Décidabilité de la déduction et de l'équivalence statique pour le chiffrement homomorphique.

Rapport du stage de L3, ENS Cachan

Jérémy Dubut
sous la direction de Véronique Cortier
équipe CASSIS, LORIA

16 septembre 2011

Table des matières

1. Introduction

Avec le développement des échanges (communications, paiements, votes, ...) se faisant sur des réseaux à large échelle comme Internet, il est important de pouvoir garantir certaines propriétés de sécurité telles que le secret, l'anonymat, ... Les protocoles cryptographiques ont pour but d'assurer ces propriétés. C'est pourquoi, il est important de vérifier que cela est bien le cas. Néanmoins, il reste difficile de prouver que tels protocoles sont sûrs car ils sont sensibles à plusieurs types d'attaques. La première catégorie provient du fait que la méthode de chiffrement n'est pas parfaite (par exemple, un intrus avec assez de capacités de calcul pouvant déchiffrer un message sans connaitre la clé de déchiffrement). La deuxième est due à une suite d'actions (interception, analyse, envoi de messages) de l'intrus qui lui permet de briser l'une des propriétés de sécurité (par exemple, se faire passer pour quelqu'un d'autre), on appelle ces attaques des "failles logiques". Ce type de failles est très difficile à déceler car elles demandent souvent des raisonnements subtiles.

Afin de mieux comprendre, voici une attaque de ce type trouvée par G. Lowe [Low96] sur le protocole de R. Needham et M. Schroeder.
Tout d'abord, présentons le protocole :

- l'agent A envoie à l'agent B son nom et un nombre aléatoire N_a chiffrés avec la clé publique de B.
$$A \to B \ : \ \{A, N_a\}_{pub(B)}$$
- l'agent B déchiffre le message avec sa clé privée et renvoie à A, N_a et un nombre aléatoire N_b chiffrés avec la clé publique de A.
$$B \to A \ : \ \{N_a, N_b\}_{pub(A)}$$
- l'agent A déchiffre le message, reconnait son nombre N_a et renvoie à B son nombre N_b chiffré avec sa clé publique.
$$A \to B \ : \ \{N_b\}_{pub(B)}$$

Ainsi, à la fin du protocole, les agents A et B pensent être les seuls à connaitre N_b et donc pouvoir l'utiliser comme authentification.
Néanmoins, voici une attaque de type "man-in-the-middle", où un troisième agent malhonnête C peut se faire passer pour A auprès de B :

- l'agent A veut communiquer avec l'agent C :
$$A \to C \ : \ \{A, N_a\}_{pub(C)}$$
- l'agent C déchiffre ce message, le chiffre avec la clé publique de l'agent B et lui envoie le tout pour se faire passer pour l'agent A.
$$C \to B \ : \ \{A, N_a\}_{pub(B)}$$
- l'agent B pensant communiquer avec l'agent A lui envoie :
$$B \to A \ : \ \{N_a, N_b\}_{pub(A)}$$
- l'agent pensant que cette réponse provient de l'agent C répond :
$$A \to C \ : \ \{N_b\}_{pub(C)}$$
- l'agent C n'a plus qu'à récupérer N_b et à l'envoyer à l'agent B qui croit être sûr de communiquer avec l'agent A.
$$C \to B \ : \ \{N_b\}_{pub(B)}$$

A la fin du protocole, l'agent C peut s'authentifier auprès de l'agent B comme étant l'agent A.

Ainsi, savoir si un protocole est sûr semble être un problème difficile. En effet, des résultats théoriques existent lorsque les primitives (fonctions apparaissant dans le déroulement du protocole) sont relativement simples (chiffrement, signature, hash) : [MDL+99] montre que le problème général est indécidable et [RT03] que le problème reste NP-complet si l'on borne le nombre de sessions (ie le nombre d'utilisations du protocole). Néanmoins, il existe des outils de vérification automatique des protocoles comme AVISPA ([ABB+05]) et ProVerif ([Bla01]) qui fonctionnent bien en pratique et ont pu être utilisés pour analyser de nombreux protocoles. Ainsi, ProVerif a servi à vérifier des protocoles utilisés à grande échelle comme JFK ([ABF04]).

A cause des primitives utilisées, ces résultats et ces outils ne s'appliquent pas pour les protocoles de vote. En effet, ceux-ci n'utilisent pas en général les primitives les plus standards comme le chiffrement ou les signatures. Ils font souvent appel à des primitives dédiées comme les signatures en aveugle ou le chiffrement homomorphique. Pour cette raison, tous les résultats concernant la sécurité des protocoles de vote ont été établis "à la main" comme c'est le cas par exemple pour le protocole de vote en ligne Helios ([CS11]). La principale difficulté de ces protocoles est qu'ils utilisent des propriétés algébriques et qu'aucun résultat n'existe pour de telles primitives. Ainsi, le protocole de vote Helios utilise de manière cruciale une fonction de chiffrement homomorphique (dans le sens où $\{v_1\}_{pub(S)} * \{v_2\}_{pub(S)} = \{v_1 + v_2\}_{pub(S)}$). En effet, pour connaitre les résultats d'une élection, il suffit de multiplier tous les votes chiffrés puis de seulement déchiffrer le résultat à la fin.

Pour mettre au point des outils d'analyse automatique, il faut pouvoir résoudre le problème de base qui consiste à savoir quelles sont les informations que peut connaitre un intrus passif. Nous pouvons formaliser l'idée de "connaissance" de l'intrus par deux notions. La première est une notion de déduction. Intuitivement, un intrus passif intercepte un certain nombre de messages et nous voulons savoir si, à partir de ces messages, l'intrus est capable de déterminer un secret (une clé privée par exemple). Cependant, cette notion n'est pas suffisante. En effet, dans un protocole de vote, où les messages interceptés sont des votes chiffrés, il ne serait pas pertinent de définir comme sureté, le fait que l'intrus puisse déduire la valeur d'un vote car ces valeurs sont connues de tous (nom des candidats lors d'une élection, "oui" ou "non", pour un référendum). L'idée serait ici de savoir si un intrus peut "distinguer" le fait qu'un agent vote "oui" ou "non" par exemple. C'est sur ce principe que repose la deuxième notion intéressante, l'équivalence statique.

L'objet du stage s'est porté sur l'étude de ces deux notions. Nous avons montré la décidabilité de la déduction, surtout dans le cadre de primitives ayant des propriétés algèbriques plus précisément, associativité, commutativité et homomorphisme (dans le sens d'Helios) puis dans un cadre un peu plus simple, la décidabilité de l'équivalence statique. La motivation derrière ce problème, est de pouvoir travailler sur Helios, qui est l'un des seuls protocoles de vote (avec Civitas) à être réellement implémentés (et utilisable par tous sur `http://heliosvoting.org/`) et à offrir de bonnes propriétés de sécurité prouvées (comme par exemple de pouvoir vérifier son vote).

Dans la section 2, nous allons commencer par introduire les définitions utiles que nous allons illustrer par un exemple de primitives (chiffrement, déchiffrement, paire, projections) avant de présenter les résultats existants sur ces notions. Enfin, nous décrirons formellement la problématique du stage. Dans la section 3, nous démontrerons le premier résultat de décidabilité de la déduction pour le chiffrement homomorphique en utilisant les notions d'AC-convergence et de localité afin d'arriver à un algorithme NP. Enfin, dans la section 4, nous démontrerons le second résultat de décidabilité, cette fois pour l'équivalence statique, en utilisant une technique de preuve restriction-transformation afin d'arriver à un algorithme EXPTIME.

2. Enoncé du problème

2.1 Messages et termes

Pour commencer, il faut définir une représentation des messages. Nous allons les voir ici sous la forme de termes.

Définition 1 (Ensemble des termes $T(\Sigma, \mathcal{N}, \mathcal{X})$)**.**
Soient Σ, une signature (ensemble de symboles de fonctions avec arité ou primitives), $\mathcal{N}$, un ensemble de noms et $\mathcal{X}$, un ensemble de variables.
On note $T(\Sigma, \mathcal{N}, \mathcal{X})$, l'ensemble des termes sur Σ, $\mathcal{N}$ et $\mathcal{X}$, le plus petit ensemble défini par :
- $\mathcal{N} \cup \mathcal{X} \subseteq T(\Sigma, \mathcal{N}, \mathcal{X})$
- $\forall f \in \Sigma$, *d'arité n, si* M_1, ..., $M_n \in T(\Sigma, \mathcal{N}, \mathcal{X})$ *alors* $f(M_1, ..., M_n) \in T(\Sigma, \mathcal{N}, \mathcal{X})$.

Pour tout terme M, on note $fn(M)$ l'ensemble des noms de M et $fv(M)$, celui des variables.
On dit que M est clos si $fv(M) = \varnothing$.

Exemple.
Prenons $\Sigma_{enc} = \{dec(2), enc(2), < ., . > (2), \Pi_1(1), \Pi_2(1)\}$.
Intuitivement, enc est une fonction de chiffrement, dec, de déchiffrement, $< ., . >$, de création de paires et Π_1 et Π_2, de projection suivant la première ou deuxième composante d'un paire.
Si a, b et $k \in \mathcal{N}$ et $x, y \in \mathcal{X}$, alors par exemple, $\{s\}_k$ (qui représente le chiffrement d'un message s avec la clé k), $dec(< a, \Pi_1(x) >, \Pi_2(y))$ et $\Pi_2(enc(b, x))$ sont des termes.

2.2 Théories équationnelles

Les primitives ont souvent des propriétés comme par exemple quand on déchiffre un texte chiffré, on obtient le texte en clair ou bien toute sorte de proriétés algèbriques (commutativité, associativité,...). Pour représenter ces propriétés, on quotiente l'algèbre des termes par des équations, c'est-à-dire que l'on crée des classes dans lesquelles les termes correspondent au même message.

Définition 2 (Théorie équationnelle)**.**
On dit que $H = (\Sigma, E)$ est une théorie équationnelle si E est un ensemble d'équations de la forme $M_1 = M_2$ avec $fn(M_1) = fn(M_2) = \varnothing$.
Enfin, on note $=_E$, la plus petite congruence contenant E et stable par contexte et substitution de variables.

