

PROJET LOGICAL
INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
LABORATOIRE D'INFORMATIQUE DE L'ECOLE POLYTECHNIQUE

Une nouvelle présentation de l'arithmétique de Heyting

Lisa Allali
Ens Cachan

Stage de Master 2 MPRI
Sous la direction de Gilles Dowek

Résumé : Une théorie modulo \mathcal{T}, \equiv est constituée d'un ensemble d'axiomes \mathcal{T} et d'une congruence \equiv définie par des *règles de réécriture*. Notre travail a porté sur l'étude de l'égalité dans l'arithmétique de Heyting, et plus particulièrement comment intégrer un algorithme de décidabilité de l'égalité sous forme d'un ensemble de règles de réécritures à une présentation modulo de l'arithmétique de Heyting. En particulier, nous avons remplacé l'axiome de Leibniz : $\forall x \forall y \ x = y \Rightarrow P(x) \Rightarrow P(y)$ par un ensemble de règles de réécriture définissant l'égalité, et avons prouvé que notre théorie était équivalente à la présentation usuelle de l'arithmétique de Heyting qui inclut cette axiome. Nous avons aussi montré que cette théorie a la propriété d'élimination des coupures.



Projet LogiCal



Introduction

La déduction modulo est un formalisme qui a pour objectif de distinguer dans une preuve les étapes de *raisonnement* et les étapes de *calcul*. Une théorie modulo \mathcal{T}, \equiv est constituée d'un ensemble d'axiomes \mathcal{T} et d'une congruence \equiv définie par des *règles de réécriture*. Dans une preuve en déduction modulo, une étape de raisonnement est un pas de déduction, alors qu'une étape de calcul ne nécessite aucun pas supplémentaire. Une théorie modulo sans axiome, définie uniquement par un ensemble de règles de réécriture, est dite *purement calculatoire*. L'intérêt de construire de telles théories est non seulement d'en simplifier les preuves, mais aussi l'aspect calculatoire, *algorithmique*, des règles de réécriture.

Une présentation *purement calculatoire* de l'arithmétique de Heyting a été proposée par Gilles Dowek et Benjamin Werner [2]. Cependant, cette présentation ne tirait pas partie d'un aspect essentiel de l'arithmétique : *la décidabilité de l'égalité sur les termes*. Notre travail a donc consisté à bien comprendre cette présentation afin de pouvoir la modifier de façon à intégrer cette décidabilité dans nos règles de réécriture.

La principale étape du travail effectué a été de définir l'égalité par un ensemble de règles de réécriture - et non plus par la propriété de Leibniz - puis de prouver que la théorie ainsi obtenue, où l'égalité est définie par une congruence, était équivalente à la présentation usuelle de l'arithmétique de Heyting, où l'égalité est donnée axiomatiquement. Puis nous avons encapsulé cette théorie dans une succession de théories jusqu'à obtenir une extension conservatrice de l'arithmétique de Heyting qui soit *purement calculatoire*. Ce procédé de stratification des théories permet d'obtenir des preuves plus simples pour prouver les équivalences entre chaque théorie intermédiaire.

Si cette technique de stratification avait aussi été utilisée dans la présentation initiale [2], les preuves sont très différentes. En effet, les preuves de conservativité ici sont presque toutes syntaxiques, alors qu'elles étaient sémantiques dans la présentation précédente. Une autre différence importante est que notre théorie a été allégée, le langage est plus simple, il y a moins de règles de réécriture, et cela permet d'avoir des preuves plus courtes.

Nous avons enfin prouvé que cette nouvelle présentation purement calculatoire de l'arithmétique de Heyting a la propriété d'élimination des coupures. Cette preuve d'élimination des coupures est sémantique. Basée sur le fait que la normalisation forte implique l'élimination des coupures, elle utilise un nouveau résultat [3] qui se sert de façon implicite de l'algèbre des candidats, ce qui simplifie beaucoup la preuve de normalisation forte de la théorie.

Table des matières

1 Définitions	7
1.1 Réécriture et congruence	7
1.2 La déduction modulo	7
1.3 Théorie axiomatique et théorie modulo en déduction naturelle	9
1.4 Preuves d'équivalence de théories	10
1.4.1 Les preuves syntaxiques	10
1.4.2 Les preuves sémantiques	10
1.5 Les modèles d'une théorie purement calculatoire	10
1.6 Les modèles de la logique intuitioniste	10
1.6.1 Les algèbres de Heyting	10
1.6.2 Les pseudo-algèbres de Heyting comme modèles de la déduction modulo intuitioniste	11
1.6.3 Pourquoi cette distinction entre algèbre de Heyting et pseudo-algèbre de Heyting?	12
1.6.4 Les pseudo-algèbres de Heyting complètes et ordonnées	12
1.7 La super-consistance :	
une façon de prouver la normalisation forte	13
1.8 Présentation usuelle de l'arithmétique de Heyting	13
1.8.1 Syntaxe du langage	13
1.8.2 les axiomes de la théorie	13
1.9 Une présentation modulo de l'arithmétique de Heyting	14
2 Une nouvelle présentation modulo de l'arithmétique de Heyting	15
2.1 Introduction	15
2.2 HA_R , une théorie équivalente à HA	17
2.2.1 Motivations	17
2.2.2 Présentation de HA_R	17
2.2.3 HA_R étend HA	17
2.2.4 HA étend HA_R	22
2.3 HA_N , une extension conservative de HA_R	27
2.3.1 Motivations	27
2.3.2 Présentation de HA_N	27
2.3.3 Extension	28
2.3.4 Conservativité	31
2.4 HA_K , une extension conservative de HA_N	33
2.4.1 Motivations	33
2.4.2 Présentation de HA_K	33
2.4.3 Extension	34
2.4.4 Conservativité	36
2.5 HA_{\rightarrow} , une présentation purement calculatoire de l'Arithmétique de Heyting	39
2.5.1 Motivations	39
2.5.2 La théorie	39
2.5.3 HA_{\rightarrow} , une théorie équivalente à HA_K	39
2.6 HA_{\rightarrow} , une théorie qui a la propriété d'élimination des coupures	40
2.6.1 Motivations	40
2.6.2 Rappels	40

2.6.3	HA_{\rightarrow} , une théorie super-consistance	40
3	Réalisations en Coq	45
3.1	Introduction	45
3.2	Nouvelle définition des entiers et fonctions sur les entiers	45
3.2.1	Les entiers	45
3.2.2	L'addition	45
3.2.3	La multiplication	45
3.2.4	L' égalité sur les entiers	45
3.3	Les propriétés prouvées	46
3.3.1	Les propriétés de l'égalité	46
3.3.2	Les propriétés de l'addition	46
3.3.3	Les propriétés de la multiplication	46
3.4	Comparaisons sur la taille des termes de preuves	47
3.4.1	Exemple 1 avec notre nouvelle égalité	47
3.4.2	Exemple 2 avec l'égalité usuelle dans Coq	47
3.5	Premières conclusions	48
4	Conclusion	49
A	Arbres de preuves et démonstrations	51
A.1	Lemme 2.3.3	51
A.1.1	$\vdash_{\text{HA}_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x + y)$	51
A.1.2	$\vdash_{\text{HA}_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x \times y)$	51
A.1.3	Preuve du lemme 2.3.3	52
A.2	Propriétés de l'égalité dans HA_R	53
A.2.1	Réflexivité : $\forall x x = x$	53
A.2.2	Symétrie : $\forall x \forall y x = y \Rightarrow y = x$	53
A.2.3	Transitivité : $\forall x \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z$	54
A.3	Propriétés de l'addition dans HA_R	55
A.3.1	0 neutre pour addition $O_neutre : \forall y y = y + 0$	55
A.3.2	Passage du successeur à droite $Suc_right : \forall x \forall y x + S(y) = S(x + y)$	55
A.3.3	Commutativité de l'addition $Comm : \forall x \forall y x + y = y + x$	55
A.3.4	Associativité de l'addition $Assoc : \forall z \forall x \forall y (x + y) + z = x + (y + z)$	56
A.3.5	Congruence à gauche pour l'addition : $add_left : \forall x \forall y \forall z x = y \Rightarrow x + z = y + z$	56
A.3.6	Congruence à droite pour l'addition : $add_right : \forall x \forall y \forall z x = y \Rightarrow z + x = z + y$	57
A.3.7	Congruence parallèle de l'addition : $add_par : \forall x \forall y \forall u \forall t x = y \Rightarrow u = t \Rightarrow x + u = y + t$	57
A.4	Propriétés de la multiplication dans HA_R	57
A.4.1	0 absorbant pour la multiplication à droite : $0_abs_r : \forall x x \times 0 = 0$	57
A.4.2	Multiplication successeur à droite : $mult_succ : \forall y \forall x y \times S(x) = y \times x + y$	58
A.4.3	Commutativité de la multiplication : $mult_com : \forall x \forall y x \times y = y \times x$	58
A.4.4	Congruence à gauche pour la multiplication : $mul_left : \forall z \forall x \forall y x = y \Rightarrow z \times x = z \times y$	58
A.4.5	Congruence à droite pour la multiplication : $mul_right : \forall z \forall x \forall y x = y \Rightarrow x \times z = y \times z$	59
A.4.6	Congruence parallèle de la multiplication : $mul_par : \forall x \forall y \forall u \forall t x = y \Rightarrow u = t \Rightarrow x \times u = y \times t$	59

Chapitre 1

Définitions

1.1 Réécriture et congruence

Définition 1.1.1 (Règle de réécriture)

Une règle de réécriture est un couple de termes ou de propositions A et B , noté $A \longrightarrow B$.

Si ce sont des termes alors A n'est pas une variable, si ce sont des propositions A doit être atomique.

Remarque : on peut, dans d'autres cadres, s'intéresser à réécrire des propositions non atomiques, dans notre étude, nous nous restreignons à la définition ci-dessus.

Définition 1.1.2 (Système de réécriture)

Un système de réécriture est un ensemble de règles de réécriture.

Définition 1.1.3 (Congruence)

Une congruence est une relation d'équivalence (reflexive, symétrique et transitive) qui est close par contexte i.e. pour tout contexte $C[]$, si $a \equiv b$ alors $C[a] \equiv C[b]$.

On peut dire d'un système de réécriture qu'il définit une congruence : C'est la plus petite congruence qui contient \longrightarrow .

1.2 La déduction modulo

La déduction modulo est un système basé sur la déduction naturelle, à laquelle on ajoute une congruence \equiv qu'on prend en compte dans nos règles tel que le montre la figure 1.1.

Dans ce système de déduction, on code

la négation $\neg A$ par $A \Rightarrow \perp$

l'équivalence $A \Leftrightarrow B$ par $A \Rightarrow B \wedge B \Rightarrow A$.

La règle Ax s'applique aussi si B est un axiome de la théorie.

Pour mieux comprendre la spécificité d'une théorie modulo, nous allons en donner un exemple :

Soit la théorie \mathcal{T} définie par la logique propositionnelle à laquelle on ajoute la congruence \equiv définie par les règles de réécriture suivantes :

(1) $A \longrightarrow B$

(2) $B \longrightarrow C$

Voici une preuve en déduction modulo de $\vdash_{\equiv} A \Rightarrow C$:

$$\begin{array}{c}
\frac{}{\Gamma \vdash_{\equiv} B} \text{Ax si } A \in \Gamma \text{ et } A \equiv B \\
\frac{}{\Gamma \vdash_{\equiv} \top} \text{Ax} \\
\frac{\Gamma, A \vdash_{\equiv} B}{\Gamma \vdash_{\equiv} C} \Rightarrow \text{i si } C \equiv A \Rightarrow B \\
\frac{\Gamma \vdash_{\equiv} C \quad \Gamma \vdash_{\equiv} A}{\Gamma \vdash_{\equiv} B} \Rightarrow \text{e si } C \equiv A \Rightarrow B \\
\frac{\Gamma \vdash_{\equiv} A \quad \Gamma \vdash_{\equiv} B}{\Gamma \vdash_{\equiv} C} \wedge \text{i si } C \equiv A \wedge B \\
\frac{\Gamma \vdash_{\equiv} C}{\Gamma \vdash_{\equiv} A} \wedge \text{e si } C \equiv A \wedge B \\
\frac{\Gamma \vdash_{\equiv} C}{\Gamma \vdash_{\equiv} B} \wedge \text{e si } C \equiv A \wedge B \\
\frac{\Gamma \vdash_{\equiv} A}{\Gamma \vdash_{\equiv} C} \vee \text{i si } C \equiv A \vee B \\
\frac{\Gamma \vdash_{\equiv} B}{\Gamma \vdash_{\equiv} C} \vee \text{i si } C \equiv A \vee B \\
\frac{\Gamma \vdash_{\equiv} D \quad \Gamma, A \vdash_{\equiv} C \quad \Gamma, B \vdash_{\equiv} C}{\Gamma \vdash_{\equiv} C} \vee \text{e si } D \equiv A \vee B \\
\frac{\Gamma \vdash_{\equiv} B}{\Gamma \vdash_{\equiv} A} \vee \text{i si } B \equiv \perp \\
\frac{\Gamma \vdash_{\equiv} A}{\Gamma \vdash_{\equiv} B} \forall \text{i si } B \equiv \forall x A \text{ et } x \notin FV(\Gamma) \\
\frac{\Gamma \vdash_{\equiv} B}{\Gamma \vdash_{\equiv} C} \forall \text{e si } B \equiv \forall x A \text{ et } C \equiv A\{x := t\} \\
\frac{\Gamma \vdash_{\equiv} C}{\Gamma \vdash_{\equiv} B} \exists \text{i si } B \equiv \exists x A \text{ et } C \equiv A\{x := t\} \\
\frac{\Gamma \vdash_{\equiv} C \quad \Gamma, A \vdash_{\equiv} B}{\Gamma \vdash_{\equiv} B} \exists \text{e si } C \equiv \exists x A \text{ et } x \notin FV(\Gamma, B)
\end{array}$$

FIG. 1.1 – La deduction naturelle modulo

$$\frac{\frac{}{A \vdash_{\equiv} C} \text{Ax } A \equiv C}{\vdash_{\equiv} A \Rightarrow C} \Rightarrow i$$

Cette preuve est de longueur deux : deux règles de déduction ont été appliquées : Ax et $\Rightarrow i$.

A est congrue à C car on commence par réécrire A en B par application de la règle de réécriture 1, puis B en C par la règle 2. Ces réécritures ne comptent pas dans la longueur de la preuve, on peut cependant les représenter sur l'arbre de preuve comme suit :

$$\frac{\frac{\frac{}{A \vdash_{\equiv} A} \text{Ax } A \equiv A}{A \vdash_{\equiv} B} 1}{A \vdash_{\equiv} C} 2}{\vdash_{\equiv} A \Rightarrow C} \Rightarrow i$$

Cette preuve est toujours de longueur deux. On a juste ajouté à l'arbre des informations sur les règles de réécriture utilisées. Les étapes étiquetées d'un numéro ne comptent pas dans la longueur de la preuve. Cette présentation a l'avantage de se convaincre de la congruence de deux termes, car on peut suivre pas à pas les étapes de transformations des termes par les règles de réécriture. C'est cette présentation des arbres de dérivation en déduction modulo que nous utiliserons pour la suite, en gardant bien à l'esprit que la longueur de l'arbre ne prend en compte que le nombre de règles *de déduction* appliquées, et que les labels numérotés de réécriture ne comptent pas dans le nombre de pas de déduction de l'arbre de dérivation. L'application d'une règle de réécriture est juste un calcul.

Nous insistons ici sur la façon de calculer la longueur d'un arbre de dérivation, car c'est justement tout l'enjeu de la déduction modulo : différencier les étapes de *déduction* et les étapes de *calcul* pour avoir des preuves plus courtes.

1.3 Théorie axiomatique et théorie modulo en déduction naturelle

En déduction naturelle, on peut donner les trois définitions suivantes :

Définition 1.3.1 (Théorie axiomatique)

Une théorie axiomatique est l'ensemble des théorèmes prouvables à partir d'un ensemble d'axiomes \mathcal{A} .

Définition 1.3.2 (Théorie modulo)

Une théorie modulo est l'ensemble des théorèmes prouvables à partir d'un ensemble d'axiomes \mathcal{A} et d'une congruence \equiv .

Définition 1.3.3 (Théorie purement calculatoire)

Une théorie purement calculatoire est l'ensemble des théorèmes prouvables à partir d'une congruence \equiv .

La question qui se pose est de savoir si ces définitions se rejoignent, c'est à dire par exemple de savoir si une théorie modulo peut définir la même théorie qu'un ensemble d'axiomes.

Le théorème suivant répond en partie à cette question :

Théorème 1.3.1 (Equivalence de théorie)

Soit une théorie axiomatique \mathcal{T} dont les axiomes sont \mathcal{A} . Soit \mathcal{A}' les axiomes de \mathcal{A} de la forme $B \Leftrightarrow C$ où B est atomique.

Soit \mathcal{R} l'ensemble des règles de réécriture $B \rightarrow C$ générées par \mathcal{A}' .

Soit \equiv la congruence définie par \mathcal{R} .

La théorie modulo \mathcal{T}' définie par les axiomes $\mathcal{A} \setminus \mathcal{A}'$ et la congruence \equiv est équivalente à \mathcal{T} .

1.4 Preuves d'équivalence de théories

Prouver que deux théories sont équivalentes, c'est prouver qu'elles sont composées des mêmes théorèmes. Autrement dit, \mathcal{T} et \mathcal{T}' sont deux théories équivalentes quand : A est un théorème de \mathcal{T} si et seulement si A est un théorème de \mathcal{T}' .

Si le langage de \mathcal{T}' est différent de celui de \mathcal{T} , on ne parle pas d'équivalence entre les deux théories. On dit que \mathcal{T}' est une *extension conservatrice* de \mathcal{T} . Dans ce cas, il faut que le langage de \mathcal{T} soit inclus dans \mathcal{T}' , prouver que les axiomes de \mathcal{T} sont prouvables dans \mathcal{T}' et que les axiomes de \mathcal{T}' formulés dans le langage de \mathcal{T} sont prouvables dans \mathcal{T} .

