont généralement plus simples puisque la classe des protocoles considérée est moins grande.

Dans le cas du modèle standard de Dolev-Yao, le modèle avec filtrage présente l'immense avantage d'éviter l'introduction d'une théorie équationnelle et de permettre de réaliser l'étude des protocoles dans l'algèbre libre. Dans un tel cadre, l'utilisation de ce modèle était alors entièrement justifiée. Mais, avec la prise en compte de propriétés algébriques plus complexes, il n'est dorénavant plus possible de travailler dans l'algèbre libre. L'ajout d'une théorie équationnelle semble incontournable et, il s'avère que la présence simultanée du filtrage et du raisonnement équationnel est source de complications. La solution consistant à supprimer le filtrage apparaît alors comme une solution naturelle qui a, à l'heure actuelle et au vu des résultats génériques obtenus récemment dans ce modèle (*e.g.* [DJ04a, Bau05, CR05]), déjà fait ses preuves.

Plusieurs extensions du modèle par rôles (avec filtrage ou avec tests d'égalités) sont envisageables. Dans ce modèle, toutes les communications sont réalisées en présence d'un intrus capable d'écouter, d'intercepter et d'émettre des messages sur le réseau. Or, il existe différents type de réseaux de communication et les capacités de contrôle de l'intrus dépendent bien entendu du réseau considéré. Par exemple, on ne peut pas empêcher quelqu'un d'écouter les messages envoyés sur les réseaux sans-fil, mais il est pratiquement impossible d'empêcher un message d'arriver à destination. Pour ce genre de protocole, il serait donc naturel de considérer un intrus pouvant écouter et émettre, mais ne pouvant pas intercepter les messages. En fait, il faudrait modéliser ces différents type de réseaux (réseau sans-fil, réseau câblé, ...) par des canaux de types différents et avoir un intrus ayant des capacités de contrôle différentes sur chacun de ces canaux.

Enfin, dans cette thèse, nous nous sommes intéressés à des propriétés de sécurité relativement simples. Il existe de nombreuses autres propriétés pertinentes pour les protocoles cryptographiques qu'il serait intéressant d'étudier (*e.g.* anonymat, non-répudiation, ...). Du point de vue de la modélisation, toutes ces propriétés ne sont pas encore clairement définies, un travail important de modélisation reste à faire. À long terme, il serait intéressant de définir un langage de propriétés avec une sémantique clairement définie afin de ne plus avoir à adapter la modélisation du protocole à la propriété à vérifier.

Vérification

Sur le plan de la vérification des protocoles cryptographiques, nous avons relâché l'hypothèse du chiffrement parfait afin de prendre en compte les propriétés algébriques des primitives cryptographiques. Nous avons proposé des procédures de décision pour le problème de déduction de l'intrus et pour le problème de la recherche d'attaques pour un nombre borné de sessions.

Ces deux problèmes sont étudiés en présence de primitives cryptographiques possédant des propriétés algébriques (*cf.* chapitres 5 et 6). Les types des théories équationnelles traitées dans ces deux chapitres sont très différents, ce qui nous a conduit à proposer des algorithmes assez différents. Par analogie avec les procédures permettant de résoudre le problème d'unification modulo une théorie équationnelle (*cf.* [BS01]), la procédure proposée dans le chapitre 5 est syntaxique alors que l'approche utilisée dans le chapitre 6 pourrait être qualifiée de sémantique. En effet, dans le chapitre 5, la classe de théories équationnelles traitée est définie syntaxiquement et l'algorithme proposé pour résoudre le problème de la sécurité est une procédure non-déterministe basée sur une série de règles de transformations. En cela, elle s'apparente aux procédures d'unification basées sur des techniques de surréduction. En revanche, dans le chapitre 6, l'approche utilisée est plutôt sémantique et s'apparente aux algorithmes d'unification proposées pour résoudre les problème d'unification dans les théories monoïdales [Nut90] ou commutatives [Baa93]. Pour de telles théories, les algorithmes d'unification, mais aussi ceux que nous proposons dans cette thèse, se réduisent à la résolution de systèmes d'équations dans la structure algébrique correspondante.