Exemple (Théorie Enc)**.**

$$\begin{aligned} \Sigma_{enc} &= \{dec(2), enc(2), < ., . > (2), \Pi_1(1), \Pi_2(1)\} \\ E_{enc} &= dec(enc(x, y), y) = x \quad (D) \\ &\quad \Pi_1(< x, y >) = x \quad (P_1) \\ &\quad \Pi_2(< x, y >) = y \quad (P_2) \end{aligned}$$

(D) représente le fait que le déchiffrement d'un texte chiffré est le texte en clair et (P_i), que la i-ième projection d'une paire correspond à la i-ième composante de la paire.
Par exemple,

$$\Pi_1(dec(enc(< a, d >, b), \Pi_2(< c, b >))) =_{E_{enc}} \Pi_1(dec(enc(< a, d >, b), b)) =_{E_{enc}} \Pi_1(< a, d >) =_{E_{enc}} a$$

2.3 Déduction

La première notion est une notion de déduction, c'est-à-dire une notion qui consiste à savoir quelles sont les informations déductibles par un intrus ayant intercepté un certain nombre de messages. Tout d'abord, il faut modéliser les connaissances de base de l'intrus :

Définition 3 (Frame)**.**
On dit que $\phi = \nu\tilde{n}.\sigma$ est une frame si $\tilde{n} \subset \mathcal{N}$ est fini et σ est une substitution de variables.

Intuitivement, la substitution de la frame représente les messages interceptés et $\tilde{n}$ les noms auxquels l'intrus n'a pas accès.

Exemple.
On reprend Σ_{enc}.
Par exemple, $\nu\{k_1\}.\{enc(a,k_1)/x, enc(b,k_2)/y, <c,b>/z\}$ est une frame.
Intuitivement, l'intrus intercepte des messages codés dont il ne connait pas la clé (x), des messages codés dont il connait la clé (y) et d'autres messages en clair (z).

Maintenant, nous pouvons définir la notion de déductibilité :

Définition 4 (Déduction)**.**
On définit les règles de déduction suivantes (avec ϕ une frame et M, un terme clos) :

$$\frac{\exists x \in dom(\sigma)\ tq\ x\sigma = M}{\phi \vdash_E M}(Ax) \qquad \frac{M \in \mathcal{N} \setminus \tilde{n}}{\nu\tilde{n}.\sigma \vdash_E M}(Nom)$$

$$\frac{\phi \vdash_E M_1 \quad ... \quad \phi \vdash_E M_l \quad f \in \Sigma \quad ar(f) = l}{\phi \vdash_E f(M_1, ..., M_l)}(F) \qquad \frac{\phi \vdash_E M \quad M =_E M'}{\phi \vdash_E M'}(E)$$

Dans cette définition, on illustre le fait qu'un intrus connaissant une frame ϕ peut déduire les messages qu'il a interceptés, les noms publics et tous les messages qu'il peut construire à partir des primitives du protocole (par exemple, il intercepte un message chiffré, construit la clé de déchiffrement et déchiffre le message avec la clé).

Exemple.
On reprend l'exemple de la théorie Enc.
Soit $\phi = \nu s, k.\{enc(s,k)/x_1, enc(<k, n>, k')/x_2\}$. Alors $\phi \vdash_{E_+} s$. En effet,

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\phi \vdash_{E_{enc}} k'}\ (Nom) \quad \dfrac{}{\phi \vdash_{E_{enc}} enc(<k,n>,k')}\ (Ax)}{\phi \vdash_{E_{enc}} <k,n>}\ (D)}{\phi \vdash_{E_{enc}} k}\ (P_1) \quad \dfrac{}{\phi \vdash_{E_{enc}} enc(s,k)}\ (Ax)}{\phi \vdash_{E_{enc}} s}\ (D)$$

Définition 5 (Problème de la déduction)**.** *On définit le problème de décision suivant :*
données : une frame ϕ et un terme clos M
question : $\phi \vdash_E M$?

L'idée du problème est de savoir si l'on peut décider si un intrus a accès à un secret.

2.4 Equivalence statique

La deuxième notion est une notion d'"indistinguabilité". La motivation derrière cette notion est la suivante : supposons qu'un intrus intercepte une frame $\phi = \nu\{k\}.\{enc(vote,k)/x\}$. La première idée serait de savoir si $\phi \vdash vote$. Cependant, le terme "vote" est souvent un nom ou une constante publique (nom d'un candidat par exemple) et donc $\phi \vdash vote$ est toujours vrai. La notion d'équivalence statique permet de résoudre ce problème en essayant de savoir si l'intrus peut distinguer ϕ de $\phi' = \nu\{k\}.\{enc(vote',k)/x\}$.

Définition 6 (Equivalence statique)**.**
Soient ϕ, une frame, M et M' deux termes. On dit que M et M' sont égaux dans ϕ, noté $(M =_E M')\phi$, si $\phi = \nu\tilde{n}.\sigma$, $(fn(M) \cup fn(M')) \cap \tilde{n} = \varnothing$ et $M\sigma =_E M'\sigma$.
On note $Eq(\phi) = \{(M, M')\ tq\ (M =_E M')\phi\}$.
On dit que deux frames ϕ et ϕ' sont statiquement équivalentes, noté $\phi \approx_E \phi'$, si
- *$dom(\phi) = dom(\phi')$*
- *$Eq(\phi) = Eq(\phi')$.*

On note $\phi' \models_E K$ si $K \subseteq Eq(\phi')$.

L'idée est de savoir si un intrus peut distinguer deux séries de messages (des séries de votes chiffrés pour le protocole de vote par exemple) juste en en construisant d'autres et en les comparant.

Exemple.
On reprend l'exemple de la théorie Enc.
- *On pose* $\phi = \nu k.\{enc(s_0, k)/x_1, k/x_2\}$ *et* $\phi' = \nu k.\{enc(s_1, k)/x_1, k/x_2\}$.
  Alors, $\phi \not\approx_E \phi'$ *car* $(dec(x_1, x_2), s_0) \in Eq(\phi) \setminus Eq(\phi')$.
  L'idée ici est que l'intrus a juste besoin de déchiffrer le premier message à l'aide du deuxième pour se rendre compte que les frames sont différentes.
- *Par contre si on prend* $\phi = \nu k.\{enc(s_0, k)/x_1\}$ *et* $\phi' = \nu k.\{enc(s_1, k)/x_1\}$ *alors* $\phi \approx_E \phi'$.
  L'idée est que l'intrus n'a aucun moyen de récupérer s_0 *ou* s_1 *à partir de* x *donc de distinguer les deux frames.*

Définition 7 (Problème de l'équivalence statique)**.** *On définit le problème de décision suivant :*
données : ϕ *et* ϕ'*, deux frames*
question : $\phi \approx_E \phi'$ *?*

2.5 Résultats existants

Des résultats d'indécidabilité et de décidabilité existent déjà pour ces deux problèmes.

2.5.1 Indécidabilité

Théorème 1 ([AC06])**.**
- *Le problème de la déduction peut être indécidable.*
- *Le problème de la déduction pour* E *sur* Σ *se réduit au problème de l'équivalence statique pour* E *sur* $\Sigma \uplus \{h(1)\}$.
- *Le problème de l'équivalence statique peut-être indécidable.*

2.5.2 Combinaison

Théorème 2 ([CD11])**.**
Soient deux théories équationnelles E_1 *sur* Σ_1 *et* E_2 *sur* Σ_2 *telles que* Σ_1 *et* Σ_2 *sont disjoints. Alors :*
- *Si le problème de la déduction pour* E_i *sur* Σ_i *est décidable alors celui pour* $E_1 \cup E_2$ *sur* $\Sigma_1 \cup \Sigma_2$ *l'est aussi.*
- *Si le problème de la déduction et le problème de l'équivalence statique pour* E_i *sur* Σ_i *sont décidables alors le problème de l'équivalence statique pour* $E_1 \cup E_2$ *sur* $\Sigma_1 \cup \Sigma_2$ *l'est aussi.*

2.5.3 Théories sous-termes convergentes

Définition 8 (Théorie sous-terme convergente)**.**
Une théorie équationnelle $E = \bigcup\{M_i = N_i\}$ *est dite sous-terme convergente si le système de réécriture* $\mathcal{R} = \bigcup\{M_i \to N_i\}$ *est convergent et si* $\forall i$, N_i *est un sous-terme propre de* M_i *ou une constante (primitive d'arité 0).*

Exemple.
La théorie E_{enc} *précédente est sous-terme convergente.*

Théorème 3 ([AC06])**.**
Les problèmes de la déduction et de l'équivalence statique sont décidables pour les théories sous-termes convergentes (PTIME).

2.5.4 Théories monoïdales

Définition 9 (Théorie monoïdale)**.**
Une théorie E *sur* Σ *est dite monoïdale si :*
- $\Sigma = \{0(0), +(2), h_1(1), ..., h_n(1)\}$
- $AC \subset E$
- $x + 0 = 0\ (U) \in E$
- $\forall i$, $\{h_i(x + y) = h_i(x) + h_i(y), h_i(0) = 0\} \subset E$

Exemple.

ACU :
- $\Sigma = \{0(0), +(2)\}$
- $E = AC \cup U$

ACUN (ou exclusif) :
- $\Sigma = \{0(0), +(2)\}$
- $E = ACU \cup \{x + x = 0(N)\}$

ACUI :
- $\Sigma = \{0(0), +(2)\}$
- $E = ACU \cup \{x + x = x(I)\}$

AG (groupe abélien) :
- $\Sigma = \{0(0), +(2), -(1)\}$
- $E = ACU \cup \{x + -(x) = 0(Inv)\}$ *(les propriétés d'homomorphisme de $-$ se déduisent de ces équations)*

ACUh, ACUNh, ACUIh :
- $\Sigma_h = \{0(0), +(2), h(1)\}$
- $E_h = E \cup \{h(x + y) = h(x) + h(y), h(0) = 0\}$

AG(h_1,...,h_n) :
- $\Sigma_h = \{0(0), +(2), h_1(1), ..., h_n(1)\}$
- $E_h = E \cup \left(\bigcup_i \{h_i(x + y) = h_i(x) + h_i(y), h_i(0) = 0\}\right) \cup \left(\bigcup_{i \neq j} \{h_i(h_j(x)) = h_j(h_i(x))\}\right)$

De nombreux résultats existent déjà pour la déduction et l'équivalence statique de ces théories ([CD11]), obtenus par des arguments algébriques. En voici un récapitulatif :

Théorie	Déduction	Equivalence statique
ACU	NP-complet	PTIME
ACUN	PTIME	PTIME
ACUI	décidable	décidable
AG	PTIME	PTIME
ACUh	NP-complet	décidable
ACUNh	PTIME	décidable
ACUIh	décidable	?
AGh	PTIME	décidable
AG(h_1,...,h_n)	décidable	décidable

En particulier, nous utiliserons dans la suite le théorème 2 et la NP-complètude de la déduction pour ACU.