Il y a deux techniques possibles pour de telles preuves : syntaxique et sémantique.

1.4.1 Les preuves syntaxiques

On prouve d'abord pour tout théorème A de \mathcal{T} qu'à partir de la preuve Π de A dans \mathcal{T} on peut construire une preuve Π' de A dans \mathcal{T}' .

On prouve réciproquement pour tout théorème A de \mathcal{T}' qu'à partir de la preuve Π' de A dans \mathcal{T}' on peut construire une preuve Π de A dans \mathcal{T} .

1.4.2 Les preuves sémantiques

Ces preuves s'appuient sur le théorème de complétude de Gödel :

Théorème 1.4.1 (Théorème de complétude de Gödel)

Soient deux théories \mathcal{T} et \mathcal{T}' . Si pour tout modèle \mathcal{M} tel que $\mathcal{M} \models \mathcal{T}$ on a $\mathcal{M} \models \mathcal{T}'$ alors $\mathcal{T} \vdash \mathcal{T}'$.

Pour prouver que deux théories \mathcal{T} et \mathcal{T}' sont équivalentes, on prouve successivement que $\mathcal{T} \vdash \mathcal{T}'$ et $\mathcal{T}' \vdash \mathcal{T}$ de façon sémantique, à l'aide du théorème de complétude de Gödel.

Pour utiliser les preuves sémantiques d'équivalence de deux théories, il faut d'abord avoir une notion de modèle pour ces théories, c'est pourquoi nous allons donner une définition de ce qu'est un modèle pour une théorie purement calculatoire, puis rappeler la définition de modèle de la logique intuitioniste qui nous servira pour présenter les modèles de la déduction modulo intuitioniste.

1.5 Les modèles d'une théorie purement calculatoire

Définition 1.5.1

Un modèle d'une théorie purement calculatoire dont les règles de réécriture sont

$$\begin{array}{c} R_1 \longrightarrow R'_1 \\ \vdots \\ R_n \longrightarrow R'_n \end{array}$$

est tel que pour toute interprétation Φ , et pour $i \in \{1, n\}$

$$\llbracket R_i \rrbracket_{\Phi} = \llbracket R'_i \rrbracket_{\Phi}$$

1.6 Les modèles de la logique intuitioniste

1.6.1 Les algèbres de Heyting

Définition 1.6.1 (Algèbre de Heyting)

Une structure $\langle B, D, \leq, \tilde{\wedge}, \tilde{\vee}, \perp, \tilde{\top}, \tilde{\forall}, \tilde{\exists}, \Rightarrow \rangle$ est une Algèbre de Heyting si

- D est un sous-ensemble de $\wp(B)$,
- \leq une relation d'ordre réflexive, transitive en antisymétrique

- $\tilde{\perp}$ est le minimum
- $\tilde{\top}$ est le maximum
- $x\tilde{\wedge}y$ est la borne inférieure de x et y
- $x\tilde{\vee}y$ est la borne supérieure de x et y
- $\tilde{\forall}$ et $\tilde{\exists}$ (bornes inférieure et supérieure infinies) sont des fonctions de D dans A telles que :
 - Si $x \in a$ alors $\tilde{\forall}a \leq x$,
 - Si pour tout a dans A $c \leq a$, alors $c \leq \tilde{\forall}A$
 - Si $x \in a$ alors $x \leq \tilde{\exists}a$,
 - Si pour tout a dans A $a \leq b$, alors $\tilde{\exists}a \leq b$
- $x \leq y \Rightarrow z$ si et seulement si $x\tilde{\wedge}y \leq z$.

Définition 1.6.2 (Modèle intuitioniste)

Soit \mathcal{L} un langage. Un modèle intuitioniste \mathcal{M} de \mathcal{L} est formé par :

- un ensemble M ,
- une algèbre de Heyting B ,
- pour tout symbole de fonction f d'arité n une fonction \hat{f} de M^n dans M ,
- pour tout symbole de prédicat P d'arité n une fonction \hat{P} de M^n dans B .

Définition 1.6.3 (Dénotation)

Soit un modèle \mathcal{M} , une proposition A et une assignation ϕ , on définit $\llbracket A \rrbracket_\phi$ comme suit :

- $\llbracket x \rrbracket_\phi = \phi(x)$,
- $\llbracket f(t_1, \dots, t_n) \rrbracket_\phi = \hat{f}(\llbracket t_1 \rrbracket_\phi, \dots, \llbracket t_n \rrbracket_\phi)$,
- $\llbracket P(t_1, \dots, t_n) \rrbracket_\phi = \hat{P}(\llbracket t_1 \rrbracket_\phi, \dots, \llbracket t_n \rrbracket_\phi)$,
- $\llbracket \perp \rrbracket_\phi = \tilde{\perp}$,
- $\llbracket \top \rrbracket_\phi = \tilde{\top}$,
- $\llbracket A \Rightarrow B \rrbracket_\phi = \llbracket A \rrbracket_\phi \Rightarrow \llbracket B \rrbracket_\phi$,
- $\llbracket A \wedge B \rrbracket_\phi = \llbracket A \rrbracket_\phi \tilde{\wedge} \llbracket B \rrbracket_\phi$,
- $\llbracket A \vee B \rrbracket_\phi = \llbracket A \rrbracket_\phi \tilde{\vee} \llbracket B \rrbracket_\phi$,
- $\llbracket \forall x A \rrbracket_\phi = \tilde{\forall}\{\llbracket A \rrbracket_{\phi, x:=v} \mid v \in M\}$,
- $\llbracket \exists x A \rrbracket_\phi = \tilde{\exists}\{\llbracket A \rrbracket_{\phi, x:=v} \mid v \in M\}$.

1.6.2 Les pseudo-algèbres de Heyting comme modèles de la déduction modulo intuitioniste**Définition 1.6.4 (Pseudo-algèbre de Heyting)**

Une structure $\langle B, D, \leq, \tilde{\wedge}, \tilde{\vee}, \tilde{\perp}, \tilde{\top}, \tilde{\forall}, \tilde{\exists}, \Rightarrow \rangle$ est une pseudo-algèbre de Heyting si

- D est un sous-ensemble de $\wp(B)$,
- \leq est une relation réflexive et transitive
- $\tilde{\perp}$ est le minimum
- $\tilde{\top}$ est le maximum
- $x\tilde{\wedge}y$ est la borne inférieure de x et y
- $x\tilde{\vee}y$ est la borne supérieure de x et y
- $\tilde{\forall}$ et $\tilde{\exists}$ (bornes inférieure et supérieure infinies) sont des fonctions de D dans A telles que :
 - Si $x \in a$ alors $\tilde{\forall}a \leq x$,
 - Si pour tout a dans A $c \leq a$, alors $c \leq \tilde{\forall}A$
 - Si $x \in a$ alors $x \leq \tilde{\exists}a$,
 - Si pour tout a dans A $a \leq b$, alors $\tilde{\exists}a \leq b$

– $x \leq y \Rightarrow z$ si et seulement si $x \tilde{\wedge} y \leq z$.

Remarque 1 : une *pseudo-algèbre de Heyting* a les mêmes propriétés qu'une algèbre de Heyting sauf concernant la relation \leq qui n'est pas antisymétrique, \leq **n'est pas nécessairement une relation d'ordre**.

Remarque 2 : Toutes les algèbres de Heyting sont des pseudo-algèbres de Heyting.

Définition 1.6.5 (Modèle intuitioniste modulo)

Soit \mathcal{L} un langage. Un modèle intuitioniste modulo \mathcal{M} de \mathcal{L} est formé par :

- un ensemble M ,
- une pseudo-algèbre de Heyting B ,
- pour tout symbole de fonction f d'arité n une fonction \hat{f} de M^n dans M ,
- pour tout symbole de prédicat P d'arité n une fonction \hat{P} de M^n dans B .

La dénotation est la même que pour les algèbres de Heyting.

Théorème 1.6.1

L'algèbre des candidats est une pseudo-algèbre de Heyting.

Théorème 1.6.2

L'algèbre des candidats n'est une algèbre de Heyting.

Preuves dans [3].

1.6.3 Pourquoi cette distinction entre algèbre de Heyting et pseudo-algèbre de Heyting ?

La différence est, rappelons-le, le symbole \leq antisymétrique dans les algèbres de Heyting et pas dans les pseudo-algèbres de Heyting.

Cette caractérisation des modèles des théories modulo est intéressante car elle permet de distinguer les propositions *équivalentes* des propositions *congruentes*.

En effet, si $A \Leftrightarrow B$ dans une théorie, on a $\llbracket A \rrbracket \leq \llbracket B \rrbracket$ et $\llbracket B \rrbracket \leq \llbracket A \rrbracket$, mais pas nécessairement $\llbracket A \rrbracket = \llbracket B \rrbracket$.

(Ce qui serait évidemment le cas dans une algèbre de Heyting puisque la relation \leq est antisymétrique).

Si $A \equiv B$, alors on aura $\llbracket A \rrbracket = \llbracket B \rrbracket$.

Par ailleurs, cette distinction est essentielle car l'algèbre des candidats est une pseudo-algèbre de Heyting, mais pas une algèbre de Heyting. C'est la fondation de l'article [3] qui donne une méthode plus simple pour prouver la normalisation forte d'une théorie, méthode que nous utiliserons pour prouver la normalisation forte de la théorie que nous présentons.

1.6.4 Les pseudo-algèbres de Heyting complètes et ordonnées

Une *pseudo-algèbre de Heyting ordonnée complète* est une pseudo-algèbre de Heyting à laquelle on ajoute une relation d'ordre \sqsubseteq complète telle que

- $\tilde{\wedge}, \tilde{\vee}, \tilde{\exists}$ sont croissants
- $\tilde{\Rightarrow}$ est décroissante à gauche, croissante à droite.

L'introduction de cette relation d'ordre vient du fait que nous allons être amenés à vouloir calculer des points fixes dans nos modèles, et pour ça nous aurons besoin d'une relation d'ordre complète, qui ne peut plus être \leq puisqu'elle n'est plus nécessairement relation d'ordre, c'est pourquoi on introduit \sqsubseteq .

1.7 La super-consistance : une façon de prouver la normalisation forte

On commence par définir la notion de super-consistance :

Définition 1.7.1 (Super-consistance)

Une théorie \mathcal{T}, \equiv en déduction modulo est super-consistante si, pour toute pseudo-algèbre de Heyting ordonnée \mathcal{B} , on peut construire un \mathcal{B} -modèle de cette théorie.

Théorème 1.7.1 (Normalisation) Si une théorie \mathcal{T}, \equiv est super-consistante, alors toute preuve dans \mathcal{T}, \equiv est fortement normalisable.

Preuve : Si on peut construire un modèle pour toute pseudo-algèbre de Heyting complète et ordonnée, on le peut en particulier pour l'algèbre des candidats. Or, d'après [3], si on peut construire un modèle à une théorie à partir de l'algèbre des candidats, cette théorie a la propriété d'élimination des coupures.

1.8 Présentation usuelle de l'arithmétique de Heyting

1.8.1 Syntaxe du langage

Les symboles de fonction

- 0 d'arité 0
- S d'arité 1
- + et \times d'arité 2

Le symbole de prédicat

- = d'arité 2

1.8.2 les axiomes de la théorie

Le schéma d'axiomes de récurrence

$$(P\{x := 0\} \wedge \forall y (P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow \forall n P\{x := n\}$$

Le schéma d'axiomes dit "de Leibniz"

$$\forall x \forall y x = y \Rightarrow P(x) \Leftrightarrow P(y)$$

La réflexivité de l'égalité

$$\forall x x = x$$

Les axiomes sur le successeur

$$\forall x 0 = S(x) \Rightarrow \perp \qquad \forall x \forall y (S(x) = S(y) \Rightarrow x = y)$$

Les axiomes sur + et \times

$$\forall y (0 + y = y) \qquad \forall x \forall y (S(x) + y = S(x + y))$$

$$\forall y (0 \times y = 0) \qquad \forall x \forall y (S(x) \times y = x \times y + y)$$

Pour comprendre les motivations du travail effectué, prouvons dans cette théorie que $S(S(0)) + S(0) = S(S(S(0)))$

$$\frac{\frac{\frac{}{\vdash_{HA} 0 + y = y} Ax}{!!!!!!} \forall e \quad \frac{\frac{\frac{}{\vdash_{HA} x = y \Rightarrow S(0) + S(0) = S(x) \Rightarrow S(0) + S(0) = S(y)} Instance\ du\ schéma\ de\ Leibniz} Ax}{\vdash_{HA} 0 + S(0) = S(0) \Rightarrow S(0) + S(0) = S(0 + S(0)) \Rightarrow S(0) + S(0) = S(S(0))} \forall ex2}{\vdash_{HA} S(0) + S(0) = S(0 + S(0)) \Rightarrow S(0) + S(0) = S(S(0))} \Rightarrow e \quad \frac{\frac{\frac{}{\vdash_{HA} \forall x S(x) + y = S(x + y)} Ax}{\vdash_{HA} S(0) + S(0) = S(0 + S(0))} \forall ex}{\vdash_{HA} S(S(0)) + S(0) = S(S(S(0)))} \Rightarrow e$$

Les points d'exclamations ne font évidemment pas partie de la preuve. Ils sont là pour spécifier un endroit important de la preuve. Ici, on additionne deux nombres proches de zéro, la preuve est courte. Si on additionnait

$3+1$ au lieu de $2+1$, à l'emplacement des points d'exclamation, au lieu de pouvoir appliquer un axiome comme ici, on aurait un nouvel arbre d'égalité (de $1+1=2$) qui viendrait se greffer, de la même forme que l'arbre ci dessus. Et surtout, avec le même noeud critique. Ainsi, si on sommait un grand nombre à gauche de l'addition, on aurait pour preuve un arbre de preuve en forme d'un peigne gigantesque !

1.9 Une présentation modulo de l'arithmétique de Heyting

Nous donnons ici la présentation purement calculatoire de l'arithmétique de Heyting proposée par Gilles Dowek et Benjamin Werner [2].

Le langage est enrichi, on introduit :

- les prédicats N , $etNull$, d'arité 1, qui s'interprètent respectivement par le fait d'être un entier et le fait d'être zéro.
- le symbole de fonction $Pred$, d'arité 1, qui donne le prédécesseur
- le symbole de prédicat \in qui modélise l'appartenance, et une infinité de symboles de fonction f_p (autant que de propositions P énonçables dans le langage de l'arithmétique).
On modélise l'ensemble des x qui vérifient une proposition P par un nouveau symbole de fonction f_p .
Si un certain x vérifie une proposition P on aura $x \in f_p$

La théorie est définie par les règles de réécriture suivantes :

L'appartenance :

$$x \in f_{z,y_1,\dots,y_n,P}(y_1, \dots, y_n) \longrightarrow P\{z := x\}$$

La récurrence :

$$N(n) \longrightarrow \forall p (0 \in p \Rightarrow \forall y (N(y) \Rightarrow y \in p \Rightarrow S(y) \in p) \Rightarrow n \in p)$$

Leibniz :

$$y = z \longrightarrow \forall p (y \in p \Rightarrow z \in p)$$

Le successeur : (dont les propriétés sont prouvés à l'aide de $Null$ et $Pred$)

$$\begin{array}{ll} Null(0) \longrightarrow \top & Pred(0) \longrightarrow 0 \\ Null(S(x)) \longrightarrow \perp & Pred(S(x)) \longrightarrow x \end{array}$$

L'addition et la multiplication :

$$\begin{array}{ll} 0 + y \longrightarrow y & S(x) + y \longrightarrow S(x + y) \\ 0 \times y \longrightarrow 0 & S(x) \times y \longrightarrow x \times y + y \end{array}$$

Cette présentation purement calculatoire de l'arithmétique de Heyting est une *extension conservative* de l'arithmétique de Heyting (HA).

Les gains par rapport à HA : voici la preuve de $2+1=3$

$$\frac{\frac{\frac{2+1 \in p \vdash_{HA_R} 3 \in p}{\vdash_{HA_R} 2+1 \in p \Rightarrow 3 \in p} \Rightarrow i}{\vdash_{HA_R} \forall p 2+1 \in p \Rightarrow 3 \in p} \forall i}{\vdash_{HA_R} 2+1=3} \text{réécriture - Leibniz}$$

Quelles que soient les valeurs additionnées, la preuve est la même que ci-dessus. Le gain est conséquent.

Regardons maintenant comment on prouve qu'on n'a pas $0=1$.

Il faut construire une proposition qui invalide l'égalité $0=1$. On choisit $Null(x)$. On a $Null(0)$ qui se réécrit en \top et $Null(1)$ qui se réécrit en \perp .

Pour invalider $1=2$, on prend $Null(pred(x))$.

Plus généralement, pour invalider $x=y$, avec $x \leq y$, si $x=0$ on choisira $Null(x)$, sinon $Null(pred^x(x))$. Cette construction a un coût. C'est ce coût qu'on va économiser dans notre nouvelle présentation.

Chapitre 2

Une nouvelle présentation modulo de l'arithmétique de Heyting

2.1 Introduction

Notre objectif est de définir l'égalité par des règles de réécriture.

On garde les règles de réécriture sur l'addition et la multiplication, et on ajoute les quatre règles suivantes pour définir l'égalité :

$$\begin{aligned}0 = 0 &\longrightarrow \top \\S(x) = S(y) &\longrightarrow x = y \\S(x) = 0 &\longrightarrow \perp \\0 = S(x) &\longrightarrow \perp\end{aligned}$$

Regardons ce qu'est devenue notre preuve de $2 + 1 = 3$

$$\frac{\overline{\vdash_{HA_R} \top}}{\vdash_{HA_R} 2 + 1 = 3} \text{ réécriture}$$

Avec la réécriture sur l'addition, on met $2 + 1$ en une sorte de forme normale (une forme qui ne contient ni $+$ ni \times mais que des S), puis, à coup de réécriture par $S(x) = S(y) \longrightarrow x = y$, on arrive à $0 = 0$ qui se réécrit enfin en \top .