Les procédures, décrites au chapitre 6, pour résoudre le problème de déduction de l'intrus et le problème de la sécurité, permettent de traiter des théories équationnelles complexes faisant intervenir un opérateur AC et une propriété d'homomorphisme sur cet opérateur. Une caractérisation algébrique des problèmes considérés a permis de les réduire à la résolution de systèmes d'équations, et d'établir des procédures de décision. Ces procédures de décision sont génériques, dans le sens où elles s'adaptent dès lors que la structure algébrique associée à la théorie équationnelle considérée vérifie quelques bonnes propriétés. Dans le cas du problème de déduction de l'intrus, la procédure proposée permet de traiter un grand nombre de théories équationnelles parmi lesquelles ACh, ACUNh et AGh, mais aussi AC, ACUN, AG. Dans le cas du problème de la sécurité, les propriétés requises sont plus spécifiques. La procédure proposée permet de traiter les théories équationnelles ACUN et ACUNh, et devrait permettre de traiter la théorie AG. Nous avons montré que le problème était en fait indécidable pour les théories ACh et AGh. Le cas de l'opérateur AC est un problème encore ouvert à l'heure actuelle, dont l'intérêt va bien au delà de l'étude de cette « simple » théorie.

Nous avons défini au chapitre 7 une propriété appelée *propriété de variants finis* qui, lorsqu'elle est satisfaite par une théorie équationnelle, permet de la simplifier et de se débarrasser d'un certain nombre d'axiomes. Dans le cas du problème de la sécurité, cela permet de se ramener à l'étude du problème de la sécurité pour une théorie équationnelle plus simple. D'autre part, nous avons montré que de nombreuses théories équationnelles satisfont cette propriété, et que dans la majorité des cas, la propriété de variants finis permet de réduire la théorie considérée à la théorie vide ou à la théorie AC. Ainsi, la résolution du problème de la sécurité en présence d'un opérateur AC aura des répercussions considérables. Dans le cas où la propriété de variants finis permet de faire disparaître entièrement la théorie équationnelle, des conditions suffisantes sur le système d'inférence ont été mis en place [Ber06]. Elles permettent d'assurer la décidabilité du problème de la sécurité pour un nombre borné de sessions. Le système d'inférence obtenu après calcul des variants doit vérifier quelques conditions syntaxiques et une propriété de *localité*. Une poursuite naturelle de ces travaux est de réaliser un travail similaire en présence d'un ou plusieurs opérateurs associatifs et commutatifs, et de mettre au point des conditions syntaxiques suffisantes sur le système d'inférence afin d'assurer la décidabilité du problème de la sécurité.

Enfin, dans l'optique d'obtenir des procédures de vérification entièrement automatiques, il serait intéressant de développer des techniques permettant d'automatiser les preuves de localité et de mettre au point un ou plusieurs critères décidables qui permettraient de décider si une théorie équationnelle donnée satisfait la propriété de variants finis.

Bibliographie

[AC04] M. Abadi et V. Cortier : Deciding knowledge in security protocols under equational theories. *In Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP'04)*, volume 3142 de *Lecture Notes in Computer Science*, pages 46–58, Turku (Finland), 2004. Springer-Verlag.

[AC05] M. Abadi et V. Cortier : Deciding knowledge in security protocols under (many more) equational theories. *In Proceedings of the 18th IEEE Computer Security Foundations Workshop (CSFW'05)*, pages 62–76, Aix-en-Provence (France), 2005. IEEE Computer Society Press.

[AF01] M. Abadi et C. Fournet : Mobile values, new names, and secure communication. *In Proceedings of the 28th ACM Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115, London (United Kingdom), 2001. ACM Press.

[AG97] M. Abadi et A. Gordon : A calculus for cryptographic protocols : The spi calculus. *In Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 36–47, Zurich (Switzerland), 1997. ACM Press.

[ALV02] R. Amadio, D. Lugiez et V. Vanackère : On the symbolic reduction of processes with cryptographic functions. *Theoretical Computer Science*, 290(1): 695–740, 2002.

[Asc04] M. Aschenbrenner : Ideal membership in polynomial rings over the integers. *Journal of the American Mathematical Society*, 17:407–441, 2004.

[Baa93] F. Baader : Unification in commutative theories, Hilbert's basis theorem, and Gröbner bases. *Journal of the ACM*, 40(3):477–503, 1993.

[Bau05] M. Baudet : Deciding security of protocols against off-line guessing attacks. *In Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05)*, pages 16–25, Alexandria, (Virginia, USA), 2005. ACM Press.

[BB03] M. Boreale et M. G. Buscemi : Symbolic analysis of crypto-protocols based on modular exponentiation. *In Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS'03)*, volume 2747 de *Lecture Notes in Computer Science*, pages 269–278, Bratislava (Slovak Republic), 2003. Springer-Verlag.

[BDKV05] L. Bozga, S. Delaune, F. Klay et L. Vigneron : Retour d'expérience sur la validation du porte-monnaie électronique. Rapport Technique 5, projet RNTL PROUVÉ, 2005. 29 pages.