2.6 Exposé du problème

Notre but est d'étudier les problèmes de la déduction et de l'équivalence statique dans le cadre d'Helios. Ce protocole de vote utilise un chiffrement El Gamal dont voici la description :
- on considère un groupe cyclique de cardinal q et de générateur g
- les clés privées sont les éléments de $\{0, ..., q-1\}$ et les clés publiques les g^x avec $x \in \{0, ..., q-1\}$
- $\{v\}_{g^x} = (g^r, g^{v+x*r})$ avec r aléatoire

On remarque que ce chiffrement a la propriété d'homomorphisme :

$$\{v_1\}_{g^x} * \{v_2\}_{g^x} = (g^{r_1} * g^{r_2}, g^{v_1+x*r_1} g^{v_2+x*r_2}) = (g^{r_1+r_2}, g^{v_1+v_2+x*(r_1+r_2)}) = \{v_1 + v_2\}_{g^x}$$

(ou par exemple v_1 et v_2 sont des votes) et c'est ce qui motive l'étude de cette théorie équationnelle :

Définition 10 (Théorie DHAC)**.**

$$\begin{aligned}
\Sigma &= \{\{.\}_.(2), dec(2), pub(1), *(2), +(2)\} \\
E &= dec(\{x\}_{pub(y)}, y) = x \quad (D) \\
&\quad \{x_1\}_{pub(y)} * \{x_2\}_{pub(y)} = \{x_1 + x_2\}_{pub(y)} \quad (H) \\
&\quad x + (y + z) = (x + y) + z \quad (A_+) \\
&\quad x + y = y + x \quad (C_+) \\
&\quad x * (y * z) = (x * y) * z \quad (A_*) \\
&\quad x * y = y * x \quad (C_*)
\end{aligned}$$

En plus de la propriété d'homomorphisme du chiffrement, on ajoute les propriétés algèbriques de $+$ et $*$, l'associativité et la commutativité.

La première partie du stage consistera à prouver que le problème de la déduction est décidable pour la théorie DHAC (NPTIME).

Le problème de l'équivalence statique étant plus complexe nous allons étudier une théorie plus simple :

Définition 11 (Théorie HAC_{+*})**.**

$$\begin{aligned}
\Sigma &= \{*(2), +(2), h(1)\} \\
E &= h(x_1) * h(x_2) = h(x_1 + x_2) \quad (H) \\
&\quad x + (y + z) = (x + y) + z \quad (A_+) \\
&\quad x + y = y + x \quad (C_+) \\
&\quad x * (y * z) = (x * y) * z \quad (A_*) \\
&\quad x * y = y * x \quad (C_*)
\end{aligned}$$

Cette théorie est intuitivement DHAC dans laquelle on ne pourrait utiliser qu'une seule clé publique dont on ne connaitrait pas la clé privée associée.

La deuxième partie de ce stage sera de démontrer que le problème de l'équivalence statique est décidable pour HAC_{+*} (EXPTIME).

3. Décidabilité du problème de la déduction pour DHAC

La preuve de décidabilité de la déduction est ici très proche de ce qui a été fait dans [CT03], [BCD07], [Laf06] et [Del06] : nous allons introduire un nouveau système de déduction équivalent à celui de départ mais dans lequel on peut trouver une preuve (si elle existe) ne faisant intervenir que des "sous-termes" (localité), avec deux bonnes notions de "sous-termes". La première nous permettra d'avoir un algorithme EXPTIME procédant par saturation et la deuxième, un algorithme NPTIME, qui consiste à se ramener à la déduction d'une théorie bien connue (AC).

3.1 Système de réécriture et règles de déduction normalisées

La première idée est de normaliser les termes pour un bon système de réécriture. Le problème étant que l'associativité et la commutativité ne permettent pas de trouver un système confluent. On va donc utiliser la notion de réécriture modulo AC (cf [Del06]).

Définition 12 (Réécriture modulo AC)**.**
Soit R, un système de réécriture.
La réécriture modulo AC est la relation $\rightarrow_{R/AC}$ définie par : $s \rightarrow_{R/AC} t$ s'il existe un contexte C, une règle $l \rightarrow r$ de R et σ une substitution tels que $s =_{AC} C[l\sigma]$ et $t =_{AC} C[r\sigma]$ où $=_{AC}$ est $=_{E'}$ avec $E' = \{A_+, C_+, A_*, C_*\}$ et on note $\overline{M} = \{N \ tq \ M =_{AC} N\}$, la classe de M modulo AC.
Soit $\rightarrow^*_{R/AC}$ la clôture transitive et reflexive modulo AC (ie $s \rightarrow^0_{R/AC} t$ ssi $s =_{AC} t$) de $\rightarrow_{R/AC}$. Une forme normale de s est alors un terme t tel que $s \rightarrow^*_{R/AC} t$ et $t \not\rightarrow_{R/AC}$.

Dans notre cas, nous allons considérer le système de réécriture HD suivant (il consiste juste à orienter les équations D et H) :
- $dec(\{x\}_{pub(y)}, y) \rightarrow x$
- $\{x_1\}_{pub(y)} * \{x_2\}_{pub(y)} \rightarrow \{x_1 + x_2\}_{pub(y)}$.

Définition 13 (AC-convergence)**.**
Soit R/AC un système de réécriture modulo AC.
On dit que R/AC est AC-confluent si pour tout terme s, si $s \rightarrow^*_{R/AC} t_1$ et $s \rightarrow^*_{R/AC} t_2$ alors il existe t'_1 et t'_2 tels que $t_1 \rightarrow^*_{R/AC} t'_1$, $t_2 \rightarrow^*_{R/AC} t'_2$ et $t'_1 =_{AC} t'_2$. On dit qu'il est AC-convergent si, de plus, $\rightarrow_{R/AC}$ termine.

C'est la notion dont nous avons besoin pour pouvoir parler de la forme normale d'un terme modulo AC. En effet, on appellera alors la forme normale d'un terme, l'ensemble de ses formes normales ce qui est par AC-convergence la classe d'équivalence modulo AC de l'une de ses formes normales pour $\rightarrow_{R/AC}$.

Lemme 1.
HD/AC est AC-convergent.

On notera alors $\overline{M}_\downarrow$, la forme normale de M et $\overline{\phi}_\downarrow$ pour $\nu\tilde{n}.\{\overline{M_1}_\downarrow/x_1, ..., \overline{M_n}_\downarrow/x_n\}$.

Maintenant, nous pouvons définir un nouveau système de déduction en normalisant les règles de déduction initiales, ce qui nous permettra de ne considérer que des formes normales dans les preuves et de supprimer la règle (E).

Définition 14.
On définit les règles de déduction suivantes :

$$\frac{\exists x \in dom(\overline{\sigma}_\downarrow) \ tq \ x\overline{\sigma}_\downarrow = \overline{M}}{\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow} \qquad \frac{m \in \mathcal{N} \setminus \tilde{n}}{\overline{\phi}_\downarrow \vdash_2 \overline{m}_\downarrow}$$

$$\frac{\overline{\phi}_\downarrow \vdash_2 \overline{M_1} \ \ ... \ \ \overline{\phi}_\downarrow \vdash_2 \overline{M_l} \ \ f \in \Sigma \ \ ar(f) = l}{\overline{\phi}_\downarrow \vdash_2 \overline{f(M_1, ..., M_l)}_\downarrow}$$

Propriété 1 (Equivalence des systèmes de déductions)**.**
$\vdash_2$ *a les propriétes suivantes :*
(i) si $\overline{\phi}_\downarrow \vdash_2 \overline{M}$ *alors* $\overline{M}_\downarrow = \overline{M}$.
(ii) $\phi \vdash_E M$ *ssi* $\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow$.

Démonstration.
(i) évident par la forme des conclusions des règles.
(ii) ($\Leftarrow$) évident
($\Rightarrow$) par récurrence sur la taille de la preuve de $\phi \vdash_E M$ et distiction de cas sur la dernière règle utilisée :

- si $\dfrac{\exists x \in dom(\sigma)\ tq\ x\sigma = M}{\phi \vdash_E M}$ alors $\dfrac{\exists x \in dom(\overline{\sigma}_\downarrow)\ tq\ x\overline{\sigma}_\downarrow = \overline{M}_\downarrow}{\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow}$
- si $\dfrac{m \in \mathcal{N} \setminus \tilde{n}}{\phi \vdash_E m}$ alors c'est évident.
- si $\dfrac{\phi \vdash_E M_1 \ \ ... \ \ \phi \vdash_E M_l \ \ f \in \Sigma \ \ ar(f) = l}{\phi \vdash_E f(M_1, ..., M_l)}$ alors par HR, $\overline{\phi}_\downarrow \vdash_2 \overline{M_1}_\downarrow, ..., \overline{\phi}_\downarrow \vdash_2 \overline{M_l}_\downarrow$ donc $\overline{\phi}_\downarrow \vdash_2 \overline{f(M_1', ..., M_l')}_\downarrow$ avec $M_i' \in \overline{M_i}_\downarrow$, et par AC-confluence, $\overline{f(M_1', ..., M_l')}_\downarrow = \overline{f(M_1, ..., M_l)}_\downarrow$.
- si $\dfrac{\phi \vdash_E M \ \ M =_E M'}{\phi \vdash_E M'}$ alors par HR, $\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow$ et par AC-confluence, $\overline{M}_\downarrow = \overline{M'}_\downarrow$.

