Logique du Premier Ordre

Notes de Cours

DESS Developpement de logiciels sûrs

Valeur C Construction Rigoureuse de Logiciel

Marianne Simonot
CNAM
simonot@cnam.fr

October 7, 2004

Contents

Chapter 1

Le Calcul des Propositions

1.1 Introduction

Pour exprimer nos idées, nous énoncons des jugements, par exemple : "une carré est constitué de quatre cotés égaux". La logique vise à formaliser le concept de raisonnement (mathématique) et par là même celui d'énoncés, de jugements. Considérons les deux exemples suivants :

1 Si Pierre a une cicatrice, c'est qu'il a été blessé.

2 Si Pierre a une cicatrice, c'est qu'il a été blessé. Or il a une cicatrice, donc il a été blessé.

Au premier abord, ces deux énoncés semblent vrais. Substituons maintenant "Pierre a les cheveux blonds" à Pierre a une cicatrice, dans ces deux énoncés.

3 Si Pierre a les cheveux blonds, c'est qu'il a été blessé.

4 Si Pierre a les cheveux blonds, c'est qu'il a été blessé. Or il a les cheveux blonds, donc il a été blessé.

Le premier nous parait maintenant faux, le second reste vrai. Ceci indique une différence de statut entre les deux enoncés. Le premier est une vérité du monde dans lequel nous vivons, le second exprime une vérité dans tous les mondes possibles.

La logique a pour but de donner un statut clair à ces notions de vérité, de correction des raisonnements Elle formalise la notion de substitution que nous venons d'utiliser intuitivement pour différencier les énoncés en désignant :

1. des places où la substitution est possible (variables)
2. des constantes représentant les "outils" de base nécessaires à la construction des énoncés.

Le système logique particulier que nous allons étudier maintenant -le Calcul des Propositions- est une formalisation rudimentaire du raisonnement : les constantes seront les liens "et" ($\wedge$), "ou" ($\vee$) , "non" ($\neg$), "si ... alors" ($\rightarrow$), "si et seulement si" ($\leftrightarrow$). Tout énoncé non divisible au moyen de ces liens sera considéré comme une entité indivisible -une *place*- que nous apellerons une *proposition* . Les énoncés 1 et 3 de notre exemple seront traduits en Calcul des Propositions par $A \rightarrow B$ qui n'est ni vraie ni fausse, seulement *satisfaisable*. Les énoncés 2 et 4 seront traduits par $((A \rightarrow B) \wedge A) \rightarrow B$ qui est une formule *vraie* ou *valide*.

1.2 Définir le langage

Un langage propositionnel est la donnée de :

1. un alphabet :
   (a) un ensemble de *connecteurs* : $\{\wedge, \vee, \neg, \rightarrow, \}$
   (b) un ensemble de symbole de ponctuation : $\{(,)\}$
   (c) un ensemble de *variables propositionnelles* : tous les identificateurs ne contenant ni connecteurs, ni parenthèses.
2. une grammaire définie par les deux règles suivantes :
   (a) Les variables propositionnelles sont des formules.
   (b) Si A et B sont des formules alors $(A \vee B)$, $(A \wedge B)$, $(A \rightarrow B)$, $\neg A$ et (A) sont des formules.

Remarque 1.2.1 *L'appartenance à l'ensemble des formules est décidable (ie il existe un algorithme permettant de décider si un mot quelconque est une formule)*

1.3 Sémantique: validité, satisfaisabilité

1.3.1 Interprétation

Il s'agit d'*interpréter* les variables propositionnelles d'une formule dans l'ensemble des booléens $\{vrai, faux\}$. Grace à cela on pourra calculer la valeur de vérité de la formule par rapport à cette interprétation.
exemple :

$$\begin{array}{cccc} & vrai & & vrai \\ I_1 & \uparrow & & \uparrow \\ & A & \wedge & B \\ I_2 & \downarrow & & \downarrow \\ & vrai & & faux \end{array}$$

L'interprétation I_1 satisfait la formule $A \wedge B$, car pour qu'une formule dont le connecteur principal est $\wedge$ soit vraie, il faut que les deux sous formules soient vraies. En revanche, l'interprétation I_2 ne satisfait pas la formule $A \wedge B$.
$A \wedge B$ est donc satisfaisable mais pas valide.

Définition 1.3.1 *Une* Interprétation *d'une formule F est une fonction de l'ensemble des variables propositionnelles de F dans l'ensemble $\{vrai, faux\}$.*

Remarque 1.3.1 *si F contient n variables propositionnelles alors il existe exactement 2^n interprétations de F.*

1.3.2 tables de vérité des connecteurs

Dans notre exemple, nous avons pu conclure sur la valeur de vérité de $A \wedge B$ pour I_1 et I_2 parceque nous connaissons intuitivement le comportement du connecteur $\wedge$ qui correspond à l'usage du mot "et". Le rôle des *tables de verité* des connecteurs est de fixer formellement le comportement de chacun des connecteurs.

s	$(\neg$ s$)$
v	f
f	v

s	t	(s $\wedge$ t)
v	v	v
v	f	f
f	v	f
f	f	f

s	t	(s $\vee$ t)
v	v	v
v	f	v
f	v	v
f	f	f

s	t	(s $\rightarrow$ t)
v	v	v
v	f	f
f	v	v
f	f	v

$\wedge$, $\vee$ et $\rightarrow$ correspondent respectivement aux expressions "et", "ou" et "si ... alors" de la langue naturelle. Leur table de vérité correspond à leur usage courant aux deux remarques suivantes près :

- La table de vérité du $\vee$ correspond a l'usage inclusif du mot "ou".
- La valeur de verité *vrai* attribuée au cas où la condition de l'implication est fausse est peu naturelle. Dans le langage courant,on ne s'intéresse à la vérité d'un tel énoncé que lorsque la condition est vraie : "s'il fait beau je vais à la pêche " n'a d'interêt pratique que s'il fait beau ... Attribuer la valeur *vrai* dans le cas ou la prémisse est fausse correspond a peu près à l'usage du si .. alors dans la phrase suivante : "S'il y a des porte-manteaux dans cette salle, je veux bien etre pendu".

Remarque 1.3.2 *Donner les tables de vérité des connecteurs sous forme de tableaux revient à définir pour chacun des connecteurs* $\wedge, \vee, \neg, \rightarrow$ *une fonction des booléens vers les booléens, que nous nommerons respectivement* $T_\wedge, T_\vee, T_\neg, T_\rightarrow$.
Ces fonctions sont définies de la façon suivante :
$T_\wedge(x,y) = vrai$ *ssi* $x = vrai$ *et* $y = vrai$
$T_\vee(x,y) = vrai$ *ssi* $x = vrai$ *ou* $y = vrai$
$T_\rightarrow(x,y) = faux$ *ssi* $x = vrai$ *et* $y = faux$
$T_\neg(x) = vrai$ *ssi* $x = faux$

1.3.3 Valeur de vérité d'une formule

Les notions d'interprétation et de table de vérité des connecteurs permettent de calculer la valeur de verité pour une interprétation donnée de la formule toute entière.
exemple :

i_1	s	t	(s $\vee$ t)	(t $\wedge$ s)	(s $\vee$ t) $\rightarrow$ (t $\wedge$ s)
i_1	v	v	v	v	v
i_2	v	f	v	f	f
i_3	f	v	v	f	f
i_4	f	f	f	f	v

Chaque ligne correspond à une interprétation.

Autrement dit :

Définition 1.3.2 *étant données une formule* F *et une interprétation* I *de cette formule, On définit la* valeur de vérité de F *relativement à* I *comme une fonction des formules vers les booléens* $V_I(F)$ *définie par :*

1. *Si* F *est une variable propositionnelle alors* $V_I(F) = I(F)$
2. *Si* $F = \neg G$ *alors* $V_I(F) = T_\neg(V_I(G))$
3. *Si* $F = (G \vee H)$ *alors* $V_I(F) = T_\vee(V_I(G), V_I(H))$
4. *Si* $F = (G \wedge H)$ *alors* $V_I(F) = T_\wedge(V_I(G), V_I(H))$
5. *Si* $F = (G \rightarrow H)$ *alors* $V_I(F) = T_\rightarrow(V_I(G), V_I(H))$

exemple : Pour l'exemple précédent, la valeur de vérité relativement à i_2 est la suivante :
$V_{i_2}((s \vee t) \rightarrow (t \wedge s)) =$
$T_\rightarrow(V_{i_2}(s \vee t), V_{i_2}(t \wedge s)) =$
$T_\rightarrow(T_\vee(V_{i_2}(s), V_{i_2}(t)), T_\wedge(V_{i_2}(t), V_{i_2}(s))) =$
$T_\rightarrow(T_\vee(v,f), T_\wedge(f,v)) =$
$T_\rightarrow(v,f) =$
f

1.3.4 Validité, satisfaisabilité, modèle

Définition 1.3.3 *Une interprétation I satisfait une formule F ssi $V_I(F) = vrai$*

Définition 1.3.4 *Une formule F est* satisfaisable *ssi il existe une interprétation I telle que $V_I(F) = vrai$ (noté $I \models F$)*

Définition 1.3.5 *Une formule F est* valide *(ou est une* tautologie*) ssi pour toute interprétation I on a : $V_I(F) = vrai$ (noté $\models F$)*

Remarque 1.3.3 *Les notions de validité et de satisfaisabilité en calcul des propositions est décidable. algorithme : etant donné une formule F ayant n variables propositionnelles Calculer les 2^n interprétations possibles. Pour chacune d'entre elles calculer la valeur de vérité de F. Si toutes les valeurs de vérité de F obtenues sont égales à vrai alors F est valide. Si au moins une est vraie F est réalisable.*

1.4 La notion de Déduction

Quand dirons nous qu'une proposition P se *déduit* des propositions $P_1,\ldots P_n$? On pourrait adopter la position suivante : supposant $P_1, \ldots P_n$ formalisées dans le calcul des propositions, P se déduit de $P_1,\ldots,\ P_n$ si la proposition $P_1,\ldots,$ $P_n \rightarrow P$ est une tautologie. Prenons l'exemple suivant de déduction :
Si l'accusé est coupable, il était à Paris au moment du crime. Or il n'était pas à Paris, donc il n'est pas coupable.
La première approche proposée est de montrer la correction de la formule :
$((C \rightarrow P) \wedge \neg P) \rightarrow \neg C$

Une autre approche est possible : pour justifier une déduction, on en montre une *Preuve*, qui conduit des hypothèses à la conclusion par une suite d'étapes évidentes. Au lieu de s'intéresser à ce qui est vrai, on s'intéresse à ce qui est démontrable. Ce que l'on va chercher à formaliser sera cette fois la notion de *preuve*.
Pour notre exemple, on cherchera à prouver que l'on peut déduire $\neg C$ à partir des hypothèses $C \rightarrow P$ et $\neg P$.

Les deux formalisations de la notion de preuve les plus connues et les plus utilisées dans les systèmes de preuves sont *la déduction naturelle* et *le calcul des séquents*. Nous allons étudier la seconde.

1.4.1 Séquent, règle d'inférence et preuve formelle

Une preuve aura la forme d'un arbre :

$$\dfrac{\dfrac{\begin{array}{ccc} d_1 & & d_2 \\ \vdots & & \vdots \\ \text{S1} & \ldots & \text{S2} \end{array}}{S3}\, r1}{S4}\, r2$$

et s'interprétera comme suit : pour prouver S4, il suffit de prouver S3. Pour prouver S3 il suffit de prouver S1 et S2 ...

séquent

Chaque noeud de l'arbre est appelé un *sequent* et est constitué d'un ensemble d'*hypothèses* : $A_1, \ldots, A_n$ et d'un ensemble de but : $B_1, \ldots B_m$. Un séquent sera noté : $A_1, \ldots, A_n \vdash B_1, \ldots B_m$.
Une preuve de notre exemple sera donc un arbre dont la racine est le séquent :
$C \rightarrow P, \neg P \vdash \neg C$

Obtenir une preuve de $A_1, \ldots, A_n \vdash B_1, \ldots B_m$ correspond intuitivement à s'être assuré de la vérité de *l'un des* B_i en ayant supposé que *tous les* A_i sont vrais. En d'autres termes (Nous démontrerons ce résultat page 17 :
$A_1, \ldots, A_n \vdash B_1, \ldots B_m$ si et seulement si $\vdash (A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee, \ldots, \vee B_m)$

règles d'inférence

Le passage d'un niveau à un autre de l'arbre est codifié. Il se fait au moyen de règles de preuves (les *règles d'inférence*).

$$\frac{S_1 \ldots S_n}{S_m} R$$

S_m est la *conclusion* de la règle, et $S_1 \ldots S_n$ sont les *prémisses* de la règle.

$S_1 \ldots S_n, S_m$ sont des *patrons* de séquents : ils contiennent des "jokers" de deux sortes :

- Les lettres majuscules grecques : Γ , Δ ... qui désignent des ensembles de formules
- Les autres lettres majuscules : A, B ... qui désignent une formule

Une règle est *applicable* sur un séquent concret si en donnant des valeurs aux jokers de la conclusion de la règle, cette conclusion devient identique au séquent sur lequel on veut l'appliquer.

Voici une règle d'inférence :

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta} \wedge_d$$

Cette règle est applicable au séquent S :$\neg A, B \wedge C \vdash (E \rightarrow F) \wedge F$
car en donnant aux jokers les valeurs :
$\Gamma = \{\neg A, B \wedge C\}$
$\Delta = \emptyset$
$A = (E \rightarrow F)$
$B = F$
$\Gamma \vdash A \wedge B, \Delta$ et S deviennent identiques.

Cette règle n'est en revanche pas applicable au séquent T :
$\neg A, B \wedge C \vdash (E \rightarrow F) \vee G, \neg H$.

Cette règle est applicable sur tout séquent contenant une formule à droite dont le connecteur principal est un $\wedge$.

l'application d'une règle sur un séquent concret (sur lequel elle est applicable) consiste à créer un niveau de l'arbre de preuve de ce séquent en appliquant les valeurs trouvées pour les jokers sur les prémisses de la règle.

Pour notre exemple, appliquer la règle $\wedge_d$ sur $\neg A, B \wedge C \vdash (E \rightarrow F) \wedge F$ produit :

$$\frac{\overbrace{\neg A, B \wedge C}^{\Gamma} \vdash \overbrace{E \rightarrow F}^{A} \qquad \overbrace{\neg A, B \wedge C}^{\Gamma} \vdash \overbrace{F}^{B}}{\underbrace{\neg A, B \wedge C}_{\Gamma} \vdash \underbrace{E \rightarrow F}_{A} \wedge \underbrace{F}_{B}} \wedge_d$$

Une règle d'inférence

$$\frac{\Gamma_1 \vdash \Delta_1 \ldots \Gamma_n \vdash \Delta_n}{\Gamma_m \vdash \Delta} R$$

peut se lire du haut vers le bas ou du bas vers le haut.

- haut $\rightarrow$ bas :
  "Si Δ_1 est démontrable sous les hypothèses de $\Gamma_1 \ldots$ et si Δ_n est démontrable sous les hypothèses de Γ_n alors Δ est démontrable sous les hypothèses de Γ_m".
- bas $\rightarrow$ haut :
  "Pour démontrer Δ sous les hypothèses de Γ_m, il suffit de démontrer Δ_1 sous les hypothèses de $\Gamma_1 \ldots$ et Δ_n sous les hypothèses de Γ_n".

Les règles d'inférence constituent les étapes élémentaires du raisonnement et correspondent au maniement des connecteurs. Nous aurons pour chaque connecteur 2 règles : une (la règle gauche) qui permet de *déduire* de nouvelles hypothèses à partir de celles que l'on a. Par exemple "si $A \wedge B$ est en hypothèse, alors je peux ajouter à mon contexte d'hypothèses A et B" . L'autre règle (la règle droite) disent comment manipuler les buts. Par exemple "pour démontrer $A \wedge B$ dans un certain contexte d'hypothèses, il suffit de démontrer A dans ce contexte puis de démontrer B dans ce même contexte."

Par exemple, les règles pour le connecteur $\wedge$ sont les suivantes :

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma \vdash B, \Delta}{\Gamma \vdash \boxed{A \wedge B}} \wedge_d \qquad \frac{\Gamma, A \wedge B, A, B \vdash \Delta}{\Gamma, \boxed{A \wedge B} \vdash \Delta} \wedge_g$$

Par convention, Γ et Δ représentent des ensembles de formules.
Certaines règles sont terminales : elles ne produisent pas de fils. Elles expriment un fait toujours vrai.

preuve

Une preuve de $\Gamma \vdash F$ est est une suite d'applications de règles d'inférence commençant par $\Gamma \vdash F$ et se terminant par des axiomes.

théorème

Un *théorème* est une formule prouvable

1.4.2 Les règles du Calcul des Séquents

2 types de règles sur chaque connecteurs :

- des règles droites permettant de raisonner sur la(es) formule(s) à prouver.
- Des règles gauches permettant de raisonner sur les hypothèses.

La présentation que nous faisons ici est identique à celle effectuée dans [3]. C'est une présentation ascendante du système adaptée à la recherche de preuve. Vous trouverez par exemple dans [**?**] une présentation descendante du même système.

$$\frac{}{\Gamma, \boxed{A} \vdash \boxed{A}, \Delta}\ axiome$$

$$\frac{\Gamma \vdash \boxed{A}, \Delta \qquad \Gamma \vdash \boxed{B}, \Delta}{\Gamma \vdash \boxed{A \wedge B}, \Delta}\ \wedge_d \qquad \frac{\Gamma, \boxed{A}, \boxed{B} \vdash \Delta}{\Gamma, \boxed{A \wedge B} \vdash \Delta}\ \wedge_g$$

$$\frac{\Gamma \vdash \boxed{A}, \boxed{B}, \Delta}{\Gamma \vdash \boxed{A \vee B}, \Delta}\ \vee_d \qquad \frac{\Gamma, \boxed{A} \vdash \Delta \qquad \Gamma, \boxed{B} \vdash \Delta}{\Gamma, \boxed{A \vee B} \vdash \Delta}\ \vee_g$$

$$\frac{\Gamma, \boxed{A} \vdash \Delta}{\Gamma \vdash \boxed{\neg A}, \Delta}\ \neg_d \qquad \frac{\Gamma \vdash \boxed{A}, \Delta}{\Gamma, \boxed{\neg A} \vdash \Delta}\ \neg_g$$

$$\frac{\Gamma, \boxed{A} \vdash \boxed{B}, \Delta}{\Gamma \vdash \boxed{A \rightarrow B}, \Delta}\ \rightarrow_d \qquad \frac{\Gamma \vdash \boxed{A}, \Delta \qquad \Gamma, \boxed{B} \vdash \Delta}{\Gamma, \boxed{A \rightarrow B} \vdash \Delta}\ \rightarrow_g$$

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \boxed{A}, \Delta}\ aff_d \qquad \frac{\Gamma \vdash \Delta}{\Gamma \boxed{A}, \vdash \Delta}\ aff_g$$

$$\frac{\Gamma \vdash \boxed{A}, \boxed{A}, \Delta}{\Gamma \vdash \boxed{A}, \Delta}\, contraction_d \qquad \frac{\Gamma, \boxed{A}, \boxed{A} \vdash \Delta}{\Gamma, \boxed{A}, \vdash \Delta}\, contraction_g$$

$$\frac{\Gamma \vdash \boxed{A}, \Delta \qquad \Gamma, \boxed{A} \vdash \Delta}{\Gamma \vdash \Delta}\, cut$$

Le fragment $\wedge$, $\vee$

Les règles pour ces connecteurs s'interprètent comme suit :

$\wedge_d$ pour démontrer $A \wedge B$ dans un certain contexte d'hypothèses, il suffit de démontrer A dans ce contexte puis de démontrer B dans ce même contexte.