[Ber06] V. BERNAT : *Théories de l'intrus pour la vérification des protocoles cryptographiques*. Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France, 2006.

[BGW01] N. BORISOV, I. GOLDBERG et D. WAGNER : Intercepting mobile communications : The insecurity of 802.11. *In Proceedings of the 7th Annual International Conference on Mobile Computing and Networking (MOBICOM'01)*, pages 180–188, Rome (Italy), 2001. ACM Press.

[Big05] R. BIGOT : La diffusion des technologies de l'information dans la société française. *Enquête « Conditions de vie et Aspirations des Français »,* Centre de Recherche pour l'Etude et l'Observation des Conditions de Vie (CRÉDOC), 2005. http ://www.arcep.fr/publications/etudes/et-credoc2005.pdf.

[Bla01] B. BLANCHET : An efficient cryptographic protocol verifier based on prolog rules. *In Proceedings of the 14th Computer Security Foundations Workshop (CSFW'01)*, pages 82–96, Cape Breton (Canada), 2001. IEEE Computer Society Press.

[Bor01] M. BOREALE : Symbolic trace analysis of cryptographic protocols. *In Proceedings of the 28th International Colloquium on Automata, Languages, and Programming (ICALP'01)*, volume 2076 de *Lecture Notes in Computer Science*, pages 667–681, Crete (Greece), 2001. Springer-Verlag.

[BS01] F. BAADER et W. SNYDER : Unification theory. *In* A. ROBINSON et A. VORONKOV, éditeurs : *Handbook of Automated Reasoning*, chapitre 8. Elsevier, 2001.

[CD05] H. COMON-LUNDH et S. DELAUNE : The finite variant property : How to get rid of some algebraic properties. *In Proceedings of the 16th International Conference on Rewriting Techniques and Applications (RTA'05)*, volume 3467 de *Lecture Notes in Computer Science*, pages 294–307, Nara (Japan), 2005. Springer-Verlag.

[CDL+99] I. CERVESATO, N. DURGIN, P. LINCOLN, J. MITCHELL et A. SCEDROV : A meta-notation for protocol analysis. *In Proceedings of the 12th Computer Security Foundations Workshop (CSFW'99)*, pages 55–69, Mordano (Italy), 1999. IEEE Computer Society Press.

[CDL06] V. CORTIER, S. DELAUNE et P. LAFOURCADE : A survey of algebraic properties used in cryptographic protocols. *Journal of Computer Security*, 14(1):1–43, 2006.

[Cha84] D. CHAUM : Blind signature system. *In* Plenum PRESS, éditeur : *Proceedings of Advances in Cryptology (CRYPTO'83)*, New York (USA), 1984.

[CJ97] J. CLARK et J. JACOB : A survey of authentication protocol literature. `http://www.cs.york.ac.uk/~jac/papers/drareviewps.ps`. 1997.

[CKRT03a] Y. CHEVALIER, R. KÜSTERS, M. RUSINOWITCH et M. TURUANI : Deciding the security of protocols with Diffie-Hellman exponentiation and product in exponents. *In Proceedings of the 23rd Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS'03)*, volume 2914 de *Lecture Notes in Computer Science*, pages 124–135, Mumbai (India), 2003. Springer-Verlag.

[CKRT03b] Y. Chevalier, R. Küsters, M. Rusinowitch et M. Turuani : An NP decision procedure for protocol insecurity with XOR. *In Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science (LICS'03)*, pages 261–270, Ottawa (Canada), 2003. IEEE Computer Society Press.

[CKRT04] Y. Chevalier, R. Küsters, M. Rusinowitch et M. Turuani : Deciding the security of protocols with commuting public key encryption. *In Proceedings of the Workshop on Automated Reasoning for Security Protocol Analysis (ARSPA'04)*, pages 53–63, Cork (Ireland), 2004.

[CLC03] H. Comon-Lundh et V. Cortier : New decidability results for fragments of first-order logic and application to cryptographic protocols. *In Proceedings of the 14th International Conference on Rewriting Techniques and Applications (RTA'2003)*, volume 2706 de *Lecture Notes in Computer Science*, pages 148–164, Valencia (Spain), 2003. Springer-Verlag.

[CLS03] H. Comon-Lundh et V. Shmatikov : Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. *In Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science (LICS'03)*, pages 271–280, Ottawa (Canada), 2003. IEEE Computer Society Press.