□

3.2 Lemme de localité et décidabilité

Maintenant, on va définir deux notions de "sous-termes", dans l'idée de considérer des preuves dans notre nouveau système de déduction n'utilisant que des "sous-termes".

Définition 15 ("Sous-termes")**.**
On définit les deux ensembles suivants :
- $S_1(\overline{M}) = S_1'(\overline{M}) \cup S_1''(\overline{M})$ *avec :*
  - $S_1'(\overline{M}) = \{\overline{M'}\ tq\ M'\ est\ un\ sous-terme\ syntaxique\ de\ M'' \in \overline{M}\}$
  - $S_1''(\overline{M}) = \{\overline{\{a\}_b} \mid \exists c\ tq\ \overline{\{a+c\}_b} \in S_1'(\overline{M})\}$

  S_1 *est donc l'ensemble des sous-termes syntaxiques auxquels on ajoute les termes de la forme* $\overline{\{a\}_b}$ *si* $\overline{\{a+c\}_b}$ *est un sous-terme car* $\overline{\{a\}_b}$ *est un sous-terme de* $\overline{\{a\}_b * \{c\}_b} =_E \overline{\{a+c\}_b}$.
- $S_2(\overline{M}) = \overline{M} \cup S_2'(\overline{M})$ *avec* $S_2'(\overline{M})$ *définit comme suit :*
  - *si* $\overline{M}$ *est un nom,* $S_2'(\overline{M}) = \varnothing$
  - *si* $\overline{M} = \overline{dec(M_1, M_2)}$, $S_2'(\overline{M}) = S_2(\overline{M_1}) \cup S_2(\overline{M_2})$
  - *si* $\overline{M} = \overline{\{M_1\}_{M_2}}$, $S_2'(\overline{M}) = S_2(\overline{M_1}) \cup S_2(\overline{M_2})$
  - *si* $\overline{M} = \overline{\sum_i M_i}$ *avec* M_i *qui n'est pas une somme,* $S_2'(\overline{M}) = \bigcup_i S_2(\overline{M_i})$
  - *si* $\overline{M} = \overline{\prod_i M_i}$ *avec* M_i *qui n'est pas un produit,* $S_2'(\overline{M}) = \bigcup_i S_2(\overline{M_i})$

  S_2 *est ce qui est appelé ensemble des sous-termes dans [CD11], c'est-à-dire l'ensemble des sous-termes aliens (un sous-terme syntaxique est sous-terme si son symbole de tête est différent de celui juste au dessus dans le terme).*

On notera $S_i(\overline{\phi}_\downarrow)$ *pour* $\bigcup_{M_j \in \sigma} S_i(\overline{M_j}_\downarrow)$

On remarque que ces ensembles sont stables, c'est-à-dire que $S_i(S_i(\overline{M})) = S_i(\overline{M})$.

Exemple.
Prenons $M = \{(a+b)+c\}_{pub(d)}$.

$$\begin{aligned} S_1(\overline{M}) &= \{\overline{\{(a+b)+c\}_{pub(d)}}, \overline{\{a+b\}_{pub(d)}}, \overline{\{b+c\}_{pub(d)}}, \overline{\{a+c\}_{pub(d)}}, \overline{\{a\}_{pub(d)}}, \overline{\{b\}_{pub(d)}}, \overline{\{c\}_{pub(d)}}, \overline{(a+b)+c}, \\ &\quad \overline{a+b}, \overline{b+c}, \overline{a+c}, \overline{a}, \overline{b}, \overline{c}, \overline{pub(d)}, \overline{d}\} \\ S_2(\overline{M}) &= \{\overline{\{(a+b)+c\}_{pub(d)}}, \overline{(a+b)+c}, \overline{a}, \overline{b}, \overline{c}, \overline{pub(d)}, \overline{d}\} \end{aligned}$$

Définition 16 (Preuve S-locale)**.**
Soit S un ensemble de termes.
On dit qu'une preuve est S-locale si tous les termes apparaissant dans la preuve appartiennent à S.

On en arrive donc au lemme principal duquel on déduira la décidabilité de la déduction.

Lemme 2 (de localité)**.**
Si $\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow$ *alors il en existe une preuve* P*, Sat-locale avec* $Sat = S_1(\overline{M}_\downarrow) \cup S_1(\overline{\phi}_\downarrow)$ *et telle que pour toute sous-preuve de* P *si la dernière règle est de la forme :*

$$\frac{\overline{\phi}_\downarrow \vdash_2 \overline{\{u\}_{pub(v)}} \quad \overline{\phi}_\downarrow \vdash_2 \overline{v}}{\overline{\phi}_\downarrow \vdash_2 \overline{u}}$$

alors elle est $S_1(\overline{\phi}_\downarrow)$*-locale et* $\overline{\{u\}_{pub(v)}} \in S_2(\overline{\phi}_\downarrow)$*.*

Démonstration.
Soit P, une preuve de $\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow$. On applique la transformation t suivante :

- si P est de la forme :

$$\frac{\dfrac{\dfrac{P_u}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u}}} \quad \dfrac{P_{pub(v)}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{pub(v)}}}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u\}_{pub(v)}}}} \quad \dfrac{P_v}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{v}}}}{\overline{\phi}_\downarrow \vdash_2 \overline{u}}$$

  alors on remplace par $t\left(\dfrac{P_u}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u}}}\right)$.

- si P est de la forme :

$$\frac{\dfrac{\dfrac{P_1}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_1\}_{pub(v)}}}} \quad \dfrac{P_2}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_2\}_{pub(v)}}}}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_1+u_2\}_{pub(v)}}}} \quad \dfrac{P_v}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{v}}}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u_1+u_2}}}$$

  on remplace par :

$$\frac{t\left(\dfrac{P_1'}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u_1}}}\right) \quad t\left(\dfrac{P_2'}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u_2}}}\right)}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u_1+u_2}}} \quad avec \quad P_i' = \frac{\dfrac{\dfrac{P_i}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_i\}_{pub(v)}}}} \quad \dfrac{P_v}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{v}}}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{u_i}}}}{}$$

- sinon P est de la forme :

$$\frac{\dfrac{P_1}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{M_1}}} \quad ... \quad \dfrac{P_n}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{M_n}}}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{M}}}$$

  alors on remplace par :

$$\frac{t\left(\dfrac{P_1}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{M_1}}}\right) \quad ... \quad t\left(\dfrac{P_n}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{M_n}}}\right)}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{M}}}$$

Cette transformation termine car on l'applique récursivement à des preuves strictement plus petite (en nombre de termes).
Montrons maintenant par récurrence sur la taille de la preuve et distinction de cas suivant la dernière règle utilisée dans P' obtenue après transformation :

- si la dernière règle est un axiome, c'est évident
- si la dernière règle est une application de fonction sans pas de réduction, on utilise juste l'hypothèse de récurrence et la définition de S_1.
- si la dernière règle est :

$$\frac{\dfrac{P_1}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_1\}_{pub(v)}}}} \quad \dfrac{P_2}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_2\}_{pub(v)}}}}}{\overline{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_1+u_2\}_{pub(v)}}}}$$

  alors on conclut toujours par récurrence et définition de S_1.

– le cas intéressant est si la dernière règle est :

$$\frac{\overline{\phi}_\downarrow \vdash_2 \overline{\{u\}_{pub(v)}} \quad \overline{\phi}_\downarrow \vdash_2 \overline{v}}{\overline{\phi}_\downarrow \vdash_2 \overline{u}}$$

Regardons les différents cas selon la dernière règle de la preuve de $\overline{\phi}_\downarrow \vdash_2 \overline{\{u\}_{pub(v)}}$:
– soit c'est un axiome et c'est OK.
– soit c'est $\frac{\overline{\phi}_\downarrow \vdash_2 \overline{u} \quad \overline{\phi}_\downarrow \vdash_2 \overline{pub(v)}}{\overline{\phi}_\downarrow \vdash_2 \overline{\{u\}_{pub(v)}}}$ et c'est impossible par construction.
– soit c'est $\frac{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_1\}_{pub(v)}} \quad \overline{\phi}_\downarrow \vdash_2 \overline{\{u_2\}_{pub(v)}}}{\overline{\phi}_\downarrow \vdash_2 \overline{\{u_1+u_2\}_{pub(v)}}}$ et c'est impossible par construction.
– soit c'est $\frac{\overline{\phi}_\downarrow \vdash_2 \overline{\{\{u\}_{pub(v)}\}_{pub(w)}} \quad \overline{\phi}_\downarrow \vdash_2 \overline{w}}{\overline{\phi}_\downarrow \vdash_2 \overline{\{u\}_{pub(v)}}}$. Alors, par hypothèse de récurrence cette preuve est $S_1(\overline{\phi}_\downarrow)$-locale et $\overline{\{\{u\}_{pub(v)}\}_{pub(w)}} \in S_2(\overline{\phi}_\downarrow)$ donc $\overline{\{u\}_{pub(v)}}$ est dans $S_2(\overline{\phi}_\downarrow)$ et $\overline{u}$ et $\overline{v}$ sont dans $S_1(\overline{\phi}_\downarrow)$. Ainsi, par hypothèse de récurrence, la preuve de $\overline{\phi}_\downarrow \vdash_2 \overline{v}$ est $S_1(\overline{\phi}_\downarrow)$-locale et donc P' aussi.

□

On en déduit :

Théorème 4 (Décidabilité de la déduction)**.**
Le problème de la déduction pour la théorie DHAC est décidable (EXPTIME).

Démonstration.
On décide $\overline{\phi}_\downarrow \vdash_2 \overline{M}_\downarrow$ en construisant par saturation l'ensemble des termes déductibles appartenant à Sat. Par le lemme de localité, cet algorithme est correct et comme Sat peut être de taille exponentielle, on a un algorithme de décision en EXPTIME. □

3.3 Algorithme de déduction NP

En réalité dans l'algorithme, on n'utilise que la première partie du lemme de localité. Dans cette partie, on va encore améliorer le système de déduction afin de pouvoir démontrer un nouveau lemme de localité plus puissant, nous permettant ainsi d'avoir un algorithme NP.