$\wedge_g$ Si $A \wedge B$ est supposé vrai (ie est dans les hypothèses) alors A et B sont également vrais (ie on peut les ajouter aux hypothèses).

$\vee_d$ Pour démontrer $A \vee B$ il suffit de démontrer A ou bien de démontrer B.

$\vee_g$ Si $A \vee B$ est supposé vrai (ie est dans les hypothèses) alors ou bien A ou bien B est vrai. En d'autres termes on utilise un énoncé de la forme $A \vee B$ en raisonnant par cas : premier cas : A est vrai. deuxième cas : B est vrai.

Exemple 1.4.1 *Voici un exemple de preuve :*

$$\dfrac{\dfrac{\dfrac{}{C \vdash A, C}\,ax \quad \dfrac{\dfrac{}{A, B \vdash A, C}\,ax}{A \wedge B \vdash A, C}\,\wedge_g}{(A \wedge B) \vee C \vdash A, C}\,\vee_g}{(A \wedge B) \vee C \vdash (A \vee C)}\,\vee_d \qquad \dfrac{\dfrac{\dfrac{}{C \vdash B, C}\,ax \quad \dfrac{\dfrac{}{A, B \vdash B, C}\,ax}{A \wedge B \vdash B, C}\,\wedge_g}{(A \wedge B) \vee C \vdash B, C}\,\vee_g}{(A \wedge B) \vee C \vdash (B \vee C)}\,\vee_d}{(A \wedge B) \vee C \vdash (A \vee C) \wedge (B \vee C)}\,\wedge_d$$

On peut représenter les arbres de preuves de façon linéaire et de haut en bas. Notre exemple se représente de la façon suivante :

$$\begin{array}{ll}
(A \wedge B) \vee C \vdash (A \vee C) \wedge (B \vee C) & \wedge_d \\
\quad (A \wedge B) \vee C \vdash (A \vee C) & \vee_d \\
\quad\quad (A \wedge B) \vee C \vdash A, C & \vee_g \\
\quad\quad\quad A \wedge B \vdash A, C & \wedge_g \\
\quad\quad\quad\quad A, B \vdash A, C & ax \\
 & \\
\quad\quad\quad C \vdash A, C & ax \\
 & \\
\quad (A \wedge B) \vee C \vdash (B \vee C) & \vee_d \\
\quad\quad (A \wedge B) \vee C \vdash B, C & \vee_g \\
\quad\quad\quad A \wedge B \vdash B, C & \wedge_g \\
\quad\quad\quad\quad A, B \vdash B, C & ax
\end{array}$$

$$C \vdash B, C \qquad ax$$

remarque
Dans l'exemple précédent, nous avons systématiquement indenté. Il est possible sans nuire à la lisibilité de ne pas indenter lorsque la règle appliquée ne produit qu'un seul fils. Notre exemple deviendrait :

$$\begin{array}{ll}
(A \wedge B) \vee C \vdash (A \vee C) \wedge (B \vee C) & \wedge_d \\
\quad (A \wedge B) \vee C \vdash (A \vee C) & \vee_d \\
\quad (A \wedge B) \vee C \vdash A, C & \vee_g \\
\quad\quad A \wedge B \vdash A, C & \wedge_g \\
\quad\quad A, B \vdash A, C & ax \\
 & \\
\quad\quad C \vdash A, C & ax \\
 & \\
\quad (A \wedge B) \vee C \vdash (B \vee C) & \vee_d \\
\quad (A \wedge B) \vee C \vdash B, C & \vee_g \\
\quad\quad A \wedge B \vdash B, C & \wedge_g \\
\quad\quad A, B \vdash B, C & ax \\
 & \\
\quad\quad C \vdash B, C & ax
\end{array}$$

L'implication

$\rightarrow_d$ Pour démontrer $A \rightarrow B$ on démontre B en supposant A.
Cette règle énonce la similitude de comportement entre le connecteur $\rightarrow$ et le symbole $\vdash$. Dans le sens de la recherche de preuve (bas $\rightarrow$ haut) cette règle permet de créer des hypothèses en déconstruisant le but.

$\rightarrow_g$ Si $A \rightarrow B$ est en hypothèses, il suffit de démontrer A pour pouvoir ajouter B en hypothèses.

Exemple 1.4.2 *Voici un exemple de preuve :*

$$\dfrac{\dfrac{\begin{array}{c}d_1\\ \vdots \\ (A \wedge B) \vee C \vdash (A \vee C)\end{array} \quad \ldots \quad \begin{array}{c}d_2\\ \vdots \\ (A \wedge B) \vee C \vdash (B \vee C)\end{array}}{(A \wedge B) \vee C \vdash (A \vee C) \wedge (B \vee C)}\wedge_d}{\vdash ((A \wedge B) \vee C) \rightarrow ((A \vee C) \wedge (B \vee C))}\rightarrow_d$$

d_1 et d_2 sont les sous arbres de la preuves de l'exemple précédent (deuxième ligne en partant du bas).

Exemple 1.4.3

$A \to B, A \vdash B$ $\to_g$
$\quad A \vdash A$ ax

$\quad B, A \vdash B$ ax

Exemple 1.4.4

$\vdash ((A \wedge B) \to C) \to (A \to (B \to C))$ $\to_d$
$(A \wedge B) \to C \vdash A \to (B \to C)$ $\to_d$
$(A \wedge B) \to C, A \vdash B \to C$ $\to_d$
$(A \wedge B) \to C, A, B \vdash C$ $\to_g$
$\quad A, B \vdash A \wedge B$ $\wedge_d$
$\quad\quad A, B \vdash A$ ax

$\quad\quad A, B \vdash B$ ax

$\quad C, A, B \vdash C$ ax

La négation

Il est moins facile de paraphraser les règles de la négation. On peut justifier le raisonnement sous-jacent à ces règles en constatant que les formules correspondant à la prémisse et à la conclusion de la règle sont équivalentes. En effet la proposition 1.4.1 énoncée page 10 et démontrée page 17 nous dit que le sequent conclusion de la règle $\neg_g$: $A_1, \ldots, A_n, \neg A \vdash B_1, \ldots B_m$ est équivalent au séquent $\vdash (A_1 \wedge \ldots \wedge A_n \wedge \neg A) \to (B_1 \vee \ldots \vee B_m)$
De même le séquent prémisse $A_1, \ldots, A_n \vdash A, B_1, \ldots, B_m$ équivaut à : $\vdash (A_1 \wedge \ldots \wedge A_n) \to (A \vee (B_1 \vee \ldots \vee B_m))$. Or il est facile de montrer par les tables de vérité que $((A_1 \wedge \ldots \wedge A_n \wedge \neg A) \to (B_1 \vee \ldots \vee B_m)) \leftrightarrow ((A_1 \wedge \ldots \wedge A_n) \to (A \vee (B_1 \vee \ldots \vee B_m)))$.

La règle $\neg_d$ se motive de façon analogue.

Exemple 1.4.5

$$\dfrac{\dfrac{}{\Gamma, A, \vdash A, B}\, ax}{\Gamma, A, \neg A \vdash B}\, \neg_g$$

Cet exemple montre que dès que l'on possède une contradiction en hypothèse, la preuve réussit. Ceci correspond au fait que du faux (ie $A \wedge \neg A$ en hypothèse), on déduit n'importe quoi.

Exemple 1.4.6 (Le tiers exclu)

$$\dfrac{\dfrac{\dfrac{}{\Gamma, A, \vdash A}\,ax}{\Gamma \vdash A, \neg A}\,\neg_d}{\Gamma \vdash A \vee \neg A}\,\vee_d$$

Attention, si le fait d'avoir A et $\neg A$ en hypothèse signifie que l'on a une contradiction dans les hypothèses, le fait d'avoir $A, \neg A$ a droite de $\vdash$ signifie au contraire que l'on a une formule toujours vraie comme but ($A \vee \neg A$).

Exemple 1.4.7

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A, \vdash A}\,ax}{\Gamma \vdash \neg A, A}\,\neg_d}{\Gamma\neg\neg A \vdash A}\,\neg_g}{\Gamma \vdash \neg\neg A \to A}\,\to_d$$

La coupure

Cette règle permet d'utiliser des résultats déjà prouvés. Elle dit que l'on peut toujours ajouter en hypothèse une formule A du moment que l'on sait en produire une preuve. Supposons par exemple que nous voulions montrer que $pred(x) \leq x$ pour tout entier x , avec comme définition de $pred$: $x = 0 \to pred(x) = 0$ et $\neg(x = 0) \to pred(x) = x - 1$. La façon la plus simple de procéder est de raisonner par cas suivant que $x = 0$ ou non. Cela signifie que l'on va utiliser le tiers exclu que nous venons de démontrer. Nous allons faire une *coupure* sur ce résultat :
Posons $H_1 = x = 0 \to pred(x) = 0$ et $H_2 = \neg(x = 0) \to pred(x) = x - 1$

$$\dfrac{\dfrac{\vdots}{H_1, H_2, x = 0 \vee \neg(x = 0) \vdash pred(x) \leq x} \quad \dfrac{\vdots}{H_1, H_2 \vdash x = 0 \vee \neg(x = 0)}}{H_1, H_2 \vdash pred(x) \leq x}\,cut$$

La preuve de la prémisse droite se fera par $\to_g$ sur H_1 et H_2, la preuve de la prémisse droite n'est autre que la preuve de l'exemple 1.4.6.

Nous pouvons maintenant démontrer la propriété énoncée page 10 :

Proposition 1.4.1 *$A_1, \ldots, A_n \vdash B_1, \ldots B_m$ si et seulement si $\vdash (A_1 \wedge \ldots \wedge A_n) \to (B_1 \vee, \ldots, \vee B_m)$*

Démonstration
$\rightarrow$ Supposons que nous ayons une preuve $\mathcal{D}$ de $A_1, \ldots, A_n \vdash B_1, \ldots B_m$. Nous en déduisons une preuve de $\vdash (A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee, \ldots, \vee B_m)$ de la façon suivante :

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ \cdots\end{array}}{A_1, \ldots, A_n \vdash B_1, \ldots, B_m}}{A_1, \ldots, A_n \vdash (B_1 \vee, \ldots, \vee B_m)} \vee_d(mfois)}{(A_1 \wedge \ldots \wedge A_n) \vdash (B_1 \vee, \ldots, \vee B_m)} \wedge_g(nfois)}{\vdash (A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee, \ldots, \vee B_m)} \rightarrow_g$$

$\leftarrow$ Supposons que nous ayons une preuve $\mathcal{D}$ de $\vdash (A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee, \ldots, \vee B_m)$. Nous en déduisons une preuve de $A_1, \ldots, A_n \vdash B_1, \ldots B_m$ de la façon suivante : Posons $\Gamma = \{A_1, \ldots, A_n\}$ et $\Delta = \{B_1, \ldots, B_m\}$

$$\begin{array}{ll}
\Gamma \vdash \Delta & \text{cut} \\
\quad \vdash (A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee, \ldots, \vee B_m) & \mathcal{D} \\
 & \\
\quad \Gamma, (A_1 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee, \ldots, \vee B_m) \vdash \Delta & \rightarrow_g \\
\qquad \Gamma \vdash (A_1 \wedge \ldots \wedge A_n) & \wedge_d(nfois) \\
\qquad\quad \Gamma \vdash A_1 & \text{ax} \\
\qquad\quad \vdots & \\
\qquad\quad \Gamma \vdash A_n & \text{ax} \\
 & \\
\qquad \Gamma, (B_1 \vee, \ldots, \vee B_m) \vdash \Delta & \vee_g(mfois) \\
\qquad\quad \Gamma, B_1 \vdash \Delta & \text{ax} \\
\qquad\quad \vdots & \\
\qquad\quad \Gamma, B_m \vdash \Delta & \text{ax}
\end{array}$$

1.5 Les résultats fondamentaux

Nous allons énoncer les résultats fondamentaux du Calcul des propositions et esquisser leurs preuves. Ces résultats sont :

1. La correction : tout ce qui est prouvable est vrai
2. la complétude : tout ce qui est vrai est prouvable
3. La décidabilité du calcul des séquents.

Avant de commencer, remarquons que les règles d'affaiblissement sont redondantes. Il est facile de démontrer par récurrence sur la longueur de la preuve,

qu'une preuve qui utilise ces règles peut etre transformée en une preuve qui ne les utilise pas.
Dans la suite, nous allons donc considérer le calcul des séquents sans ces deux règles.
Nous allons dans un premier temps omettre la règle de coupure.

1.5.1 Le calcul des séquents sans coupure

Théorème 1.5.1 (Correction) *Soit* Δ *et* Γ *deux ensembles de formules, Alors,*

Si $\Gamma \vdash \Delta$ *alors toute interprétation rendant vraie toutes les formules de* Γ *rend aussi vraie l'une des formules de* Δ

Démonstration
Le preuve se fait sans difficulté par récurrence sur la longueur de la dérivation. Cas de base : si on a $\dfrac{}{\Gamma \vdash \Delta}\, ax$ alors c'est que Γ et Δ ont au moins une formule A en commun et donc une interprétation rendant vraie toutes les formules de Δ rend vraie A qui est une des formules de Δ.

étape inductive :Examinons le cas du $\vee_d$, les autres cas se traitant de façon similaire. Si on a $\dfrac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \vee B, \Delta} \vee_d$ On a par hypothèse de récurrence une interprétation rendant vraie toute formule de Γ et A ou B ou l'une des formules de Δ. Si c'est A ou B qui est rendue vraie alors $A \vee B$ l'est aussi, et si c'est une des formules de Δ le résultat est trivial.

Théorème 1.5.2 (Complétude) *Soit* Δ *et* Γ *deux ensembles de formules, Alors :*
Si toute interprétation rendant vraie toutes les formules de Γ *rend aussi vraie l'une des formules de* Δ*,*
alors $\Gamma \vdash \Delta$

Démonstration
Supposons donc que l'on ait une interprétation rendant vraie toute formule de Γ et l'une des formules de Δ (1).
On construit une dérivation de $\Gamma \vdash \Delta$ de la façon suivante : on choisit la formule de $\Gamma \vdash \Delta$ la plus longue. Si elle est dans Γ et on lui applique la règle gauche correspondant à son connecteur principal et si elle est dans Δ et on lui applique la règle droite correspondant à son connecteur principal. On itère ce processus sur les sous buts obtenus tant que c'est possible. On obtient donc une dérivation dont toutes les feuilles sont de la forme $\Gamma' \vdash \Delta'$ ne contenant plus que des variables propositionnelles. Si tous ces séquents se terminent par la règle axiome, cette dérivation est la preuve recherchée de $\Gamma \vdash \Delta$. Sinon, on a donc au moins un noeud $\Gamma' \vdash \Delta'$ qui n'est pas un axiome c'est à dire que

$\Gamma' \cap \Delta' = \emptyset$. On construit alors l'interprétation I sur $\Gamma' \cup \Delta'$ par : $I(F) =$ vrai si $F \in \Delta$ et $I(F) =$ faux si $F \in \Gamma$. En examinant chacune des règles du calcul des séquents sans coupure et sans affaiblissement, il est facile de voir que pour chaque séquent $\Gamma'' \vdash \Delta''$ en dessous de $\Gamma' \vdash \Delta'$, I rend vraie toute formule de Γ'' et fausse toutes formule de Δ''. Ceci est donc vrai en particulier de $\Gamma \vdash \Delta$ ce qui contredit (1). Il est donc impossible que la dérivation construite ne soit pas une preuve.

Théorème 1.5.3 (Décidabilité) *Le calcul des séquents propositionnel sans coupure est décidable i.e. il existe un algorithme qui s'arrête toujours et qui reussit si et seulement si son entrée est prouvable dans ce système.*

Démonstration
Le cas du Calcul des séquents sans coupure a la propriété suivante : pour toutes ses règles, les formules des séquents prémisses (en haut de la règle) sont des sous formules des formules du séquent conclusion (en bas de la règle). Ceci nous donne la *propriété de la sous formule* :

Proposition 1.5.1 (propriété de la sous formule) *Les formules figurant dans une preuve en Calcul des Séquents propositionnel sans coupure sont des sous formules du séquent conclusion de la preuve.*

Cette propriété nous assure que l'espace de recherche d'une preuve se réduit aux sous formules du but. Or, l'ensemble des sous formules d'un séquent est un ensemble fini. Il est donc possible d'énumerer en un temps fini toutes les preuves possibles d'un séquent, ce qui rend le problème décidable.

Plus précisemment, pour prouver un énoncé du Calcul des Propositions avec le Calcul des Séquents, on peut procéder de la façon suivante : appliquer sur chacune des formules, la règle correspondant à son connecteur principal. Itérerer ce processus jusquà avoir des sous buts ne contenant que des formules atomiques. Si chacun d'eux se termine par la règle `ax`, alors on a une preuve.
Si ce n'est pas le cas alors chaque séquent qui ne se termine pas par `ax` donne une interprétation qui falsifie l'énoncé de départ : il suffit de donner l'interprétation `vrai` aux formules qui sont à gauche, et la valeur `faux` à celles qui sont à droite.

Exemple 1.5.1

$\vdash B \to ((A \to B) \wedge A)$ $\to_d$
$B \vdash ((A \to B) \wedge A)$ $\wedge_d$
$\quad B \vdash A \to B$ $\to_d$
$\quad B, A \vdash B$ ax

$\quad B \vdash A$ *echec*

l'interprétation { (A,f),(B,v) } rend la formule $B \to ((A \to B) \wedge A)$ *fausse.*

1.5.2 le Cas de la coupure

Nous avons travaillé sans la règle de coupure. Le fait d'avoir démontré la complétude sans cette règle nous dit que cette règle n'est pas nécessaire au système au sens où chaque formule vraie est démontrable sans coupure. Il reste cependant à nous assurer que cette règle est correcte, c'est à dire que l'on continue à ne prouver que des formules vraies avec le Calcul des séquents avec coupure. Remarquons aussi que cette règle rompt la propriété de la sous formule et rend infini l'espace de recherche. Il nous faut donc nous assurer que si cette règle ajoute des preuves, elle n'ajoute que des preuves de buts déjà démontrable dans le système sans coupure. Ceci nous est donné par le *théorème délimination des coupures* de *Gentzen* (1934) qui démontre qu'il est possible de transformer effectivement toute preuve utilisant la coupure en une preuve sans coupure.