[CLT03] H. Comon-Lundh et R. Treinen : Easy intruder deductions. *In Verification : Theory & Practice, Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday*, volume 2772 de *Lecture Notes in Computer Science*, pages 225–242. Springer-Verlag, 2003.

[CM96] E. Contejean et C. Marché : CiME : Completion Modulo E. *In Proceedings of the 7th International Conference on Rewriting Techniques and Applications (RTA'96)*, volume 1103 de *Lecture Notes in Computer Science*, pages 416–419, New Brunswick (New Jersey, USA), 1996. Springer-Verlag.

[Com04] H. Comon-Lundh : Intruder theories (ongoing work). *In Proceedings of the 7th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'04)*, Barcelona (Spain), 2004.

[Cor03] V. Cortier : *Vérification automatique des protocoles cryptographiques*. Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France, 2003.

[Cor05] V. Cortier : Vérifier les protocoles cryptographiques. *Technique et Science Informatiques*, 24(1):115–140, 2005.

[CR05] Y. Chevalier et M. Rusinowitch : Combining intruder theories. *In Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 de *Lecture Notes in Computer Science*, pages 639–651, Lisbon (Portugal), 2005. Springer-Verlag.

[Del06a] S. Delaune : Easy intruder deduction problems with homomorphisms. *Information Processing Letters*, 97(6):213–218, 2006.

[Del06b] S. Delaune : An undecidability result for AGh. Rapport de Recherche LSV-06-02, Laboratoire Spécification et Vérification, ENS Cachan, France, 2006. 15 pages.

[DH76] W. Diffie et M. Hellman : New directions in cryptography. *IEEE Transactions on Information Society*, 22(6):644–654, 1976.

[DJ90] N. Dershowitz et J.-P. Jouannaud : Rewrite systems. *In* Jan van Leeuwen, éditeur : *Handbook of Theoretical Computer Science*, volume B, chapitre 6. Elsevier, 1990.

[DJ04a] S. Delaune et F. Jacquemard : A decision procedure for the verification of security protocols with explicit destructors. *In Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS'04)*, pages 278–287, Washington, (D.C., USA), 2004. ACM Press.

[DJ04b] S. Delaune et F. Jacquemard : A theory of dictionary attacks and its complexity. *In Proceedings of the 17th IEEE Computer Security Foundations Workshop (CSFW'04)*, pages 2–15, Asilomar (California, USA), 2004. IEEE Computer Society Press.

[DJ06] S. Delaune et F. Jacquemard : Decision procedures for the security of protocols with probabilistic encryption against offline dictionary attacks. *Journal of Automated Reasoning*, 2006. À paraître.

[DK04] S. Delaune et F. Klay : Vérification automatique appliquée à un protocole de commerce électronique. *In Actes des 6èmes Journées Doctorales Informatique et Réseau (JDIR'04)*, pages 260–269, Lannion (France), 2004.

[DKR06] S. Delaune, S. Kremer et M. D. Ryan : Coercion-resistance and receipt-freeness in electronic voting. *In Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW'06)*, Venice (Italy), 2006. IEEE Computer Society Press. À paraître.

[DLLT05] S. Delaune, P. Lafourcade, D. Lugiez et R. Treinen : Symbolic protocol analysis in presence of a homomorphism operator and *exclusive or*. Rapport de Recherche LSV-05-20, Laboratoire Spécification et Vérification, ENS Cachan, France, 2005. 44 pages.

[DLLT06] S. Delaune, P. Lafourcade, D. Lugiez et R. Treinen : Symbolic protocol analysis in presence of a homomorphism operator and *exclusive or*. *In Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP'06)*, Lecture Notes in Computer Science, pages 132–143, Venice (Italy), 2006. Springer-Verlag. À paraître.

[DLMS99] N. Durgin, P. Lincoln, J. Mitchell et A. Scedrov : Undecidability of bounded security protocols. *In Proceedings of the Workshop on Formal Methods and Security Protocols (FMSP'99)*, Trento (Italy), 1999.

[DMR76] M. Davis, Y. Matijasevich et J. Robinson : Hilbert's tenth problem, diophantine equations : positive aspects of a negative solution. *In Proceedings of the Symposia in Pure Mathematics*, pages 323–378, 1976.

[DS81] D. Denning et G. Sacco : Timestamps in key distribution protocols. *Communications of the ACM*, 24(8):533–536, 1981.

[DY81] D. Dolev et A. C. Yao : On the security of public key protocols. *In Proceedings of the 22nd Symposium on Foundations of Computer Science (FCS'81)*, pages 350–357, Nashville (Tennessee, USA), 1981. IEEE Computer Society Press.