Définition 17.
On définit les règles de déduction suivantes :

$$\frac{\exists x \in dom(\overline{\sigma}_\downarrow)\ tq\ x\overline{\sigma}_\downarrow = \overline{M}}{\overline{\phi}_\downarrow \vdash_3 \overline{M}_\downarrow}(Ax) \qquad \frac{m \in \mathcal{N} \setminus \tilde{n}}{\overline{\phi}_\downarrow \vdash_3 \overline{m}_\downarrow}(Nom) \qquad \frac{\overline{\phi}_\downarrow \vdash_3 \overline{M}}{\overline{\phi}_\downarrow \vdash_3 \overline{pub(M)}}(Pub)$$

$$\frac{\overline{\phi}_\downarrow \vdash_3 \overline{M_1} \quad \overline{\phi}_\downarrow \vdash_3 \overline{M_2}}{\overline{\phi}_\downarrow \vdash_3 \overline{\{M_1\}_{M_2}}}(Cod) \qquad \frac{\overline{\phi}_\downarrow \vdash_3 \overline{M_1} \quad \overline{\phi}_\downarrow \vdash_3 \overline{M_2}}{\overline{\phi}_\downarrow \vdash_3 \overline{dec(M_1,M_2)}_\downarrow}(Dec) \qquad \frac{\overline{\phi}_\downarrow \vdash_3 \overline{M_1} \cdots \overline{\phi}_\downarrow \vdash_3 \overline{M_n}}{\overline{\phi}_\downarrow \vdash_3 \overline{M_1+...+M_n}}(+)$$

$$\frac{\overline{\phi}_\downarrow \vdash_3 \overline{M_1} \cdots \overline{\phi}_\downarrow \vdash_3 \overline{M_n}}{\overline{\phi}_\downarrow \vdash_3 \overline{M_1 * ... * M_n}_\downarrow}(*) \qquad \frac{\overline{\phi}_\downarrow \vdash_3 \overline{\{M_1\}_{pub(N)}} \cdots \overline{\phi}_\downarrow \vdash_3 \overline{\{M_n\}_{pub(N)}}}{\overline{\phi}_\downarrow \vdash_3 \overline{\{M_1+...+M_n\}_{pub(N)}}}(combi1)$$

$$\frac{\overline{\phi}_\downarrow \vdash_3 \overline{pub(N)} \quad \overline{\phi}_\downarrow \vdash_3 \overline{M_1} \cdots \overline{\phi}_\downarrow \vdash_3 \overline{M_n} \quad \overline{\phi}_\downarrow \vdash_3 \overline{\{M_{n+1}\}_{pub(N)}} \cdots \overline{\phi}_\downarrow \vdash_3 \overline{\{M_m\}_{pub(N)}}}{\overline{\phi}_\downarrow \vdash_3 \overline{\{M_1+...+M_m\}_{pub(N)}}}(combi2)$$

Propriété 2.
$\vdash_3$ *est équivalent à* $\vdash_2$.

Lemme 3 (de localité 2)**.**
Si $\overline{\phi}_\downarrow \vdash_3 \overline{M}_\downarrow$ *alors il en existe une preuve Sat-locale avec* $Sat = S_2(\overline{M}_\downarrow) \cup S_2(\overline{\phi}_\downarrow)$.

Démonstration.
Soit P la preuve de $\overline{\phi}_{\downarrow} \vdash_2 \overline{M}_{\downarrow}$ du premier lemme de localité. On remplace les $\vdash_2$ par des $\vdash_3$ puis on effectue les transformations suivantes :

$$\frac{\overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1}} \ ... \ \frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_j}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i + ... + M_j}}(+) \ ... \ \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 + ... + M_n}}(+)$$

$$\Downarrow$$

$$\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 + ... + M_n}}(+)$$

$$\frac{\overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1}} \ ... \ \frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_j}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i * ... * M_{j\downarrow}}}(*) \ ... \ \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 * ... * M_{n\downarrow}}}(*)$$

$$\Downarrow$$

$$\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 * ... * M_{n\downarrow}}}(*)$$

$$\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_i} = \overline{\{M_i'\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_j} = \overline{\{M_j'\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 * ... * M_{n\downarrow}}}(*)$$

$$\Downarrow$$

$$\frac{\overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1}} \ ... \ \frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_i'\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_j'\}_{pub(N)}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_i' + ... + M_j'\}_{pub(N)}}}(combi1) \ ... \ \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 * ... * M_{n\downarrow}}}(*)$$

$$\frac{\overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1\}_{pub(N)}}} \ ... \ \frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_i\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_j\}_{pub(N)}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_i + ... + M_j\}_{pub(N)}}}(combi1 \text{ ou } *) \ ... \ \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_n\}_{pub(N)}}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_n\}_{pub(N)}}}(combi1)$$

$$\Downarrow$$

$$\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_n\}_{pub(N)}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_n\}_{pub(N)}}}(combi1)$$

idem si on avait un (combi2) à la place de (combi1) ou $(*)$ sauf que l'on passe en (combi2)

$$\frac{\overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1\}_{pub(N)}}} \ ... \ \frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{pub(N)}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_i\}_{pub(N)}}}(Cod) \ ... \ \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_n\}_{pub(N)}}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_n\}_{pub(N)}}}(combi1)$$

$$\Downarrow$$

$$\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{pub(N)} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_i} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_n\}_{pub(N)}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_n\}_{pub(N)}}}(combi2)$$

idem pour (combi2)

$$\frac{\overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{pub(N)}} \quad \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1}} ... \frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_j}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i + ... + M_j}} ... \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_m}} \quad \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_{m+1}\}_{pub(N)}}} ... \overline{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_n\}_{pub(N)}}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_n\}_{pub(N)}}}(combi2)$$

$$\Downarrow$$

$$\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{pub(N)} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{M_1} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_m} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_{m+1}\}_{pub(N)}} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_n\}_{pub(N)}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_n\}_{pub(N)}}}(combi2)$$

Soit P', une preuve de $\overline{\phi}_{\downarrow} \vdash_3 \overline{M}_{\downarrow}$ obtenue en appliquant les transformations sur P tant que cela est possible. On remarque que cette transformation conserve les propriétés de P, surtout que si $\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{u\}_{pub(v)}} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{v}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{u}}(Dec)$ apparait dans P', alors $\overline{\{u\}_{pub(v)}} \in S_2(\overline{\phi}_{\downarrow})$.
Montrons alors par récurrence sur la taille de la preuve et distinction de cas que P' convient.

- (Ax) et (Nom) sont évidents
- (Pub), (Cod) et (Dec) sans pas de réduction sont de simples applications de l'hypothèse de récurrence
- (Dec) avec pas de réduction est une conséquence de la remarque pécédente
- $(+)$: $\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1} \ ... \ \overline{\phi}_{\downarrow} \vdash_3 \overline{M_n}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{M_1 + ... + M_n}}(+)$. Montrons que $\forall i$, $\overline{M_i} \in Sat$. Distinguons les cas de la dernière de la preuve de $\overline{\phi}_{\downarrow} \vdash_3 \overline{M_i}$:
  - (Ax) et (Nom) sont évidents
  - $(+)$ est impossible par construction
  - (Dec) avec pas de réduction, par la remarque, $\overline{M_i} \in S_2(\overline{\phi}_{\downarrow})$
  - toute autre règle implique que $\overline{M_i} \in S_2(\overline{M_1 + ... + M_n})$

  On conclut alors par récurrence.
- $(*)$: par construction, on ne peut pas avoir de pas de réduction, donc ce cas est analogue au $(+)$
- (combi2) ((combi1) est similaire) :
  $\frac{\overline{\phi}_{\downarrow} \vdash_3 \overline{pub(N)} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{M_1} \cdots \overline{\phi}_{\downarrow} \vdash_3 \overline{M_n} \quad \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_{n+1}\}_{pub(N)}} \cdots \overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_m\}_{pub(N)}}}{\overline{\phi}_{\downarrow} \vdash_3 \overline{\{M_1 + ... + M_m\}_{pub(N)}}}(combi2)$.
  On montre de manière analogue à $(+)$ que $\overline{pub(N)}$, $\overline{M_i}$ et $\overline{\{M_j\}_{pub(N)}}$ sont dans Sat et on conclue par récurrence.

□

Théorème 5.
Le problème de la déduction pour la théorie DHAC est décidable (NPTIME).

Démonstration.
On suppose que l'on a une fonction $deduc_{AC}$ qui décide le problème de la déduction pour la théorie AC suivante :

- $\Sigma = \{+(2)\}$
- $E = \{x + y = y + x, (x + y) + z = x + (y + z)\}$

On sait que ce problème est NP-complet donc on peut supposer que $deduc_{AC}$ est NP.
Soient $transformer_+$ et $transformer_*$, deux fonctions qui transforment un terme de DHAC en un de AC de cette façon :

$transformer_+(M)$: – si $\overline{M} = \overline{M_1 + ... + M_n}$ avec M_i qui n'est pas une somme alors on renvoie $a_{M_1} + ... + a_{M_n}$ avec $a_{M_i} \in \mathcal{N}$
– sinon a_M

$transformer_*(M)$: – si $\overline{M} = \overline{M_1 * ... * M_n}$ avec M_i qui n'est pas un produit alors on renvoie $a_{M_1} + ... + a_{M_n}$ avec $a_{M_i} \in \mathcal{N}$
– sinon a_M

Maintenant, considérons l'algorithme suivant :

- on initialise ded à $\{\overline{M_i}_{\downarrow}\} \cup \{n \in (fn(M) \cup fn(\phi)) \setminus \tilde{n}\}$
- on initialise S à $S_2(\overline{M}_{\downarrow}) \cup S_2(\overline{\phi}_{\downarrow}) \setminus ded$
- tant que l'on peut ajouter des termes de S à ded alors pour tout $\overline{t} \in S$ si l'on se trouve dans l'un des cas suivants :
  (1) si $\exists \overline{N} \in ded$ tel que $\overline{\{t\}_{pub(N)}} \in ded$
  (2) si $\overline{t} = \overline{pub(N)}$ et $\overline{N} \in ded$
  (3) si $\overline{t} = \overline{dec(N_1, N_2)}$ et $\overline{N_1}, \overline{N_2} \in ded$
  (4) si $\overline{t} = \overline{\{N_1\}_{N_2}}$ et $\overline{N_1}, \overline{N_2} \in ded$
  (5) si $\overline{t} = \overline{N_1 + ... + N_m}$ et si $deduc_{AC}(transformer_+(\overline{t}), transformer_+(ded))$
  (6) si $\overline{t} = \overline{N_1 * ... * N_m}$ et si $deduc_{AC}(transformer_*(\overline{t}), transformer_*(ded))$