1.6 Rendre le système utilisable

L'ensemble des règles définissant le Calcul des Séquents ne constitue pas un "bon" système pour qui veut effectivement écrire des preuves formelles. L'élémentarité des règles rend l'ecriture des preuves longue et fastidieuse.
Nous voudrions maintenant nous donner les moyens d'ecrire des preuves de façon un peu plus confortable en ajoutant des règles. Mais il ne faut pas sortir du Calcul des Séquents : si tel était le cas, nous perdrions toute assurance sur la correction de nos preuves. Les extensions que nous voulons proposer se doivent donc d'être transparentes du point de vue théorique. Nous nous assurerons donc scrupuleusement qu'une preuve avec les extensions que nous proposerons est toujours traduisible en une preuve en Calcul des Séquents purs. Nous allons étendre le système de façon à permettre

1. la définition de nouveaux objets
2. la définition de règles dérivées

1.6.1 Définitions

Les objets primitifs du langage sont peu nombreux. Il serait utile de pouvoir définir de nouveaux objets à partir des objets primitifs. Il ne s'agit de rien d'autres que de donner un nom à des expressions du langage.
C'est le cas par exemple du connecteur $\leftrightarrow$.
$A \leftrightarrow B$ désigne la formule $(A \rightarrow B) \wedge (B \rightarrow A)$.
Pour effectuer de telles définitions, nous utiliserons le symbole $\stackrel{\text{def}}{=}$. Ainsi :

$$A \leftrightarrow B \stackrel{\text{def}}{=} (A \rightarrow B) \wedge (B \rightarrow A)$$

Dans les preuves, à chaque fois que nous rencontrerons un objet défini par $\stackrel{\text{def}}{=}$, nous pourrons le remplacer par sa définition.

Nous introduisons donc dans les preuves, la notation suivante qui permet dans une preuve d'ouvrir une definition :

$$\frac{(\Gamma \vdash \Delta)[O//E]}{\Gamma \vdash \Delta} \stackrel{\text{def}}{=} (O)*$$

(*) : si $O \stackrel{\text{def}}{=} E$ est présente dans la bibliothèque de définitions.

La notation $(\Gamma \vdash \Delta)[O//E]$ désigne le remplacement de une ou plusieurs occurrences de O par E dans $\Gamma \vdash \Delta$.
A partir d'une preuve comportant des définitions et l'utilisation de def, on obtient une preuve en Calcul des Séquents pur en remplacant dans chaque séquent les objets par leur définition et et enlevant les utilisations de la "macro" $\stackrel{\text{def}}{=}$.

Exemple 1.6.1 *Montrons* $(P \to Q) \leftrightarrow (\neg P \vee Q)$

$$\begin{array}{ll}
\vdash (P \to Q) \leftrightarrow (\neg P \vee Q) & \stackrel{\text{def}}{=} (\leftrightarrow) \\
\quad \vdash (P \to Q) \to (\neg P \vee Q) & \to_d \\
\quad P \to Q \vdash (\neg P \vee Q) & \vee_d \\
\quad P \to Q \vdash \neg P, Q & \to_g \\
\qquad \vdash P, \neg P, Q & \neg_d \\
\qquad P \vdash P, Q & ax \\
 & \\
\qquad Q \vdash \neg P, Q & ax \\
 & \\
\quad \vdash (\neg P \vee Q) \to (P \to Q) & \to_d \\
\quad \neg P \vee Q \vdash P \to Q & \to_d \\
\quad \neg P \vee Q, P \vdash Q & \vee_g \\
\qquad \neg P, P \vdash Q & contr \\
 & \\
\qquad Q, P \vdash Q & ax
\end{array}$$

Exercice 1.6.1 *Montrer que* $\vdash A \leftrightarrow A$

Exemple 1.6.2 *Montrons :* $\vdash \neg\neg P \leftrightarrow P$

$$\begin{array}{ll}
\vdash \neg\neg P \leftrightarrow P & \stackrel{\text{def}}{=} (\leftrightarrow) \\
\quad \vdash \neg\neg P \to P & exemple\ \ 1.4.7 \\
 & \\
\quad P \vdash \neg\neg P & \neg_d \\
\quad P, \neg P \vdash & contr
\end{array}$$

Exemple 1.6.3 *Montrons* $\vdash \neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$

$$
\begin{array}{ll}
\vdash \neg(A \vee B) \leftrightarrow \neg A \wedge \neg B & \stackrel{\text{def}}{=} (\leftrightarrow) \\
\quad \vdash \neg(A \vee B) \rightarrow \neg A \wedge \neg B & \rightarrow_d \\
\quad \neg(A \vee B) \vdash \neg A \wedge \neg B & \neg_g \\
\quad \vdash A \vee B, \neg A \wedge \neg B & \vee_d \\
\quad \vdash A, B, \neg A \wedge \neg B & \wedge_d \\
\qquad \vdash A, B, \neg A & \neg_d \\
\qquad A \vdash A, B & ax \\
 & \\
\qquad \vdash A, B, \neg B & non_d \\
\qquad B \vdash A, B & ax \\
 & \\
\quad \vdash (\neg A \wedge \neg B) \rightarrow \neg(A \vee B) & \rightarrow_d \\
\quad \neg A \wedge \neg B \vdash \neg(A \vee B) & \wedge_g \\
\quad \neg A, \neg B \vdash \neg(A \vee B) & \neg_d \\
\quad \neg A, \neg B, A \vee B \vdash & \neg_g \textit{(2x)} \\
\quad A \vee B \vdash A, B & \vee_g \\
\qquad A \vdash A, B & ax \\
 & \\
\qquad B \vdash A, B & ax
\end{array}
$$

1.6.2 Définir ses propres règles

Les règles de base expriment des raisonnements tout à fait élémentaires. On voudrait pouvoir ajouter des règles correspondant à des raisonnements plus complexes. Nous pouvons le faire à condition de démontrer que ces règles sont correctes, c'est à dire qu'elles sont définissables à partir des règles de base. Nous appellerons de telle règles des règles dérivées :

Définition 1.6.1 *Une* règle dérivée *est une règle :*

$$\frac{\Gamma \vdash \phi_1 \qquad \ldots \qquad \Gamma \vdash \phi_n}{\Gamma \vdash \phi}$$

telle que si $\Gamma \vdash \phi_1 \ldots \Gamma \vdash \phi_n$ *sont prouvables en calcul des séquents alors* $\Gamma \vdash \phi$ *est aussi prouvable en calcul des séquents.*

Pour ajouter une règle dérivée, il suffit de la formuler et de prouver sa correction. Cette preuve nous assure qu'une règle dérivée n'est rien d'autre qu'une notation pour un bout de preuve (la preuve de correction de la règle). La définition que nous venons de donner nous assure donc que toute preuve faite avec des règles dérivées peut se traduire en une preuve du Calcul des Séquents pur : il suffit de remplacer chaque application de règles dérivées par sa preuve de correction.

Le raisonnement par l'absurde

Le raisonnement par l'absurde correspond à : pour montrer A, je suppose que A est faux et je montre qu'on arrive à une contradiction. Une règle correspondant à ceci serait donc :

$$\boxed{\dfrac{\Gamma, \neg A \vdash \Delta}{\Gamma \vdash A, \Delta}\, abs}$$

Montrons que c'est une règle dérivée. Il suffit de produire une preuve de $\Gamma \vdash A, \Delta$ en calcul des séquents purs où l'unique branche non fermée est $\Gamma, \neg A \vdash \Delta$:

$$\dfrac{\dfrac{\dfrac{}{\Gamma, A \vdash A, \Delta}\, ax \quad \ldots \quad \dfrac{\Gamma, \neg A \vdash \Delta}{\Gamma \vdash \neg\neg A, \Delta}\, \neg_d}{\Gamma, \neg\neg A \to A \vdash A, \Delta}\, \to_g \qquad \begin{array}{c} d_1 \\ \vdots \\ \Gamma \vdash \neg\neg A \to A, \Delta \end{array}}{\Gamma \vdash A, \Delta}\, cut$$

où d_1 n'est autre que la preuve de l'exercice 1.4.7

Des hypothèses contradictoires

Montrer que la règle suivante est une règle dérivée du calcul des Propositions :

$$\boxed{\dfrac{}{\Gamma, A, \neg A \vdash \Delta}\, contr}$$

Reflexivité de l'équivalence

Voici encore une règle terminale

$$\boxed{\dfrac{}{\Gamma \vdash A \leftrightarrow A, \Delta}\, reflequiv}$$

dont la correction est directement issues de l'exercice 1.6.1.

Une nouvelle règle pour $\to$

Lorsque nous avons A et $A \to B$ en hypothèse, nous pourons ajouter B en hypothèse.

$$\boxed{\dfrac{\Gamma A, A \to B, B \vdash \Delta}{\Gamma, A, A \to B \vdash \Delta}\, MP}$$

La preuve de correction de cette règle est triviale: c'est l'application de $\rightarrow_g$, puis de ax dans la branche correspondant à la preuve de A :

utiliser des théorèmes

Une fois prouvé un théorème, il est possible grace à la règle *cut* de le réutiliser dans une preuve ultérieure. Toute preuve d'un énonçé quelque peu complexe se construit humainement de cette façon. Tout système effectif de preuve formelle se doit donc de gérer une bibliothèque de théorèmes déjà prouvés. Si nous pouvons laisser implicite la gestion de la bibliothèque proprement dite, en supposant par exemple qu'a un instant t de ce cours la bibliothèque de théorèmes est constituée de toutes les propriétés et exercices déjà effectués, nous nous devons d'avoir des règles permettant une utilisation aisée des théorèmes. Or, cela n'est pas le cas de la règle *cut*. Invoquer un théorème par un cut scinde la preuve en deux sous preuves, dont l'une est toujours fermée par la preuve du théorème invoqué. Il serait beaucoup plus économique d'avoir une régle *cut* paramétrée par le nom d'un théorème et qui deviendrait par la même unaire :

$$\frac{\Gamma, th \vdash \Delta}{\Gamma \vdash \Delta}\, cut_1(th)*$$

(*) : th est un théorème déjà prouvé.

La correction d'une telle "macro" est triviale puisque si la condition (*) est respectée nous sommes en possession d'une preuve d_1 de th. C'est donc la notation de la preuve "pure" :

$$\dfrac{\dfrac{\begin{array}{c} d_1 \\ \vdots \\ \vdash th \end{array}}{\Gamma \vdash th}\, aff_g \qquad \Gamma, th \vdash \Delta}{\Gamma \vdash \Delta}\, cut$$

utiliser des théorèmes de la forme $A \leftrightarrow B$

De nombreux théorèmes sont des équivalences. Depuis le collège, nous sommes habitués à prouver des résultats par substitution successive de formule par une formule équivalente comme dans l'exemple suivant : Si nous avons déjà que

1. $\vdash (P \rightarrow Q) \leftrightarrow \neg P \vee Q$
2. $\vdash \neg(A \vee B) \leftrightarrow \neg A \vee \neg B$
3. $\vdash \neg\neg A \leftrightarrow A$

Nous montrons $\vdash (P \wedge \neg Q) \leftrightarrow \neg(P \rightarrow Q)$ de la façon suivante :
$\neg(P \rightarrow Q) \leftrightarrow$ (par 1.)
$\neg(\neg P \vee Q) \leftrightarrow$ (par 2.)
$\neg\neg P \wedge \neg Q \leftrightarrow$ (par 3.)
$P \wedge \neg Q$
CQFD

La définition de l'équivalence que nous avons, ne nous permet pas de procéder comme cela. Par elle, nous ne pouvons utiliser une équivalence qu' en la scindant en deux implications.

Nous voulons donc étendre notre système avec une règle nous permettant de raisonner comme au collège, en substituant à n'importe quelle sous formule d'un séquent une formule qui lui est équivalente. cette règle peut par exemple se formuler comme suit :

$$\frac{(\Gamma \vdash \Delta)[A//B]}{\Gamma \vdash \Delta} \leftrightarrow (A \leftrightarrow B)*$$

(*) :si $A \leftrightarrow B$ est un théorème.

remarque : $(\Gamma \vdash \Delta)[A//B]$ désigne le séquent $\Gamma \vdash \Delta$ dans lequel une ou plusieurs occurrences de la sous formule A est remplacée par la formule B.
Notons que cette règle est pour le moment la seule qui permettent d'agir sur autre chose que les connecteurs principaux des formules du séquent.

Pour montrer la correction de cette règle, nous allons procéder en deux étapes :

1. Nous formulerons une règle dérivée $\leftrightarrow_1$ et montrerons sa correction
2. nous utiliserons cette règle dérivée pour pour montrer la correction de $\leftrightarrow$.

la règle $\leftrightarrow_1$

$$\frac{\Gamma \vdash A \leftrightarrow B}{\Gamma \vdash F[A] \leftrightarrow F[B]} \leftrightarrow_1$$

Pour montrer la correction de cette règle, il faut montrer que :

Proposition 1.6.1 *Pour toutes formules A et B si $\vdash A \leftrightarrow B$ alors, pour toute formule F contenant A il existe une preuve en calcul des Séquents pur de $\vdash F(A) \leftrightarrow F(B)$.*

La démonstration se fait par récurrence sur la complexité de la formule F. Nous ne la détaillerons pas ici.

correction de $\leftrightarrow (A \leftrightarrow B)$

Pour simplifier, nous allons traiter le cas ou A n'a qu'une seule occurrence dans le séquent $\Gamma \vdash \Delta$ et raisonnerons par cas suivant que cette occurrence est dans une hypothèse ou dans un but. Si la règle $\leftrightarrow (A \leftrightarrow B)$ est applicable, nous avons une preuve d_1 de $A \leftrightarrow B$

Premier cas : Si A apparait dans une hypothèse notre règle correspond à la preuve suivante :

$\Gamma, F[A] \vdash \Delta$ cut
$\quad \Gamma \vdash F[A] \leftrightarrow F[B]$ $\leftrightarrow_1$
$\quad \Gamma \vdash A \leftrightarrow B$ d_1

$\quad \Gamma, F[A], F[A] \leftrightarrow F[B] \vdash \Delta$ $\stackrel{\text{def}}{=} (\leftrightarrow)$
$\quad \Gamma, F[A], F[A] \rightarrow F[B], F[B] \rightarrow F[A] \vdash \Delta$ MP
$\quad \Gamma, F[B] \vdash \Delta$ MP

La seule branche non fermée est bien la prémisse de notre règle.

Deuxième cas : Si A apparait dans un but notre règle correspond à la preuve suivante :

$\Gamma \vdash F[A], \Delta$ cut
$\quad \Gamma \vdash F[A] \leftrightarrow F[B]$ $\leftrightarrow_1$
$\quad \Gamma \vdash A \leftrightarrow B$ d_1

$\quad \Gamma, F[A] \leftrightarrow F[B] \vdash F[A], \Delta$ $\stackrel{\text{def}}{=} (\leftrightarrow)$
$\quad \Gamma, F[A] \rightarrow F[B], F[B] \rightarrow F[A] \vdash F[A], \Delta$ $\rightarrow_g + aff_g$
$\quad\quad \Gamma, \vdash F[B], \Delta$
$\quad\quad \Gamma, F[A] \vdash F[A], \Delta$ ax

La seule branche non fermée est bien la prémisse de notre règle.

Exemple 1.6.4 $\vdash (P \wedge \neg Q) \leftrightarrow \neg(P \rightarrow Q)$ $\leftrightarrow$ *(ex 1.6.1)*
$\vdash (P \wedge \neg Q) \leftrightarrow \neg(\neg P \vee Q)$ $\leftrightarrow$ *(ex 1.6.3)*
$\vdash (P \wedge \neg Q) \leftrightarrow \neg\neg P \wedge \neg Q$ $\leftrightarrow$ *(ex 1.6.2)*
$\vdash (P \wedge \neg Q) \leftrightarrow P \wedge \neg Q$ *ex 1.6.1*

1.7 Quelques Propriétés

double négation	$\vdash \neg\neg A \leftrightarrow A$
tiers exclu	$\vdash A \vee \neg A$
distributivité	$\vdash ((A \wedge B) \vee C) \leftrightarrow ((A \vee C) \wedge (B \vee C))$ $\vdash (A \vee B) \wedge C \leftrightarrow (A \wedge C) \vee (B \wedge C)$ $\vdash (A \rightarrow (B \wedge C)) \leftrightarrow (A \rightarrow B) \wedge (A \rightarrow C)$
associativité	$\vdash (A \vee (B \vee C)) \leftrightarrow ((A \vee B) \vee C)$ $\vdash (A \wedge (B \wedge C)) \leftrightarrow ((A \wedge B) \wedge C)$
commutativité	$\vdash (A \vee B) \leftrightarrow (B \vee A)$ $\vdash (A \wedge B) \leftrightarrow (B \wedge A)$
Loi de Morgan	$\vdash \neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$ $\vdash \neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$ $\vdash \neg(A \rightarrow B) \leftrightarrow A \wedge \neg B$
transitivité	$A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$
	$\vdash A \leftrightarrow A$ $\vdash \neg\neg P \leftrightarrow P$ $A \wedge A \leftrightarrow A$ $A \vee A \leftrightarrow A$ $(P \rightarrow Q) \leftrightarrow (\neg P \vee Q)$ $\vdash (A \vee B) \leftrightarrow \neg(\neg A \wedge \neg B)$ $\vdash ((A \rightarrow B) \wedge A) \rightarrow B$ $\vdash ((A \wedge B) \rightarrow C) \leftrightarrow (A \rightarrow (B \rightarrow C))$

1.8 Exercices

Exercice 1.8.1 *Prouver :* $A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$

$A \to B, B \to C \vdash A \to C$ $\to_d$
$A \to B, B \to C, A \vdash C$ $\to_g$
$A \to B, B \to C, A \vdash A$ ax

$B, B \to C, A \vdash C$ $\to_g$
$B, A \vdash B$ ax

$B, C, A \vdash C$ ax

Nota bene : On a definit la regle derivee suivante :

$$\frac{\Gamma, A, , B \vdash \Delta}{\Gamma, A, A \to B \vdash \Delta}\ MP$$

En utilisant cette regle la preuve devient :

$A \to B, B \to C \vdash A \to C$ $\to_d$
$A \to B, B \to C, A \vdash C$ MP
$B \to C, A, B \vdash A$ MP
$B, C, A \vdash C$ ax

Exercice 1.8.2 *On juge 3 hommes : A B et C. On sait que :*

1. *Si A est innocent ou B coupable, alors C est coupable.*
2. *Si A est innocent alors C est innocent*

1. *Formaliser :*
   (1) $(\neg A \vee B) \to C$
   (2) $\neg A \to \neg C$
2. *Montrer que A est coupable, par un raisonnement informel, par les tables de verite et par une preuve en calcul des séquents.*
   Raisonnement informel : Si A est innocent alors par (1) C coupable et par (2) C est innocent. On ne peut donc supposer A innocent sans aboutir a une contradiction , donc A est coupable.

preuve :
$(1), (2) \vdash A$ *absurde*
$(1), (2), \neg A \vdash$ $\to_g$
$(1), \neg A \vdash \neg A$ ax

$(1), \neg A, \neg C \vdash$ $\to_g$
$\neg A, \neg C \vdash \neg A \vee B$ $\vee_d$
$\neg A, \neg C \vdash \neg A, B$ ax

$\neg A, \neg C, C \vdash$ $\neg_g$
$\neg A, C \vdash C$ ax

Exercice 1.8.3 *On juge un homme accusé d'avoir trempé dans un cambriolage. Le procureur et l'avocat disent tour à tour :*
Le procureur : Si l'accusé est coupable, alors il a un complice.
L'avocat : C'est faux!
Pourquoi est ce la pire des choses que puisse dire l'avocat ?