Comme les coups de réécriture ne comptent pour rien dans la hauteur de l'arbre, nous voici capables de prouver l'égalité de deux termes en temps constant égal à 1.

Encore mieux, on n'est capable, de la même façon, de invalider n'importe quelle inégalité de la même façon : on met en forme normale, on réécrit l'égalité de deux successeurs jusqu'à se trouver soit avec $0 = S(x)$ soit $S(x) = 0$, les deux se réécrivent sur \perp . C'est gagné.

Regardons maintenant le problème qui se pose avec cette présentation : que devient la règle de réécriture pour Leibniz ?

$$x = y \longrightarrow P(x) \Rightarrow P(y)$$

Elle crée un paire critique avec les nouvelles règles de réécriture. Il y aurait deux façons d'invalider $1 = 0$: réécrire directement en \perp avec la nouvelle règle ou avec les propositions invalidantes comme à la section précédente. On n'aurait donc rien gagné.

On veut circonscrire la réécriture de l'égalité entre deux termes par les règles ci-dessus.

Notre défi est de prouver qu'un système où l'égalité est définie par ces 4 règles est équivalent à un système où l'égalité est définie par Leibniz.

La deduction modulo permet d'intégrer la notion de *calcul* à nos preuves. Notre objectif est de construire une théorie équivalente à l'arithmétique de Heyting dont les preuves d'égalité ou de non égalité sur les termes soient de longueur 1, qu'il suffise de *calculer* sur les termes, et non plus de prouver. Il est en effet contre intuitif de devoir *raisonner* pour montrer que 3 est différents de 4, alors qu'il s'agit d'une *vérification*, une procédure, qu'un algorithme est capable de décider.

Nous partons de la présentation usuelle de l'arithmétique de Heyting HA.

On ne passe pas tout de suite à la présentation purement calculatoire qu'on note HA \rightarrow . La preuve d'équivalence entre le deux théories aurait été trop compliquée. Nous allons présenter quatre théories, qui, par transformations successives, vont nous mener de HA à HA \rightarrow . Nous donnons ci-dessous les grandes lignes des ces théories intermédiaires que nous développerons dans la suite.

- **HA $_R$** : Les axiomes sur l'addition et la multiplication sont remplacés par un ensemble de règles de réécriture de façon très intuitives (par exemple on remplace l'axiome $\forall x 0 + x = x$ par la règle de réécriture $0 + x \rightarrow x$).
On enlève l'axiome de Leibniz, c'est là le point le plus essentiel de notre travail car nous sommes privé du principe de substitutivité.
On ajoute 4 règles pour définir l'égalité.
On montre que cette théorie est équivalente à HA, en particulier que le principe de substitutivité est toujours valide, bien que l'axiome de Leibniz ne fasse plus partie de notre théorie.
- **HA $_N$** : On introduit un nouveau symbole de prédicat pour les entiers.
L'intuition est que le principe de récurrence définit les entiers.
On prouve que cette théorie est une extension conservative de HA $_R$.
- **HA $_K$** : On sorte la théorie. On ajoute des classes à nos objets, un symbole d'appartenance défini par : si un x valide une proposition P alors x appartient à la classe des objets qui valident P . Pour chaque P , on ajoute un symbole de fonction qui représente la classe des objets qui valident P .
On remplace le schéma d'axiomes de récurrence par une équivalence entre le fait d'être un entier et le principe de récurrence.
On prouve que cette théorie est une extension conservative de HA $_N$.
- **HA \rightarrow** : On transforme les axiomes en règles de réécriture.
HA \rightarrow est une présentation purement calculatoire de l'arithmétique de Heyting.

$$\frac{\frac{\frac{x = y \vdash x = y}{\vdash x = y \Rightarrow x = y} \Rightarrow \text{intro}}{\vdash S(x) = S(y) \Rightarrow x = y} 4}{\vdash \forall x \forall y (S(x) = S(y) \Rightarrow x = y)} \forall ix2$$

- $\forall x 0 = S(x) \Rightarrow \perp$
De même, en réécrivant $0 = S(x)$ par (2) on doit prouver $\forall x \perp \Rightarrow \perp$ — trivial.
- $\forall y (0 + y = y)$
De même, en réécrivant $0 + y$ par (5) on doit prouver $\forall y y = y$ — trivial.
- $\forall y (0 \times y = 0)$
De même, en réécrivant $0 \times y$ par (7) on doit prouver $\forall y 0 = 0$ — trivial.
- $\forall x \forall y (S(x) + y = S(x + y))$
De même, en réécrivant $S(x) + y$ par (6) on doit prouver $\forall x \forall y S(x + y) = S(x + y)$ — trivial.
- $\forall x \forall y (S(x) \times y = x \times y + y)$
De même, en réécrivant $S(x) \times y$ par (8) on doit prouver $\forall x \forall y x \times y + y = x \times y + y$ — trivial.
- $\forall a \forall b a = b \Rightarrow P(a) \Leftrightarrow P(b)$

C'est l'axiome Leibniz : le cas non trivial de la preuve.

Il est plus commode pour suivre cette preuve de noter explicitement la substitution :

Il nous faut trouver une preuve de

$$\vdash_{HA_R} \forall a \forall b a = b \Rightarrow P\{y := a\} \Leftrightarrow P\{y := b\}$$

Plan de la preuve :

Étape 1 :

Nous allons d'abord prouver un lemme plus faible : l'instance de ce théorème sur les termes.

Étape 2 :

Nous introduirons une nouvelle règle de déduction *Leibniz* grâce à laquelle on peut construire un arbre de dérivation dans HA_R de la proposition $\vdash_{HA_R} \forall a \forall b a = b \Rightarrow P\{y := a\} \Leftrightarrow P\{y := b\}$.

Étape 3 :

Nous prouverons que cette règle est admissible dans notre système.

Étape 1 :

Lemme 2.2.2 (substitutivité sur les termes)

Pour tout terme t on a $a = b \vdash_{HA_R} t\{y := a\} = t\{y := b\}$

La preuve de ce lemme nécessite les deux propriétés de congruence sur l'addition et la multiplication

énoncées ci-dessous dont les preuves sont données en annexe pages 57 et 59.

Lemme 2.2.3 (congruence de l'addition)

$$\vdash_{HA_R} \forall x \forall y \forall u \forall t \ x = y \Rightarrow u = t \Rightarrow x + u = y + t$$

Lemme 2.2.4 (congruence de la multiplication)

$$\vdash_{HA_R} \forall x \forall y \forall u \forall t \ x = y \Rightarrow u = t \Rightarrow x \times u = y \times t$$

On prouve le lemme de substitutivité sur les termes par récurrence structurale sur la forme de t :

* Cas de base : t est une variable

2 cas :

t est la variable y :

On veut construire une preuve de $a = b \vdash_{HA_R} y\{y := a\} = y\{y := b\}$

Ce qui revient à prouver $a = b \vdash_{HA_R} a = b$. On applique la règle d'axiome.

t est une variable différente de y (ou une constante) :

On veut construire une preuve de $a = b \vdash_{HA_R} z\{y := a\} = z\{y := b\}$

Ce qui revient à prouver $a = b \vdash_{HA_R} z = z$ - on utilise la réflexivité $\forall x \ x = x$ prouvée ci-dessus

* Cas récursifs

t est de la forme $S(t')$

On sait par hypothèse de récurrence que :

$$a = b \vdash_{HA_R} t\{y := a\} = t\{y := b\}$$

On veut montrer que

$$a = b \vdash_{HA_R} S(t)\{y := a\} = S(t)\{y := b\}$$

On fait rentrer la substitution

$$a = b \vdash_{HA_R} S(t\{y := a\}) = S(t\{y := b\})$$

On utilise la règle de réécriture 4 sur l'égalité de 2 successeurs

On retrouve $a = b \vdash_{HA_R} t\{y := a\} = t\{y := b\}$ dont on a une preuve par hypothèse de récurrence.

t est de la forme $(t1 + t2)$

On sait par hypothèse de récurrence que :

$$a = b \vdash_{HA_R} t1\{y := a\} = t1\{y := b\} \text{ et } a = b \vdash_{HA_R} t2\{y := a\} = t2\{y := b\}$$

On veut montrer que

$$a = b \vdash_{HA_R} (t1 + t2)\{y := a\} = (t1 + t2)\{y := b\}$$

On fait rentrer la substitution

$$a = b \vdash_{HA_R} (t1\{y := a\} + t2\{y := a\}) = (t1\{y := b\} + t2\{y := b\})$$

On utilise la congruence de l'addition (lemme 2.2.3) : $\forall x \forall y \forall u \forall t \ x = y \Rightarrow u = t \Rightarrow x + u = y + t$

On élimine $\forall x$ par $t1\{y := a\}$, $\forall y$ par $t1\{y := b\}$, $\forall u$ par $t2\{y := a\}$, $\forall t$ par $t2\{y := b\}$

On déduit $a = b \vdash_{HA_R} t1\{y := a\} + t2\{y := a\} = t1\{y := b\} + t2\{y := b\}$

t est de la forme $(t1 \times t2)$

On sait par hypothèse de récurrence que :

$$a = b \vdash_{HA_R} t1\{y := a\} = t1\{y := b\} \text{ et } a = b \vdash_{HA_R} t2\{y := a\} = t2\{y := b\}$$

On veut montrer que

$$a = b \vdash_{HA_R} (t1 \times t2)\{y := a\} = (t1 \times t2)\{y := b\}$$

On fait rentrer la substitution

$$a = b \vdash_{HA_R} (t1\{y := a\} \times t2\{y := a\}) = (t1\{y := b\} \times t2\{y := b\})$$

On utilise la congruence de la multiplication (lemme 2.2.4) : $\forall x \forall y \forall u \forall t \ x = y \Rightarrow u = t \Rightarrow x \times u = y \times t$

On élimine $\forall x$ par $t1\{y := a\}$, $\forall y$ par $t1\{y := b\}$, $\forall u$ par $t2\{y := a\}$, $\forall t$ par $t2\{y := b\}$

On déduit $a = b \vdash_{HA_R} t1\{y := a\} \times t2\{y := a\} = t1\{y := b\} \times t2\{y := b\}$

□

Étape 2 : Introduisons une nouvelle règle de déduction : *Leibniz*

$$\frac{\frac{\Gamma \vdash_{HA_R} a = b}{\Gamma, \Delta \vdash_{HA_R} P\{y := a\}} \quad \frac{\Delta \vdash_{HA_R} P\{y := a\}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\}}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\}} \text{Leibniz}$$

Avec cette nouvelle règle, nous construisons dans HA_R la preuve de l'axiome de Leibniz de HA comme suit :

$$\frac{\frac{\frac{\text{Ax}}{a = b \vdash_{HA_R} a = b} \quad \frac{\text{Ax}}{P\{y := a\} \vdash_{HA_R} P\{y := a\}}}{a = b, P\{y := a\} \vdash_{HA_R} P\{y := b\}} \text{Leibniz} \quad \frac{\frac{\text{Symétrie}}{a = b \vdash_{HA_R} b = a} \quad \frac{\text{Ax}}{P\{y := b\} \vdash_{HA_R} P\{y := b\}}}{a = b, P\{y := b\} \vdash_{HA_R} P\{y := a\}} \text{Leibniz}}{\frac{a = b \vdash_{HA_R} P\{y := a\} \Rightarrow P\{y := b\} \quad a = b \vdash_{HA_R} P\{y := b\} \Rightarrow P\{y := a\}}{a = b \vdash_{HA_R} P\{y := a\} \Leftrightarrow P\{y := b\}} \Rightarrow i \quad \Leftrightarrow i}}{\frac{\frac{a = b \vdash_{HA_R} P\{y := a\} \Leftrightarrow P\{y := b\}}{\vdash_{HA_R} a = b \Rightarrow P\{y := a\} \Leftrightarrow P\{y := b\}} \Rightarrow i}{\vdash_{HA_R} \forall a \forall b a = b \Rightarrow P\{y := a\} \Leftrightarrow P\{y := b\}} \forall i x 2}$$

Étape 3 : Nous allons maintenant prouver que la règle *Leibniz* est admissible, autrement dit

Lemme 2.2.5

Pour toute proposition P , si on a une preuve Π_1 de $\Gamma \vdash_{HA_R} a = b$ et une preuve Π_2 de $\Delta \vdash_{HA_R} P\{y := a\}$ alors on peut construire une preuve de $\Gamma, \Delta \vdash P\{y := b\}$

Preuve :

Nous allons faire une récurrence sur la forme de la proposition P .

Remarque : notons ici qu'on fait une dérivation sur la proposition, et pas sur une preuve

* Cas de base : P est une formule atomique.

Comme nous n'avons qu'un seul prédicat, l'égalité, une formule atomique P dans notre théorie ne peut avoir qu'une seule forme : l'égalité entre deux termes. P est donc de la forme $t1 = t2$

Il faut montrer que si on a une preuve Π_1 de $\Gamma \vdash_{HA_R} a = b$ et une preuve Π_2 de $\Delta \vdash_{HA_R} t1\{y := a\} = t2\{y := a\}$ alors on peut construire une preuve Π_3 de $\Gamma, \Delta \vdash_{HA_R} t1\{y := b\} = t2\{y := b\}$.

Nous disposons du lemme 2.2.2 de substitution sur les termes.

On se donne la transitivité et la symétrie dont les preuves sont données en annexe pages 54 et 53.

(Trans et TransEtSym sont deux règles admissibles une fois que la transitivité et la symétrie sont données).

A partir des preuves Π_1 et Π_2 on commence par construire une preuve de $\Gamma, \Delta \vdash_{HA_R} t1\{y := b\} = t2\{y := \mathbf{a}\}$, puis à l'aide de celle-ci on construit la preuve de $\Gamma, \Delta \vdash_{HA_R} t1\{y := b\} = t2\{y := \mathbf{b}\}$.

$$\frac{\frac{\Pi_1}{\Gamma \vdash_{HA_R} a = b} \quad \frac{\text{Lemme 2.2.2}}{a = b \vdash_{HA_R} t1\{y := a\} = t1\{y := b\}}}{\frac{\vdash_{HA_R} a = b \Rightarrow t1\{y := a\} = t1\{y := b\}}{\Gamma, \Delta \vdash_{HA_R} t1\{y := a\} = t1\{y := b\}}} \Rightarrow_i \quad \frac{\Pi_2}{\Delta \vdash_{HA_R} t1\{y := a\} = t2\{y := a\}}}{\Gamma, \Delta \vdash_{HA_R} t1\{y := b\} = t2\{y := a\}} \text{TransEtSym}$$

$$\frac{\frac{\text{preuve ci-dessus}}{\Gamma, \Delta \vdash_{HA_R} t1\{y := b\} = t2\{y := a\}} \quad \frac{\frac{\Pi_1}{\Gamma \vdash_{HA_R} a = b} \quad \frac{\text{Lemme 2.2.2}}{a = b \vdash_{HA_R} t2\{y := a\} = t2\{y := b\}}}{\frac{\vdash_{HA_R} a = b \Rightarrow t2\{y := a\} = t2\{y := b\}}{\Gamma, \Delta \vdash_{HA_R} t2\{y := a\} = t2\{y := b\}}} \Rightarrow_i}{\Gamma, \Delta \vdash_{HA_R} t1\{y := b\} = t2\{y := b\}} \text{Trans} \Rightarrow_e$$

* Cas récursifs

- Cas du \wedge :

Si $\Pi_1 : \Gamma \vdash_{HA_R} a = b$ et $\Pi_2 : \Delta \vdash_{HA_R} P\{y := a\} \wedge Q\{y := a\}$ alors $\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \wedge Q\{y := b\}$

Preuve formelle :

Comme on a une preuve de $\Delta \vdash_{HA_R} P\{y := a\} \wedge Q\{y := a\}$, par élimination du \wedge à droite, on a une preuve de $\Delta \vdash_{HA_R} P\{y := a\}$.

Comme par ailleurs on a une preuve de $\Gamma \vdash_{HA_R} a = b$, par application de l'hypothèse récurrence, on a une preuve de $\Gamma, \Delta \vdash_{HA_R} P\{y := b\}$.

De la même façon, par élimination du \wedge à gauche et par hypothèse de récurrence, on a une preuve de $\Gamma, \Delta \vdash_{HA_R} Q\{y := b\}$.

A partir de ces deux preuves et en introduisant un \wedge , on obtient une preuve de $\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \wedge Q\{y := b\}$.

Preuve informelle :

L'arbre ci-dessous n'est pas un arbre de dérivation - les doubles barres étiquetées HR dénotent l'application de l'hypothèse récurrence ce qui ne correspond pas à l'application d'une règle de déduction ni de réécriture. Néanmoins cette présentation plus visuelle est plus intuitive.

$$\frac{\frac{\Pi_1}{\Gamma \vdash_{HA_R} a = b} \quad \frac{\frac{\Pi_2}{\Delta \vdash_{HA_R} P\{y := a\} \wedge Q\{y := a\}}{\Delta \vdash_{HA_R} P\{y := a\}} \wedge_e}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\}} \text{HR}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \wedge Q\{y := b\}} \wedge_i \quad \frac{\frac{\Pi_1}{\Gamma \vdash_{HA_R} a = b} \quad \frac{\frac{\Pi_2}{\Delta \vdash_{HA_R} P\{y := a\} \wedge Q\{y := a\}}{\Delta \vdash_{HA_R} Q\{y := a\}} \wedge_e}{\Gamma, \Delta \vdash_{HA_R} Q\{y := b\}} \text{HR}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \wedge Q\{y := b\}} \wedge_i$$

- Le cas du \vee se prouve de façon similaire

Pour les autres cas de la preuve, nous donnons juste les dérivations informelles.