(7) si $\overline{t} = \overline{\{N_1\}_{pub(N_2)}}$, si $\overline{pub(N_2)} \in ded$ et si $deduc_{AC}(transformer_+(\overline{N_1}), transformer_+(ded'))$ avec $ded' = ded \cup \{\overline{u_i}|\overline{\{u_i\}_{pub(N_2)}} \in ded\}$

(8) $\overline{t} = \overline{\{N_1\}_{pub(N_2)}}$ et $deduc_{AC}(transformer_+(\overline{N_1}), transformer_+(ded'))$ avec $ded' = \{\overline{u_i}|\overline{\{u_i\}_{pub(N_2)}} \in ded\}$

alors ajouter $\overline{t}$ à ded et l'enlever de S

– on teste si $\overline{M}_{\downarrow} \in ded$

Comme Sat est de taille polynomiale alors la complexité de cet algorithme dépend polynomialement de la complexité de $deduc_{AC}$, à condition que les tests d'égalité entre classes d'équivalence modulo AC se font en temps polynomial. En effet, en considérant un ordre total sur les termes, on peut construire un représentant de manière unique et l'égalité de deux classes est alors équivalente à l'égalité syntaxique de leur représentant.

Maintenant, notons ded_i, l'ensemble ded à la i-ième itération et ded_f, à la fin.

Montrons que $ded_f = Sat \cap \{\overline{t}|\overline{\phi}_{\downarrow} \vdash_3 \overline{t}\}$:

$\subseteq$: par récurrence sur i, $ded_i \subseteq Sat \cap \{\overline{t}|\overline{\phi}_{\downarrow} \vdash_3 \overline{t}\}$

$\supseteq$: soit $\overline{M} \in Sat \cap \{\overline{t}|\overline{\phi}_{\downarrow} \vdash_3 \overline{t}\}$. On considère la preuve P de $\overline{\phi}_{\downarrow} \vdash_3 \overline{M}$ du lemme de localité et on va montrer par récurrence sur la taille de la preuve que si $\overline{\phi}_{\downarrow} \vdash_3 \overline{t}$ apparait dans P alors $\overline{t} \in ded$. Comme $\overline{M} \in Sat$ alors $\overline{t} \in Sat$. Distinguons les cas selon la dernière règle utilisée :

– Ax ou Nom : $\overline{t} \in ded$ par l'initialisation
– Pub : par (2) et HR
– Cod : par (4) et HR
– Dec sans pas de réduction : par (3) et HR
– Dec avec pas de réduction : par (1) et HR
– $+$: par (5) et HR
– $*$: par construction de P, cette règle n'induit pas de réduction donc par (6) et HR
– combi1 : par (8) et HR
– combi2 : par (9) et HR

□

4. Décidabilité du problème de l'équivalence statique pour HAC_{+*}

Nous allons étudier l'équivalence statique pour une théorie qui pourra nous être utiles par la suite en utilisant une technique de preuve qui revient à se ramener à des théories où l'équivalence statique est décidable.

4.1 Première étape : effacement des h et suppression du $+$

Commençons par rappeler la définition de la théorie :

Définition 18 (Théorie HAC_{+*})**.**

$$\begin{aligned}
\Sigma &= \{*(2), +(2), h(1)\} \\
E &= h(x_1) * h(x_2) = h(x_1 + x_2) \quad (H) \\
&\quad x + (y + z) = (x + y) + z \quad (A_+) \\
&\quad x + y = y + x \quad (C_+) \\
&\quad x * (y * z) = (x * y) * z \quad (A_*) \\
&\quad x * y = y * x \quad (C_*)
\end{aligned}$$

A cette théorie, on associe la règle de réécriture (confluente) suivante : $h(x_1 + x_2) \to h(x_1) * h(x_2)$.

La première idée est de restreindre $Eq(\phi)$, en limitant le nombre de h consécutifs dans les termes.

Définition 19.
Soit ϕ, une frame que l'on suppose sous forme normale ie $\forall x$, $x\sigma$ est sous forme normale.
On note :

- *$p(\phi)$, le nombre maximal de h consécutifs dans ϕ*
- *$K_p(\phi) = \{M$ sous forme normale tel que $M\sigma_\downarrow$ a au plus $p+1$ h consécutifs$\}$*
- *$Eq^p(\phi) = Eq(\phi) \cap K_p(\phi)^2$.*
- *$\tilde{Eq}^p(\phi, \phi') = Eq(\phi) \cap \{M|$ sous forme normale et tel que $M\sigma_\downarrow$ et $M\sigma'_\downarrow$ ont au plus $p+1$ h consécutifs$\}^2$.*

Lemme 4 (de restriction)**.**
$\forall \phi, \phi', \phi' \models_E Eq(\phi) \Leftrightarrow \phi' \models_E Eq^{p(\phi)}(\phi)$.

Démonstration.

($\Rightarrow$) évident

($\Leftarrow$) Soit $(M, N) \in Eq(\phi)$ ie $M\sigma =_E N\sigma$. On peut supposer que M et N sont sous forme normale. Montrons par récurrence sur $|M| + |N|$ que $(M, N) \in Eq(\phi')$.
Observons tout d'abord que si M ou $N \in K_{p(\phi)}(\phi)$ alors $(M, N) \in Eq^{p(\phi)}(\phi)$.

- si M ou N est une variable alors $(M, N) \in Eq^{p(\phi)}(\phi)$ et c'est OK.
- si $M = h(M')$ et $N = h(N')$, comme $M\sigma =_E N\sigma$ alors $M'\sigma =_E N'\sigma$ donc par HR $M'\sigma' =_E N'\sigma'$ et $M\sigma' =_E N\sigma'$.
- si $M =_{AC} \sum M_i$ et $N =_{AC} \sum N_j$ avec M_i et N_j qui ne sont pas des sommes.
  - si $(M, N) \in Eq^{p(\phi)}(\phi)$ alors c'est OK.
  - sinon $\exists M_{i'}$ et $N_{j'} \notin K_{p(\phi)}(\phi)$ donc en particulier, $M_{i'}$ et $N_{j'}$ ne sont pas des variables (ni des sommes par construction) donc $\exists M_{i'}$ et $N_{j'}$ tels que $M_{i'}\sigma =_E N_{j'}\sigma$. Ainsi, si on prend $M' = \sum_{i \neq i'} M_i$ et $N' = \sum_{j \neq j'} N_j$, alors $M'\sigma =_E N'\sigma$ et par HR, $M_{i'}\sigma' =_E N_{j'}\sigma'$ et $M'\sigma' =_E N'\sigma'$ donc $M\sigma' =_E (M' + M_{i'})\sigma' =_E (N' + N_{j'})\sigma' =_E N\sigma'$.
- si $M =_{AC} \prod M_i$ et $N =_{AC} \prod N_j$ avec M_i et N_j qui ne sont pas des produits.
  - si $(M, N) \in Eq^{p(\phi)}(\phi)$ alors c'est OK.
  - sinon $\exists M_{i'} \notin K_{p(\phi)}(\phi)$ donc en particulier, $M_{i'}$ n'est pas une variable (ni un produit par construction).
    - si $\exists M_i$ qui est une somme alors $\exists N_j$ tel que $M_i\sigma =_E N_j\sigma$ et on fait comme pour la somme.

- sinon $\forall i$, M_i est soit une variable, soit un h et $M_{i'}$ ne peut être qu'un h. Donc $M =_{AC} x_1 * ... * x_k * h(M'_{k+1}) * ... * h(M'_m)$ et $N =_{AC} y_1 * ... * y_l * h(N'_{l+1}) * ... * h(N'_n)$ avec $M_{i'} = h(M'_{k+1})$ donc M'_{k+1} n'est ni une somme (car sous forme normale), ni une variable (car $M_{i'} \notin K_{p(\phi)}(\phi)$) donc $\exists j$ tel que $M_{i'} = h(N'_j)$ et on fait comme pour la somme.
- si $M =_{AC} \prod M_i$ et $N = h(N')$, alors comme N est sous forme normale, $N = h(x) \in K_{p(\phi)}(\phi)$ et $(M, N) \in Eq^{p(\phi)}(\phi)$.

□

Corollaire 1.
$\forall \phi, \phi'$, *soit* $p = p(\phi) + p(\phi')$ *alors* $\phi \approx_E \phi' \Leftrightarrow \tilde{Eq}^p(\phi, \phi') = \tilde{Eq}^p(\phi'\phi)$

Démonstration.
Il suffit de voir que $Eq^{p(\phi)}(\phi) \subseteq \tilde{Eq}^p(\phi, \phi')$. □

Maintenant que l'on a ce lemme, on va transformer une première fois les frames, pour se ramener dans une théorie où l'on pourra enlever le + grâce à l'algorithme de combinaison de [CD11].