Le procureur dit en fait : l'accusé n'a pas agit seul. Donc l'avocat dit : si il a agit seul. Donc il dit qu'il est coupable.

Formalisation :
(1) $A \rightarrow B$
(2) $\neg(A \rightarrow B)$

Montrons :
$(2) \vdash A$ $\neg_g$
$\vdash A, A \rightarrow B$ $\rightarrow_d$
$A \vdash A, B$ ax

Cette preuve peut aussi se faire par équivalence, si on suppose que l'on a déjà montré la loi de Morgan suivante : $\neg(A \rightarrow B) \leftrightarrow A \wedge \neg B$.
$\neg(A \rightarrow B) \vdash A$ $\leftrightarrow (Morgan)$
$A \wedge \neg B \vdash A$ $\wedge_g$
$A, \neg B \vdash A$ ax

Exercice 1.8.4 *Supposons que les affirmations suivantes soient vraies :*

1. *J'aime Elisabeth ou j'aime Jeanne*
2. *Si j'aime Elisabeth , alors j'aime Jeanne*

Que peut on en deduire ?

Si j'aime E alors j'aime J par (2).
Si je n 'aime pas E alors j'aime J par (1).
Donc j'aime jeanne dans tous les cas.

Formalisation :
(1) $E \vee J$
(2) $E \rightarrow J$

Preuve :

$$\begin{array}{lr}
(1),(2) \vdash J & cut \\
\quad (1),(2) \vdash E \vee \neg E & th \\
 & \\
\quad (1),(2),E \vee \neg E \vdash J & \vee_g \\
\quad\quad (1),(2),E \vdash J & \rightarrow_g \\
\quad\quad\quad (1),E \vdash E & ax \\
 & \\
\quad\quad\quad (1),E,J \vdash J & ax \\
 & \\
\quad\quad (1),(2),\neg E \vdash J & \vee_g \\
\quad\quad\quad (2),\neg E,E \vdash J & contr \\
 & \\
\quad\quad\quad (2),\neg E,J \vdash J & ax
\end{array}$$

Exercice 1.8.5 *Un vol a ete commis. 3 malfaiteurs notoires A B C sont apprehendes. On a etablit les faits suivants :*

1. *Nul autre que A B ou C ne peuvent etre impliques*
2. *A ne travaille jamais sans un complice*
3. *C est innocent*

1. *formalisation*
   (1) $CA \vee CB \vee CC$
   (2) $CA \rightarrow (CB \vee CC)$
   (3) $\neg CC$
2. *B est il innocent ou coupable ?*
   B est coupable .

   raisonnement :
   Si A coupable : alors par (2) et comme C est innocent, B est coupable.
   Si A innocent : alors par (1) B est coupable Donc dans tous les cas B est coupable.

   table de verite

A	B	C	*1*	*2*	*3*	$1,2,3 \rightarrow B$
v	v	v	v	v	f	v
v	v	f	v	v	v	v
v	f	v	v	v	f	v
v	f	f	v	f	v	v
f	v	v	v	v	f	v
f	v	f	v	v	v	v
f	f	v	v	v	f	v
f	f	f	f	v	v	v

preuve :
Posons :
(1) $CA \vee CB \vee CC$
(2) $CA \rightarrow (CB \vee CC)$
(3) $\neg CC$

$1,2,3 \vdash CB$ *cut(tiers exclu)*
$1,2,3, CA \vee \neg CA \vdash CB$ $\vee_g$
$1,2,3, CA \vdash CB$ $MP(2 + CA)$
$1,2,3, CA, CB \vee CC \vdash CB$ $\vee_g$
$1,2,3, CA, CB \vdash CB$ *ax*

$1,2,3, CA, CC \vdash CB$ *contr*

$1,2,3, \neg CA \vdash CB$ $\vee_g$*(2* sur 1)*
$CA, 2,3, \neg CA \vdash CB$ *contr*

$CB, 2,3, \neg CA \vdash CB$ *ax*

$CC, 2,3, \neg CA \vdash CB$ *contr*

Exercice 1.8.6 *Supposons qu'on me demande : Est il vrai que si tu aimes Eve alors tu aimes Marguerite aussi ? et que je reponde : Si c'est vrai, alors j'aime Eve.*
Qu'en déduisez vous ?

Pour que mon affirmation soit vraie il faut soit que si j'aime E alors j'aime M et que j'aime Eve, soit que si j'aime Eve alors j'aime M soit faux c'est a dire que j'aime E et pas M. J'aime donc E dans tous les cas

Formalisation :
$(E \rightarrow M) \rightarrow E$

Preuve :

$$\begin{array}{ll}
(E \to M) \to E \vdash E & \to_g \\
\quad \vdash E \to M, E & \to_d \\
\quad E \vdash M, E & ax \\
 & \\
\quad E \vdash E & ax
\end{array}$$

Exercice 1.8.7 *Prouver :* $(A \vee B) \wedge C \vdash (A \wedge C) \vee (B \wedge C)$

$$\begin{array}{ll}
(A \vee B) \wedge C \vdash (A \wedge C) \vee (B \wedge C) & \vee_d \\
(A \vee B) \wedge C \vdash A \wedge C, B \wedge C & \wedge_g \\
A \vee B, C \vdash A \wedge C, B \wedge C & \vee_g \\
\quad A, C \vdash A \wedge C, B \wedge C & \wedge_d \\
\qquad A, C \vdash A, B \wedge C & ax \\
 & \\
\qquad A, C \vdash C, B \wedge C & ax \\
 & \\
\quad B, C \vdash A \wedge C, B \wedge C & \wedge_d \\
\qquad B, C \vdash A \wedge C, B & ax \\
 & \\
\qquad B, C \vdash A \wedge C, C & ax
\end{array}$$

Exercice 1.8.8 *Prouver :* $A \vee B \vdash \neg(\neg A \wedge \neg B)$

$$\begin{array}{ll}
A \vee B \vdash \neg(\neg A \wedge \neg B) & \neg_d \\
A \vee B, \neg A \wedge \neg B \vdash & \wedge_g \\
A \vee B, \neg A, \neg B \vdash & \neg_g(2) \\
A \vee B \vdash B, A & \vee_g \\
\quad A \vdash B, A & ax \\
 & \\
\quad B \vdash B, A & ax
\end{array}$$

Chapter 2

Calcul des Prédicats

Une proposition est un énoncé, un jugement pris en sa totalité.
exemple :
Le chat est noir
Il fait beau
Il ne fait pas beau
Il fait beau ou il ne fait pas beau
Les deux premières propositions sont atomiques c'est à dire indécomposables en propositions plus petites contrairement aux deux dernières qui se décomposent comme suit :
NON il fait beau
il fait beau **OU NON** il fait beau
Ce découpage en propositions est insuffisant pour rendre compte de tous les types de raisonnement. L'énoncé "le chat est noir" peut faire l'objet d'une analyse plus fine : il relie une propriété **etre noir** et un individu qui possède cette propriété **le chat**. La propriété **etre noir** s'appelle un prédicat à une place et **le chat** une constante d'individu et ces nouveaux éléments formeront le socle (les objets primitifs) sur lequel on construit les formules du calcul des prédicats. Les formules atomiques ne sont plus maintenant la donnée d'un ensemble de symboles de propositions mais seront de la forme **P(t)** où **P(-)** est une symbole de prédicat et **t** un terme (un individu).

2.1 Définir le langage

2.1.1 L'alphabet

Le langage du calcul des prédicats est formé des symboles suivants :

- un ensemble de symboles de constante : a,b,c ...
- un ensemble de symboles de prédicats d'arité fixée : A, B, P, Q, ...

- un ensemble de variables d'individus : x,y,z, ...
- un ensemble de symboles de fonctions d'arité fixée : f,g,h ...

2.1.2 les termes

Les termes sont formés à partir des règles suivantes :

1. Les constantes et les variables sont des termes.
2. Si f est un symbole de fonction d'arité n et si $t_1 \dots t_n$ sont des termes, alors $f(t_1, \dots, t_n)$ est un terme.

2.1.3 les formules atomiques

Les formules atomiques sont formées à partir de la règle suivante :

1. Si P est un symbole de prédicat d'arité n et si $t_1 \dots t_n$ sont des termes, alors $P(t_1, \dots, t_n)$ est une formule atomique.

2.1.4 les formules

En passant des propositions aux prédicats, nous avons augmenté le pouvoir d'expression de notre langage. Nous pouvons comme dans le calcul des propositions combiner les formules au moyen des connecteurs pour former des formules plus complexes. Mais nous allons aussi ajouter deux nouveaux constructeurs de formules qui permettent de dire quelque chose sur les termes dont parlent les formules. Ce sont les quantificateurs. Le quantificateur universel ($\forall$) permettra d'exprimer le fait que tout les individus possèdent une propriété et le quantificateur existenciel ($\exists$) permettra d'exprimer le fait qu'il existe un individu qui possèdent une propriété. Les formules sont formées à partir des règles suivantes :

- les formules atomiques sont des formules
- Si A et B sont des formules alors $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, (A) et $\neg A$ sont des formules
- Si A est une formule et x une variable alors $\forall x A$ et $\exists x A$ sont des formules.

2.2 Sémantique: validité, satisfaisabilité

Comme pour le calcul des propositions, nous allons définir la notion de vérité, en nous appuyant sur la notion d'interprétation. Mais, les formules faisant intervenir des individus, les *termes*, cela sera plus complexe. Nous définirons d'abord la notion de *réalisation*, qui permet d'interpréter les termes et les prédicats, puis

celle d'*interprétation* permettant de donner une valeur aux variables, et finalement celles de *satisfaction* et de *validité*.

Définition 2.2.1 *Soit L un langage du calcul des prédicats. Une* réalisation *de L est la donnée de :*

- *Un ensemble D non vide appelé domaine.*
- *Un élément de D pour chaque constante de L*
- *Une application de $D^n \to D$ pour chaque symbole de fonction de L*
- *Un ensemble de n-uplets pour chaque symbole de prédicat d'arité n.*

Définition 2.2.2 *Soit L un langage du calcul des prédicats et M une réalisation de L. Une* interprétation *pour M et L est une fonction de l'ensemble des variables de L vers le domaine de M.*

Nous noterons $\bar{c}$ l'element associé a l'objet c de L par l'interprétation.

Définition 2.2.3 *On définit par récurrence sur la formation des formules la relation de* satisfaction *d'une formule par une interprétation et une réalisation (notée $M \models_I F$)*

- *Si $F = P(t_1, \dots t_n)$ alors $M \models_I F$ ssi $(t_1, \dots t_n) \in \bar{P}$*
- *Si $F = A \vee B$ alors $M \models_I F$ ssi $M \models_I A$ ou $M \models_I B$*
- *Si $F = A \wedge B$ alors $M \models_I F$ ssi $M \models_I A$ et $M \models_I B$*
- *Si $F = A \to B$ alors $M \models_I F$ ssi $M \not\models_I A$ ou $M \models_I B$*
- *Si $F = \neg A$ alors $M \models_I F$ ssi $M \not\models_I A$*
- *Si $F = \forall x P$ alors $M \models_I F$ ssi pour tout $c \in D$, $M \models_I P[x/c]$*
- *Si $F = \exists x P$ alors $M \models_I F$ ssi il existe $c \in D$ tel que $M \models_I P[x/c]$*

remarque : L'interprétation I ne sert que pour les variables libres des formules, c'est à dire les variables qui ne sont pas sous la portée d'un quantificateur. Définissons précisemment les notions de variables libres et liées :

Définition 2.2.4 *Soit x une variable et F une formule.*

- *Si $F = P(t_1, \dots t_n)$ alors toutes les occurrences de x dans F sont libres.*
- *Si $F = A * B$ (où $*$ représente un connecteur binaire), alors les occurrences libres de x dans F sont les occurrences libres de x dans A et les occurrences libres de x dans B.*

- *Si $F = \neg A$ alors les occurrences libres de x dans F sont les occurrences libres de x dans A.*
- *Si $F = \forall x P$ ou $\exists x P$ alors x n'a aucune occurrence libre dans F.*
- *Si $F = \forall y P$ ou $\exists y P$ avec $y \neq x$, alors les occurrences libres de x dans F sont les occurrences libres de x dans P*

Exemple 2.2.1 $\forall_1 x(A(x,y) \rightarrow ((\forall_2 x B(x)) \vee \exists_3 y C(x,y)))$
Les première et troisième occurrences de x sont liées par le premier quantificateur, la seconde par le second. La première occurrence de y est libre, la seconde liée.

Définition 2.2.5 *Soit x une variable et F une formule.*
Les occurrences liées *de x dans F, sont les occurrences non libres de x dans F.*
Une variable libre *de F est une variable de F dont au moins une occurrence est libre.*
Une variable liée *de F est une variable non libre de F, (ie dont toutes les occurrences sont liées).*
Une formule close *est une formule dont toutes les variables sont liées.*

remarque : Comme le nom d'une variable liée n'a pas d'importance, on peut toujours s'arranger pour qu'une variable n'ait pas à la fois des occurrences libres et liées. Par exemple la formule :
$\forall_1 x(A(x,y) \rightarrow ((\forall_2 x B(x)) \vee \exists_3 y C(x,y)))$ est logiquement équivalente à
$\forall_1 x(A(x,y) \rightarrow ((\forall_2 z B(z)) \vee \exists_3 v C(x,v)))$

Définition 2.2.6 *Soit F une formule, x une variable et t un terme.*
$F[x/t]$ désigne F dans laquelle chaque occurrence libre *de x a été remplacée par t.*

par exemple :si $F = \forall x(A(x,y) \rightarrow ((\forall x B(x)) \vee \exists y C(x,y)))$ alors
$F[x/v] = F$ et
$F[y/v] = \forall x(A(x,v) \rightarrow ((\forall x B(x)) \vee \exists y C(x,y)))$

Définition 2.2.7 *Une réalisation M est un modèle d'une formule F ssi pour toute interprétation I on a : $M \models_I F$ (noté $M \models F$)*

Définition 2.2.8 *Une formule F est* satisfaisable *ssi il existe une réalisation M telle que $M \models F$*

Définition 2.2.9 *Une formule F est* valide *(ou est une* tautologie*) ssi pour toute réalisation M on a : $M \models F$ (noté $\models F$)*

2.3 La notion de Déduction

On va pouvoir rendre compte d'un nouveau type de raisonnement dont l'exemple canonique est le suivant:
Socrate est un homme,
tous les hommes sont mortels
donc socrate est mortel.
En calcul des propositions, chacune des trois lignes précédentes correspondraient à trois propositions, et les règles d'inférence connues ne nous permettent pas d'inférer la dernière des deux premières. Pourtant le raisonnement est correct. Ce sont les règles d'inférence concernant les quantificateurs qui vont nous permettre de déduire "Socrate est mortel" à partir des deux premiers faits. Formalisons cet exemple en calcul des prédicats :
donnons nous tout d'abord deux symboles de prédicat unaire

- "Etre-un-homme" (que nous ecrirons H(-)),
- "Etre-mortel" (que nous ecrirons M(-))

et une constante : socrate.
"Socrate est un homme" correspond à $H(socrate)$ qui est une formule atomique.
"tous les hommes sont mortels" correspond à la formule : $\forall x(H(x) \rightarrow M(x))$
"Socrate est mortel" correspond à $M(socrate)$.
Intuitivement le résultat permettant de déduire $M(socrate)$ à partir de $H(x)$ et $\forall x(H(x) \rightarrow M(x))$ est le suivant :
de $\forall x(H(x) \rightarrow M(x))$ qui vaut pour tout individu, on déduit en particulier : $(H(socrate) \rightarrow M(socrate))$. De ce fait et de l'hypothèse $H(socrate)$ on déduit par modus ponens : $M(socrate)$ Nous avons ici utilisé la règle gauche du $\forall$ ou d'instanciation qui applique un fait qui vaut pour tout individu à un individu particulier. La règle droite du $\exists$ est tout aussi naturelle : Pour montrer que la formule $\exists x P(x)$ est vraie, il suffit d'exhiber un terme t, un *témoin* dont on sait montrer qu'il verifie P.

2.3.1 Calcul des séquents

$$\frac{\Gamma, P[x/t] \vdash \Delta}{\Gamma, \forall x P[x] \vdash \Delta}\ \forall_g \qquad \frac{\Gamma \vdash P[x/y], \Delta}{\Gamma \vdash \forall x P[x], \Delta}\ \forall_d(*)$$

$$\frac{\Gamma, P[x/y] \vdash \Delta}{\Gamma, \exists x P[x] \vdash \Delta}\ \exists_g(*) \qquad \frac{\Gamma \vdash P[x/t], \Delta}{\Gamma \vdash \exists x P[x], \Delta}\ \exists_d$$

(*) : y non libre dans Γ, ni dans Δ

Le $\forall_d$ correspond au raisonnement suivant : Pour montrer $\forall x P[x]$, il suffit de montrer P pour un individu *absolument quelconque*, c'est à dire une variable dont on ne sait rien, c'est à dire qui n'apparait nulle part libre dans le séquent.

De même, Le $\exists_g$ correspond au raisonnement suivant : si on a $\exists x P[x]$ en hypothèse, alors on sait aussi que P vaut d'un individu *absolument quelconque*, c'est à dire une variable non libre dans le séquent. La condition (*) est donc fondamentale à la bonne application de ces règles. Un grand nombre d'erreur de démonstration proviennent de leur non respect.