[EGS86] S. Even, O. Goldreich et A. Shamir : On the security of ping-pong protocols when implemented using the RSA. *In Proceedings of Advances in Cryptology*

(CRYPTO'85), volume 218 de *Lecture Notes in Computer Science*, pages 58–72, Santa Barbara (California, USA), 1986. Springer-Verlag.

[FOO92] A. Fujioka, T. Okamoto et K. Ohta : A practical secret voting scheme for large scale elections. *In Advances in Cryptology (AUSCRYPT'92)*, volume 718 de *Lecture Notes in Computer Science*, pages 244–251. Springer-Verlag, 1992.

[GK00] T. Genet et F. Klay : Rewriting for cryptographic protocol verification (extended version). *In Proceedings of the 17th International Conference on Automated Deduction (CADE'00)*, volume 1831 de *Lecture Notes in Artificial Intelligence*, Pittsburgh (Pennsylvania, USA), 2000. Springer-Verlag.

[GM84] S. Goldwasser et S. Micali : Probabilistic encryption. *Journal of Computer System Science*, 28(2):270–299, 1984.

[GP01] M. Girault et J.-C. Paillès : Contactless EP : A Public-Key Solution with Good Performances. 2001.

[Hul80a] J.-M. Hullot : Canonical forms and unification. *In Proceedings of the 5th Conference on Automated Deduction, (CADE'80)*, volume 87 de *Lecture Notes in Computer Science*, pages 318–324, Les Arcs (France), 1980. Springer-Verlag.

[Hul80b] J.-M. Hullot : A catalogue of canonical term rewriting systems. Rapport technique CSL-114, Computer Science Laboratory, SRI, Standford (California, USA), 1980.

[JRV00] F. Jacquemard, M. Rusinowitch et L. Vigneron : Compiling and verifying security protocols. *In Proceedings of the 7th International Conference on Logic for Programming and Automated Reasoning (LPAR'00)*, volume 1955 de *Lecture Notes in Computer Science*, pages 131–160, Reunion Island (France), 2000. Springer-Verlag.

[Kir85] C. Kirchner : *Méthodes et Outils de Conception Systématique d'Algorithmes d'Unification dans les Théories Équationnelles.* Thèse de doctorat, Université de Nancy I, 1985.

[KKS87] E. Kaltofen, M. S. Krishnamoorthy et B. D. Saunders : Fast parallel computation of Hermite and Smith forms of polynomial matrices. *SIAM Journal of Algebraic and Discrete Methods*, 8(4):683–690, 1987.

[KR05a] S. Kremer et M. D. Ryan : Analysis of an electronic voting protocol in the applied pi-calculus. *In Proceedings of the 14th European Symposium on Programming (ESOP'05)*, volume 3444 de *Lecture Notes in Computer Science*, pages 186–200, Edinburgh, (United Kingdom), 2005. Springer-Verlag.

[KR05b] S. Kremer et M.D. Ryan : Analysing the vulnerability of protocols to produce known-pair and chosen-text attacks. *In Proceedings of the 2nd International Workshop on Security Issues in Coordination Models, Languages and Systems (SecCo'04)*, volume 128 de *Electronic Notes in Theoretical Computer Science*, pages 84–107, London (United Kingdom), 2005. Elsevier Science Publishers.

[Laf05] P. LAFOURCADE : Intruder deduction for the equational theory of *exclusive-or* with commutative and distributive encryption. Rapport de Recherche LSV-05-21, Laboratoire Spécification et Vérification, ENS Cachan, France, 2005. 20 pages.

[LLT05a] P. LAFOURCADE, D. LUGIEZ et R. TREINEN : Intruder deduction for AC-like equational theories with homomorphisms. *In Proceedings of 16th International Conference on Rewriting Techniques and Applications (RTA'05)*, volume 3467 de *Lecture Notes in Computer Science*, pages 308–322, Nara (Japan), 2005. Springer-Verlag.

[LLT05b] P. LAFOURCADE, D. LUGIEZ et R. TREINEN : Intruder deduction for the equational theory of exclusive-or with distributive encryption. Rapport de Recherche LSV-05-19, Laboratoire Spécification et Vérification, ENS Cachan, France, 2005. 39 pages.

[LM04] C. LYNCH et C. MEADOWS : On the relative soundness of the free algebra model for public key encryption. *In Proceedings of the Workshop in Issues in the Theory of Security (WITS'04)*, pages 81–95, Barcelona (Spain), 2004.