Définition 20 (Transformation n°1)**.**
- *Soit M, un terme sous forme normale. On définit le transformé $t(M)$ par :*
  - *si $M \in \mathcal{X} \cup \mathcal{N}$, $t(M) = M^0$*
  - *si $M = M_1 + M_2$, $t(M) = t(M_1) + t(M_2)$*
  - *si $M = M_1 * M_2$, $t(M) = t(M_1) * t(M_2)$*
  - *si $M = h(N)$ avec $N \in \mathcal{X} \cup \mathcal{N}$, $t(M) = N^1$*
  - *si $M = h(N^i)$ avec $N \in \mathcal{X} \cup \mathcal{N}$, $t(M) = N^{i+1}$*
  - *si $M = h(h(N))$, $t(M) = t(h(t(h(N))))$*
  - *si $M = h(M_1 * M_2)$, $t(M) = t(M_1) *_1 t(M_2)$*
  - *si $M = h(M_1 *_i M_2)$, $t(M) = t(M_1) *_{i+1} t(M_2)$*
- $t(\nu\tilde{n}.\sigma, p, \sigma') = \nu t(\tilde{n}, p).t(\sigma, p, \sigma')$ *avec* $t(\tilde{n}, p) = \{n^k | n \in \tilde{n}$ *et* $0 \leq k \leq p+1\}$ *et* $t(\{M_1/x_1, ..., M_n/x_n\}, p, \{M'_1/x_1, ..., M'_n/x_n\}) = \{t(M_1)/x_1^0, ..., t(h^{p_1}(M_1)_\downarrow)/x_1^{p_1}, ..., t(M_n)/x_n^0, ..., t(h^{p_n}(M_n)_\downarrow)/x_n^{p_n}\}$ *avec* $p_i = max\{p' | h^{p'}(M_i)_\downarrow$ *et* $h^{p'}(M'_i)_\downarrow$ *ont au plus* $p+1$ *h consécutifs*$\}$.
- $t(\mathcal{N}, p) = \{n^k | n \in \mathcal{N}$ *et* $0 \leq k \leq p+1\}$
- $t(dom(\sigma), p) = \{x_i^k | x_i \in dom(\sigma)$ *et* $0 \leq k \leq p_i\}$
- $K_p(\phi, \phi') = \{M |$ *sous forme normale et tel que* $M\sigma_\downarrow$ *et* $M\sigma'_\downarrow$ *ont au plus* $p+1$ *h consécutifs*$\}$
- $t(\Sigma, p) = \{+, *, *_1, ..., *_{p+1}\}$
- $E_{*_i} = C_{*_i} \cup \{x *_i (y * z) = (x * y) *_i z\}$
- $t(E, p) = AC_+ \cup AC_* \cup \left(\bigcup_{i=1}^{p+1} E_{*_i}\right)$

Voici quelques propriétés importantes :

Propriété 3.
Soient ϕ, ϕ' deux frames et $p = p(\phi) + p(\phi')$.
(1) si $M \in K_p(\phi, \phi')$ alors $M\sigma_\downarrow \in K_p(\phi, \phi')$ et $M\sigma'_\downarrow \in K_p(\phi, \phi')$
(2) t est un isomorphisme de $K_p(\phi, \phi')$ dans $T(t(\Sigma, p), t(\mathcal{N}, p), t(dom(\sigma), p))$
(3) $\forall M \in K_p(\phi, \phi')$, $t(M\sigma_\downarrow) = t(M)t(\sigma, p, \sigma')$
(4) $\forall M, N \in K_p(\phi, \phi')$, $M =_E N \Leftrightarrow M =_{AC} N \Leftrightarrow t(M) =_{t(E,p)} t(N)$
(5) si $M, N \in K_p(\phi, \phi')$ alors $(M =_E N)\phi \Leftrightarrow (t(M) =_{t(E,p)} t(N))t(\phi, p, \sigma')$

Démonstration.
(1) évident par définition de $K_p(\phi, \phi')$
(2) par construction de t
(3) soit $M \in K_p(\phi, \phi')$ alors M s'écrit $C[h^{n_1}(x_{i_1}), ..., h^{n_m}(x_{i_m})]$ avec C qui n'a pas de h juste au-dessus de ses variables (ie il n'y a pas de $h(x)$ dans C). Alors $t(M) = t(C)[x_{i_1}^{n_1}, ..., x_{i_m}^{n_m}]$ et $M\sigma_\downarrow = C[h^{n_1}(x_{i_1}\sigma)_\downarrow, ..., h^{n_m}(x_{i_m}\sigma)_\downarrow]$ et donc $t(M\sigma_\downarrow) = t(C)[t(h^{n_1}(x_{i_1}\sigma)_\downarrow), ..., t(h^{n_m}(x_{i_m}\sigma)_\downarrow)]$ et $\forall j$, $t(h^{n_j}(x_{i_j}\sigma)_\downarrow) = x_{i_j}^{n_j} t(\sigma, p, \sigma')$.
(4) la première équivalence peut se voir en termes de AC-convergence. La deuxième se montre par récurrence sur le nombre d'étapes dans la preuve de $M =_E N$ ou de $t(M) =_{t(E,p)} t(N)$.

(5)

$$\begin{aligned}(M =_E N)\phi &\Leftrightarrow M\sigma_\downarrow =_E N\sigma_\downarrow \\ &\Leftrightarrow t(M\sigma_\downarrow) =_{t(E,p)} t(N\sigma_\downarrow) \ \ par\ (1)\ et\ (4) \\ &\Leftrightarrow t(M)t(\sigma,p,\sigma') =_{t(E,p)} t(N)t(\sigma,p,\sigma') \ \ par\ (3) \\ &\Leftrightarrow (t(M) =_{t(E,p)} t(N))t(\phi,p,\sigma')\end{aligned}$$

□

Lemme 5.
Soient ϕ, ϕ', deux frames et $p = p(\phi) + p(\phi')$.
Alors, $\phi \approx_E \phi' \Leftrightarrow t(\phi,p,\sigma') \approx_{t(E,p)} t(\phi',p,\sigma)$.

Démonstration.

$$\begin{aligned}\phi \approx_E \phi' &\Leftrightarrow \tilde{Eq}^p(\phi,\phi') = \tilde{Eq}^p(\phi',\phi) \ \ par\ corollaire\ 1 \\ &\Leftrightarrow Eq(\phi) \cap K_p(\phi,\phi')^2 = Eq(\phi') \cap K_p(\phi,\phi')^2 \\ &\Leftrightarrow \{(t(M),t(N))|(M,N) \in Eq(\phi) \cap K_p(\phi,\phi')^2\} = \{(t(M),t(N))|(M,N) \in Eq(\phi') \cap K_p(\phi,\phi')^2\} \\ &\qquad car\ t\ injective \\ &\Leftrightarrow Eq(t(\phi,p)) = Eq(t(\phi',p)) \ \ car\ t\ surjective\ et\ (5) \\ &\Leftrightarrow t(\phi,p) \approx_{t(E,p)} t(\phi',p)\end{aligned}$$

□

Maintenant, on peut remarquer que $t(E,p)$ est l'union disjointe de $AC+$ et de $E'_p = AC_* \cup \left(\bigcup_{i=1}^{p+1} E_{*_i}\right)$. Donc, il suffit d'avoir la décidabilité de la déduction et de l'équivalence statique pour E'_p pour avoir par [CD11] la décidabilité de l'équivalence statique pour $t(E,p)$.

4.2 Deuxième étape : réduction de l'équivalence statique à celle de AC

Pour l'équivalence statique, nous allons effectuer une nouvelle étape de restriction - transformation pour nous ramener à AC. Cette fois-ci, nous allons nous restreindre à des termes de la forme un produit pour $*$ de "sous-termes" de la frame :

Définition 21 ("Petits termes")**.**
Soit ϕ, une frame.
- $K_\phi = \{M\ qui\ n'est\ pas\ un * \mid \exists x \in dom(\sigma)\ tel\ que\ M\sigma =_{E'_p} x\sigma\ ou\ x\sigma =_{E'_p} (M * M')\sigma\}$
- $K'_\phi = \{M =_{E'_p} \prod M_i \mid M_i \in K_\phi \cup \mathcal{N}\}$

Lemme 6 (de restriction 2)**.**
$\forall \phi, \phi', \phi' \models_{E'_p} Eq(\phi) \Leftrightarrow \phi' \models_{E'_p} Eq(\phi) \cap K_\phi^{'2}$.

Démonstration.
($\Rightarrow$) évident
($\Leftarrow$) Soit $(M,N) \in Eq(\phi)$. Montrons pas récurrence sur ($|M|$ + nombre de $*_i$ dans M) que $(M,N) \in Eq(\phi')$.
- soit M ou N est une variable et c'est OK
- sinon si $M =_{E'_p} \prod M_i$ avec M_i qui n'est pas un produit alors $N =_{E'_p} \prod N_j$ avec N_j qui n'est pas un produit. Supposons que $M \notin K'_\phi$ alors $\exists M_i \notin K_\phi$ donc $M_i = M'_i *_k M''_i$. Alors, $\exists j$ tel que $(M_i,N_j) \in Eq(\phi)$ et si $M' = \prod_{i' \neq i} M_{i'}$ et $N' = \prod_{j' \neq j} N_{j'}$, $(M',N') \in Eq(\phi)$. Donc, par HR, $(M_i,N_j) \in Eq(\phi')$ et $(M',N') \in Eq(\phi')$ et donc $(M,N) \in Eq(\phi')$.
- sinon si $M = M' *_k M''$ alors $N = N' *_k N''$. Donc, $(M' * M'', N' * N'') \in Eq(\phi)$ et par HR, $(M' * M'', N' * N'') \in Eq(\phi')$ et donc $(M,N) \in Eq(\phi')$.

□

Maintenant, nous allons transformer les frames pour nous ramener à AC.

Définition 22 (transformation n°2)**.**

- *Soit M, un terme clos. On définit le transformé $t(M)$ par*
  - *si $M \in \mathcal{N}$, $t(M) = M$*
  - *si $M = M_1 * M_2$, $t(M) = t(M_1) * t(M_2)$*
  - *si $M = M_1 *_k M_2$, $t(M) = a_{r(M)}$, où $a_{r(M)}$ est un nom frais et $r(M)$ est un représentant de M modulo E'_p définit de manière unique.*

  Cette définition suppose que l'on puisse définir un représentant de mainière unique. Cela est fait de la même manière que pour AC dans la section 3.
- *Soient $\phi = \nu\tilde{n}.\sigma$ et $\phi' = \nu\tilde{n}.\sigma'$, deux frames. $t(\nu\tilde{n}.\sigma'', \phi, \phi') = \nu t(\tilde{n}, \phi, \phi').t(\sigma'', \phi)$ avec $t(\tilde{n}, \phi, \phi') = \tilde{n} \cup \{a_{r(M)} \mid M$ apparait dans $t(\sigma, \phi) \cup t(\sigma', \phi)\}$ et $t(\sigma'', \phi) = \{t(M\sigma'')/y_M \mid M \in K_\phi \setminus \mathcal{N}\}$.*
- *Soit $M \in K'_\phi$. On définit le transformé $t'(M)$ par :*
  - *si $M \in \mathcal{N}$, $t'(M) = M$*
  - *si $M \in \mathcal{X}$, $t'(M) = y_M$*
  - *si $M = M_1 * M_2$, $t'(M) = t'(M_1) * t'(M_2)$*
  - *si $M = M_1 *_k M_2$, $t'(M) = y_M$*

Voici quelques propriétés importantes :

Propriété 4.