- Cas du \Rightarrow

Si $\Pi_1 : \Gamma \vdash_{HA_R} a = b$ et $\Pi_2 : \Delta \vdash_{HA_R} P\{y := a\} \Rightarrow Q\{y := a\}$ alors $\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \Rightarrow Q\{y := b\}$

Remarque : il y a une petite astuce grâce à la symétrie de l'égalité. L'hypothèse de récurrence utilise $b = a$ (et non $a = b$) pour faire apparaître $P\{y := b\}$ dans le contexte.

$$\frac{\frac{\frac{\frac{\frac{\frac{\Gamma \vdash_{HA_R} a = b}{\Pi_1}}{\Delta \vdash_{HA_R} P\{y := a\} \Rightarrow Q\{y := a\}}{\Pi_2}}{\Gamma, \Delta, P\{y := b\} \vdash_{HA_R} Q\{y := a\}}}{\Gamma \vdash_{HA_R} a = b} \text{HR}}{\Gamma, \Delta, P\{y := b\} \vdash_{HA_R} Q\{y := b\}} \Rightarrow i}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \Rightarrow Q\{y := b\}} \Rightarrow i$$

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\Gamma \vdash_{HA_R} a = b}{\Pi_1}}{\Gamma \vdash_{HA_R} b = a} \text{Sym}}{\Gamma, P\{y := b\} \vdash_{HA_R} P\{y := a\}} \Rightarrow e}{\Gamma, \Delta, P\{y := b\} \vdash_{HA_R} P\{y := a\}} \Rightarrow e}{\Gamma \vdash_{HA_R} a = b} \text{Ax}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\} \Rightarrow Q\{y := b\}} \text{HR}$$

- Cas du \exists

Si $\Pi_1 : \Gamma \vdash_{HA_R} a = b$ et $\Pi_2 : \Delta \vdash_{HA_R} (\exists x P)\{y := a\}$ alors $\Gamma, \Delta \vdash_{HA_R} (\exists x P)\{y := b\}$

Remarque : On suppose qu'on a respecté la convention de Barendregt sur les variables : x n'est libre ni dans a ni dans b ni dans P .

Cela permet de à faire un échange de l'ordre des substitutions dénoté par 'subst switch' qui n'est pas l'application d'une règle (la substitution n'étant pas un élément du langage , il s'agit même de la même proposition dans les deux cas). Nous préférons expliciter pour mieux suivre l'intuition de la preuve.

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\Gamma \vdash_{HA_R} a = b}{\Pi_1}}{\Delta \vdash_{HA_R} \exists x P\{y := a\}}{\Pi_2}}{\Delta \vdash_{HA_R} P\{y := a\}\{x := x'\}} \text{Ax}}{\Delta \vdash_{HA_R} P\{y := a\}\{x := x'\}} \text{Ax}}{\Delta \vdash_{HA_R} P\{x := x'\}\{y := a\}} \text{subst switch}}{\Gamma \vdash_{HA_R} a = b} \text{HR}}{\Gamma, \Delta \vdash_{HA_R} P\{x := x'\}\{y := b\}} \text{subst switch}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\}\{x := x'\}} \text{subst switch}}{\Gamma, \Delta \vdash_{HA_R} \exists x P\{y := b\}} \exists i$$

- Cas du \forall

Si $\Pi_1 : \Gamma \vdash_{HA_R} a = b$ et $\Pi_2 : \Delta \vdash_{HA_R} \forall x P\{y := a\}$ alors $\Gamma, \Delta \vdash_{HA_R} \forall x P\{y := b\}$

$$\frac{\frac{\frac{\frac{\frac{\frac{\Gamma \vdash_{HA_R} a = b}{\Pi_1}}{\Delta \vdash_{HA_R} \forall x P\{y := a\}}{\Pi_2}}{\Delta \vdash_{HA_R} P\{y := a\}} \text{Ve}}{\Gamma \vdash_{HA_R} a = b} \text{HR}}{\Gamma, \Delta \vdash_{HA_R} P\{y := b\}} \text{Vi}}{\Gamma, \Delta \vdash_{HA_R} \forall x P\{y := b\}} \forall i$$

Nous avons prouvé que tous les axiomes de HA sont prouvables dans HA_R . Toute preuve dans HA se transforme en une preuve dans HA_R en ajoutant les preuves des axiomes aux feuilles de la preuve.

Pour prouver l'équivalence des deux théories, reste à prouver que HA étend HA_R .

2.2.4 HA étend HA_R

Lemme 2.2.6

Si A est prouvable dans HA_R alors A est prouvable dans HA

On va chercher à transformer toute preuve de HA_R en une preuve de HA. HA_R est la théorie composée du schéma d'axiomes de récurrence, qui existe aussi dans HA, et des règles de réécriture sur l'addition, la multiplication et l'égalité.

Le schéma d'axiomes de récurrence de HA_R est le même que celui de HA, ainsi, si une preuve de A dans HA_R utilise une instance de cet axiome, alors la preuve de A dans HA peut l'utiliser de la même façon.

Reste le cas des propositions congruentes. En effet, il faut être capable de montrer que pour toutes propositions

A et B telles que $\vdash_{HA_R} A \equiv B$ on peut construire une preuve de $\vdash_{HA} A \Leftrightarrow B$, et ce quel que soit le nombre de pas de réécriture effectués.

Il suffit de s'occuper du cas où $\vdash_{HA_R} A \equiv B$ en un pas de réécriture, le cas en n pas de réécriture est un corollaire par transitivité de la relation d'équivalence. (le cas en 0 pas est trivial A et B sont la même proposition et on a évidemment $\vdash_{HA} A \Leftrightarrow A$)

Lemme 2.2.7

si $A \longrightarrow B$ en une étape dans HA_R alors on peut prouver que A est équivalent à B dans HA

Corollaire :

si $A \longrightarrow B$ en n étapes dans HA_R alors on peut prouver que A est équivalent à B dans HA .

Preuve du Lemme 2.2.7

On fait une récurrence structurale sur la proposition A .

- Cas de base - A est atomique

Par hypothèse, $A \longrightarrow B$. Il y a 8 cas de propositions atomiques qui peuvent se réécrire :

- * Soit A est une des propositions de gauche des 4 règles de réécriture sur les propositions

- $0 = 0 \longrightarrow \top$
- $0 = S(x) \longrightarrow \perp$
- $S(x) = 0 \longrightarrow \perp$
- $S(x) = S(y) \longrightarrow x = y$

- * Soit A est une proposition atomique qui contient un terme qui se réécrit

- $A(0 + y) \longrightarrow A(y)$
- $A(S(x) + y) \longrightarrow A(S(x + y))$
- $A(0 \times y) \longrightarrow A(0)$
- $A(S(x) \times y) \longrightarrow A(x \times y + y)$

Ci-dessous les 8 preuves correspondant à ces 8 cas :

- * A est une des propositions de gauche des 4 règles de réécriture sur les propositions

- $0 = 0 \longrightarrow \top$

On prouve que $0 = 0$ est équivalent à \top dans HA

$$\frac{\frac{\frac{}{0 = 0 \vdash_{HA} \top} \text{Ax}}{\vdash_{HA} 0 = 0 \Rightarrow \top} \Rightarrow i \quad \frac{\frac{\frac{}{\top \vdash_{HA} \forall x x = x} \text{Ax}}{\top \vdash_{HA} 0 = 0} \forall e}{\vdash_{HA} \top \Rightarrow 0 = 0} \Rightarrow i}{\vdash_{HA} 0 = 0 \Leftrightarrow \top} \Leftrightarrow i$$

- $0 = S(x) \longrightarrow \perp$

On prouve que $0 = S(x)$ est équivalent à \perp dans HA

$$\frac{\frac{\frac{}{\vdash_{HA} \forall x 0 = S(x) \Rightarrow \perp} \text{Ax}}{\vdash_{HA} 0 = S(x) \Rightarrow \perp} \forall e \quad \frac{\frac{\frac{}{\perp \vdash_{HA} \perp} \text{Ax}}{\perp \vdash_{HA} 0 = S(x)} \perp e}{\vdash_{HA} \perp \Rightarrow 0 = S(x)} \Rightarrow i}{\vdash_{HA} 0 = S(x) \Leftrightarrow \perp} \Leftrightarrow i$$

- $S(x) = 0 \longrightarrow \perp$

On prouve que $S(x) = 0$ est équivalent à \perp dans HA

$$\frac{\frac{\text{On admet la symétrie dans HA}}{S(x) = 0 \vdash_{HA} 0 = S(x)} \text{Ax} \quad \frac{\frac{\frac{\frac{\frac{\perp \vdash_{HA} \perp}{\perp \vdash_{HA} S(x) = 0}}{\perp \vdash_{HA} \perp} \text{Ax}}{\perp \vdash_{HA} S(x) = 0} \perp e}{\perp \vdash_{HA} \perp} \text{Ax}}{\perp \vdash_{HA} S(x) = 0} \perp e}{\vdash_{HA} S(x) = 0 \Rightarrow \perp} \Rightarrow e}{\vdash_{HA} S(x) = 0 \Rightarrow \perp} \Rightarrow i}{\vdash_{HA} S(x) = 0 \Leftrightarrow \perp} \Leftrightarrow i$$

- $S(x) = S(y) \longrightarrow x = y$

On prouve que $S(x) = S(y)$ est équivalent à $x = y$ dans HA

$$\frac{\frac{\frac{\frac{\frac{\text{Leibniz avec } P(z) : S(x) = S(z)}{\vdash_{HA} \forall x \forall y x = y \Rightarrow (S(x) = S(x) \Leftrightarrow S(x) = S(y))} \text{Ax}}{\vdash_{HA} x = y \Rightarrow (S(x) = S(x) \Leftrightarrow S(x) = S(y))} \text{Ve x2}}{x = y \vdash_{HA} S(x) = S(x) \Leftrightarrow S(x) = S(y)} \Leftrightarrow e}{x = y \vdash_{HA} (S(x) = S(x) \Rightarrow S(x) = S(y)) \wedge (S(x) = S(y) \Rightarrow S(x) = S(x))} \Leftrightarrow e}{x = y \vdash_{HA} S(x) = S(x) \Rightarrow S(x) = S(y)} \wedge e}{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\vdash_{HA} \forall x \forall y (S(x) = S(y) \Rightarrow x = y)} \text{Ax}}{\vdash_{HA} S(x) = S(y) \Rightarrow x = y} \text{Ve x2}}{x = y \vdash_{HA} S(x) = S(y)} \Rightarrow i}{\vdash_{HA} x = y \Rightarrow S(x) = S(y)} \Rightarrow i}{\vdash_{HA} S(x) = S(y) \Leftrightarrow x = y} \Leftrightarrow i}}{\vdash_{HA} \forall x x = x} \text{Ax}}{\vdash_{HA} S(x) = S(x)} \text{Ve}}{\vdash_{HA} S(x) = S(x)} \Rightarrow e}}{\vdash_{HA} S(x) = S(y) \Leftrightarrow x = y} \Rightarrow e$$

Remarque : Ve x2 dénote deux applications successives du $\forall e$

* A est une proposition atomique qui contient un terme qui se réécrit

- $A(0 + y) \longrightarrow A(y)$

On prouve que $A(0 + y)$ est équivalent à $A(y)$ dans HA

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\vdash_{HA} \forall y (0 + y = y)} \text{Ax}}{\vdash_{HA} 0 + y = y} \text{Ve}}{\vdash_{HA} \forall y (0 + y = y)} \text{Ax}}{\vdash_{HA} 0 + y = y} \text{Ve}}{\vdash_{HA} 0 + y = y} \text{Ve}}{\vdash_{HA} 0 + y = y} \text{Ve}}{\vdash_{HA} 0 + y = y} \text{Ve}}{\vdash_{HA} A(0 + y) \Leftrightarrow A(y)} \Rightarrow e$$

- $A(S(x) + y) \longrightarrow A(S(x + y))$

On prouve que $A(S(x) + y)$ est équivalent à $A(S(x + y))$ dans HA

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\vdash_{HA} \forall x \forall y S(x) + y = S(x + y)} \text{Ax}}{\vdash_{HA} S(x) + y = S(x + y)} \text{Ve x2}}{\vdash_{HA} \forall x \forall y S(x) + y = S(x + y)} \text{Ax}}{\vdash_{HA} S(x) + y = S(x + y)} \text{Ve x2}}{\vdash_{HA} S(x) + y = S(x + y)} \text{Ve x2}}{\vdash_{HA} S(x) + y = S(x + y)} \text{Ve x2}}{\vdash_{HA} S(x) + y = S(x + y)} \text{Ve x2}}{\vdash_{HA} A(S(x) + y) \Leftrightarrow A(S(x + y))} \Rightarrow e$$

- $A(0 \times y) \longrightarrow A(0)$

On prouve que $A(0 \times y)$ est équivalent à $A(0)$ dans HA

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{\vdash_{HA} \forall y (0 \times y = 0)} \text{Ax}}{\vdash_{HA} 0 \times y = 0} \text{Ve}}{\vdash_{HA} \forall y (0 \times y = 0)} \text{Ax}}{\vdash_{HA} 0 \times y = 0} \text{Ve}}{\vdash_{HA} 0 \times y = 0} \text{Ve}}{\vdash_{HA} 0 \times y = 0} \text{Ve}}{\vdash_{HA} 0 \times y = 0} \text{Ve}}{\vdash_{HA} 0 \times y = 0} \text{Ve}}{\vdash_{HA} A(0 \times y) \Leftrightarrow A(0)} \Rightarrow e$$

- $A(S(x) \times y) \longrightarrow A(x \times y + y)$

On prouve que $A(S(x) \times y)$ est équivalent à $A(x \times y + y)$ dans HA

$$\frac{\frac{\frac{\vdash_{HA} \forall x \forall y (S(x) \times y) = x \times y + y}{\vdash_{HA} (S(x) \times y) = x \times y + y} \text{Ax}}{\forall e \ x2} \quad \frac{\frac{\vdash_{HA} \forall x \forall y x = y \Rightarrow A(x) \Leftrightarrow A(y)}{\vdash_{HA} (S(x) \times y = x \times y + y) \Rightarrow (A(S(x) \times y) \Leftrightarrow A(x \times y + y))} \text{Ax}}{\forall e \ x2}}{\vdash_{HA} A(S(x) \times y) \Leftrightarrow A(x \times y + y)} \Rightarrow e$$

- Cas récursifs

Pour prouver les cas récursifs $A \wedge B \longrightarrow C$, $A \vee B \longrightarrow C$, $A \Rightarrow B \longrightarrow C$, $\forall x A \longrightarrow C$ et $\exists x A \longrightarrow C$, nous avons besoin des deux lemmes suivants :

Lemme 2.2.8

Si $A \wedge B \longrightarrow C$ en un pas de réécriture alors

- soit $A \longrightarrow A'$ et $C \equiv A' \wedge B$

- soit $B \longrightarrow B'$ et $C \equiv A \wedge B'$

De même si $A \vee B$ ou $A \Rightarrow B$.

Si $\forall x A \longrightarrow C$ alors $A \longrightarrow A'$ et $C \equiv \forall x A'$.

Si $\exists x A \longrightarrow C$ alors $A \longrightarrow A'$ et $C \equiv \exists x A'$.

Preuve :

Il n'y a pas de règle de réécriture qui contienne de connecteur ni de quantificateur à gauche, donc si on réécrit une proposition non atomique, c'est nécessairement une sous formule qui se réécrit.

Lemme 2.2.9

Si $A \Leftrightarrow A'$ alors $(A \wedge B) \Leftrightarrow (A' \wedge B)$, $(B \wedge A) \Leftrightarrow (B \wedge A')$, $(A \vee B) \Leftrightarrow (A' \vee B)$, $(B \vee A) \Leftrightarrow (B \vee A')$, $(A \Rightarrow B) \Leftrightarrow (A' \Rightarrow B)$, $(B \Rightarrow A) \Leftrightarrow (B \Rightarrow A')$, $(\forall x A) \Leftrightarrow (\forall x A')$ et $(\exists x A) \Leftrightarrow (\exists x A')$

Preuve : trivial

Retour aux cas récursifs

* $A \wedge B \longrightarrow C$

Il y a deux cas d'après le lemme 2.2.8

- $A \longrightarrow A'$ et $C \equiv A' \wedge B$

Par hypothèse de récurrence, $A \Leftrightarrow A'$. D'après le lemme 2.2.9 $A \wedge B \Leftrightarrow C$

- $B \longrightarrow B'$ et $C \equiv A \wedge B'$

Par hypothèse de récurrence, $B \Leftrightarrow B'$. D'après le lemme 2.2.9 $A \wedge B \Leftrightarrow C$

* $A \vee B \longrightarrow C$

Il y a deux cas d'après le lemme 2.2.8

- $A \longrightarrow A'$ et $C \equiv A' \vee B$

Par hypothèse de récurrence, $A \Leftrightarrow A'$. D'après le lemme 2.2.9 $A \vee B \Leftrightarrow C$

- $B \longrightarrow B'$ et $C \equiv A \vee B'$

Par hypothèse de récurrence, $B \Leftrightarrow B'$. D'après le lemme 2.2.9 $A \vee B \Leftrightarrow C$

* $A \Rightarrow B \longrightarrow C$

Il y a deux cas d'après le lemme 2.2.8

- $A \longrightarrow A'$ et $C \equiv A' \Rightarrow B$

Par hypothèse de récurrence, $A \Leftrightarrow A'$. D'après le lemme 2.2.9 $A \Rightarrow B \Leftrightarrow C$
 - $B \longrightarrow B'$ et $C \equiv A \Rightarrow B'$
 Par hypothèse de récurrence, $B \Leftrightarrow B'$. D'après le lemme 2.2.9 $A \Rightarrow B \Leftrightarrow C$

* $\forall x A \longrightarrow C$

Il y a un cas d'après le lemme 2.2.8

- $A \longrightarrow A'$ et $C \equiv \forall x A'$

Par hypothèse de récurrence, $A \Leftrightarrow A'$. D'après le lemme 2.2.9 $\forall x A \Leftrightarrow C$

* $\exists x A \longrightarrow C$

Il y a un cas d'après le lemme 2.2.8

- $A \longrightarrow A'$ et $C \equiv \exists x A'$

Par hypothèse de récurrence, $A \Leftrightarrow A'$. D'après le lemme 2.2.9 $\exists x A \Leftrightarrow C$

On a prouvé qu'on pouvait transformer toute preuve de HA en une preuve de HA_R et toute preuve de HA_R en une preuve de HA. HA et HA_R sont donc deux théories équivalentes.