2.4 Les résultats fondamentaux

Théorème 2.4.1 (Correction) *Soit F une formule alors :*
Si $\vdash F$ alors $\models F$

Théorème 2.4.2 (Complétude) *Soit F une formule alors :*
Si $\models F$ alors $\vdash F$

Théorème 2.4.3 (Indécidabilité) *Le calcul des prédicats est indécidable i.e. il n'existe pas d'algorithme qui s'arrête toujours et qui reussit si et seulement si son entrée est un théorème du calcul des prédicats.*

2.5 Quelques propriétés

$\vdash \forall x(P(x) \wedge Q(x)) \leftrightarrow \forall x P(x) \wedge \forall x Q(x)$
$\vdash \exists x(P(x) \vee Q(x)) \leftrightarrow \exists x P(x) \vee \exists x Q(x)$
$\vdash \forall x(P(x) \rightarrow Q(x)) \rightarrow (\forall x P(x) \rightarrow \forall x Q(x))$
$\vdash \forall x(\exists y Q(x,y) \rightarrow P(x)) \rightarrow \forall x \forall y(Q(x,y) \rightarrow P(x))$
$\vdash \neg \forall x P(x) \leftrightarrow \exists x \neg P(x)$
$\vdash \neg \exists x P(x) \leftrightarrow \forall x \neg P(x)$

2.6 Exercices

Exercice 2.6.1 *Formaliser en Calcul des prédicats les phrases suivantes :*

1. *Les baleines sont des mammifères.*
2. *Les entiers sont pairs ou impairs.*
3. *Il existe un entier pair*

Corrigé :

1. $\forall x(Baleine(x) \rightarrow Mamm(x))$
2. $\forall x(Entier(x) \rightarrow (Pair(x) \vee Impair(x)))$
3. $\exists x(Entier(x) \wedge Pair(x))$

Exercice 2.6.2 *Il s'agit de construire un modèle partiel du fonctionnement d'une banque. Considérons les règles informelles suivantes.:*

- *Une banque gère pour ses clients deux types de comptes : les comptes* courant *et les comptes* épargne*.*
- *Chaque compte appartient à un unique client.*
- *Un client peut posséder plusieurs comptes courants mais un seul compte épargne.*

Formaliser les règles précédentes en Calcul des Prédicats
Il s'agit donc de se donner des symboles de prédicats et d'énoncer les règles au moyen de ceux ci. l'utilisation du connecteur $\exists!$ *est autorisée.*

- $\forall x(C(x) \rightarrow Courant(x) \vee Epargne(x))$
  $\forall x(Courant(x) \rightarrow C(x) \wedge \neg Epargne(x))$
  $\forall x(Epargne(x) \rightarrow C(x) \wedge \neg Courant(x))$

- $\forall x(C(x) \rightarrow \exists! y(Client(y) \wedge possede(y, x)))$
- $\forall x \forall y(Client(x), Epargne(y), possede(x, y) \rightarrow \exists! y(Epargne(y) \wedge possede(x, y)))$

Exercice 2.6.3 *Prouver :* $\forall x P(x) \wedge \forall x Q(x) \vdash \forall x(P(x) \wedge Q(x))$

$$\begin{array}{ll}
\forall x P(x) \wedge \forall x Q(x) \vdash \forall x(P(x) \wedge Q(x)) & \forall_d \\
\forall x P(x) \wedge \forall x Q(x) \vdash P(y) \wedge Q(y) & \wedge_g \\
\forall x P(x), \forall x Q(x) \vdash P(y) \wedge Q(y) & \wedge_d \\
\quad \forall x P(x), \forall x Q(x) \vdash P(y) & \forall_g(x \leftarrow y) \\
\quad P(y), \forall x Q(x) \vdash P(y) & ax
\end{array}$$

$\forall x P(x), Q(y) \vdash Q(y)$ $\forall_g$
$\forall x P(x), Q(y) \vdash Q(y)$ ax

Exercice 2.6.4 *Prouver :* $\exists x P(x) \vee \exists x Q(x) \vdash \exists x (P(x) \vee Q(x))$

$\exists x P(x) \vee \exists x Q(x) \vdash \exists x (P(x) \vee Q(x))$ $\vee_g$
$\exists x P(x) \vdash \exists x (P(x) \vee Q(x))$ $\exists_g$
$P(y) \vdash \exists x (P(x) \vee Q(x))$ $\exists_d$
$P(y) \vdash P(y) \vee Q(y)$ $\vee_d$
$P(y) \vdash P(y), Q(y)$ ax

$\exists x Q(x) \vdash \exists x (P(x) \vee Q(x))$ $\exists_g$
$Q(y) \vdash \exists x (P(x) \vee Q(x))$ $\exists_d$
$Q(y) \vdash P(y) \vee Q(y)$ $\vee_d$
$Q(y) \vdash P(y), Q(y)$ ax

Exercice 2.6.5 *Prouver :* $\forall x (P(x) \to Q(x)) \vdash \forall x P(x) \to \forall x Q(x)$

$\forall x (P(x) \to Q(x)) \vdash \forall x P(x) \to \forall x Q(x)$ $\to_d$
$\forall x (P(x) \to Q(x)), \forall x P(x) \vdash \forall x Q(x)$ $\forall_d$
$\forall x (P(x) \to Q(x)), \forall x P(x) \vdash Q(y)$ $\forall_g$
$\forall x (P(x) \to Q(x)), P(y) \vdash Q(y)$ $\forall_g$
$P(y) \to Q(y), P(y) \vdash Q(y)$ $\to_g$
$Q(y), P(y) \vdash Q(y)$ ax

$P(y) \vdash P(y), Q(y)$ ax

Exercice 2.6.6 *Prouver :* $\forall x (\exists y Q(x,y) \to P(x)) \vdash \forall x \forall y (Q(x,y) \to P(x))$

$\forall x (\exists y Q(x,y) \to P(x)) \vdash \forall x \forall y (Q(x,y) \to P(x))$ $\forall_d(2)$
$\forall x (\exists y Q(x,y) \to P(x)) \vdash Q(x,y) \to P(x)$ $\forall_g$
$\exists y Q(x,y) \to P(x) \vdash Q(x,y) \to P(x)$ $\to_d$
$\exists y Q(x,y) \to P(x), Q(x,y) \vdash P(x)$ $\to_g$
$Q(x,y) \vdash \exists y Q(x,y)$ $\exists_d$
$Q(x,y) \vdash Q(x,y)$ ax

$P(x), Q(x,y) \vdash P(x)$ ax

Chapter 3

La théorie de l'égalité

3.1 Qu'est ce qu'une théorie ?

Les formules valides ou démontrables en calcul des prédicats sont les pures vérités logiques, les énoncés vrais dans tous les mondes possibles. Si l'on veut utiliser la logique pour modéliser des domaines d'application, la programmation par exemple, cela est insuffisant. On aura par exemple besoin de parler des types de données comme les entiers, les tableaux ..., d'énoncer des faits qui sont relatifs à ces données.
La notion de *théorie* permet cela. Prenons un exemple: imaginons que nous ayons besoin d'un prédicat particulier P qui à la propriété d'être une relation transitive et anti reflexive. La théorie permettant de définir ce prédicat sera constituée des formules : $\forall x \forall y \forall z (P(x,y) \wedge P(y,z) \rightarrow P(x,z))$ et $\forall x \neg P(x,x)$.
Nous pouvons maintenant nous intéresser non pas à tous les mondes possibles, mais uniquement aux mondes dans lequel ces deux formules sont vraies, c'est à dire les univers qui admettent une relation d'ordre stricte.
D'un point de vue sémantique, cela signifie que l'on va s'intéresser à tous les *modèles de cette théorie*. On aura par exemple que tous ces modèles vérifient que P est anti symétrique.
D'un point de vue syntaxique, on s'intéressera aux formules *démontrables dans cette théorie* c'est à dire démontrable en Calcul des Prédicats dans lequel on suppose vraies ces deux formules. Ces formules ne sont pas des théorèmes, car elles ne sont pas démontrées, ce sont des *axiomes*. Mise a part cette différence fondamentale, elles s'utilisent dans les preuves exactement de la même façon, au moyen de la règle *cut*.

Définition 3.1.1 *Une* théorie *est un ensemble T de formules closes.*
Une réalisation $\mathcal{M}$ est un modèle d'une théorie *T ssi $\mathcal{M} \models \mathcal{F}$ pour chaque formule F de T.*
Une formule ϕ est une conséquence *de T ssi tous modèles de T est un modèle*

de ϕ (noté $T \models \phi$).
Une formule ϕ est démontrable dans une théorie T ssi elle est démontrable au moyen des règles d'inférence du Calcul des Prédicats plus la possibilité de faire une coupure sur les axiomes de la théorie.

3.2 les axiomes de la théorie de l'égalité

Nous allons exprimer les propriétés d'un prédicat particulier : l'égalité (=). L'égalité est un prédicat qui est reflexif, symétrique, transitif, et qui a la propriété de substitution des égaux par les égaux dans n'importe quel prédicat. Cette dernière propriété est appelée *loi de Leibniz*. En fait seules la reflexivité et la loi de Leibniz suffisent à définir l'égalité. La symétrie et la transitivité seront démontrables à partir de ces deux axiomes.
La théorie de l'égalité est donc constituée des deux axiomes suivants:

1. (refl :) $\forall e(e = e)$
2. (leibniz:) Pour tout prédicat P , $\forall x \forall y(x = y \wedge P(x) \rightarrow P(y))$

L'égalité est un prédicat indispensable à toutes modélisations. C'est pourquoi la majeure partie des théories contiennent les axiomes de l'égalité. C'est le cas par exemple de la théorie des ensembles. Nous parlerons alors de théories du Calcul des Prédicats *avec égalité*.

3.3 symétrie de l'égalité

La loi de Leibniz est un schéma d'axiomes, c'est à dire qu'il représente une infinité d'axiomes : un pour chacune des formules du langage. On va utiliser son instance suivante :$(L1) \forall x \forall y(x = y \wedge x = x \rightarrow y = x)$. $P(y)$ a pris comme valeur $y = x$ et $P(x)$ la valeur $x = x$.

$$\vdash \forall x \forall y(x = y \rightarrow y = x) \quad \forall_d$$
$$\vdash x = y \rightarrow y = x \quad \rightarrow_d$$
$$x = y \vdash y = x \quad cut(L1)$$
$$x = y, \forall x \forall y((x = y \wedge x = x) \rightarrow y = x) \vdash y = x \quad \forall_g$$
$$x = y, (x = y \wedge x = x) \rightarrow y = x \vdash y = x \quad \rightarrow_g$$
$$x = y \vdash (x = y \wedge x = x), y = x \quad \wedge_d$$
$$x = y \vdash x = y, y = x \quad ax$$
$$x = y \vdash x = x, y = x \quad refl$$
$$x = y, y = x \vdash y = x \quad ax$$

3.4 Deux règles dérivées pour l'egalité

3.4.1 une règle terminale : $refl$

$$\frac{}{\Gamma \vdash e = e, \Delta}\ refl$$

3.4.2 utilisation d'une égalité

L'axiome *leibniz* permet de substituer des égaux par égaux. Autrement dit, l'égalité est aux termes ce que l'équivalence est aux formules.
Donnons nous une règle pour l'égalité similaire à la règle pour l'équivalence.

$$\frac{(\Gamma \vdash \Delta)[t//u]}{(\Gamma \vdash \Delta)[t]} = (H)*$$

(*) :si H est de la forme $t = u$ et est un théorème ou une hypothèse.

Pour simplifier, nous allons montrer la correction de cette règle dans le cas ou on ne réecrit qu'une seule formule $P(t)$ et en supposant que $t = u$ est en hypothèse. Si la formule est à gauche cette règle correspond à la dérivation suivante :

$\Gamma, P(t), t = u \vdash \Delta$ $cut(leibniz)$
$\Gamma, P(t), t = u, \forall x \forall y (x = y \wedge P(x) \rightarrow P(y)) \vdash \Delta$ $\forall_g$
$\Gamma, P(t), t = u, (t = u \wedge P(t) \rightarrow P(u)) \vdash \Delta$ $\rightarrow_g$
$\Gamma, P(t), t = u \vdash t = u \wedge P(t), \Delta$ $\wedge_d$
$\Gamma, P(t), t = u \vdash t = u, \Delta$ ax

$\Gamma, P(t), t = u \vdash P(t), \Delta$ ax

$\Gamma, P(t), P(u) \vdash \Delta$ aff
$\Gamma, P(u) \vdash \Delta$ premisse de la regle

Si la formule est à droite :
$\Gamma, t = u \vdash P(t), \Delta$ $cut(leibniz)$
$\Gamma, t = u, \forall x \forall y ((x = y \wedge P(x)) \rightarrow P(y)) \vdash P(t), \Delta$ $\forall_g$
$\Gamma, t = u, (u = t \wedge P(u)) \rightarrow P(t) \vdash P(t), \Delta$ $\rightarrow_g$
$\Gamma, P(t) \vdash P(t), \Delta$ ax

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, t = u \vdash t = u}\, ax}{\Gamma, t = u \vdash u = t}\, cut(sym)}{\Gamma, \forall x \forall y(x = y \to y = x), t = u \vdash u = t}\, \forall_g, \to_g}{\Gamma, t = u \vdash u = t \wedge P(u)}\, \wedge_d$$

premisse de la règle : $\Gamma, t = u \vdash P(u), \Delta$

3.5 transitivité de l'égalité

$$\dfrac{\dfrac{\dfrac{\dfrac{}{a = b, b = c \vdash b = c}\, ax}{a = b, b = c \vdash a = c}\, = (H1)}{a = b \wedge b = c \vdash a = c}\, \wedge_g}{\vdash (a = b \wedge b = c) \to a = c}\, \to_d$$

$$\vdash \forall a \forall b \forall c(a = b \wedge b = c \to a = c)\ \forall_d(3*)$$

3.6 Quelques propriétés

symétrie	$\vdash \forall x \forall y(x = y \to y = x)$
transitivité	$\vdash \forall x \forall y \forall z(x = y \wedge y = z \to x = z)$
Un point$_\forall$	$\vdash \forall e(\forall x(x = e \to P(x))) \leftrightarrow P(e))$
Un point$_\exists$	$\vdash \forall e(\exists x(x = e \wedge P(x))) \leftrightarrow P(e))$

3.7 Exercices

Exercice 3.7.1 *On veut modéliser le fonctionnement d'une bibliothèque : On observe les règles suivantes :*

1. *Un exemplaire est toujours associé à un livre. Celui ci est unique.*
2. *Un même exemplaire de livre ne peut etre emprunté par différents abonnés.*
3. *Un même abonné ne peut emprunter plus d'un exemplaire d'un même livre*

formaliser les règles en Calcul des prédicats.

On se donne :
$Ex(\_)$*(etre un exemplaire),*
$L(\_)$ *(etre un livre)*
$A(\_)$ *(etre un abonné)*
$Emp(a, e)$ *(a emprunte l'exemplaire e) et*
$Exde(e, l)$ *(e est un exemplaire du livre l)*

1. $\forall e(Ex(e) \rightarrow \exists! y(L(y) \wedge Exde(e, l)))$
2. $\forall e, aa, ab(Ex(e), A(aa), A(ab), Emp(aa, e) \wedge Emp(ab, e) \rightarrow aa = ab)$
3. *Si deux exemplaires sont empruntés par un même abonnés, ils concernent des livres différents :*
   $\forall ea, eb((Ex(ea) \wedge Ex(eb) \wedge ea \neq eb \wedge \exists a(A(a) \wedge Emp(a, ea) \wedge Emp(a, eb))) \rightarrow$
   $\forall la, lb(Exde(ea, la) \wedge Exde(eb, lb) \wedge L(la) \wedge L(lb) \rightarrow la \neq lb))$

Exercice 3.7.2 *Prouver* $\vdash \forall e(\forall x(x = e \rightarrow P(x)) \rightarrow P(e))$

$$\begin{array}{ll}
\vdash \forall e(\forall x(x = e \rightarrow P(x)) \rightarrow P(e)) & \forall_d \\
\vdash \forall x(x = e \rightarrow P(x)) \rightarrow P(e) & \rightarrow_d \\
\forall x(x = e \rightarrow P(x)) \vdash P(e) & \forall_g \\
e = e \rightarrow P(e) \vdash P(e) & \rightarrow_g \\
\quad \vdash e = e & refl \\
\\
\quad P(e) \vdash P(e) & ax
\end{array}$$

Exercice 3.7.3 *Prouver* $\vdash \forall e(P(e) \rightarrow (\forall x(x = e \rightarrow P(x))))$

$$\begin{array}{ll}
\vdash \forall e(P(e) \rightarrow (\forall x(x = e \rightarrow P(x)))) & \forall_d \\
\vdash P(e) \rightarrow (\forall x(x = e \rightarrow P(x))) & \rightarrow_d \\
P(e) \vdash \forall x(x = e \rightarrow P(x)) & \forall_d \\
P(e) \vdash x = e \rightarrow P(x) & \rightarrow_d \\
P(e), x = e \vdash P(x) & = (x = e)
\end{array}$$

$P(e), x = e \vdash P(e)$ ax

Exercice 3.7.4 *Prouver* $\vdash \forall e(\exists x(x = e \wedge P(x)) \rightarrow P(e))$

$\vdash \forall e(\exists x(x = e \wedge P(x)) \rightarrow P(e))$ $\forall_d$
$\vdash \exists x(x = e \wedge P(x)) \rightarrow P(e)$ $\rightarrow_d$
$\exists x(x = e \wedge P(x)) \vdash P(e)$ $\exists_g$
$x_0 = e \wedge P(x_0) \vdash P(e)$ $\wedge_g$
$x_0 = e, P(x_0) \vdash P(e)$ $= (x_0 = e)$
$x_0 = e, P(e) \vdash P(e)$ ax

Exercice 3.7.5 *Prouver* $\vdash \forall e(P(e) \rightarrow (\exists x(x = e \wedge P(x))))$

$\vdash \forall e(P(e) \rightarrow (\exists x(x = e \wedge P(x))))$ $\forall_d$
$\vdash P(e) \rightarrow (\exists x(x = e \wedge P(x)))$ $\rightarrow_d$
$P(e) \vdash \exists x(x = e \wedge P(x))$ $\exists_d$
$P(e) \vdash e = e \wedge P(e)$ $\wedge_d$
$P(e) \vdash e = e$ $refl$

$P(e) \vdash P(e)$ ax

Chapter 4

quelques tactiques de preuve

Avec les moyens que nous nous sommes donnés, l' écriture et la lecture des preuves formelles à la main est une tache presqu'impossible, tant le niveau de détail demandé est élevé.
La possibilité de définir des règles dérivées a certes permis de raccourcir un peu les preuves, mais ceci reste insuffisant.
Pour poursuivre cet effort, nous allons maintenant définir des petites stratégies de preuves, que nous appellerons des *tactiques*. Une tactique ne s'exprime pas nécessairement sous la forme -statique- d'une règle dérivée, car son comportement est le plus souvent dynamique, épousant la forme des formules.
Les tactiques seront des algorithmes pour engendrer des bouts de preuves. Elles prennent donc en entrée un séquent et des informations propres et produisent une dérivation.
Les tactiques produisent de façon déterministe une dérivation du sequent sur lesquelles on les applique. Une fois leurs corrections dmontres, nous pouvons donc les considérer comme des règles dérivées, dont la conclusion est le séquent de départ, et les prémisses les sous buts non fermés de la dérivation produite.

Notre but n'est pas dans ce chapitre de définir des stratégies élaborées et puissantes de preuve, mais au contraire de définir des stratégies de preuve permettant de faire autant que possible le travail non inventif et trivial.