[Low96] G. LOWE : Breaking and fixing the Needham-Schroeder public-key protocol using FDR. *In Proceedings of the 2nd International Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, volume 1055 de *Lecture Notes in Computer Science*, pages 147–166, Berlin (Germany), 1996. Springer-Verlag.

[Low97] G. LOWE : A hierarchy of authentication specifications. *In Proceedings of the 10th Computer Security Foundations Workshop (CSFW'97)*, pages 18–30, Rockport (Massachusetts, USA), 1997. IEEE Computer Society Press.

[LR97] G. LOWE et A. W. ROSCOE : Using CSP to detect errors in the TMN protocol. *IEEE Transactions on Software Engineering*, 23(10):659–669, 1997.

[McA93] D. MCALLESTER : Automatic recognition of tractability in inference relations. *Journal of the ACM*, 40(2):284–303, 1993.

[Mil03] J. MILLEN : On the freedom of decryption. *Information Processing Letters*, 86(6):329–333, 2003.

[MR00] J. MILLEN et H. RUESS : Protocol-independent secrecy. *In Proceedings of the 21st Symposium in Security and Privacy (SSP'00)*, pages 110–209, Oakland (California, USA), 2000. IEEE Computer Society Press.

[MS03] J. MILLEN et V. SHMATIKOV : Symbolic protocol analysis with products and Diffie-Hellman exponentiation. *In Proceedings of the 16th Computer Security Foundation Workshop (CSFW'03)*, pages 47–62, Pacific Grove (California, USA), 2003. IEEE Computer Society Press.

[MS05] J. MILLEN et V. SHMATIKOV : Symbolic protocol analysis with an abelian group operator or Diffie-Hellman exponentiation. *Journal of Computer Security*, 13(3):515–564, 2005.

[MW04] D. MICCIANCIO et B. WARINSCHI : Soundness of formal encryption in the presence of active adversaries. *In Prooceedings of the 1st Theory of Cryptography Conference (TCC'04)*, volume 2951 de *Lecture Notes in Computer Science*, pages 133–151, Cambridge (Massachusetts, USA), 2004. Springer-Verlag.

[Nar96] P. NARENDRAN : Solving linear equations over polynomial semirings. *In Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 466–472, New Brunswick (New Jersey), 1996. IEEE Computer Society Press.

[NPS97] P. NARENDRAN, F. PFENNING et R. STATMAN : On the unification problem for cartesian closed categories. *Journal of Symbolic Logic*, 62(2):636–647, 1997.

[NS78] R. NEEDHAM et M. SCHROEDER : Using encryption for authentification in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

[Nut90] W. NUTT : Unification in monoidal theories. *In Proceedings of the 10th International Conference on Automated Deduction, (CADE'90)*, volume 449 de *Lecture Notes in Computer Science*, pages 618–632, Kaiserslautern (Germany), 1990. Springer-Verlag.

[Pau97] L. PAULSON : Mechanized proofs for a recursive authentication protocol. *In Proceedings of the 10th Computer Security Foundations Workshop (CSFW'97)*, pages 84–95, Rockport (Massachusetts, USA), 1997. IEEE Computer Society Press.

[RS98] P. RYAN et S. SCHNEIDER : An attack on a recursive authentication protocol : A cautionary tale. *Information Processing Letters*, 65(1):7–10, 1998.

[RSA78] R. RIVEST, A. SHAMIR et L. ADLEMAN : A method for obtaining digital signatures and public-key cryptosystems. *Communication of the ACM*, 21(2):120–126, 1978.

[RT03] M. RUSINOWITCH et M. TURUANI : Protocol insecurity with a finite number of sessions, composed keys is NP-complete. *Theoretical Computer Science*, 1-3(299):451–475, 2003.

[Sch86] A. SCHRIJVER : *Theory of Linear and Integer Programming*. Wiley, 1986.

[Sch96] S. SCHNEIDER : Security properties and CSP. *In Proceedings of the Symposium on Security and Privacy*, pages 174–187, Oakland (California, USA), 1996. IEEE Computer Society Press.

[Shm04] V. SHMATIKOV : Decidable analysis of cryptographic protocols with products and modular exponentiation. *In Proceedings of the 13th European Symposium On Programming (ESOP'04)*, volume 2986 de *Lecture Notes in Computer Science*, pages 355–369, Barcelona (Spain), 2004. Springer-Verlag.

[Sim94] G. SIMMONS : Cryptoanalysis and protocol failures. *Communications of the ACM*, 37(11):56–65, 1994.

[Spo] Security protocols open repository. `http://www.lsv.ens-cachan.fr/spore/index.html`.