2.3 HA_N , une extension conservative de HA_R

2.3.1 Motivations

HA_R est la théorie composée du schéma d'axiomes de récurrence et des règles de réécriture sur l'addition, la multiplication et l'égalité. Notre objectif est, rappelons-le, d'obtenir une théorie purement calculatoire, soit sans axiome avec uniquement des règles de réécriture.

Intuitivement, on cherche maintenant où mettre une flèche dans :

$$(P\{x := 0\} \wedge \forall y (P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow \forall n P\{x := n\}$$

A priori on ne voit pas comment transformer cet axiome en une règle de réécriture.

On aborde donc la question différemment. Le principe de récurrence définit d'une certaine façon les entiers. On ajoute un symbole de prédicat N qui intuitivement correspond au fait d'être un entier. Le schéma d'axiomes de récurrence est réécrit de la façon suivante :

$$\forall n N(n) \Rightarrow (P\{x := 0\} \wedge \forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow P\{x := n\}$$

L'intuition est de réécrire le fait d'être un entier en la récurrence elle-même. Pour l'instant, dans HA_N , on se concentre sur le fait de prouver que ce nouveau schéma est équivalent à l'ancien. On s'attachera ensuite à transformer l'implication ci-dessus en une équivalence.

Le langage a été enrichi, HA_N est une *extension conservative* de HA_R .

2.3.2 Présentation de HA_N

On ajoute le prédicat N qui correspond intuitivement au fait d'être un entier.

On modifie le schéma d'axiome comme suit

$$\forall n N(n) \Rightarrow (P\{x := 0\} \wedge \forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow P\{x := n\}$$

On ajoute 2 axiomes :

$$N(0)$$

$$\forall x N(x) \Rightarrow N(S(x))$$

Et on garde les règles de réécriture

- | | |
|---|--|
| (1) $0 = 0 \longrightarrow \top$ | (5) $0 + y \longrightarrow y$ |
| (2) $0 = S(x) \longrightarrow \perp$ | (6) $S(x) + y \longrightarrow S(x + y)$ |
| (3) $S(x) = 0 \longrightarrow \perp$ | (7) $0 \times y \longrightarrow 0$ |
| (4) $S(x) = S(y) \longrightarrow x = y$ | (8) $S(x) \times y \longrightarrow x \times y + y$ |

Le langage est différent. Pour prouver que HA_N est une extension conservative de HA_R , nous allons avoir besoin d'une traduction de HA_R vers HA_N et une autre traduction de HA_N vers HA_R

Traduction | | de HA_R vers HA_N

$$|P| = P, \text{ si } P \text{ est atomique,}$$

$$|\top| = \top,$$

$$|\perp| = \perp,$$

$$|A \wedge B| = |A| \wedge |B|,$$

$$|A \vee B| = |A| \vee |B|,$$

$$|A \Rightarrow B| = |A| \Rightarrow |B|,$$

$$|\forall x A| = \forall x (N(x) \Rightarrow |A|),$$

$$|\exists x A| = \exists x (N(x) \wedge |A|).$$

Traduction * de HA_N vers HA_R

$$N(x)^* = \top,$$

$$P^* = P, \text{ pour les autres propositions atomiques,}$$

$$\top^* = \top,$$

$$\perp^* = \perp,$$

$$\begin{aligned}
 (A \wedge B)^* &= A^* \wedge B^*, \\
 (A \vee B)^* &= A^* \vee B^*, \\
 (A \Rightarrow B)^* &= A^* \Rightarrow B^*, \\
 (\forall x A)^* &= \forall x (A^*), \\
 (\exists x A)^* &= \exists x (A^*).
 \end{aligned}$$

Nous allons prouver que pour toute proposition close A , $\vdash_{HA_R} A$ si et seulement si $\vdash_{HA_N} |A|$. HA_N n'est pas une extension conservative de HA_R au sens propre, mais une extension conservative modulo la traduction $| \cdot |$. Cette traduction, très simple, ne fait que relativiser les quantificateurs avec le symbole de prédicat N . On se permettra donc un abus de langage quand on dira que HA_N est une extension conservative de HA_R .

2.3.3 Extension

Lemme 2.3.1

Pour toute proposition close A , si $\vdash_{HA_R} A$ alors $\vdash_{HA_N} |A|$.

Lemme 2.3.2

Pour toute proposition A et tout ensemble de variables \vec{z} tel que les variables libres de A sont incluses dans \vec{z} , si $\Gamma \vdash_{HA_R} A$ alors $|\Gamma|, N(\vec{z}) \vdash_{HA_N} |A|$.

Prouvons que le lemme 2.3.1 est une conséquence du lemme 2.3.2

On suppose le lemme 2.3.2 :

si $\vdash_{HA_R} A$ alors $N(\vec{z}) \vdash_{HA_N} |A|$

Soit A une proposition close, \vec{z} est un vecteur de variables quelconque.

Pour chaque variable de \vec{z} , on fait une étape d'élimination du \exists comme suit :

$$\frac{\frac{\overline{\vdash_{HA_N} N(0)} \text{Ax}}{\vdash_{HA_N} \exists x N(x)} \exists i \quad \frac{\Pi}{N(z_0), N(\vec{z}') \vdash_{HA_N} |A|} \exists e}{N(\vec{z}') \vdash_{HA_N} |A|} \exists e$$

On élimine de cette façon tous les $N(x)$ du contexte, et on prouve finalement $\vdash_{HA_N} |A|$.

Grâce au lemme 2.3.2, on a une preuve du lemme 2.3.1.

Prouvons maintenant le lemme 2.3.2

Preuve par récurrence sur la taille de la dérivation (nombre de règles appliquées) :

- Cas de base : On prouve $\Gamma \vdash_{HA_R} A$ en une étape de dérivation
Il y a deux possibilités :
 - A est une occurrence du schéma de récurrence
Dans ce cas, on a une preuve en une étape de $|\Gamma| \vdash_{HA_N} |A|$ car la traduction du schéma de récurrence de HA_R est le schéma de récurrence de HA_N .
 - C'est une instance de la règle d'axiome : $B \in \Gamma$ et $B \equiv A$. Les règles de réécritures de HA_R étant les mêmes que celles de HA_N , si $B \equiv A$ dans HA_R alors $|B| \equiv |A|$ dans HA_N . On a donc $|\Gamma| \vdash_{HA_N} |A|$ en une étape avec $|B| \in |\Gamma|$ et $|B| \equiv |A|$
- Cas rékursifs

Les cas où la dernière règle appliquée n'est pas un quantificateur se prouvent facilement. On en donne deux à titre d'exemple, on se concentrera ensuite sur les cas où la dernière règle appliquée est un règle

d'introduction ou d'élimination d'un quantificateur.

Deux exemples des cas récursifs simples

* La dernière règle appliquée dans la dérivation est $\wedge e$ à droite

$$\frac{\frac{\Pi}{\Gamma \vdash_{HA_R} A \wedge B}}{\Gamma \vdash_{HA_R} A} \wedge e$$

L'arbre suivant

$$\frac{\Pi}{\Gamma \vdash_{HA_R} A \wedge B}$$

est de taille inférieure. On lui applique l'hypothèse de récurrence :

$$\frac{|\Pi|}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} |A \wedge B|}$$

Or $|A \wedge B| = |A| \wedge |B|$, il ne reste plus qu'à faire un $\wedge e$ pour avoir la conclusion que l'on voulait

$$\frac{\frac{|\Pi|}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} |A| \wedge |B|}}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} |A|} \wedge e$$

* La dernière règle appliquée dans la dérivation est $\Rightarrow i$

$$\frac{\frac{\Pi}{\Gamma, A \vdash_{HA_R} B}}{\Gamma \vdash_{HA_R} A \Rightarrow B} \Rightarrow i$$

On applique l'hypothèse de récurrence sur un arbre de taille inférieure, il ne reste plus qu'à faire un $\Rightarrow i$

$$\frac{\frac{|\Pi|}{|\Gamma|, N(\vec{y}), |A| \vdash_{HA_N} |B|}}{\Gamma, N(\vec{y}) \vdash_{HA_N} |A| \Rightarrow |B|} \Rightarrow i$$

Les cas récursifs non triviaux

Pour les cas avec quantificateur, nous aurons besoin du lemme suivant dont la preuve est donnée en annexe page 52 :

Lemme 2.3.3

$N(\vec{z}) \vdash_{HA_N} N(t)$ pour tout t avec $FV(t) = \vec{z}$

* La dernière règle appliquée dans la dérivation est $\forall e$

$$\frac{\frac{\Pi}{\Gamma \vdash_{HA_R} \forall x A}}{\Gamma \vdash_{HA_R} A\{x := t\}} \forall e$$

On applique l'hypothèse de récurrence sur un arbre de taille inférieure,

$$\frac{\frac{\frac{|\Pi|}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} \forall x (N(x) \Rightarrow |A|)}}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} (N(t) \Rightarrow |A|\{x := t\})} \forall e - FV(t) = \vec{z}}{|\Gamma|, N(\vec{y}), N(\vec{z}) \vdash_{HA_N} N(t) \Rightarrow |A|\{x := t\}} \text{WL}}{\frac{\frac{|\Gamma|, N(\vec{y}) \vdash_{HA_N} (N(t) \Rightarrow |A|\{x := t\})}{|\Gamma|, N(\vec{y}), N(FV(t)) \vdash_{HA_N} N(t)} \text{Lemme 2.3.3}}{|\Gamma|, N(\vec{y}), N(FV(t)) \vdash_{HA_N} |A|\{x := t\}} \Rightarrow e} \Rightarrow e$$

La règle WL d'affaiblissement à gauche est admissible dans notre système de déduction.

* La dernière règle appliquée dans la dérivation est $\forall i$

$$\frac{\frac{\Pi}{\Gamma \vdash_{HA_R} A}}{\Gamma \vdash_{HA_R} \forall x A} \forall i$$

On applique l'hypothèse de récurrence sur un arbre de taille inférieure,

$$\frac{\frac{\frac{|\Pi|}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} |A|}}{|\Gamma|, N(\vec{y}), N(x) \vdash_{HA_N} |A|} \text{WL}}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} N(x) \Rightarrow |A|} \Rightarrow i}{|\Gamma|, N(\vec{y}) \vdash_{HA_N} \forall x (N(x) \Rightarrow |A|)} \forall i$$

Remarque : si $x \in \vec{y}$ alors on ne fait pas l'affaiblissement et $N(x)$ sort du contexte

* La dernière règle appliquée dans la dérivation est $\exists i$

$$\frac{\frac{\Pi}{\Gamma \vdash_{HA_R} A\{x := t\}}}{\Gamma \vdash_{HA_R} \exists x A} \exists i$$

On applique l'hypothèse de récurrence sur un arbre de taille inférieure,

$$\frac{\frac{\frac{\Pi}{|\Gamma|, N(\vec{y}), N(FV(t)) \vdash_{HA_N} |A|\{x := t\}}}{|\Gamma|, N(\vec{y}), N(FV(t)) \vdash_{HA_N} N(t) \wedge |A|\{x := t\}} \text{Lemme 2.3.3}}{|\Gamma|, N(\vec{y}), N(FV(t)) \vdash_{HA_N} \exists x N(x) \wedge |A|} \exists i}{|\Gamma|, N(\vec{y}), N(FV(t)) \vdash_{HA_N} \exists x N(x) \wedge |A|} \wedge i$$

* La dernière règle appliquée dans la dérivation est $\exists e$

$$\frac{\frac{\Pi_1}{\Gamma \vdash_{HA_R} \exists x A} \quad \frac{\Pi_2}{\Gamma, A \vdash_{HA_R} B}}{\Gamma \vdash_{HA_R} B} \exists e$$

Remarque : si on a pu appliquer cette règle, on a respecté la condition de bord : x n'apparaît ni dans Γ ni dans B

On applique l'hypothèse de récurrence sur deux arbres de taille inférieure,

$$\frac{\frac{\frac{|\Pi_1|}{|\Gamma|, N(\vec{y}) \vdash_{HA_R} \exists x (N(x) \wedge |A|)}}{|\Gamma|, N(\vec{y}) \vdash_{HA_R} \exists x |A|} \quad \frac{\frac{\frac{N(x) \wedge |A| \vdash_{HA_R} N(x) \wedge |A|}{N(x) \wedge |A| \vdash_{HA_R} |A|} \text{Ax} \quad \wedge e}{N(x) \wedge |A| \vdash_{HA_R} \exists x |A|} \exists i}{|\Gamma|, N(\vec{y}) \vdash_{HA_R} \exists x |A|} \exists e \quad \frac{|\Pi_2|}{|\Gamma|, N(\vec{y}), |A| \vdash_{HA_R} |B|} \exists e}{|\Gamma|, N(\vec{y}) \vdash_{HA_R} |B|} \exists e$$

2.3.4 Conservativité

Lemme 2.3.4

Pour toute proposition close A de HA_R , si $|A|$ est prouvable dans HA_N alors A est prouvable dans HA_R

La preuve se fait en trois étapes :

- On prouve : si A est prouvable dans HA_N alors A^* est prouvable dans HA_R
- On en déduit le corollaire : pour toute proposition A de HA_R , si $|A|$ est prouvable dans HA_N alors $|A|^*$ est prouvable dans HA_R .
- On prouve dans HA_R que $|A|^*$ est équivalent A

Lemme 2.3.5

Si B est prouvable dans HA_N alors B^* est prouvable dans HA_R

Traduisons les axiomes de HA_N :

$$N(0)^* = \top$$

$$(\forall x N(x) \Rightarrow N(S(x)))^* = \forall x \top \Rightarrow \top$$

Ces deux propositions sont trivialement prouvables dans HA_R

$$\begin{aligned}
& (\forall n N(n) \Rightarrow (P\{x := 0\} \wedge (\forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\}))) \Rightarrow P\{x := n\})^* \\
& = \forall n \top \Rightarrow (P\{x := 0\} \wedge \forall y (\top \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow P\{x := n\}
\end{aligned}$$

La traduction du schéma de récurrence est équivalente à l'énoncé du schéma de récurrence dans HA_R

Les traductions des axiomes de HA_N sont prouvables dans HA_R .

Les règles de réécritures sont les mêmes.

On prouve par récurrence sur la dérivation qu'à toute preuve de A dans HA_N correspond une preuve de A^* dans HA_R .

Corollaire :

Pour toute proposition A de HA_R , si $|A|$ est prouvable dans HA_N alors $|A|^*$ est prouvable dans HA_R .

Lemme 2.3.6

$|A|^*$ et A sont équivalentes dans HA_R

Démonstration par récurrence sur la forme de A

- Cas de base - A est atomique (ou \top ou \perp)
 - $|A|$ ne contient pas de N car la traduction d'une proposition atomique est l'identité
 - $|A| = A$
 - $|A|^* = A^* = A$
 - On a bien $\vdash_{HA_R} |A|^* \Leftrightarrow A$

- Cas récurrents

- A est de la forme $(A \wedge B)$

$$|(A \wedge B)| = |A| \wedge |B|$$

$$(|A| \wedge |B|)^* = |A|^* \wedge |B|^*$$

Par hypothèse de récurrence, $\vdash_{HA_R} |A|^* \Leftrightarrow A$ et $\vdash_{HA_R} |B|^* \Leftrightarrow B$

d'où $\vdash_{HA_R} (|A|^* \wedge |B|^*) \Leftrightarrow (A \wedge B)$

- A est de la forme $(A \vee B)$

$$|(A \vee B)| = |A| \vee |B|$$

$$(|A| \vee |B|)^* = |A|^* \vee |B|^*$$

Par hypothèse de récurrence, $\vdash_{HA_R} |A|^* \Leftrightarrow A$ et $\vdash_{HA_R} |B|^* \Leftrightarrow B$

d'où $\vdash_{HA_R} (|A|^* \vee |B|^*) \Leftrightarrow (A \vee B)$

- A est de la forme $(A \Rightarrow B)$

$$|(A \Rightarrow B)| = |A| \Rightarrow |B|$$

$$(|A| \Rightarrow |B|)^* = |A|^* \Rightarrow |B|^*$$

Par hypothèse de récurrence, $\vdash_{HA_R} |A|^* \Leftrightarrow A$ et $\vdash_{HA_R} |B|^* \Leftrightarrow B$

d'où $\vdash_{HA_R} (|A|^* \Rightarrow |B|^*) \Leftrightarrow (A \Rightarrow B)$

- A est de la forme $\forall x A$

$$|\forall x A| = \forall x (N(x) \Rightarrow |A|),$$

$$(\forall x (N(x) \Rightarrow |A|))^* = \forall x (\top \Rightarrow |A|^*)$$

Par hypothèse de récurrence, $\vdash_{HA_R} |A|^* \Leftrightarrow A$

Or $\vdash_{HA_R} (\top \Rightarrow A) \Leftrightarrow A$

d'où $\vdash_{HA_R} \forall x (\top \Rightarrow |A|^*) \Leftrightarrow \forall x A$

- A est de la forme $\exists x A$

$$|\exists x A| = \exists x (N(x) \wedge |A|^*)$$

$$(\exists x (N(x) \wedge |A|^*))^* = \exists x (\top \wedge |A|^*)$$

Par hypothèse de récurrence, $\vdash_{HA_R} |A|^* \Leftrightarrow A$

Or $\vdash_{HA_R} (\top \wedge A) \Leftrightarrow A$

d'où $\vdash_{HA_R} \exists x (\top \wedge |A|^*) \Leftrightarrow \exists x A$

On a prouvé que pour toute proposition close A de \mathbf{HA}_R , si $\vdash_{HA_R} A$ alors $\vdash_{HA_N} |A|$, et si $\vdash_{HA_N} |A|$ alors $\vdash_{HA_R} A$. \mathbf{HA}_N est bien une extension conservative (modulo $| \cdot |$) de \mathbf{HA}_R .