(1) t' est un isomorphisme de K'_ϕ dans $T(\{*\}, \mathcal{N}, \{y_M\})$
(2) $\forall M \in K'_\phi$, $\psi = \phi$ ou ϕ', $t'(M)t(\sigma_\psi, \phi) = t(M\sigma_\psi)$
(3) $\forall M, N$, $M =_{E'_p} N \Leftrightarrow t(M) =_{AC} t(N)$
(4) $\forall M \in K'_\phi$, $\psi = \phi$ ou ϕ', $(M =_{E'_p} N)\psi \Leftrightarrow (t'(M) =_{AC} t'(N))t(\psi, \phi, \phi')$

Démonstration.

(1) par construction de t'
(2) par construction des transformations
(3) par construction de t
(4)

$$\begin{aligned}
(M =_{E'_p} N)\psi &\Leftrightarrow M\sigma_\psi =_{E'_p} N\sigma_\psi \\
&\Leftrightarrow t(M\sigma_\psi) =_{AC} t(N\sigma_\psi) \ \ par\ (3) \\
&\Leftrightarrow t'(M)t(\sigma_\psi, \phi) =_{AC} t'(N)t(\sigma_\psi, \phi) \ \ par\ (2) \\
&\Leftrightarrow (t'(M) =_{AC} t'(N))t(\psi, \phi, \phi')
\end{aligned}$$

□

Lemme 7.
$\forall \phi, \phi', \phi' \models_{E'_p} Eq(\phi) \Leftrightarrow t(\phi', \phi, \phi') \models_{AC} Eq(t(\phi, \phi, \phi'))$.

Démonstration.
par lemme de restriction, (1) et (4). □

Corollaire 2.
Le problème de l'équivalence statique pour E'_p est décidable (EXPTIME).

Le temps exponentiel vient du fait que K_ϕ peut être de taille exponentielle. En effet, c'est le cas si $\phi = \nu\varnothing\{(\prod a_i) *_k (\prod b_j)/x, a_i/y_i, b_j/z_j\}$, par exemple.

4.3 Troisième étape : réduction de la déduction à l'équivalence statique

Ainsi, il nous manque plus que la décidabilité de la déduction. Mais, par un argument analogue à ce qui permet de démontrer le théorème 1, on peut montrer que la déduction se réduit à l'équivalence statique :

Lemme 8.
Pour toute frame $\phi = \nu\tilde{n}.\sigma$ et tout terme M,

$$\phi \vdash_{E'_p} M \Leftrightarrow \nu\tilde{n} \cup \{n_2\}.\sigma \cup \{M *_1 n_1/y\} \not\approx_{E'_p} \nu\tilde{n} \cup \{n_2\}.\sigma \cup \{n_2/y\}$$

avec n_1, n_2 et y frais.

Démonstration.
Pour commencer, il est facile de voir que :

$$\phi \vdash_{E'_p} M \Leftrightarrow \exists M',\ fn(M') \cap \tilde{n} = \varnothing \ et\ M'\sigma =_{E'_p} M$$

Montrons donc que :

$$\exists M',\ fn(M') \cap \tilde{n} = \varnothing \ et\ M'\sigma =_{E'_p} M \Leftrightarrow \nu\tilde{n} \cup \{n_2\}.\sigma \cup \{M *_1 n_1/y\} = \phi_1 \not\approx_{E'_p} \nu\tilde{n} \cup \{n_2\}.\sigma \cup \{n_2/y\} = \phi_2$$

($\Rightarrow$) $(M' *_1 n_1, y) \in Eq(\phi_1) \setminus Eq(\phi_2)$
($\Leftarrow$) $\exists (M', M'') \in Eq(\phi_1) \setminus Eq(\phi_2)$ et y apparait dans l'un des deux. Donc, soit $\exists C'$ et C'' des contextes tels que $M' = C'[y]$, $M'' = C''[y]$ et si m est un nom frais, $(C'[m], C''[m]) \in Eq(\phi_1) \setminus Eq(\phi_2)$ et y apparait dans l'un des contextes, soit $\exists C'$ et C'' des contextes, N, un terme tel que $M' = C'[y]$, $M'' = C''[N]$ avec si m est un nom frais $(C'[m], C''[m]) \in Eq(\phi_1) \setminus Eq(\phi_2)$ et $(N, y) \in Eq(\phi_1) \setminus Eq(\phi_2)$.
Donc, $\exists N = (\prod N_i) *_1 (\prod N'_j)$ apparaissant dans M'' et tel que $(N, y) \in Eq(\phi_1)$ et l'un des N_i ou N_j est n_1 (par ce qui précède et le fait que $n_1 \notin \sigma$), disons N_k. Alors, $M =_{E'_p} (\prod_{i \neq k} N_i * \prod_{j \neq k} N'_j)\sigma$.

□

Ainsi, nous avons tous les outils pour conclure par le théorème 2 :

Théorème 6.
Le problème de l'équivalence statique pour HAC_{+*} est décidable (EXPTIME).

5. Conclusion

Durant ce stage, nous avons donc montré la décidabilité du problème de la déduction pour la théorie $DHAC$ (NPTIME) et la décidabilité de l'équivalence statique pour HAC_{+*} (EXPTIME). Savoir décider l'équivalence statique pour HAC_{+*} induit la décidabilité de l'équivalence statique pour $DHAC$ si on se restreint à des frames utilisant une seule et même clé, non déductible, pour chiffrer. Nous espérons que ce dernier résultat puisse être complété afin de montrer la décidabilité de l'équivalence statique pour DHAC. En effet, la première étape serait de considérer plusieurs homomorphismes de la forme $h_i(x + y) = h_i(x) * h_i(y)$ (ce qui représenterait le fait que l'on ait plusieurs clés non déductibles). Pour cette étape, on remarque que toute la partie transformation précédente se conserve plutôt bien à condition de montrer un lemme de restriction similaire au lemme 4. Ensuite, il s'agirait de représenter le décodage dans le cas où l'une des clés serait déductible. Cela reviendrait à considérer HAC_{+*} avec une fonction "inverse" pour h. L'une des difficultés pour réutiliser la preuve précédente est de pouvoir définir un système de réécriture ayant de bonnes propriétés. En effet, on aimerait que $h^{-1}(h(x)) \to x$, $h(x+y) \to h(x)*h(y)$ et que $\to$ soit confluent, ce qui nous pose le problème de $h^{-1}(h(\sum a_i)) \to \sum a_i$ et $h^{-1}(\prod h(a_i))$ ce qui nous obligerait à avoir une infinité de règles pour avoir la confluence.

Nous avons également été amenés à utiliser le type de preuve restriction-transformation sur des théories monoïdales auxquelles on ajoute un homomorphisme de la forme $h(x+y) = h(x)+h(y)$. Nous avons vu que si l'on arrivait à borner les piles d'homomorphismes, nous pouvions nous ramener à la théorie monoïdale elle-même. Cependant, cette voie a été abandonnée car les lemmes de restrictions comme le lemme 4 sont très difficiles à montrer (même pour ACU).

Bibliographie

[ABB+05] Alessandro Armando, David A. Basin, Yohan Boichut, Yannick Chevalier, Luca Compagna, Jorge Cuéllar, Paul Hankes Drielsma, Pierre-Cyrille Héam, Olga Kouchnarenko, Jacopo Mantovani, Sebastian Mödersheim, David von Oheimb, Michaël Rusinowitch, Judson Santiago, Mathieu Turuani, Luca Viganò, and Laurent Vigneron. The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In *CAV*, pages 281–285, 2005.

[ABF04] Martín Abadi, Bruno Blanchet, and Cédric Fournet. Just Fast Keying in the Pi Calculus. In David Schmidt, editor, *Programming Languages and Systems : Proceedings of the 13th European Symposium on Programming (ESOP'04)*, volume 2986 of *Lecture Notes on Computer Science*, pages 340–354, Barcelona, Spain, March 2004. Springer Verlag.

[AC06] Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1-2) :2–32, 2006.

[BCD07] Sergiu Bursuc, Hubert Comon-Lundh, and Stéphanie Delaune. Deducibility constraints, equational theory and electronic money. In Hubert Comon-Lundh, Claude Kirchner, and Hélène Kirchner, editors, *Rewriting, Computation and Proof — Essays Dedicated to Jean-Pierre Jouannaud on the Occasion of his 60th Birthday*, volume 4600 of *Lecture Notes in Computer Science*, pages 196–212, Cachan, France, June 2007. Springer.

[Bla01] Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *CSFW*, pages 82–96, 2001.

[CD11] Véronique Cortier and Stéphanie Delaune. Decidability and combination results for two notions of knowledge in security protocols. *Journal of Automated Reasoning*, 2011. To appear.

[CS11] Véronique Cortier and Ben Smyth. Attacking and fixing Helios : An analysis of ballot secrecy. In *Proceedings of the 16th European Symposium on Research in Computer Security*, Cernay, France, 2011. IEEE Computer Society Press. to appear.

[CT03] Hubert Comon-Lundh and Ralf Treinen. Easy Intruder Deductions. In Nachum Dershowitz, editor, *Verification : Theory and Practice, Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday*, volume 2772 of *Lecture Notes in Computer Science*, pages 225–242. Springer, February 2003. Invited paper.

[Del06] Stéphanie Delaune. *Vérification des protocoles cryptographiques et propriétés algébriques.* Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France, June 2006.

[Laf06] Pascal Lafourcade. *Vérification des protocoles cryptographiques en présence de théories équationnelles.* Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France, September 2006. 209 pages.

[Low96] Gavin Lowe. Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR. In *TACAS*, pages 147–166, 1996.

[MDL+99] Durgin Lincoln Mitchell, N. A. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov. Undecidability of Bounded Security Protocols. 1999.

[RT03] Michaël Rusinowitch and Mathieu Turuani. Protocol insecurity with a finite number of sessions, composed keys is NP-complete. *Theor. Comput. Sci.*, 1-3(299) :451–475, 2003.