4.1 la tactique de simplification : *simpl*

Qu'est ce qui est trivial dans une preuve ? Autrement dit, à quel moment appliquons nous des règles sans réflechir ? La première chose qui vient à l'esprit, est ce que nous faisons au début d'une preuve : nous simplifions notre séquent en appliquant à droite les $\forall_d$, $\rightarrow_d$, $\vee_d$, $\neg_d$ et à gauche les $\exists_g$, $\wedge_g$, $\neg_g$.

Nous n'appliquons les règles $\vee_g$, $\rightarrow_g$ que le plus tard possible dans la preuve car elles produisent plusieurs sous buts. Nous ne les appliquons donc que sur les hypothèses dont nous sommes surs qu'elles vont nous servir.
Pour les mêmes raisons, nous n'appliquons $\wedge_d$ que lorsque nous sommes surs d'avoir produit les hypothèses nécessaires communes aux preuves des deux parties de la disjonction.
L'application des règles $\forall_g$, $\exists_d$, *cut* n'est jamais triviale puisque en elles réside la partie inventive des preuves. Le raisonnement équationnel et par équivalence n'est pas non plus un travail trivial car leur application en aveugle produit facilement des boucles infinies.

La tactique *simpl* va prendre en charge ce premier niveau de travail dans une preuve.

Soient $\Gamma \vdash \Delta$ un séquent. L'application de $Simpl$ sur $\Gamma \vdash \Delta$ consiste à choisir l'une des formules F du séquent et à appliquer l'algorithme suivant :

- Si $F \in \Delta$:
  - Si F est aussi dans Γ, on applique la règle axiome.
  - Si $\neg F$ est aussi dans Δ, on applique les règles $\neg_d$ et axiome.
  - Si F est de la forme $A \leftrightarrow A$ ou $a = a$ on applique respectivement les règles $refl equiv$ et $refl$.
  - Si F est de la forme $A \vee B$, $A \rightarrow B$, $\neg A$, $\forall x A$,
    appliquer la règle droite correspondant au connecteur ou quantificateur principal de F , puis
    appliquer *simpl* sur chacun des sequents produit par l'application de la règle.
  - Sinon : choisir une autre formule du séquent et appliquer le même algorithme.
- Si $F \in \Gamma$:
  - Si F est aussi dans Δ, on applique la règle axiome.
  - Si $\neg F$ est aussi dans Γ, on applique la règle contr.
  - Si F est de la forme $A \wedge B$, $\neg A$, $\exists x A$,
    appliquer la règle gauche correspondant au connecteur ou quantificateur principal de F , puis
    appliquer $Simpl$ sur chacun des sequents produit par l'application de la règle.
  - Sinon : choisir une autre formule du séquent et appliquer le même algorithme.

Simpl s'arrête lorsque qu'il n'y a plus aucune formule du séquent sur laquelle elle produit un effet.

Remarque : Simpl termine car chaque appel récursif de *simpl* se fait sur un séquent dont la somme des tailles de ses formules est plus petite que la somme des tailles des formules du séquent de départ ou dont la somme des tailles des formules est égale à celles du séquent de départ mais dont le nombre de formules à simplifier non analysées est plus petit.
Simpl est correcte : à chaque fois que l'algorithme applique une règle, elle est applicable.

Exemple 4.1.1 *L'application de la tactique simpl sur* $\vdash \neg(A \vee B) \rightarrow \neg A \wedge \neg B$ *produit :*

$\vdash \neg(A \vee B) \rightarrow \neg A \wedge \neg B$ $\rightarrow_d$
$\neg(A \vee B) \vdash \neg A \wedge \neg B$ $\neg_g$
$\vdash A \vee B, \neg A \wedge \neg B$ $\vee_d$
$\vdash A, B, \neg A \wedge \neg B$

Une preuve du théorème précédent pourra donc s'ecrire :

$\vdash \neg(A \vee B) \rightarrow \neg A \wedge \neg B$ *Simpl*
$\vdash A, B, \neg A \wedge \neg B$ $\wedge_d$
$\vdash A, B, \neg A$ *simpl*
Ok

$\vdash A, B, \neg B$ *simpl*
Ok

Exemple 4.1.2 *L'application de simpl sur* $\vdash (\forall x P(x) \wedge \exists x Q(x)) \rightarrow \exists P(x)$ *produit :*
$\vdash (\forall x P(x) \wedge \exists x Q(x)) \rightarrow \exists x P(x)$ $\rightarrow_d$
$\forall x P(x) \wedge \exists x Q(x) \vdash \exists x P(x)$ $\wedge_g$
$\forall x P(x), \exists x Q(x) \vdash \exists x P(x)$ $\exists_g$
$\forall x P(x), Q(x_0) \vdash \exists x P(x)$

Une preuve de ce théorème pourra donc s'ecrire :

$\vdash (\forall x P(x) \wedge \exists x Q(x)) \rightarrow \exists P(x)$ *simpl*
$\forall x P(x), Q(x_0) \vdash \exists x P(x)$ $\forall_g(x \leftarrow x_0)$
$P(x_0), Q(x_0) \vdash \exists x P(x)$ $\exists_d(x \leftarrow x_0)$
$P(x_0), Q(x_0) \vdash P(x_0)$ *ax*

Exemple 4.1.3 *l'application de simpl sur* $\vdash \forall e(\exists x(x = e \wedge P(x)) \rightarrow P(e))$ *produit :*

$\vdash \forall e(\exists x(x = e \wedge P(x)) \rightarrow P(e))$ $\forall_d$
$\vdash \exists x(x = e \wedge P(x)) \rightarrow P(e)$ $\rightarrow_d$

$\exists x(x = e \wedge P(x)) \vdash P(e)$ $\exists_g$
$x_0 = e \wedge P(x_0) \vdash P(e)$ $\wedge_g$
$x_0 = e, P(x_0) \vdash P(e)$

Une preuve de ce théorème pourra donc s'écrire :

$\vdash \forall e(\exists x(x = e \wedge P(x)) \rightarrow P(e))$ *simpl*
$x_0 = e, P(x_0) \vdash P(e)$ $= (x_0 = e)$
$x_0 = e, P(e) \vdash P(e)$ *ax*

4.2 la tactique d'utilisation des théorèmes et hypothèses : $use(H)$

Cette tactique a pour but de prendre en charge la partie facile du travail lorsqu'on cherche à utiliser une hypothèse ou un théorème déjà prouvé. Il ne s'agit plus, comme pour *simpl*, de faire un travail à l'aveugle sur toutes les formules du séquent, mais de faire de la déduction sur une formule *choisie* par la personne qui fait la preuve. Dans un certains nombre de cas, la seule partie intelligente du travail est précisemment ce choix.
C'est le cas par exemple lorsque la formule est une formule du Calcul des Propositions. Utiliser une telle formule c'est, si c'est une hypothèse, appliquer toutes les règles correspondant à ses connecteurs, en consommant graduellement l'arbre de syntaxe abstraite, tant qu'aucune règle terminale n'est applicable. Autrement dit, contrairement à *simpl*, on n'interdira plus les règles $\rightarrow_g$, $\wedge_d$ et $\vee_g$. Utiliser par exemple une hypothèse de la forme $A \vee B$ consiste bien à faire un raisonnement par cas puis à déduire dans chacune des branches tout ce que l'on peut de la partie de la disjonction que l'on a en hypothèse. Si la formule (propositionnelle) à utiliser est un théorème, il s'agit de faire une coupure sur cette formule, puis de procéder de la même façon.
De même, utiliser une hypothèse ou un théorème de la forme $a = b$ ou $A \leftrightarrow B$ consiste a remplacer une ou plusieurs occurrences de l'une des parties par l'autre (règles $\leftrightarrow$ et $=$).
Lorsque le théorème ou l'hypothèse est de la forme :$\forall x P(x)$, la partie intelligente du travail ne réside pas seulement dans le choix du théorème. Elle réside aussi dans le choix des *instances de ce théorème*. Nous laissons donc ce choix à la personne qui ecrit la preuve. Autrement dit, $use(\forall x F)$ sera sans aucun effet. En revanche $use(G)$ si G est une instance d'un théorème universel sera permis. L'argument de *use* doit être :

- une hypothèse du séquent ou un théorème ou alors
- une instance d'une hypothèse universelle ou d'un théorème universel

Une formule universelle est une formule de la forme $\forall x_1 \ldots \forall x_n A(x_1, \ldots, x_n)$. Une instance de cette formule est de la forme $A[t_1/x_1, \ldots, t_n/x_n]$. Pour ne pas trop alourdir la syntaxe de use, nous noterons les arguments de *use* qui sont des instances de la façon suivante : $th(t1, \ldots, t_n)$.
th est le nom du theorème de la forme : $\forall x_1 \ldots \forall x_n A(x_1, \ldots, x_n)$. On voit en quelques sortes les variables $x_1, \ldots x_n$ comme les paramètres formels du théorème, et l'instanciation comme l'application du théorème à des valeurs.

Exemple 4.2.1 *Soit* $trans \equiv \forall x \forall y \forall z((P(x,y) \wedge P(y,z)) \rightarrow P(x,z))$
$trans(f(a), a, b))$ *désigne la formule* $(P(f(a), a) \wedge P(a, b)) \rightarrow P(f(a), b)$

Soient $\Gamma \vdash \Delta$ un séquent et F une formule de Γ ou bien un théorème déjà prouvé.
L'application de $use(F)$ sur $\Gamma \vdash \Delta$ est définie de la façon suivante :

- Si F est une hypothèse :

  1. Si F est aussi dans Δ, on applique la règle axiome.
  2. Si $\neg F$ est aussi dans Γ, on applique la règle contr.
  3. Si F est de la forme $A \wedge B$, l'application de use(F) produit la dérivation suivante :
     $\Gamma \vdash \Delta$ $\quad \wedge_g$
     $\Gamma, A, B \vdash \Delta$ $\quad use1([A, B])$ [1]
     $\vdots$

  4. Si F est de la forme $A \vee B$, l'application de use(F) produit la dérivation suivante :
     $\Gamma \vdash \Delta$ $\quad \vee_g$
     $\Gamma, A \vdash \Delta$ $\quad use(A)$
     $\vdots$
     $\Gamma, B \vdash \Delta$ $\quad use(B)$
     $\vdots$

  5. Si F est de la forme $\neg A$, l'application de use(F) produit la dérivation suivante :
     $\Gamma \vdash \Delta$ $\quad \vee_g$
     $\Gamma \vdash A, \Delta$ $\quad simpl$
     $\vdots$

[1] use1([F1,..Fn]) applique use(F1) puis applique use1([F2,..Fn])sur chacun des sous buts non prouvés de la dérivation obtenue

6. Si F est de la forme $A \leftrightarrow B$, l'application de use(F) produit :
   $\Gamma \vdash \Delta$ $\leftrightarrow (F)$
   $(\Gamma \vdash \Delta)[A//B]$ $simpl$
   $\vdots$

7. Si F est de la forme $a = b$, l'application de use(F) produit :
   $\Gamma \vdash \Delta$ $= (F)$
   $(\Gamma \vdash \Delta)[a//b]$ $simpl$
   $\vdots$

8. Si F est de la forme $A \rightarrow B$, l'application de use(F) produit la dérivation suivante :
   $\Gamma \vdash \Delta$ $\rightarrow_g$
   $\Gamma \vdash A, \Delta$ $simpl$
   $\vdots$
   $\Gamma, B \vdash \Delta$ $use(B)$
   $\vdots$

9. si F est de la forme $\exists x A$, l'application de use(F) produit la dérivation suivante (où x_0 est non libre dans le séquent):
   $\Gamma \vdash \Delta$ $\exists_g$
   $\Gamma, A[x/x_0] \vdash \Delta$ $use(A[x/x_0])$
   $\vdots$

10. si F est de la forme $H(t_1, \ldots, t_n)$ où H est le nom d'une hypothèse de la forme $\forall x_1, \ldots, \forall x_n A$, alors l'application de use(F) produit la dérivation suivante :
    $\Gamma \vdash \Delta$ $\forall_g$
    $\Gamma, A[x_1/t_1, \ldots, x_n/t_n] \vdash \Delta$ $use(A[x_1/t_1, \ldots, x_n/t_n])$
    $\vdots$

11. Dans tous les autres cas : ne rien faire.

- Si F est un théorème : l'application de $use(F)$ produit la dérivation :
  $\Gamma \vdash \Delta$ $cut(F)$
  $\Gamma, F \vdash \Delta$ $use(F)$
  $\vdots$

Exemple 4.2.2 *Voici une preuve qui utilise nos tactiques :*

$\vdash \forall x(P(x) \to Q(x)) \to (\forall x P(x) \to \forall x Q(x))$ *simpl*

$\forall x(P(x) \to Q(x)), \forall x P(x) \vdash Q(y)$ $use(H2(y))$

$\forall x(P(x) \to Q(x)), P(y) \vdash Q(y)$ $use(H1(y))$

Ok

En effet: simpl applique les règles $\to_d$, $\to_d$ *et* $\forall_d$.

$use(H2(y))$ *ne fait que le* $\forall_g$ *sur* $H2$ *en donnant à* x *la valeur* y. *l'appel récursif* $use(P(y))$ *ne produit rien car aucun cas ne s'applique.*

$use(H1(y))$ *fait le* $\forall_g$ *sur* $H1$ *en donnant à* x *la valeur* y. *l'appel récursif* $use(P(y) \to Q(y))$ *fait le* $\to_g$ *et clot par axiomes chacunes des deux branches résultantes (appels de simpl et de* $use(Q(y))$*).*

Plus précisemment, $use(H1(y))$ *correspond à la dérivation suivante :*

$\forall x(P(x) \to Q(x)), P(y) \vdash Q(y)$ $\forall_g$

$P(y) \to Q(y), P(y) \vdash Q(y)$ $\to_g$

$Q(y), P(y) \vdash Q(y)$ ax

$P(y) \vdash P(y), Q(y)$ ax

Exercice 4.2.1 *Prouver* $\vdash \forall e(\forall x(x = e \to P(x)) \to P(e))$ *en utilisant les tactiques.*

$\vdash \forall e(\forall x(x = e \to P(x)) \to P(e))$ *simpl*

$\forall x(x = e \to P(x)) \vdash P(e)$ $use(H1(e))$

OK

Exercice 4.2.2 *Prouver* $\vdash \forall e(P(e) \to (\forall x(x = e \to P(x))))$ *en utilisant les tactiques.*

$\vdash \forall e(P(e) \to (\forall x(x = e \to P(x))))$ *simpl*

$P(e), x = e \vdash P(x)$ $use(H2)$

ok

Exercice 4.2.3 *Prouver* $\vdash \forall e(\exists x(x = e \wedge P(x)) \to P(e))$ *en utilisant les tactiques.*

$\vdash \forall e(\exists x(x = e \wedge P(x)) \to P(e))$ *simpl*

$x_0 = e, P(x_0) \vdash P(e)$ $use(H1)$

Ok

Exercice 4.2.4 *Prouver* $\vdash \forall e(P(e) \to (\exists x(x = e \wedge P(x)))$ *en utilisant les tactiques.*

$\vdash \forall e(P(e) \to (\exists x(x = e \wedge P(x))))$ *simpl*

$P(e) \vdash \exists x(x = e \wedge P(x))$ $\exists_d$

$P(e) \vdash e = e \wedge P(e)$ $\wedge_d$

$P(e) \vdash e = e$ $refl$

$P(e) \vdash P(e)$ ax

Cet exercice nous permet de voir que nos tactiques ne font pas tout ...

Chapter 5

Théories des Entiers Naturels et des listes

5.1 Les Entiers : les axiomes de Péano

Cette théorie axiomatise l'ensemble des entiers naturels.

5.1.1 le langage

Le langage de cette théorie est constitué du langage du Calcul des prédicats, auquel on ajoute :

- un symbole de prédicat unaire : $\mathcal{N}$ (être un entier)
- une constante 0
- un symbole de fonction unaire :s

5.1.2 les axiomes

Notation
$A_1, \ldots A_n \rightarrow B$ désignera dans la suite de ce cours la formule :
$A_1 \rightarrow (\ldots \rightarrow (A_n \rightarrow B) \ldots)$.
On notera par ailleurs que cette formule est équivalente à $(A_1 \wedge \ldots \wedge A_n) \rightarrow B$
Par exemple $F, G \rightarrow H$ désigne $F \rightarrow (G \rightarrow H)$.

Les axiomes sont les suivants :

1. 0 est un entier : $(pea_1) : N(0)$
2. Si x est un entier, son successeur aussi : $(pea_2) : \forall x(N(x) \rightarrow N(s(x)))$

3. 0 n'est le successeur de personne : (pea_3):
   $\forall x(N(x) \rightarrow \neg(s(x) = 0))$

4. 2 entiers ayant même successeur sont égaux : (pea_4) :
   $\forall x \forall y(N(x), N(y), s(x) = s(y) \rightarrow x = y)$

5. Le schéma de récurrence : (pea_5) : pour tout formule F :
   $(F(0) \wedge \forall y((N(y), F(y)) \rightarrow F(s(y)))) \rightarrow \forall x(N(x) \rightarrow F(x))$

5.1.3 Une premiere preuve

$(0ous) : \forall x(N(x) \rightarrow x = 0 \vee \exists n(N(n) \wedge x = s(n)))$

$\vdash \forall x(N(x) \rightarrow x = 0 \vee \exists n(N(n) \wedge x = s(n)))$ $use(pea_5)$
$\vdash (0 = 0 \vee \exists n(N(n) \wedge 0 = s(n))) \wedge \forall y(\mathcal{N}(y), y = 0 \vee \exists n(N(n) \wedge y = s(n)) \rightarrow$
$(s(y) = 0 \vee \exists n(N(n) \wedge s(y) = s(n))))$ $\wedge_d$
  $\vdash 0 = 0 \vee \exists n(N(n) \wedge 0 = s(n))$ simpl
  Ok

$\vdash \forall y((\mathcal{N}(y) \wedge y = 0 \vee \exists n(N(n) \wedge y = s(n))) \rightarrow (s(y) = 0 \vee \exists n(N(n) \wedge s(y) =$
$s(n)))$ simpl
$\mathcal{N}(y), y = 0 \vee \exists n(N(n) \wedge y = s(n)) \vdash s(y) = 0, \exists n(N(n) \wedge s(y) = s(n)))$
use(H2)
  $\mathcal{N}(y), y = 0 \vdash s(y) = 0, \exists n(N(n) \wedge s(y) = s(n)))$ $\exists_d(n == 0)$
  $\mathcal{N}(y), y = 0 \vdash s(y) = 0, N(0) \wedge s(y) = s(0)$ $\wedge_d$
    $\mathcal{N}(y), y = 0 \vdash s(y) = 0, N(0)$ $use(pea_1)$
    Ok

    $\mathcal{N}(y), y = 0 \vdash s(y) = 0, s(y) = s(0)$ use(H2)
    Ok

  $\mathcal{N}(y), N(a), y = s(a)) \vdash s(y) = 0, \exists n(N(n) \wedge s(y) = s(n)))$ $\exists_d(n == s(a))$
  $\mathcal{N}(y), N(a), y = s(a) \vdash s(y) = 0, N(s(a)) \wedge s(y) = s(s(a))$ $\wedge_d$
    $\mathcal{N}(y), N(a), y = s(a) \vdash s(y) = 0, N(s(a))$ $use(pea_2(a))$
    Ok

    $\mathcal{N}(y), N(a), y = s(a) \vdash s(y) = 0, s(y) = s(s(a))$ $use(H3)$
    Ok

5.1.4 Une règle dérivée : *Rec*

La correction de cette règle se déduit directement de pea_5.

$$\frac{\Gamma \vdash F[x/0], \Delta \qquad\qquad \Gamma, N(y), F(x/y) \vdash F[x/s(y)], \Delta}{\Gamma, N(x) \vdash F(x), \Delta}\ Rec(x)*$$

(*) : y nouvelle variable

5.1.5 Définir des fonctions et des Prédicats

On peut définir des fonctions et des prédicats en étendant le language et en ajoutant des axiomes. Lorsque ces objets sont définis récursivement, il faut bien sur s'assurer que ces définitions sont correctes. Pour les fonctions cela revient à s'assurer que ce sont bien des fonctions (image unique) qui terminent.