2.4 HA_K , une extension conservative de HA_N

2.4.1 Motivations

L'idée est de remplacer le schéma d'axiomes de récurrence de HA_N

$$\forall n N(n) \Rightarrow (P\{x := 0\} \wedge \forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow P\{x := n\}$$

par une règle de réécriture. Pour pouvoir mettre une règle de réécriture à la place d'un axiome et obtenir une théorie équivalente, il faut que cet axiome soit sous forme d'une équivalence (d'après le théorème 1.3.1 page 9 sur l'équivalence entre théories modulo et théories axiomatiques).

Intuitivement, cette équivalence est :

$$\forall n (N(n) \Leftrightarrow \forall P (P\{x := 0\} \wedge \forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\})) \Rightarrow P\{x := n\})$$

Quel est le problème de cette proposition ?

Elle quantifie sur deux variables, n et P .

n parcourt les termes et P parcourt les propositions. On veut être capable de distinguer, syntaxiquement, ces deux types de variables. Pour cela, on va sortir notre théorie et ajouter un symbole \in pour l'appartenance :

- ι est la sorte des éléments
- κ est la sorte des ensembles

Pour chaque proposition P qu'on peut former dans le langage, on ajoute un symbole de fonction f_P qui modélise l'ensemble des éléments qui vérifient la proposition P .

$x \in f_P \Leftrightarrow P(x)$, où $f_P = \{x \mid P(x) \text{ est vrai}\}$

Cette équivalence vaut si P n'a qu'une variable libre.

Prenons maintenant une proposition P dont les variables libres sont z, y_1, \dots, y_n . On aurait une équivalence entre deux propositions, $x \in f_P$ et $P(z, y_1, \dots, y_n)\{z := x\}$ dont les variables libres sont différentes. C'est un problème. Il ne suffit donc pas de créer un symbole de fonction f_P d'arité 0, il faut que ce symbole soit d'une arité une fois inférieur au nombre de variables libres de P . L'équivalence devient

$$x \in f_{y_1, \dots, y_n, P}(y_1, \dots, y_n) \Leftrightarrow P(z, y_1, \dots, y_n)\{z := x\}$$

On se permettra d'écrire $P\{z := x\}$ à la place de $P(z, y_1, \dots, y_n)\{z := x\}$

2.4.2 Présentation de HA_K

Le langage

On ajoute 2 sortes ι et κ , et un symbole \in

$$\begin{array}{ll} 0 : \iota & S : \langle \iota, \iota \rangle \\ + : \langle \iota, \iota, \iota \rangle & \times : \langle \iota, \iota, \iota \rangle \\ = : \langle \iota, \iota \rangle & N : \langle \iota \rangle \\ \in : \langle \iota, \kappa \rangle & \end{array}$$

Pour toute proposition P de HA_N , avec $FV(P) = z, y_1, \dots, y_n$, on ajoute un nouveau symbole de fonction $f_{z, y_1, \dots, y_n, P}$ de rang $\underbrace{\langle \iota, \dots, \iota, \kappa \rangle}_{n \text{ fois}}$.

Remarquons que P est dans HA_N , il ne contient donc pas de symbole de fonction comme ceux introduits ci dessus.

La théorie

On ajoute un schéma d'axiomes pour l'appartenance - dit *schéma de compréhension*

$$\forall x \forall y_1 \dots \forall y_n (x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \Leftrightarrow P\{z := x\})$$

On passe du schéma d'axiomes de récurrence à un seul axiome

$$\forall n (N(n) \Leftrightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f))$$

on garde les règles de réécriture

- | | |
|---|--|
| (1) $0 = 0 \longrightarrow \top$ | (5) $0 + y \longrightarrow y$ |
| (2) $0 = S(x) \longrightarrow \perp$ | (6) $S(x) + y \longrightarrow S(x + y)$ |
| (3) $S(x) = 0 \longrightarrow \perp$ | (7) $0 \times y \longrightarrow 0$ |
| (4) $S(x) = S(y) \longrightarrow x = y$ | (8) $S(x) \times y \longrightarrow x \times y + y$ |

2.4.3 Extension

Lemme 2.4.1

Si A est prouvable dans HA_N alors A est prouvable dans HA_K

Les règles de réécriture sont les mêmes.

On prouve les axiomes de HA_N avec les axiomes de HA_K .

On pourra à partir d'une preuve de A dans HA_N en construire une dans HA_K . Ce sera la preuve de HA_N à laquelle on aura greffé aux feuilles les preuves des axiomes de HA_N dans HA_K .

Les axiomes de HA_N sont :

- $N(0)$
- $\forall x N(x) \Rightarrow N(S(x))$
- $\forall n N(n) \Rightarrow P\{x := 0\} \Rightarrow \forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\}) \Rightarrow P\{x := n\}$

On utilisera les trois règles suivantes :

$$\frac{\Gamma \vdash_{HA_K} N(t)}{\Gamma \vdash_{HA_K} \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow t \in f)} \text{Eq1}$$

$$\frac{\Gamma \vdash_{HA_K} \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow t \in f)}{\Gamma \vdash_{HA_K} N(t)} \text{Eq2}$$

$$\frac{\Gamma \vdash_{HA_K} 0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow t \in f}{\Gamma \vdash_{HA_K} P\{x := 0\} \Rightarrow \forall y (N(y) \Rightarrow P\{x := y\} \Rightarrow P\{x := S(y)\}) \Rightarrow P\{x := n\}} \text{Eq3}$$

Les deux premières règles Eq1 et Eq2 sont admissibles dans HA_K car l'équivalence suivante est un axiome de HA_K :

$$\forall n (N(n) \Leftrightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f))$$

La règle Eq3 est admissible en utilisant 4 fois le schéma de compréhension :

$$\forall x \forall y_1 \dots \forall y_n (x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \Leftrightarrow P\{z := x\})$$

Nous allons prouver les 3 axiomes de HA_N dans HA_K

2.4.4 Conservativité

Lemme 2.4.2

Si A , proposition close du langage de HA_N , est prouvable dans HA_K , alors A est prouvable dans HA_N

Nous proposons ici une preuve par modèle. On utilise les algèbres de Heyting comme modèles de la logique intuitioniste.

Définition 2.4.1 (Fonction définissable dans HA_N)

Soit un modèle \mathcal{M} de HA_N . Une fonction γ de M dans B est définissable s'il existe une proposition P du langage de HA_N avec $FV(P) = \{x, y_1, \dots, y_n\}$ et une assignation Φ qui attribue des valeurs b_1, \dots, b_n de M à tous les y_1, \dots, y_n telles que : $\gamma(a) = \llbracket P \rrbracket_{\Phi, x:=a}$

Plan de la preuve :

On part d'un modèle \mathcal{M}_N quelconque de HA_N . On montre qu'on peut étendre ce modèle à un modèle \mathcal{M}_K de HA_K . Comme A est un théorème de HA_K , \mathcal{M}_K valide A . Or on aura construit l'extension de \mathcal{M}_N à \mathcal{M}_K de façon à ne rien changer à la sémantique de A . On conclut que A était valide dans \mathcal{M}_N . La proposition A étant valide dans tous les modèles de HA_N , c'est un théorème de HA_N .

Soit \mathcal{M}_N un modèle de HA_N .

Soient M_N le domaine de \mathcal{M}_N et B son algèbre de Heyting.

Extension de \mathcal{M}_N à \mathcal{M}_K

Soit $M_l = M_N$.

Soit M_κ l'ensemble des fonctions définissables de M_l dans B . Le domaine M_K de \mathcal{M}_K est composé des ensembles M_κ et M_l . Les variables s'interprètent dans M_l .

Tous les symboles de HA_N ont la même dénotation dans \mathcal{M}_K que dans \mathcal{M}_N .

Interprétation des symboles de HA_K qui n'apparaissent pas dans HA_N :

- Interprétation des symboles de fonctions :

On interprète le symbole de fonction $f_{z, y_1, \dots, y_n, P}$ par une fonction F de M_l^n dans M_κ . F reçoit n éléments de M_l et en fait une assignation ϕ pour renvoyer la fonction définissable de M_κ associée à P avec l'assignation ϕ .

Il n'est pas inutile ici de rappeler comment ces symboles ont été ajoutés à notre langage : chaque symbole de fonction a été en effet associé à une proposition P dont les variables libres sont z, y_1, \dots, y_n .

Les n premiers arguments de F définissent une assignation Φ pour les variables y_1, \dots, y_n . Le résultat de $F(\Phi(y_1), \dots, \Phi(y_n))$ est une fonction définissable γ (élément de M_κ). Cette fonction définissable recevra une valeur a (de M_l) pour z et renverra la valeur de vérité de $P(a)$ (avec les autres variables libres de P instanciées par Φ).

- Interprétation du symbole \in

On interprète l'appartenance par l'application suivante :

$$\llbracket x \in E \rrbracket = \llbracket E \rrbracket \llbracket x \rrbracket$$

Prouvons que \mathcal{M}_K est un modèle de HA_K

Pour cela, on montre que les axiomes de HA_K sont valides dans \mathcal{M}_K .

- $\forall x \forall y_1 \dots \forall y_n (x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \Leftrightarrow P\{z := x\})$

On veut montrer que

$$\llbracket \forall x \forall y_1 \dots \forall y_n (x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \Leftrightarrow P\{z := x\}) \rrbracket = \tilde{\top}$$

Cela revient à montrer que pour tout a, b_1, \dots, b_n de M_l

$$\llbracket x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \Leftrightarrow P\{z := x\} \rrbracket_{x:=a, y_1:=b_1, \dots, y_n:=b_n} = \tilde{\top}$$

Appelons Φ cette assignation.

On doit montrer maintenant que

$$\llbracket x \in f_{z,y_1,\dots,y_n,P}(y_1,\dots,y_n) \rrbracket_{\Phi} = \llbracket P\{z := x\} \rrbracket_{\Phi}$$

Concentrons nous sur la première partie de l'égalité :

$$\llbracket x \in f_{z,y_1,\dots,y_n,P}(y_1,\dots,y_n) \rrbracket_{\Phi} = \llbracket f_{z,y_1,\dots,y_n,P}(y_1,\dots,y_n) \rrbracket_{\Phi} \llbracket x \rrbracket_{\Phi}$$

On a :

$$\llbracket f_{z,y_1,\dots,y_n,P}(y_1,\dots,y_n) \rrbracket_{\Phi} = \llbracket f_{z,y_1,\dots,y_n,P} \rrbracket_{\Phi}(\llbracket y_1 \rrbracket_{\Phi}, \dots, \llbracket y_n \rrbracket_{\Phi}) = \llbracket f_{z,y_1,\dots,y_n,P} \rrbracket_{\Phi}(b_1, \dots, b_n)$$

Or, par l'interprétation qu'on a donné aux symboles de fonctions, $\llbracket f_{z,y_1,\dots,y_n,P} \rrbracket_{\Phi}(b_1, \dots, b_n)$ est la fonction définissable γ de \mathcal{M}_K associée à P avec l'affectation Φ' b_1, \dots, b_n à y_1, \dots, y_n .

On a donc :

$$(\llbracket f_{z,y_1,\dots,y_n,P} \rrbracket_{\Phi}(b_1, \dots, b_n)) \llbracket x \rrbracket_{\Phi} = \gamma a$$

Et par définition des fonctions définissables,

$$\gamma a = \llbracket P \rrbracket_{\Phi', z:=a}$$

Comme x n'apparaît pas libre dans P , on peut ajouter $x := a$ à Φ' et retomber sur Φ , car les valeurs assignées à y_1, \dots, y_n par Φ et Φ' sont identiques. On a

$$\gamma a = \llbracket P \rrbracket_{\Phi, z:=a}$$

Regardons maintenant ce qu'il en est de $\llbracket P\{z := x\} \rrbracket_{\Phi}$

$$\llbracket P\{z := x\} \rrbracket_{\Phi} = \llbracket P \rrbracket_{\Phi, z:=\llbracket x \rrbracket_{\Phi}} = \llbracket P \rrbracket_{\Phi, z:=a}$$

On a finalement pour toute interprétation Φ

$$\llbracket x \in f_{z,y_1,\dots,y_n,P}(y_1,\dots,y_n) \rrbracket_{\Phi} = \llbracket P\{z := x\} \rrbracket_{\Phi}$$

On peut conclure que

$$\llbracket \forall x \forall y_1 \dots \forall y_n (x \in f_{z,y_1,\dots,y_n,P}(y_1, \dots, y_n) \Leftrightarrow P\{z := x\}) \rrbracket = \tilde{\top}$$

- $\forall n (N(n) \Leftrightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f))$

On veut montrer que cette proposition est valide dans \mathcal{M}_K . Autrement dit, que pour tout a de M_L ,

$$\llbracket N(n) \Leftrightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket_{n:=a} = \tilde{\top}$$

On va montrer successivement que

$$\llbracket N(n) \Rightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket_{n:=a} = \tilde{\top}$$

et

$$\llbracket \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \Rightarrow N(n) \rrbracket_{n:=a} = \tilde{\top}$$

* $\llbracket N(n) \Rightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket_{n:=a}$
 Comme f n'apparaît pas dans $N(n)$, on peut sortir le $\forall f$ de l'implication

$$\llbracket \forall f (N(n) \Rightarrow (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f)) \rrbracket_{n:=a}$$

Autrement dit, pour tout γ de M_κ (γ défini par une proposition P dont les variables libres sont z, y_1, \dots, y_n et une assignation b_1, \dots, b_n pour les y_1, \dots, y_n)

$$\llbracket N(n) \Rightarrow (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket_{n:=a, f:=\gamma, \Phi}$$

Comme γ est associée à P , on sait que $\llbracket x \in \gamma \rrbracket_\Phi = \llbracket P\{z := x\} \rrbracket_\Phi$

On peut donc substituer

$$\llbracket N(n) \Rightarrow (P\{z := 0\} \Rightarrow \forall y (N(y) \Rightarrow P\{z := y\} \Rightarrow P\{z := S(y)\}) \Rightarrow P\{z := S(y)\}) \rrbracket_{n:=a, \Phi}$$

Cette proposition est prouvable dans HA_N , elle est donc valide dans \mathcal{M}_N . Et comme les propositions des HA_N ont la même dénotation dans \mathcal{M}_N et \mathcal{M}_K , elle est aussi valide dans \mathcal{M}_K .

* $\llbracket \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \Rightarrow N(n) \rrbracket_{n:=a}$

Dans cette proposition, le $\forall f$ est en position négative (à gauche d'une implication).

La proposition est de la forme $(\forall x A(x)) \Rightarrow B$.

On peut montrer en déduction naturelle intuitioniste que $(\exists x (A(x) \Rightarrow B) \Rightarrow ((\forall x, A(x)) \Rightarrow B))$.

Nous allons nous intéresser à la sémantique de la proposition 'sous sa forme' $(\exists x (A(x) \Rightarrow B))$, et nous pourrons en déduire que la sémantique de $(\forall x A(x)) \Rightarrow B$ est vraie aussi.

On veut montrer

$$\llbracket \exists f ((0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \Rightarrow N(n)) \rrbracket_{n:=a} = \tilde{\top}$$

Nous allons prendre comme valeur de M_κ pour f la fonction γ définie par la proposition $N(z)$ avec cette seule variable libre.

$$\llbracket (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \Rightarrow N(n) \rrbracket_{n:=a, f:=\gamma} = \tilde{\top}$$

Comme γ est associée à $N(z)$, on sait que $\llbracket x \in \gamma \rrbracket_\Phi = \llbracket N(z)\{z := x\} \rrbracket$. On a donc

$$\llbracket (N(0) \Rightarrow \forall y (N(y) \Rightarrow N(y) \Rightarrow N(S(y))) \Rightarrow N(n)) \Rightarrow N(n) \rrbracket_{n:=a} = \tilde{\top}$$

La proposition $(N(0) \Rightarrow \forall y (N(y) \Rightarrow N(y) \Rightarrow N(S(y))) \Rightarrow N(n)) \Rightarrow N(n)$ est prouvable dans HA_N . Son interprétation dans \mathcal{M}_K est donc $\tilde{\top}$. Comme nous avons construit \mathcal{M}_K tel que les interprétations des formules de HA_N soient les mêmes dans \mathcal{M}_K et \mathcal{M}_N , cette proposition s'interprète aussi en $\tilde{\top}$ dans \mathcal{M}_K .

Nous avons prouvé la validité de tous les axiomes de HA_K dans \mathcal{M}_K . \mathcal{M}_K est donc un modèle de HA_K .

Récapitulons : Nous avons étendu un modèle \mathcal{M}_N quelconque de HA_N en un modèle \mathcal{M}_K . Nous venons de prouver que \mathcal{M}_K est un modèle de HA_K . Soit A une proposition de HA_N telle que \mathcal{M}_K valide A .

Comme \mathcal{M}_K est une extension de \mathcal{M}_N , A est aussi valide dans \mathcal{M}_N .

2.5 HA_{\rightarrow} , une présentation purement calculatoire de l'Arithmétique de Heyting

2.5.1 Motivations

Ce qui nous intéresse, c'est l'aspect calculatoire intrinsèque aux théories purement calculatoire. Les preuves sont simplifiées, elles sont plus courtes. L'arithmétique est un exemple où l'introduction du calcul dans le raisonnement est complètement intuitif. Nous avons réussi à définir tous les symboles de fonction et de prédicat de la théorie de l'arithmétique avec des règles de réécriture, et ce de façon plus efficace que la présentation précédente [2], car notre définition de l'égalité tient compte de sa décidabilité par le calcul, tout en conservant le principe de substitutivité.