[SR96] V. SHOUP et A. RUBIN : Session key distribution using smart cards. *In Proceedings of the International Conference on the Theory and Application of Cryptographic Technique (EUROCRYPT'96)*, volume 1070 de *Lecture Notes in Computer Science*, pages 321–331, Saragossa (Spain), 1996. Springer-Verlag.

[THG99] F. THAYER, J. HERZOG et J. GUTTMAN : Strand spaces : Proving security protocols correct. *Journal of Computer Security*, 7(1), 1999.

[TMN89] M. Tatebayashi, N. Matsuzaki et D. B. Newman : Key distribution protocol for digital mobile communication systems. *In Proceedings of the 9th Annual International Cryptology Conference (CRYPTO'89)*, volume 435 de *Lecture Notes in Computer Science*, pages 324–333, Santa Barbara (California, USA), 1989. Springer-Verlag.

[Tra05] J. Traoré : Are blind signatures suitable for on-line voting ? (extended abstract). *In Proc. of Workshop Frontiers in Electronic Elections (FEE'05)*, Milan, Italy, 2005.

[Tur03] M. Turuani : *Sécurité des protocoles cryptographiques : décidabilité et complexité*. Thèse de doctorat, Université Henri Poincaré, Nancy (France), 2003.

[Ver03] K. N. Verma : Two-way equational tree automata for AC-like theories : Decidability and closure properties. *In Proceedings of the 14th International Conference on Rewriting Techniques and Applications (RTA'03)*, volume 2706 de *Lecture Notes in Computer Science*, pages 180–196, Valencia (Spain), 2003. Springer-Verlag.

[Vio01] E. Viola : E-unifiability via narrowing. *In Proceedings of the 7th Italian Conference on Theoretical Computer Science, (ICTCS'01)*, volume 2202 de *Lecture Notes in Computer Science*, pages 426–438, Torino (Italy), 2001. Springer-Verlag.

Index

Symboles

$=_{\mathsf{E}}$ 31
Λ 30
$\mathcal{VF}_{\mathcal{I}}$, $\mathcal{I}_{\mathcal{VF}}$ 39
$\rightsquigarrow_{\sigma}$, $\overset{*}{\rightsquigarrow}_{\sigma}$ 159
$\xrightarrow{\leq n}_{\mathsf{E}\backslash\mathcal{R}}$ 159
$\odot$ **126**
$\rightarrow_{\mathcal{R}}$, $\overset{*}{\rightarrow}_{\mathcal{R}}$, $\overset{*}{\leftrightarrow}_{\mathcal{R}}$ 32
$\rightarrow_{\mathcal{R}/\mathsf{E}}$, $\overset{*}{\rightarrow}_{\mathcal{R}/\mathsf{E}}$, $\overset{*}{\leftrightarrow}_{\mathcal{R}/\mathsf{E}}$ 32
$\rightarrow_{\mathsf{E}\backslash\mathcal{R}}$ 158
$\vec{t}$ **126**
t^{ρ} 125
$t\downarrow$, $S\downarrow$ 33
$t|_p$, $t[s]_p$ 30

A

accessible (trace) **61**
associativité 25

B

$\mathcal{B}_i(\mathcal{C})$, $\mathcal{B}(\mathcal{C})$ 127
bien formé
  rôle **44**
  système de contraintes **69**
boundedness (propriété de) **166**

C

chiffrement 22
  asymétrique 22
  involutif 79
  probabiliste 80
  symétrique 22
commutativité 25
concaténation 22
conservatrice (substitution) **116**
$\mathcal{H}$-contexte **39**
contrainte
  définissante 127
  de déduction **62**

E-convergent **32**

D

décomposé (terme) **117**
déductibilité en un pas 56, **59**
dépendant (vecteur) **127**
déterministe (rôle) 46
Diffie-Hellman (Théorie de) 29
distributivité 28
$dom(\sigma)$ 31

F

facteur **110**
facteurs-préservant (système) **130**
$Fact_{\mathsf{E}}(t)$ **110**
forme normale **33**

G

groupe abélien 26

H

head(t) 30
homomorphisme 28

I

indépendant (vecteur) **127**
initialisation (propriété) **69**
instance
  d'un rôle **43**
intrus
  actif 36, 60
  passif 36, 56

L

$L_i(\mathcal{C})$, $L(\mathcal{C})$ 127
local (système) **58**
localité 56, 110
  lemme (de) 82, 111