Pour définir l'addition et la multiplication, on ajoute au language les symboles de fonctions $+$ et $*$ et les axiomes suivants :
$(+_0) : \forall y(N(y) \rightarrow 0 + y = y)$
$(+_1) : \forall x \forall y(N(x), N(y) \rightarrow s(x) + y = s(x + y))$
$(*_0) : \forall y(N(y) \rightarrow 0 * y = 0)$
$(*_1) : \forall x \forall y(N(x), N(y) \rightarrow s(x) * y = (x * y) + y)$

Pour définir le prédicat $\leq$ on ajoute par exemple l'axiome :
$(inf =) : \forall a \forall b(N(a), N(b) \rightarrow (a \leq b \leftrightarrow \exists n(b = n + a)))$

5.1.6 quelques propriétés

$+$ et $*$ sont des fonctions des entiers dans les entiers

$(tot+) : \forall x \forall y(N(x), N(y) \rightarrow N(x + y))$
$(tot*) : \forall x \forall y(N(x), N(y) \rightarrow N(x * y))$

associativité et commutativité de $+$ et $*$

$(assoc_+) :\vdash \forall m \forall n \forall p(N(m), \mathcal{N}(n), \mathcal{N}(p) \rightarrow m + (n + p) = (m + n) + p)$
$(commut_+) : \forall x \forall y(N(x), N(y) \rightarrow x + y = y + x)$
$(assoc_*) :\vdash \forall m \forall n \forall p(N(m), \mathcal{N}(n), \mathcal{N}(p) \rightarrow m * (n * p) = (m * n) * p)$
$(commut_*) : \forall x \forall y(N(x), N(y) \rightarrow x * y = y * x)$

Voici la preuve de $(assoc_+)$

$\vdash \forall m \forall n \forall p(N(m), \mathcal{N}(n), \mathcal{N}(p) \rightarrow m + (n + p) = (m + n) + p)$ *simpl*
$\mathcal{N}(m), \mathcal{N}(n), \mathcal{N}(p) \vdash m + (n + p) = (m + n) + p$ *Rec(m)*
$\quad \mathcal{N}(n), \mathcal{N}(p) \vdash 0 + (n + p) = (0 + n) + p$ *use($+_0(n + p)$)*

$\mathcal{N}(n), \mathcal{N}(p) \vdash n + p = (0 + n) + p$ $use(+_0(n))$
OK

$\mathcal{N}(n), \mathcal{N}(p) \vdash N(n + p)$ $use(tot + (n, m))$
OK

$\mathcal{N}(n), \mathcal{N}(p), (HR)m + (n + p) = (m + n) + p$
$\vdash S(m) + (n + p) = (S(m) + n) + p \; use(+_1(m, n + p)), use(+_1(m, n))$
$\ldots \vdash S(m + (n + p)) = (S(m + n) + p)$ $use(+_1(m + n, p))$
$\ldots \vdash S(m + (n + p)) = S((m + n) + p)$ $use(HR)$
OK

$\ldots \vdash N(n + p)$ $use(tot + (m, n))$
OK

Quelques Propriétés de $\leq$

$(\leq_0) : \forall x(N(x) \rightarrow 0 \leq x)$
$(\leq_1) : \forall x \forall y(N(x), N(y), s(x) \leq s(y) \rightarrow x \leq y)$
$(trans_{\leq}) : \forall x \forall y \forall z(N(x), N(y), N(z), x \leq y, y \leq z \rightarrow x \leq z)$

5.2 Théorie des Listes d'entiers

On procède comme pour les entiers

5.2.1 le langage

Le langage de cette théorie est constitué de la théorie des entiers auquel on ajoute :

- un symbole de prédicat unaire : Ł (être une liste)
- une constante []
- un symbole de fonction binaire ::

5.2.2 les axiomes

Les axiomes sont les suivants :

1. [] est une liste : $(li_1) : L([])$
2. $(li_2) : \forall x \forall l(N(x), L(l) \rightarrow L(x : l))$

3. (li_3):
   $\forall x \forall l(N(x), L(l) \rightarrow \neg(x : l = []))$

4. égalité de 2 listes : (li_4) :
   $\forall x \forall y \forall l_1 \forall l_2(N(x), N(y), L(l_1), L(l_2), x : l_1 = y : l_2 \rightarrow x = y \wedge l_1 = l_2)$

5. Le schéma de récurrence : (rec) : pour tout formule F :
   $(F([]) \wedge \forall l((L(l) \wedge F(l)) \rightarrow \forall x(N(x) \rightarrow F(x : l)))) \rightarrow \forall l(L(l) \rightarrow F(l))$

5.2.3 Une règle dérivée : *Rec*

$$\frac{\Gamma \vdash F[x/[]], \Delta \qquad \Gamma, L(y), F[x/y], N(a) \vdash F[x/a : y], \Delta}{\Gamma, L(x) \vdash F(x), \Delta}\ Rec(x)*$$

(*) : y et a nouvelles variables

5.2.4 définir des fonctions

On peut ensuite définir par exemple les fonctions @, *le*,*pos* :
$(le_0) : le([] = 0)$
$(le_1) : \forall a \forall l(N(a), L(l) \rightarrow le(a : l) = s(le(l)))$

$(@_0) : \forall l(L(l) \rightarrow []@l = l)$
$(@_1) : \forall a \forall l \forall m(N(a), L(l), L(m) \rightarrow (a : l)@m = a : (l@m))$

$(pos_0) : \forall n(N(n) \rightarrow pos(n, []) = 0)$
$(pos_1) : \forall a \forall b \forall l(N(a), N(b), L(l) \rightarrow (a = b \rightarrow pos(a, b : l) = 1))$
$(pos_2) : \forall a \forall b \forall l(N(a), N(b), L(l) \rightarrow (a \neq b \rightarrow pos(a, b : l) = s(pos(a, l))))$

On peut auusi définir le prédicat $\in$ par les axiomes suivants :
$(\in_1) : \forall x(x \notin [])$
$(\in_2) : \forall x \forall a \forall l(N(x), N(a), L(l) \rightarrow x \in a : l \leftrightarrow x = a \vee x \in l)$

5.2.5 quelques propriétés

$(totle) : \forall x(L(x) \rightarrow N(le(x)))$
$(tot@) : \forall x \forall y(L(x), L(y) \rightarrow L(x@y))$
$(totpos) : \forall x \forall y(N(x), L(y) \rightarrow N(pos(x, y)))$

$(assoc_@) :\vdash \forall m \forall n \forall p(L(m), L(n), L(p) \rightarrow m@(n@p) = (m@n)@p)$
$(p_4) : \forall x \forall y(L(x), L(y) \rightarrow le(x@y) = le(x) + le(y))$

$(p_5) :\vdash \forall x \forall l_1 \forall l_2 (N(x), L(l_1), L(l_2), x \notin l_1 \rightarrow pos(x, l_1 @ l_2) = le(l_1) + pos(x, l_2))$

Voici la preuve de $tot@$:

$\vdash \forall x \forall y (L(x), L(y) \rightarrow L(x@y))$ simpl
$L(x), L(y) \vdash L(x@y)$ Rec(x)
$L(y) \vdash L([]@y)$ $use(@_0(y))$
OK

$L(z), N(a), L(y), (HR) L(z@y) \vdash L((a : z)@y)$ $use(@_1(a, z, y))$
$L(z), N(a), L(y), (HR) L(z@y) \vdash L(a : (z@y))$ $use(li_2(a, z@y))$
Ok (HR utilisée par use)

Voici la preuve de $totpos$:

$\vdash \forall x \forall y (N(x), L(y) \rightarrow N(pos(x, y)))$ simpl
$N(x), L(y) \vdash N(pos(x, y))$ Rec(y)
$N(x) \vdash N(pos(x, []))$ $use(pos_0(x))$
$N(x) \vdash N(0)$ $use(pea_1)$
OK

$N(x), L(y), N(m), N(pos(x, y)) \vdash N(pos(x, m : y))$ $use(x = m \vee x \neq m)$
$N(x), L(y), N(m), N(pos(x, y)), x = m \vdash N(pos(x, m : y))$ $use(pos_1(x, m, y))$
$N(x), L(y), N(m), N(pos(x, y)), x = m \vdash N(1)$ $use(pea_2(0))$
$N(x), L(y), N(m), N(pos(x, y)), x = m \vdash N(0)$ $use(pea_1)$
OK

$N(x), L(y), N(m), N(pos(x, y)), x \neq m \vdash N(pos(x, m : y))$ $use(pos_2(x, m, y))$
$N(x), L(y), N(m), N(pos(x, y)), x \neq m \vdash N(s(pos(x, y)))$ $use(pea_2(pos(x, y)))$
Ok (HR utilisée par use)

Nous nous autoriserons dans la suite à ne noter à chaque étape que les nouvelles hypothèses. La preuve précédente devient :

$\vdash \forall x \forall y (N(x), L(y) \rightarrow N(pos(x, y)))$ simpl
$N(x), L(y) \vdash N(pos(x, y))$ Rec(y)
$N(x) \vdash N(pos(x, []))$ $use(pos_0(x))$
$\ldots \vdash N(0)$ $use(pea_1)$
OK

$N(x), L(y), N(m), N(pos(x, y)) \vdash N(pos(x, m : y))$ $use(x = m \vee x \neq m)$
$\ldots, x = m \vdash N(pos(x, m : y))$ $use(pos_1(x, m, y))$
$\ldots \vdash N(1)$ $use(pea_2(0))$
$\ldots \vdash N(0)$ $use(pea_1)$

OK

$\ldots, x \neq m \vdash N(pos(x, m : y))$ $use(pos_2(x, m, y))$
$\ldots \vdash N(s(pos(x, y)))$ $use(pea_2(pos(x, y)))$
Ok

Voici la preuve de p_4 :
$\vdash \forall x \forall y(L(x), L(y) \rightarrow le(x@y) = le(x) + le(y))$ Simpl
$L(x), L(y) \vdash le(x@y) = le(x) + le(y)$ Rec(x)
$L(y) \vdash le([]@y) = le([]) + le(y)$ $use(le_0)$
$\ldots \vdash le([]@y) = 0 + le(y)$ $use(@_0(y))$
$\ldots \vdash le(y) = 0 + le(y)$ $use(+_0(le(y)))$
$\ldots \vdash N(le(y))$ $use(totle)$
OK

$L(y), L(x), N(a), (HR)le(x@y) = le(x) + le(y) \vdash$
$le((a : x)@y) = le(a : x) + le(y)$ $use(@_1(a, x, y))$
$\ldots \vdash le(a : (x@y)) = le(a : x) + le(y)$ $use(le_1(a, x))$
$\ldots \vdash le(a : (x@y)) = s(le(x)) + le(y)$ $use(+_1(le(x), le(y)))$
$\ldots \vdash N(le(x))$ $use(totle(x))$
OK

$\ldots \vdash N(le(y))$ $use(totle(y))$
OK

$\ldots \vdash le(a : (x@y)) = s(le(x) + le(y))$ $use(le_1(a, x@y))$
$\ldots \vdash L(x@y)$ $use(tot@(x, y))$
OK

$\ldots \vdash s(le(x@y)) = s(le(x) + le(y))$ $use(HR)$
OK

Voici la preuve de p_5 :
$\vdash \forall x \forall l_1 \forall l_2(N(x), L(l_1), L(l_2), x \notin l_1 \rightarrow pos(x, l_1@l_2) = le(l_1)+pos(x, l_2))$ $\forall_d(3*)$
$\vdash N(x), L(l_1), L(l_2), x \notin l_1 \rightarrow pos(x, l_1@l_2) = le(l_1) + pos(x, l_2)$ $\rightarrow_d$ $(3*)$
$N(x), L(l_1), L(l_2) \vdash x \notin l_1 \rightarrow pos(x, l_1@l_2) = le(l_1) + pos(x, l_2)$ $Rec(l_1)$
$N(x), L(l_2) \vdash x \notin [] \rightarrow pos(x, []@l_2) = le([]) + pos(x, l_2)$ *simpl*
$\ldots, x \notin [] \vdash pos(x, []@l_2) = le([]) + pos(x, l_2)$ $use(@_0(l_2))$
$\ldots, x \notin [] \vdash pos(x, l_2) = le([]) + pos(x, l_2)$ $use(le_0)$
$\ldots, x \notin [] \vdash pos(x, l_2) = 0 + pos(x, l_2)$ $use(+_0(l_2))$
OK

$N(x), L(l_1), L(l_2), N(a), (HR)x \notin l_1 \rightarrow pos(x, l_1@l_2) = le(l_1) + pos(x, l_2) \vdash$
$x \notin a : l_1 \rightarrow pos(x, a : l_1@l_2) = le(a : l_1) + pos(x, l_2)$ *simpl*

$\ldots x \notin a : l_1 \vdash pos(x, a : l_1@l_2) = le(a : l_1) + pos(x, l_2)$ $use(\in_2 (x, a, l_1))$
$\ldots x \neq a, x \notin l_1 \vdash pos(x, a : l_1@l_2) = le(a : l_1) + pos(x, l_2)$ $use(le_1(a, l_1))$
$\ldots x \neq a, x \notin l_1 \vdash pos(x, a : l_1@l_2) = s(le(l_1)) + pos(x, l_2)$ $use(@_1(a, l_1, l_2))$
$\ldots x \neq a, x \notin l_1 \vdash pos(x, a : (l_1@l_2)) = s(le(l_1))+pos(x, l_2)$ $use(pos_2(x, a, l_1@l_2))$
$\ldots x \neq a, x \notin l_1 \vdash L(l_1@l_2)$ $use(tot@(l_1, l_2))$
OK

$\ldots x \neq a, x \notin l_1 \vdash s(pos(x, l_1@l_2)) = s(le(l_1))+pos(x, l_2)$ $use(+_1(le(l_1), pos(x, l_2)))$
$\ldots x \neq a, x \notin l_1 \vdash N(le(l_1))$ $use(totle(l_1))$
OK

$\ldots x \neq a, x \notin l_1 \vdash N(pos(x, l_2))$ $use(totpos(x, l_2))$
OK

$\ldots x \neq a, x \notin l_1 \vdash s(pos(x, l_1@l_2)) = s(le(l_1) + pos(x, l_2))$ $use(HR)$
OK

5.3 preuve de programme fonctionnel : le tri par insertion

Les théories précédentes permettent de formaliser des types de données et de définir des fonctions et des prédicats sur ces types de données. Les axiomes que nous avons donnés pour définir nos fonctions correspondent aux définitions récursives de fonctions dans un langage fonctionnel comme **Ocaml**. On peut dès lors les considérer comme des programmes fonctionnels.
Ce genre de théorie est donc suffisant pour faire de la preuve de programmes fonctionnels : nos fonctions sont nos programmes et pour montrer leur correction, il faut exprimer dans notre langage formel les propriétés désirées de ces programmes, puis montrer que nos fonctions possèdent ces propriétés. Les propriétés attendues des programmes forment ce qu'on appelle leur spécification.

A titre d'exemple, nous allons définir une fonction faisant le tri par insertion d'une liste et montrer sa correction. Pour y parvenir , il nous faut

- Etablir la spécification du tri.
- Définir notre programme de tri par insertion
- Montrer que notre programme établit sa spécification

Nous insistons sur le fait que l'élaboration de la specification et du programme sont deux activités distinctes : on n'a pas besoin de savoir comment calculer quelque chose pour savoir ce que l'on doit calculer.

5.3.1 La spécification

Notre fonction doit prendre en entrée une liste d'entiers quelconque et la trier c'est à dire, produire en sortie une liste qui contient exactement les mêmes éléments que la liste de départ, mais rangés dans un ordre croissant (par exemple). Ceci peut s'exprimer par deux propriétés : être ordonné et être une permutation de la liste de départ.

être ordonné

Cette propriété peut se définir de différentes façons.
par exemple en disant que tout élément de la liste dont la position est inférieure à celle d'un autre élément est inferieur à cet élément :
$(ord) : \forall l(L(l) \rightarrow ord(L) \leftrightarrow \forall a \forall b(N(a), N(b), a \in L, b \in L, pos(a, L) \leq pos(b, l) \rightarrow a \leq b))$

Une fois posée cette définition, on peut montrer les propriétés suivantes :
$(ord_0) : ord([])$
$(ord_1) : \forall a(N(a) \rightarrow ord(a : []))$
$(ord_2) : \forall a \forall b \forall x(N(a), N(b), L(x) \rightarrow (ord(a : b : x)) \leftrightarrow a \leq b \wedge ord(b : x))$
$(ord_3) : \forall a \forall x(N(a), L(x) \rightarrow (ord(a : x) \leftrightarrow \forall b(b \in l \rightarrow a \leq b) \wedge ord(x)))$

On peut aussi poser ord_0, ord_1 et ord_2 comme axiomes et en déduire ord et ord_4.