2.5.2 La théorie

$$x \in f_{z,y_1,\dots,y_n,P}(y_1,\dots,y_n) \longrightarrow P\{z := x\}$$

$$N(n) \longrightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f)$$

- | | |
|---|--|
| (1) $0 = 0 \longrightarrow \top$ | (5) $0 + y \longrightarrow y$ |
| (2) $0 = S(x) \longrightarrow \perp$ | (6) $S(x) + y \longrightarrow S(x + y)$ |
| (3) $S(x) = 0 \longrightarrow \perp$ | (7) $0 \times y \longrightarrow 0$ |
| (4) $S(x) = S(y) \longrightarrow x = y$ | (8) $S(x) \times y \longrightarrow x \times y + y$ |

2.5.3 HA_{\rightarrow} , une théorie équivalente à HA_K

On a remplacé deux axiomes sous forme d'équivalence en deux règles de réécriture. D'après le théorème 1.3.1 sur l'équivalence de théories modulo et théories axiomatiques, les deux théories sont équivalentes.

2.6 HA \longrightarrow , une théorie qui a la propriété d'élimination des coupures

2.6.1 Motivations

Un intérêt de la propriété d'élimination des coupures est de s'assurer qu'une théorie ne contient pas \perp dans ces théorèmes. Or, notre théorie est une extension conservative de la présentation axiomatique de l'arithmétique de Heyting qu'on sait consistante. On sait déjà que notre théorie ne contient pas \perp .

Un autre intérêt de l'élimination des coupures, dans le cadre de théories intuitionistes, comme c'est le cas pour nous, est qu'elle donne la *propriété du témoin*. En effet, dans une preuve sans coupure d'une proposition de la forme $\exists x P(x)$, la dernière règle appliquée a nécessairement été l'introduction du \exists . On a le *témoin* de cette existence juste au-dessus dans l'arbre de preuve.

On peut citer encore, dans le domaine de la démonstration automatique, il est confortable de pouvoir se ramener à la recherche d'une preuve sans coupure.

2.6.2 Rappels

Définition 2.6.1 (Super-consistance)

Une théorie \mathcal{T}, \equiv en déduction modulo est super-consistante si, pour toute pseudo-algèbre de Heyting ordonnée et complète \mathcal{B} , on peut construire un \mathcal{B} -modèle de cette théorie.

Théorème 2.6.1 (Normalisation) Si une théorie \mathcal{T}, \equiv est super-consistante, alors toute preuve dans \mathcal{T}, \equiv est fortement normalisable.

Définition 2.6.2

Un modèle d'une théorie purement calculatoire dont les règles de réécriture sont

$$\begin{array}{c} R_1 \longrightarrow R'_1 \\ \vdots \\ R_n \longrightarrow R'_n \end{array}$$

est tel que pour toute interprétation Φ , $\llbracket R_i \rrbracket_\Phi = \llbracket R'_i \rrbracket_\Phi$ pour $i \in \{1, n\}$.

On sait que la normalisation forte d'un système implique l'élimination des coupures. On sait aussi que la super-consistance implique la normalisation forte.

Nous allons donner une preuve sémantique d'élimination des coupures qui utilise les pseudo algèbres de Heyting ordonnées et complètes en prouvant que HA \longrightarrow est super-consistante

2.6.3 HA \longrightarrow , une théorie super-consistance

Lemme 2.6.1

HA \longrightarrow est super-consistante.

Plan de la preuve :

A partir d'une pseudo-algèbre de Heyting ordonnée et complète \mathcal{B} , nous allons construire un \mathcal{B} -modèle \mathcal{M} de HA \longrightarrow .

On construira \mathcal{M} tel que pour toute interprétation Φ que si $A \longrightarrow A'$ est une règle qui définit la congruence de la théorie, alors $\llbracket A \rrbracket_\Phi = \llbracket A' \rrbracket_\Phi$. \mathcal{M} sera un modèle de HA \longrightarrow d'après la définition 2.6.2.

Soit $\mathcal{B} = \langle B, D, \leq, \tilde{\wedge}, \tilde{\vee}, \tilde{\perp}, \tilde{\top}, \tilde{\forall}, \tilde{\exists}, \tilde{\Rightarrow}, \sqsubseteq \rangle$

Soient M_ι et M_κ le domaine de \mathcal{M} .

Construction de \mathcal{M}

- Les domaines de \mathcal{M} sont $M_l = \mathbb{N}$ et $M_\kappa = B^{\mathbb{N}}$.
- L'interprétation du symbole de fonction 0 est le $0_{\mathbb{N}}$ des entiers.
- \perp et \top s'interprètent respectivement par $\tilde{\perp}$ et $\tilde{\top}$.
- Les symboles $+$ et \times s'interprètent respectivement comme l'addition et la multiplication dans les entiers qu'on notera $+_{\mathbb{N}}$ et $\times_{\mathbb{N}}$.
- Le symbole S s'interprète comme le successeur des entiers, soit la fonction qui à tout x associe $x +_{\mathbb{N}} 1_{\mathbb{N}}$
Exemples : $\llbracket S(0) + S(0) \rrbracket = (0_{\mathbb{N}} +_{\mathbb{N}} 1_{\mathbb{N}}) +_{\mathbb{N}} (0_{\mathbb{N}} +_{\mathbb{N}} 1_{\mathbb{N}}) = 2_{\mathbb{N}}$
- On interprète l'appartenance et les symboles de fonction κ comme dans la preuve de conservativité de HA_K :
La dénotation de \in est la fonction qui associe n et f à $f(n)$.
La dénotation d'un symbole de fonction de rang κ est une fonction qui reçoit une assignation pour les n variables libres de la proposition associée à f , et renvoie une fonction de \mathbb{N} dans B .
- L'interprétation de l'égalité, \cong , est définie par le tableau infini suivant

\cong	0	1	2	3	4	5	6	...
0	$\tilde{\top}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$...
1	$\tilde{\perp}$	$\tilde{\top}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$...
2	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\top}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$...
3	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\top}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$...
4	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\top}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$...
5	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\top}$	$\tilde{\perp}$	$\tilde{\perp}$...
6	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\perp}$	$\tilde{\top}$	$\tilde{\perp}$...
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\ddots

soit $\tilde{\top}$ sur la diagonale, et $\tilde{\perp}$ sur le reste du tableau.

- Interprétation du prédicat N C'est le symbole dont la construction est la plus technique. En effet, il apparait de façon récursive dans la règle de réécriture

$$N(n) \longrightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f)$$

Gardons donc à l'esprit qu'on cherche une interprétation de N telle que :

$$\llbracket (N(n)) \rrbracket = \llbracket \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket$$

On cherche une certaine fonction f_N de \mathbb{N} dans B telle que

$$\llbracket (N(n)) \rrbracket_{N:=f_N} = \llbracket \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket^{N:=f_N}$$

Soit un premier modèle \mathcal{M}_1 où tous les symboles sont interprétés comme expliqué ci-dessus et f_N est la fonction qui à x associe

$$\llbracket \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow x \in f) \rrbracket_{x:=n}^{N:=f_N}$$

Il faudrait pouvoir maintenant calculer le point fixe de f_N . Pour pouvoir calculer un point fixe, il faut vérifier que :

- Nous avons un ordre sur les fonctions :
oui, c'est l'ordre point à point : $f \sqsubseteq g$ si pour tout x , $f(x) \sqsubseteq g(x)$

– La fonction f_N est croissante :

oui, il suffit de constater que le prédicat N est en position positive (car \Rightarrow est croissante à droite et décroissante à gauche, et que N apparaît à gauche de la gauche d'une implication).

Nous pouvons donc appliquer le théorème de Knasser-Tarski sur les points fixes et déduire qu'il y a un point fixe F à cette fonction.

Nous interprétons le prédicat N par ce point fixe F , et on a, par construction,

$$(1) \quad \llbracket (N(n)) \rrbracket_{N:=F} = \llbracket \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f) \rrbracket_{N:=F}$$

Vérification

Vérifions que \mathcal{M} est un modèle de HA_{\rightarrow} , autrement dit, d'après la définition 2.6.2, pour toute règle de réécriture $A \rightarrow A'$ de HA_{\rightarrow} , on a $\llbracket A \rrbracket^{\mathcal{M}} = \llbracket A' \rrbracket^{\mathcal{M}}$

- $x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \rightarrow P\{z := x\}$
On a $\llbracket x \in f_{z, y_1, \dots, y_n, P}(y_1, \dots, y_n) \rrbracket = \llbracket P\{z := x\} \rrbracket$ par définition des interprétations des symboles de fonctions et de l'appartenance (cf preuve de conservativité de HA_K)
- $(N(n) \rightarrow \forall f (0 \in f \Rightarrow \forall y (N(y) \Rightarrow y \in f \Rightarrow S(y) \in f) \Rightarrow n \in f))$
Par construction de l'interprétation de N (1).
- $0 = 0 \rightarrow \top$
D'après le tableau, $\llbracket 0 = 0 \rrbracket = \tilde{\top}$ et par définition du modèle $\llbracket \top \rrbracket = \tilde{\top}$
On a bien $\llbracket 0 = 0 \rrbracket = \llbracket \top \rrbracket$
- $0 = S(x) \rightarrow \perp$
D'après le tableau, $\llbracket 0 = S(x) \rrbracket = \tilde{\perp}$ et par définition du modèle $\llbracket \perp \rrbracket = \tilde{\perp}$
On a bien $\llbracket 0 = S(x) \rrbracket = \llbracket \perp \rrbracket$
- $S(x) = 0 \rightarrow \perp$
D'après le tableau, $\llbracket S(x) = 0 \rrbracket = \tilde{\perp}$ et par définition du modèle $\llbracket \perp \rrbracket = \tilde{\perp}$
On a bien $\llbracket S(x) = 0 \rrbracket = \llbracket \perp \rrbracket$
- $S(x) = S(y) \rightarrow x = y$
Deux cas possible :
Soit la case de coordonnées $(S(x), S(y))$ dans le tableau de l'interprétation de l'égalité est située sur la diagonale, auquel cas la case (x, y) est aussi sur la diagonale et $\llbracket S(x) = S(y) \rrbracket = \llbracket x = y \rrbracket = \tilde{\top}$
Soit la case de coordonnées $(S(x), S(y))$ dans le tableau n'est pas sur la diagonale, auquel cas la case (x, y) n'est pas non plus sur la diagonale et $\llbracket S(x) = S(y) \rrbracket = \llbracket x = y \rrbracket = \tilde{\perp}$

Dans les deux cas $\llbracket S(x) = S(y) \rrbracket = \llbracket x = y \rrbracket$
- $0 + y \rightarrow y$
Par définition de l'interprétation du 0 et du + comme les symboles usuels de \mathbb{N} , on a $\llbracket 0 + y \rrbracket = \llbracket y \rrbracket$

- $S(x) + y \longrightarrow S(x + y)$

Par définition de l'interprétation du S et du $+$ comme le successeur et l'addition de \mathbb{N} , on a $\llbracket S(x) + y \rrbracket = \llbracket S(x + y) \rrbracket$

- $0 \times y \longrightarrow 0$

Par définition de l'interprétation du 0 et du \times comme les symboles usuels de \mathbb{N} , on a $\llbracket 0 \times y \rrbracket = \llbracket 0 \rrbracket$

- $S(x) \times y \longrightarrow x \times y + y$

Par définition de l'interprétation du S et du \times comme le successeur et la multiplication de \mathbb{N} , on a $\llbracket S(x) \times y \rrbracket = \llbracket x \times y + y \rrbracket$

A partir d'une algèbre d'une pseudo-algèbre de Heyting ordonnée et complète B , on a su construire un \mathcal{B} -modèle de HA_{\rightarrow} . D'après la définition 2.6.1, HA_{\rightarrow} est *super-consistante*. On conclut par le lemme 2.6.1 que HA_{\rightarrow} est une théorie où toute proposition qui a une preuve a une preuve sans coupure.

Chapitre 3

Réalisations en Coq

3.1 Introduction

Nous avons voulu nous donner un aperçu de traduction de nos définitions vers Coq. Les définitions suivantes sont un premier essai d'implémentation de HA_{\rightarrow} . On a redéfini les entiers pour être sûr de ne pas interférer avec la définition usuelle dans Coq nat.

A partir de ces définitions, on a implémenté en Coq tous les arbres de dérivations des sections 4.2,4.3 et 4.4, sans jamais utiliser l'égalité de Coq.

3.2 Nouvelle définition des entiers et fonctions sur les entiers

3.2.1 Les entiers

```
Inductive my_int :Set :=
| Z : my_int
| S : my_int -> my_int.
```

3.2.2 L'addition

```
Fixpoint my_plus (n m : my_int) struct n : my_int :=
match n with
| Z => m
| S t => S ( my_plus t m)
end.
```

3.2.3 La multiplication

```
Fixpoint my_mult (n m : my_int) struct n : my_int :=
match n with
| Z => Z
| S t => my_plus (my_mult t m) m
end.
```

3.2.4 L'égalité sur les entiers

```
Fixpoint my_eq (n m : my_int) struct n : Prop :=
match n with
| Z => match m with
| Z => True
| _ => False
end
| S t => match m with
```

```

|Z => False
| S u => my_eq t u
end
end.

```

3.3 Les propriétés prouvées

On a prouvé en Coq toutes les propriétés sur l'égalité, l'addition et la multiplication nécessaires à prouver qu'on peut simuler Leibniz dans HA_{\rightarrow} .

3.3.1 Les propriétés de l'égalité

```

Lemma refl :
forall (n : my_int) , my_eq n n.

```

```

Lemma sym :
forall (n m : my_int) , my_eq n m -> my_eq m n.

```

```

Lemma trans :
forall (a b c : my_int) , my_eq a b -> my_eq b c -> my_eq a c.

```

3.3.2 Les propriétés de l'addition

```

Lemma neutre_0 :
forall x : my_int, my_eq (my_plus x Z) x.

```

```

Lemma succ_right :
forall (x y :my_int), my_eq (my_plus x (S y)) (S (my_plus x y)).

```

```

Lemma comm :
forall (x y :my_int), my_eq (my_plus x y) (my_plus y x).

```

```

Lemma add_left :
forall (a b c :my_int), my_eq a b -> my_eq (my_plus a c) (my_plus b c).

```

```

Lemma add_right :
forall (a b c :my_int), my_eq a b -> my_eq (my_plus c a) (my_plus c b).

```

```

Lemma add_par :
forall (a b c d :my_int), my_eq a b -> my_eq c d -> my_eq (my_plus a c) (my_plus b d).

```

3.3.3 Les propriétés de la multiplication

```

Lemma abs_0 :
forall x : my_int, my_eq (my_mult x Z) Z.

```

```

Lemma mult_right :
forall (x y :my_int), my_eq (my_mult y (S x)) (my_plus (my_mult y x) y).

```

```

Lemma comm :
forall (x y :my_int), my_eq (my_mult x y) (my_mult y x).

```

```

Lemma mul_left :
forall (a b c :my_int), my_eq a b -> my_eq (my_mult a c) (my_mult b c).

```

```
Lemma mul_right :
forall (a b c :my_int), my_eq a b -> my_eq (my_mult c a) (my_mult c b).
```

```
Lemma mul_par :
forall (a b c d :my_int), my_eq a b -> my_eq c d -> my_eq (my_mult a c) (my_mult b d).
```

3.4 Comparaisons sur la taille des termes de preuves

La notion de *terme de preuve* est liée à la correspondance de Curry-Howard : soit un terme t du λ -calcul simplement typé, de type T . On peut envisager le type T comme une proposition, et le terme t comme la fonction qui prouve cette proposition. Cette notion peut être étendue à des systèmes de type plus raffinés, comme le Calcul des Constructions Inductives, sur lequel est fondé Coq. La commande `Print` permet d'afficher le terme de preuve associé à un lemme.

Nous donnons dans cette section deux termes de preuves de la proposition $\text{not}(5 = 9)$, 5 et 9 étant codés avec notre nouvelle définition des entiers.

Le lemme `lemme1` correspond à l'énoncé de cette proposition avec notre nouvelle égalité, `lemme2` est la même proposition énoncée avec l'égalité de Coq.

On prouve `lemme1` en *calculant*, avec la tactique `compute`. On prouve `lemme2` avec la tactique `discriminate`. Cette tactique est employée pour invalider les égalités fausses.

Nous donnons ici ces énoncés, leur preuve en Coq et leur terme de preuve.

3.4.1 Exemple 1 avec notre nouvelle égalité

Le lemme et sa preuve

```
Lemma lemme1 :
not (my_eq (S (S (S (S (S Z)))))) (S (S (S (S (S (S (S (S (S (S Z)))))) )))) ).
Proof.
compute.
intro.
assumption.
Qed.
```

Le terme de preuve

```
lemme1 =
fun H : False => H
  : my_eq (S (S (S (S (S Z)))))) (S (S (S (S (S (S (S (S (S (S Z)))))))))
```

3.4.2 Exemple 2 avec l'égalité usuelle dans Coq

Le lemme et sa preuve

```
Lemma lemme2 :
not ( S (S (S (S (S Z)))) = S (S (S (S (S (S (S (S (S (S Z))))))))) ).
Proof.
discriminate.
Qed.
```

Le terme de preuve

```
lemme2 =
fun H : S (S (S (S (S Z)))) = S (S (S (S (S (S (S (S (S (S Z))))))) =>
let H0 :=
```

```

eq_ind (S (S (S (S (S Z))))))
  (fun ee : my_int =>
    match ee with
    | Z => False
    | S m =>
      match m with
      | Z => False
      | S m0 =>
        match m0 with
        | Z => False
        | S m1 =>
          match m1 with
          | Z => False
          | S m2 =>
            match m2 with
            | Z => False
            | S m3 => match m3 with
            | Z => True
            | S _ => False
            end
          end
        end
      end
    end
  end) I (S (S (S (S (S (S (S (S (S Z)))))))))) H in
False_ind False H0
: S (S (S (S (S Z)))) <> S (S (S (S (S (S (S (S (S Z))))))))

```

3.5 Premières conclusions

Le terme preuve de `lemme1` est beaucoup plus simple et court que celui de `lemme2`. De plus, on voit comment vont croître ces termes de preuve avec des entiers plus grands : `lemme1` sera toujours le plus petit terme de preuve pour une inégalité, et le terme de preuve d'une inégalité avec `discriminate` augmentera beaucoup plus vite. Cette comparaison est donnée à titre indicatif, et n'a pas valeur formelle quant à la comparaison dans le cas général entre l'égalité usuelle de Coq et notre nouvelle définition.