M

M_{E} (schéma) 109

modèle de Dolev-Yao . . . 37
monotone (système de contraintes) . . . **69**
monotonie (propriété) . . . **69**

N
$\mathcal{N}(f,t)$ . . . 144
nombre d'occurrences . . . **144**
non-effondrante (substitution) . . . **123**
non-standard (sous-terme) . . . **130**
$NSt_{\mathsf{E}}(t)$ . . . **130**

O
$\mathcal{O}(t)$, $\bar{\mathcal{O}}(t)$ . . . 30
opération $\odot$ . . . **126**
ordonnancement . . . **63**
« ou » exclusif . . . 26

P
position basique . . . 168
préfixe (propriété) . . . 27
preuve
  arbre de . . . 37
  minimale . . . **37**
  par composition . . . **81**
  par contexte . . . 39
  par décomposition . . . **81**, **111**
  taille . . . **37**
primitives
  chiffrement . . . 22
  chiffrement involutif . . . 79
  chiffrement probabiliste . . . 80
  concaténation . . . 22
problème
  3-SAT . . . **95**
  de déductibilité en un pas . . . **59**, 112
  de déduction de l'intrus . . **56**, **64**, 80, 108
  Hilbert ($10^{\text{ème}}$) . . . 144
propriété de boundedness . . . **166**
propriété de variants finis . . . **162**
propriétés algébriques
  distributivité . . . 28
  associativité . . . 25
  commutativité . . . 25
  Diffie-Hellman . . . 29
  groupe abélien . . . 26
  homomorphisme . . . 28
  « ou » exclusif . . . 26
  préfixe . . . 27
  signature en aveugle . . . 27
protocole
  Denning-Sacco . . . 96
  Needham-Shroeder . . . 65
  porte-monnaie électronique . . . 154
  TMN . . . 106
  WEP . . . 105
public-effondrant (système de réécriture) . **76**
publique-effondrante (théorie) . . . **77**

Q
$Q_{max}(\mathcal{C})$ . . . 128

R
$\mathcal{R}_{\mathsf{ACh}}$,$\mathcal{R}_{\mathsf{ACUNh}}$,$\mathcal{R}_{\mathsf{AGh}}$ . . . 33, 108
réécriture (modulo) . . . 32, 158
rôle
  bien formé . . . **44**
  déterministe . . . **46**
  dans le modèle avec égalités . . . **43**
  dans le modèle avec filtrage . . . **41**
radical . . . 32

S
séquence Λ . . . 30
3-SAT . . . **95**
schéma $\mathsf{M_E}$ . . . 109
$S_{\mathsf{E}}(t)$ . . . 121
$sig(\mathsf{E})$ . . . 31
signature en aveugle . . . 27
sous-terme
  ACh,ACUNh, AGh . . . **111**
  non-standard . . . **130**
  notion de . . . **58**
  syntaxique . . . **30**
standard (terme) . . . **110**
$St_{\mathsf{E}}(t)$ . . . **111**
stratégie de l'intérieur vers l'extérieur . . . 32
substitution
  conservatrice . . . **116**
  non-effondrante . . . **123**
  normalisée . . . **33**
  plus générale . . . 31
surréduction . . . 159
  basique . . . 168
  modulo E . . . 159
symbole de tête (d'un terme) . . . **30**
système d'inférence simple . . . 38

système de contraintes
de déduction **62**
facteurs-préservant **130**
simple avec équations **62**
système de réécriture 32
E-confluent **32**
E-convergent **32**
$\mathcal{R}_{\mathsf{ACh}}$,$\mathcal{R}_{\mathsf{ACUNh}}$,$\mathcal{R}_{\mathsf{AGh}}$ 33, 108
public-effondrant **76**

T
taille
d'un terme **30**
d'une preuve **37**
DAG d'un terme **30**
terme 30
clos 30
décomposé **117**
facteur 110
position 30
standard **110**
symbole de tête 30
taille 30
taille DAG 30
vecteur **126**
théorie équationnelle **31**
ACUNh 28
ACUN **26**
ACh 28
AGh 28
AG **26**
E_{DY} 79
publique-effondrante **77**
DH **29**
finitaire 31
régulière **159**
signature en aveugle **27**
unitaire 31
trace
accessible **61**
sans équation **60**
simple **60**
symbolique **60**

U
E-unificateur **31**
E-mgu **31**
ensemble complet **31**
plus général **31**

V
$\mathcal{V}ar(t)$ 162
variant, E-variant **161**
ensemble complet **161**
propriété de variants finis **162**
$vars(t)$ 30
vecteur
dépendant **127**
indépendant **127**