Il semble logique que la tactique `discriminate` soit plus lourde, puisqu'elle permet de résoudre les inégalités pour *n'importe quel type inductif*. Cependant notre intuition dans ce travail est qu'on pourra améliorer l'efficacité de la tactique `discriminate` : on doit pouvoir générer automatiquement, sur le même modèle que pour l'arithmétique, une nouvelle égalité pour chaque type inductif et utiliser ces nouvelles égalités dans l'implémentation de la tactique `discriminate`.

Chapitre 4

Conclusion

Nous avons donné une présentation modulo de l'arithmétique de Heyting, plus simple et plus efficace que celle qui existait déjà, en nous concentrant sur une définition modulo de l'égalité. Nous avons prouvé que notre égalité simulait l'égalité de Leibniz, c'est à dire que le principe de substitutivité était valide avec notre nouvelle égalité, bien que n'étant pas énoncé sous forme axiomatique. La plupart des résultats sur cette nouvelle égalité ont été prouvés dans Coq .

Un premier travail qu'il serait intéressant d'effectuer, maintenant que nous disposons de tous ces lemmes en Coq, serait de coder le langage des propositions et les règles de déduction de la logique intuitioniste modulo en Coq pour pouvoir implémenter notre preuve de l'axiome de Leibniz.

A plus long terme, l'objectif sera de tenter de généraliser nos résultats sur l'égalité en cherchant comment définir de façon automatique une égalité modulo pour chaque type inductif.

Annexe A

Arbres de preuves et démonstrations

A.1 Lemme 2.3.3

On commence par prouver les deux lemmes intermédiaires suivants.

$$\mathbf{A.1.1} \quad \vdash_{HA_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x + y)$$

Remarque : on utilise dans les 2 preuves suivantes une version affaiblie de l'axiome de récurrence de HA_N .

Lemme A.1.1

$$\vdash_{HA_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x + y)$$

$$\frac{\frac{\frac{N(y) \vdash_{HA_N} N(y)}{N(y) \vdash_{HA_N} N(0 + y)} \text{Ax} \quad 5 \quad \frac{\frac{N(y) \vdash_{HA_N} \forall x (N(x) \Rightarrow N(S(x)))}{N(y) \vdash_{HA_N} (N(w + y) \Rightarrow N(S(w + y)))} \forall e \quad 6 \quad \frac{N(y) \vdash_{HA_N} (N(w + y) \Rightarrow N(S(w + y)))}{N(y) \vdash_{HA_N} \forall w (N(w + y) \Rightarrow N(S(w + y)))} \forall i}{N(y) \vdash_{HA_N} \forall w (N(w + y) \Rightarrow N(S(w + y)))} \wedge i \quad \frac{N(y) \vdash_{HA_N} \forall w (N(w + y) \Rightarrow N(S(w + y)))}{\vdash_{HA_N} (N(0 + y) \wedge \forall w (N(w + y) \Rightarrow N(S(w + y))) \Rightarrow \forall w N(w) \Rightarrow N(w + y))} \text{Ax} \Rightarrow e}{\frac{\frac{N(y) \vdash_{HA_N} \forall w N(w) \Rightarrow N(w + y)}{N(y) \vdash_{HA_N} N(x) \Rightarrow N(x + y)} \forall e \quad \frac{N(y) \vdash_{HA_N} N(x) \Rightarrow N(x + y)}{\vdash_{HA_N} N(y) \Rightarrow N(x) \Rightarrow N(x + y)} \Rightarrow i}{\vdash_{HA_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x + y)} \forall i \quad x2}$$

$$\mathbf{A.1.2} \quad \vdash_{HA_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x \times y)$$

Lemme A.1.2

$$\vdash_{HA_N} \forall x \forall y N(x) \Rightarrow N(y) \Rightarrow N(x \times y)$$

$$\frac{\frac{\frac{N(y) \vdash_{HA_N} N(y)}{N(y) \vdash_{HA_N} N(0 \times y)} \text{Ax} \quad 7 \quad \frac{\frac{\text{Lemme A.1.1}}{\vdash_{HA_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x + y)} \text{Ax} \quad 8 \quad \frac{N(y) \vdash_{HA_N} (N(w \times y) \Rightarrow N(w \times y + y))}{N(y) \vdash_{HA_N} (N(w \times y) \Rightarrow N(S(w) \times y))} \forall e \quad x2 \quad \frac{N(y) \vdash_{HA_N} (N(w \times y) \Rightarrow N(S(w) \times y))}{N(y) \vdash_{HA_N} \forall w (N(w \times y) \Rightarrow N(S(w) \times y))} \forall i}{N(y) \vdash_{2.3.3 HA_N} N(0 \times y) \wedge \forall w (N(w \times y) \Rightarrow N(S(w) \times y))} \wedge i \quad \frac{N(y) \vdash_{2.3.3 HA_N} N(0 \times y) \wedge \forall w (N(w \times y) \Rightarrow N(S(w) \times y))}{\vdash_{HA_N} N(0 \times y) \wedge \forall w (N(w \times y) \Rightarrow N(S(w) \times y)) \Rightarrow \forall w N(w) \Rightarrow N(w \times y)} \text{Ax} \Rightarrow e}{\frac{\frac{N(y) \vdash_{HA_N} \forall w N(w) \Rightarrow N(w \times y)}{N(y) \vdash_{HA_N} N(x) \Rightarrow N(x \times y)} \forall e \quad \frac{N(y) \vdash_{HA_N} N(x) \Rightarrow N(x \times y)}{\vdash_{HA_N} N(y) \Rightarrow N(x) \Rightarrow N(x \times y)} \Rightarrow i}{\vdash_{HA_N} \forall x \forall y N(y) \Rightarrow N(x) \Rightarrow N(x \times y)} \forall i \quad x2}$$

A.1.3 Preuve du lemme 2.3.3

Lemme 2.3.3

$N(\vec{z}) \vdash_{HAN} N(t)$ pour tout t avec $FV(t) = \vec{z}$

Preuve par récurrence sur la forme de t :

* Cas de base

t est une variable x :

$FV(t) = x$ et par hypothèse du lemme $N(x)$ appartient au contexte

t est la constante 0 :

$N(0)$ est un axiome de la théorie

* Cas récursifs

t est de la forme $S(t')$

Par hypothèse de récurrence on sait prouver $N(t')$ - on utilise l'axiome $\forall x N(x) \Rightarrow N(S(x))$

t est de la forme $t1 + t2$

Par hypothèse de récurrence on sait prouver $N(t1)$ et $N(t2)$ - on utilise le lemme A.1.1

t est de la forme $t1 \times t2$

Par hypothèse de récurrence on sait prouver $N(t1)$ et $N(t2)$ - on utilise le lemme A.1.2

A.2.3 Transitivité : $\forall x \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z$

Cas de base : $\Pi_0 \forall y \forall z 0 = y \Rightarrow y = z \Rightarrow 0 = z$

$$\frac{\frac{\frac{\frac{}{0 = 0, 0 = z \vdash 0 = z} \text{Ax}}{0 = 0 \vdash 0 = z \Rightarrow 0 = z} \Rightarrow i}{0 = 0 \vdash \forall z 0 = z \Rightarrow 0 = z} \forall i}{\vdash \forall z 0 = 0 \Rightarrow 0 = z \Rightarrow 0 = z} \Rightarrow i}{\frac{\frac{\frac{\frac{}{\vdash \perp \Rightarrow (S(y) = z \Rightarrow 0 = z)} \text{Ax}}{\vdash 0 = S(y) \Rightarrow (S(y) = z \Rightarrow 0 = z)} 2}{\vdash \forall z 0 = S(y) \Rightarrow (S(y) = z \Rightarrow 0 = z)} \forall i}{\vdash \forall y \forall z 0 = S(y) \Rightarrow (S(y) = z \Rightarrow 0 = z)} \forall i} \wedge i}{\frac{\frac{\frac{}{\vdash (\forall z 0 = 0 \Rightarrow 0 = z \Rightarrow 0 = z)} \wedge \forall y (\forall z 0 = S(y) \Rightarrow (S(y) = z \Rightarrow 0 = z))} \wedge i}{\vdash \forall y \forall z 0 = y \Rightarrow y = z \Rightarrow 0 = z} \Rightarrow e} \text{Ax - Rec affaibli}}{\frac{}{\vdash (P(0) \wedge \forall y (P(S(y)))) \Rightarrow \forall y P(y)} \Rightarrow e}$$

Cas rec : $\Pi_S : \forall x (\forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z) \Rightarrow (\forall y \forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z)$

rec : $(P(0) \wedge (\forall z P(z) \Rightarrow P(S(z)))) \Rightarrow \forall x P(x)$

Rec sur z $\Pi_z : \forall z S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z$

$$\frac{\frac{\frac{\frac{}{\Gamma, S(x) = S(y) \vdash \perp} \text{Ax}}{\Gamma, S(x) = S(y) \vdash \perp \Rightarrow \perp} \Rightarrow i}{\frac{\frac{}{\Gamma, S(x) = S(y) \vdash S(y) = 0 \Rightarrow S(x) = 0} 3}{\Gamma \vdash S(x) = S(y) \Rightarrow S(y) = 0 \Rightarrow S(x) = 0} \Rightarrow i} \wedge i}{\frac{\frac{\frac{}{\Gamma \vdash S(x) = S(y) \Rightarrow S(y) = 0 \Rightarrow S(x) = 0} \wedge \forall z (S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z) \Rightarrow (S(x) = S(y) \Rightarrow S(y) = S(z) \Rightarrow S(x) = S(z))} \wedge i}{\Gamma \vdash \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z, \forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z \vdash \forall z S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z} \Rightarrow e} \text{Ax - dans } \Gamma}{\frac{\frac{\frac{\frac{}{\Gamma, (S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z) \vdash x = y \Rightarrow y = z \Rightarrow x = z} \text{Ax - dans } \Gamma}}{\Gamma, (S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z) \vdash (S(x) = S(y) \Rightarrow S(y) = S(z) \Rightarrow S(x) = S(z))} 4}{\Gamma \vdash (S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z) \Rightarrow (S(x) = S(y) \Rightarrow S(y) = S(z) \Rightarrow S(x) = S(z))} \Rightarrow i}{\Gamma \vdash \forall z (S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z) \Rightarrow (S(x) = S(y) \Rightarrow S(y) = S(z) \Rightarrow S(x) = S(z))} \forall i} \wedge i}{\frac{}{\Gamma \vdash \text{rec}} \text{Ax}} \Rightarrow e$$

Cas recursif de x : $\Pi_S : \forall x (\forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z) \Rightarrow (\forall y \forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z)$

$$\frac{\frac{\frac{\frac{}{\Gamma, \perp \vdash \perp} \text{Ax}}{\Gamma, \perp \vdash 0 = z \Rightarrow S(x) = z} \perp e}{\Gamma \vdash \perp \Rightarrow 0 = z \Rightarrow S(x) = z} \Rightarrow i}{\frac{\frac{}{\Gamma \vdash S(x) = 0 \Rightarrow 0 = z \Rightarrow S(x) = z} 3}{\Gamma \vdash \forall z S(x) = 0 \Rightarrow 0 = z \Rightarrow S(x) = z} \forall i} \wedge i}{\frac{\frac{\frac{}{\Gamma \vdash \forall z S(x) = 0 \Rightarrow 0 = z \Rightarrow S(x) = z} \wedge \forall y (\forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z) \Rightarrow \forall z S(x) = S(y) \Rightarrow S(y) = z \Rightarrow S(x) = z} \wedge i}{\Gamma \vdash \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z \vdash \forall y \forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z} \Rightarrow i}{\frac{\frac{}{\Gamma \vdash \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z} \Rightarrow i}{\Gamma \vdash \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z} \Rightarrow i} \wedge i}{\frac{}{\Gamma \vdash \forall x (\forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z) \Rightarrow (\forall y \forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z)} \wedge i} \text{Ax}} \Rightarrow e$$

Fin de la démonstration

$$\frac{\frac{\frac{}{\vdash \forall y \forall z 0 = y \Rightarrow y = z \Rightarrow 0 = z} \Pi_0}{\vdash \forall y \forall z 0 = y \Rightarrow y = z \Rightarrow 0 = z \wedge (\forall x (\forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z) \Rightarrow (\forall y \forall z S(x) = y \Rightarrow y = z \Rightarrow S(x) = z))} \wedge i}{\frac{}{\vdash \forall x \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z} \text{Ax}} \wedge i}{\frac{}{\vdash \forall x \forall y \forall z x = y \Rightarrow y = z \Rightarrow x = z} \text{Ax}} \Rightarrow e$$

A.3.6 Congruence à droite pour l'addition : $add_right : \forall x \forall y \forall z x = y \Rightarrow z+x = z+y$

On utilise la congruence à gauche et on la renverse grâce à la commutativité de l'addition.

A.3.7 Congruence parallèle de l'addition : $add_par : \forall x \forall y \forall u \forall t x = y \Rightarrow u = t \Rightarrow x + u = y + t$

$$\frac{\frac{\frac{}{x = y, u = t \vdash x = y} \text{Ax}}{x = y, u = t \vdash x + u = y + u} \text{Add_Left} \quad \frac{\frac{\frac{}{\vdash \forall x \forall y \forall z x = y \Rightarrow x + z = y + z} \forall e \text{ x3}}{\vdash x = y \Rightarrow x + u = y + u} \Rightarrow e}{x = y, u = t \vdash x + u = y + u} \text{Ax} \quad \frac{\frac{\frac{}{\vdash \forall x \forall y \forall z x = y \Rightarrow z + x = z + y} \forall e \text{ x3}}{\vdash u = t \Rightarrow y + u = y + t} \Rightarrow e}{x = y, u = t \vdash y + u = y + t} \text{Add_Right}}{x = y, u = t \vdash x + u = y + t} \text{Trans}}{\frac{\frac{\frac{}{x = y, u = t \vdash x + u = y + t} \Rightarrow i \text{ x2}}{\vdash x = y \Rightarrow u = t \Rightarrow x + u = y + t} \Rightarrow e}{\vdash \forall x \forall y \forall u \forall t x = y \Rightarrow u = t \Rightarrow x + u = y + t} \forall i \text{ x4}}{x = y, u = t \vdash x + u = y + u} \text{Trans}}{\vdash \forall x \forall y \forall u \forall t x = y \Rightarrow u = t \Rightarrow x + u = y + t} \forall i \text{ x4}}$$

A.4 Propriétés de la multiplication dans HA_R **A.4.1 0 absorbant pour la multiplication à droite : $0_abs_r : \forall x x \times 0 = 0$**

$$\frac{\frac{\frac{\frac{}{\vdash \forall x \forall y \forall z x = y \Rightarrow x + z = y + z} \forall e \text{ x3}}{\vdash x \times 0 = 0 \Rightarrow x \times 0 + 0 = 0 + 0} \Rightarrow e \text{ x3}}{x \times 0 = 0 \vdash x \times 0 + 0 = 0 + 0} \Rightarrow e \text{ x3} \quad \frac{\frac{\frac{}{\vdash \top} \text{Ax}}{x \times 0 = 0 \vdash 0 = 0} 1}{x \times 0 = 0 \vdash 0 + 0 = 0} 5}{x \times 0 = 0 \vdash x \times 0 + 0 = 0 + 0} \text{Trans}}{\frac{\frac{\frac{}{\vdash \top} \text{Ax}}{\vdash 0 = 0} 1}{\vdash 0 \times 0 = 0} 7 \quad \frac{\frac{\frac{}{x \times 0 = 0 \vdash x \times 0 + 0 = 0} 8}{x \times 0 = 0 \vdash S(x) \times 0 = 0} \Rightarrow}{\vdash x \times 0 = 0 \Rightarrow S(x) \times 0 = 0} \forall i}{\vdash \forall x (x \times 0 = 0 \Rightarrow S(x) \times 0 = 0)} \forall i}{\vdash 0 \times 0 = 0 \wedge \forall x (x \times 0 = 0 \Rightarrow S(x) \times 0 = 0)} \wedge i}{\vdash (P(0) \wedge \forall x (P(x) \Rightarrow P(S(x)))) \Rightarrow \forall x P(x)} \text{Ax}}{\forall x x \times 0 = 0} \Rightarrow e$$

Bibliographie

- [1] G. Dowek, T. Hardin, and C. Kirchner. Theorem proving modulo. *Journal of Automated Reasoning*, 31 :32–72, 2003.
- [2] G. Dowek and B. Werner. Arithmetic as a theory modulo. J. Giesel (Ed.), *Term rewriting and applications*, Lecture Notes in Computer Science 3467, Springer-Verlag, 2005, pp. 423-437.
- [3] G. Dowek. Truth values algebras and normalization . <http://www.lix.polytechnique.fr/~dowek/Publi/truthvalues.pdf>.
- [4] G. Dowek. La part du calcul. *Mémoire d'Habilitation à Diriger des Recherches*, Université Paris 7, 1999.
- [5] The Coq Development Team . Manuel de Référence de Coq V8.0. <http://coq.inria.fr/doc/main.html>, LogiCal Project, 2004-2006