être une permutation d'une liste

x est une pemutation de y ssi x et y ont les mêmes éléments et si le nombre d'occurrences de chacuns des éléments de x est le même dans x et dans y.
Définissons la fonction $nocc$ qui calcule le nombre d'occurence d'un éléments dans une liste en posant les axiomes suivants :
$(nocc_0) : \forall a(N(a) \rightarrow nocc(a, []) = 0)$
$(nocc_1) : \forall aptb(N(a), N(b), a = b \rightarrow nocc(a, b : x) = s(nocc(a, x)))$
$(nocc_2) : \forall aptb(N(a), N(b), a \neq b \rightarrow nocc(a, b : x) = nocc(a, x))$

L'axiome pour $permut$ est alors :
$(permut) : \forall x \forall y(L(x), L(y) \rightarrow (\forall a(a \in x \leftrightarrow a \in y)) \wedge \forall a(a \in x \rightarrow nocc(a, x) = nocc(a, y)))$

être un tri

$(tri) : \forall x \forall y(L(x), L(y) \rightarrow ord(y) \wedge permut(x, y))$

5.3.2 le programme

$(trii0) : trii([]) = []$
$(trii1) : \forall a \forall x(N(a), L(x) \rightarrow (tri(a : x) = inser(a, trii(x))))$

$(inser_0) : \forall a(N(a) \rightarrow inser(a, []) = a : [])$
$(inser_1) : \forall a \forall b \forall x(N(a), N(b), L(x), a \leq b \rightarrow inser(a, b : x) = a : b : x)$
$(inser_2) : \forall a \forall b \forall x(N(a), N(b), L(x), (a \not\leq b) \rightarrow inser(a, b : x) = b : inser(a, x))$

Ces deux fonctions vérifient :
$(totinser) : \forall a \forall x(L(x), N(a) \rightarrow L(inser(a, x)))$
$(tottrii) : \forall x(L(x) \rightarrow L(trii(x)))$

5.3.3 preuve de correction

Il nous faut montrer : $\forall x(L(x) \rightarrow tri(x, trii(x)))$

Nous découpons la preuve en isolant les trois propriétés indépendantes suivantes :
$(ordtriins) : \forall x(L(x) \rightarrow ord(trii(x)))$
$(= triins) : \forall x(L(x), \rightarrow (\forall a(a \in x \leftrightarrow a \in trii(x))))$
$(nocctriins) : \forall x(L(x), \rightarrow \forall a(a \in x \rightarrow nocc(a, x) = nocc(a, trii(x))))$

Notre résultat se déduit trivialement de ces trois propriétés :

$\vdash \forall x(L(x) \rightarrow tri(x, trii(x)))$ simpl
$L(x) \vdash tri(x, trii(x))$ $use(tri(x, trii(x)))$
$L(x) \vdash ord(trii(x)) \wedge permut(x, trii(x))$ $\wedge_d$
  $L(x) \vdash ord(trii(x))$ $use(ordtriins(x))$
  OK
  $L(x) \vdash permut(x, trii(x))$ $use(permut(x, trii(x)))$
    $L(x) \vdash L(trii(x))$ $use(tottrii)$
    OK

    $L(x) \vdash \forall a(a \in x \leftrightarrow a \in trii(x)) \wedge$
      $\forall a(a \in x \rightarrow nocc(a, x) = nocc(a, trii(x)))$ $\wedge_d$
    $L(x) \vdash \forall a(a \in x \leftrightarrow a \in trii(x))$ $use(= triins(x))$
    OK

    $L(x) \vdash \forall a(a \in x \rightarrow nocc(a, x) = nocc(a, trii(x)))$ $use(nocctriins(x))$
    OK

Il reste à démontrer les trois propriétés. ceci sera l'objet du reste de ce chapitre.

preuve de $(ordtriins)$

Nous aurons besoin des lemmes suivants :
$(ins_1) : \forall a \forall l \forall x (L(l), N(a), x \in inser(a, l) \rightarrow x = a \vee x \in l))$
$(ins_2) : \forall l (L(l) \rightarrow \forall a (N(a), ord(l) \rightarrow (ord(inser(a, l)))))$
$(\not\leq_1) : \forall a \forall b (N(a), N(b), a \not\leq b \rightarrow b \leq a)$
$(\not\leq_2) : \forall a \forall b (N(a), N(b), a \not\leq b \rightarrow a \neq b)$

$\vdash \forall x (L(x) \rightarrow ord(trii(x)))$ $simpl$
$L(x) \vdash ord(trii(x))$ $Rec(x)$
$L(x) \vdash ord(trii([]))$ $use(trii0)$
$L(x) \vdash ord([])$ $use(ord_0)$
OK

$L(y), N(a), (HR) ord(trii(y)) \vdash ord(trii(a : y))$ $use(trii1(a, y))$
$L(y), N(a), (HR) ord(trii(y)) \vdash ord(inser(a, trii(y)))$ $use(ins_2(trii(y))$
$\ldots, (H4) \forall a (N(a), ord(trii(y)) \rightarrow (ord(inser(a, trii(y)))))$
$\vdash ord(inser(a, trii(y)))$ $use(H4(a))$
OK (HR utilisee par ce use)

$L(y), N(a), (HR) ord(trii(y)) \vdash L(trii(y))$ $use(tottrii(y))$
OK

Montrons maintenant (ins_2)
$\forall l (L(l) \rightarrow \forall a (N(a), ord(l) \rightarrow (ord(inser(a, l)))))$ $\forall_d, \rightarrow_d$
$L(l) \vdash \forall a (N(a), ord(l) \rightarrow (ord(inser(a, l))))$ $Rec(l)$
$\vdash \forall a (N(a), ord([]) \rightarrow (ord(inser(a, []))))$ $simpl$
$N(a), ord([]) \vdash ord(inser(a, []))$ $use(inser_0(a))$
$N(a), ord([]) \vdash ord(a : [])$ $use(ord_1(a))$
OK

$L(l), N(b), (HR) \forall a (N(a), ord(l) \rightarrow ord(inser(a, l))) \vdash \forall a (N(a), ord(b : l) \rightarrow ord(inser(a, b : l)))$ $simpl$
$L(l), N(b), HR, N(a), (H5) ord(b : l) \vdash ord(inser(a, b : l))$ $use(a \leq b \vee a \not\leq b)$
$\ldots a \leq b \vdash ord(inser(a, b : l))$ $use(inser_1(a, b, l))$
$\ldots a \leq b \vdash ord(a : b : l)$ $use(ord2(a, b, l))$
$\ldots a \leq b \vdash a \leq b \wedge ord(b : l)$ $\wedge_d$
$\ldots a \leq b \vdash a \leq b$ ax

$\ldots a \leq b \vdash ord(b : l)$ ax

$\ldots not(a \leq b) \vdash ord(inser(a, b : l))$ $use(inser_2(a, b, l))$
$\ldots not(a \leq b) \vdash ord(b : inser(a, l))$ $use(ord_3(b, inser(a, l)))$
$\ldots not(a \leq b) \vdash L(inser(a, l)))$ $use(totinser(a, l))$
OK

$\ldots not(a \leq b) \vdash \forall x(x \in inser(a, l) \rightarrow b \leq x) \wedge ord(inser(a, l))\ tri_3(b, l)$
$\ldots not(a \leq b), (H7)ord(l), (H8)\forall x(x \in l \rightarrow b \leq x) \vdash \ldots$ $\wedge_d$
$\ldots \vdash ord(inser(a, l))$ $use(HR(a))$
Ok (se sert de H7)

$\ldots \vdash \forall x(x \in inser(a, l) \rightarrow b \leq x)$ $simpl$
$\ldots z \in inser(a, l) \vdash b \leq z$ $use(ins_1(a, l, z))$
$\ldots z = a \vdash b \leq z$ $use(z = a)$
$\ldots z = a \vdash b \leq a$ $use(\not\leq)$
OK

$\ldots z \in l \vdash b \leq z$ $use(H8(z))$
OK

Nous laissons la preuve de $(inser_1)$ en exercice.

preuve de $(= triins)$

$\forall x(L(x), \rightarrow (\forall a(a \in x \leftrightarrow a \in trii(x))))$
Laissée en exercice.

preuve de $(nocctriins)$

Nous aurons besoin des lemmes :
$(refl_{\leq}) : \forall x(N(x) \rightarrow x \leq x)$
$(noccins_1) : \forall a \forall l(N(a), L(l) \rightarrow nocc(a, inser(a, l)) = s(nocc(a, l)))$
$(noccins_2) : \forall a \forall b \forall l(N(a), N(b), L(l) a \neq b \rightarrow nocc(a, inser(b, l)) = nocc(a, l))$

$\vdash \forall x(L(x), \rightarrow \forall a(a \in x \rightarrow nocc(a, x) = nocc(a, trii(x))))$ $\forall_d, \rightarrow_d$
$L(x), \vdash \forall a(a \in x \rightarrow nocc(a, x) = nocc(a, trii(x)))$ $Rec(x)$
$\vdash \forall a(a \in [] \rightarrow nocc(a, []) = nocc(a, trii([])))$ ...

$L(x), N(b), (HR)\forall a(a \in x \rightarrow nocc(a, x) = nocc(a, trii(x)))$
$\vdash \forall a(a \in x \rightarrow nocc(a, b : x) = nocc(a, trii(b : x)))$ $simpl$
$\ldots a \in x \vdash nocc(a, b : x) = nocc(a, trii(b : x))$ $use(a = b \vee a \neq b)$
$\ldots a = b \vdash nocc(a, b : x) = nocc(a, trii(a : x))$ $use(nocc_1(a, b, x))$
$\ldots \vdash s(nocc(a, x)) = nocc(a, trii(a : x))$ $use(refl \leq)$
$\ldots a \leq a \vdash s(nocc(a, x)) = nocc(a, trii(a : x))$ $use(trii1(a, x))$
$\ldots \vdash s(nocc(a, x)) = nocc(a, inser(a, trii(x)))$ $use(HR(a))$

$\ldots \vdash s(nocc(a, trii(x))) = nocc(a, inser(a, trii(x)))$ $use(noccins_1(a, trii(x)))$
$\ldots \vdash L(trii(x))$ $use(tottrii(x))$
OK

$L(x), N(b), (HR), a \in x, a \neq b$
$\vdash nocc(a, b : x) = nocc(a, trii(b : x))$ $use(nocc_2(a, b, x))$
$\ldots \vdash nocc(a, x) = nocc(a, trii(b : x))$ $use(trii_1(b, x))$
$\ldots \vdash nocc(a, x) = nocc(a, inser(b, trii(x)))$ $use(HR(a))$
$\ldots \vdash nocc(a, trii(x)) = nocc(a, inser(b, trii(x)))$ $use(noccins_2(a, b, trii(x)))$
$\ldots \vdash L(trii(x))$ $use(tottrii(x))$
OK

preuve de lemme $noccins_1$
$\vdash \forall a \forall l(N(a), L(l) \rightarrow nocc(a, inser(a, l)) = s(nocc(a, l)))$ simpl
$N(a), L(l) \vdash nocc(a, inser(a, l)) = s(nocc(a, l))$ $Rec(l)$
$N(a) \vdash nocc(a, inser(a, [])) = s(nocc(a, []))$ ...

$N(a), L(l), N(b), (HR) nocc(a, inser(a, l)) = s(nocc(a, l))$
$\vdash nocc(a, inser(a, b : l)) = s(nocc(a, b : l))$ $use(a \leq b \vee a \not\leq b)$
$\ldots a \leq b \vdash nocc(a, inser(a, b : l)) = s(nocc(a, b : l))$ $use(inser_1(a, b, l))$
$\ldots a \leq b \vdash nocc(a, a : b : l) = s(nocc(a, b : l))$ $use(nocc_1(a, a : b : l))$
$\ldots a \leq b \vdash L(a : b : l)$ $use(pea_2(a, b : l))$
$\ldots a \leq b \vdash L(b : l)$ $use(pea_2(b, l))$
OK

$\ldots a \not\leq b \vdash nocc(a, inser(a, b : l)) = s(nocc(a, b : l))$ $use(inser_2(a, b, l))$
$\ldots a \not\leq b \vdash nocc(a, b : inser(a, l)) = s(nocc(a, b : l))$ $use(\leq_2)$
$\ldots a \not\leq b, a \neq b \vdash nocc(a, b : inser(a, l)) = s(nocc(a, b : l))$ $use(nocc_2(a, b, inser(a, l)))$
$\ldots a \not\leq b, a \neq b \vdash L(inser(a, l))$ $use(totinser(a, l))$
OK

$\ldots a \not\leq b, a \neq b \vdash nocc(a, inser(a, l)) = s(nocc(a, b : l))$ $use(nocc_2(a, b, l))$
OK (utilise HR)

La preuve des lemmes restants est laissée en exercice.

Chapter 6

La théorie des ensembles

La théorie des ensembles a pour but de définir la notion d'ensemble. Elle est constituée de 6 axiomes, qui postulent tous l'existence d'ensembles particuliers. Ces six axiomes reviennent à définir le prédicat d'appartenance $\in$. La notoriété et l'utilité de cette théorie viennent du fait qu'elle suffit à modéliser l'ensemble des mathématiques. C'est pour cette raison qu'elle est majoritairement choisie comme langage de spécification du comportement des programmes.

6.1 Les axiomes de Zermelo

6.1.1 axiome d'extensionnalité:

Deux ensembles sont égaux s'ils ont les mêmes éléments.

$$(ext) : \forall s \forall t(\forall x(x \in s \leftrightarrow x \in t) \rightarrow s = t)$$

6.1.2 axiome de la paire :

Etant donnés deux ensembles s et t, il existe un ensemble dont les éléments sont exactement s et t et noté $\{s, t\}$.

$$(paire) : \forall a \forall s \forall t(a \in \{s, t\} \leftrightarrow a = s \vee a = t)$$

6.1.3 axiome de la réunion :

Etant donné un ensemble s, il existe un ensemble dont les éléments sont les éléments des éléments de s donc qui est la réunion des ensembles qui appartiennent à s.

$$(reunion) : \forall a \forall s(a \in \bigcup_{x \in s} x \leftrightarrow \exists t(t \in s \wedge a \in t))$$

6.1.4 axiome des parties :

Etant donné un ensemble s, il existe un ensemble noté $\mathcal{P}(s)$ dont les éléments sont les sous ensembles de s.

$$(parties) : \forall a \forall s(a \in \mathcal{P}(s) \leftrightarrow \forall x(x \in a \rightarrow x \in s))$$

6.1.5 shéma d'axiomes de compréhension :

Il s'agit d'une infinité d'axiomes. Pour toute formule F, et tout ensemble s, il existe un ensemble dont les éléments sont les éléments de s qui vérifient F.

$$(comp) : \forall a \forall s(a \in \{x | x \in s \wedge F(x)\} \leftrightarrow a \in s \wedge F(a))$$

6.1.6 axiomes de l'infini et de l'ensemble vide

Tous les axiomes précédents permettent de construire des ensembles à partir d'ensembles préexistants. Il nous faut donc un axiome postulant l'existence d'au moins un ensemble. Le premier axiome qui vient à l'esprit est celui qui postule l'existence de l'ensemble vide .

$$\exists a(\forall x(x \notin a))$$

noté $\emptyset$. Grace à lui, nous pouvons construire une infinité d'ensembles. Ils sont cependant tous *finis*. Or les ensembles infinis sont absolument nécessaires à la construction des mathématiques. Il suffit de penser au rôle prédominant des ensembles des entiers, des relatifs ou des réels. Or, pour pouvoir construire tous les ensembles infinis que l'on veut, il suffit de postuler l'existence d'un ensemble infini. Une des façons de le faire est de dire qu'il existe un ensemble a qui contient l'ensemble vide et tel que si s est un élement de a alors $s \cup \{s\}$ aussi.

$$(infini) : \exists x(\emptyset \in x \wedge \forall y(y \in x \rightarrow y \cup \{y\} \in x))$$

En fait, une fois postulé l'axiome de l'infini, on peut déduire l'existence de l'ensemble vide. En effet, l'axiome de l'infini nous dit qu'il existe un ensemble (infini) X. Donc par compréhension on en déduit que $\{x | x \in X \wedge x \neq x\}$ est un ensemble. On montre ensuite qu'il n'a pas d'éléments.

Se situer dans la théorie des ensembles, c'est admettre la vérité de ces axiomes. Les preuves dans cette théorie, sont des preuves qui utilisent les règles d'inférences du Calcul des Prédicats avec égalité ainsi que les axiomes de la théorie des Ensembles que nous venons de définir.

6.2 Opérateurs ensemblistes

A partir des axiomes de la théorie des ensembles, nous pouvons construire de nombreux objets mathématiques comme l'inclusion, l'union, l'intersection, les

relations et les fonctions. Ces objets n'ajoutent aucun concepts nouveaux au système, ce sont seulement des notations pour des objets définissables dans le sytème, des abréviations métalinguistiques. Le reste du chapitre sera essentiellement consacré à cela.

6.2.1 Inclusion

$$a \subseteq b \stackrel{\text{def}}{=} a \in \mathcal{P}(b)$$

On a donc par axiome des parties :
$(\subseteq) : \forall a \forall b(a \subseteq b \leftrightarrow \forall x(x \in a \rightarrow x \in b))$
On peut montrer que l'inclusion a les propriétés suivantes :

reflexivité $(refl_{\subseteq})$	$\forall s(s \subseteq s)$
transitivité $(trans_{\subseteq})$	$\forall s,t,u(s \subseteq t, t \subseteq u \rightarrow s \subseteq u)$
anti symétrie $(asym_{\subseteq})$	$\forall s,t(s \subseteq t, t \subseteq s \rightarrow s = t)$

$\vdash \forall s(s \subseteq s)$ $\forall_d$
$\vdash s \subseteq s$ $use(\subseteq (s,s))$
OK

$\vdash \forall s,t(s \subseteq t, t \subseteq s \rightarrow s = t)$ $simpl$
$s \subseteq t, t \subseteq s \vdash s = t$ $use(\subseteq [(s,t),(t,s)])(*)$
$\forall x(x \in s \rightarrow x \in t), \forall x(x \in t \rightarrow x \in s) \vdash s = t$ $use(ext(s,t))$
$\forall x(x \in s \rightarrow x \in t), \forall x(x \in t \rightarrow x \in s) \vdash \forall x(x \in s \leftrightarrow x \in t)$ $\stackrel{\text{def}}{=} (\leftrightarrow), \wedge_d$
$\forall x(x \in s \rightarrow x \in t), \forall x(x \in t \rightarrow x \in s) \vdash \forall x(x \in s \rightarrow x \in t),$ ax

$\forall x(x \in s \rightarrow x \in t), \forall x(x \in t \rightarrow x \in s) \vdash \forall x(x \in t \rightarrow x \in s)$ ax

*** :** la notation $use(\subseteq [(s,t),(t,s)])$ désigne la suite :$use(\subseteq (s,t))$,$use(\subseteq (t,s))$.

6.2.2 Union, intersection, différence

Soient a et b deux ensembles. Par l'axiome de la paire, on peut former l'ensemble $\{a,b\}$. Puis avec l'axiome de la réunion, l'ensemble des éléments des sous ensembles de $\{a,b\}$ c'est à dire l'ensemble des éléments de a et de b. C'est *l'union* de a et de b noté $a \cup b$:

$$a \cup b \stackrel{\text{def}}{=} \bigcup_{x \in \{a,b\}} x$$

L'union vérifie :

$(\cup) : \forall a \forall b (a \cup b = \bigcup_{x \in \{a,b\}} x)$.
$(\cup_1) : \forall x \forall a \forall b (x \in a \cup b \leftrightarrow x \in a \vee x \in b)$

$\vdash \forall a \forall b (a \cup b = \bigcup_{x \in \{a,b\}} x)$	$simpl$
$\vdash a \cup b = \bigcup_{x \in \{a,b\}} x$	$\stackrel{\text{def}}{=} (\cup)$
$\vdash \bigcup_{x \in \{a,b\}} x = \bigcup_{x \in \{a,b\}} x$	$refl$

$\vdash \forall x \forall a \forall b (x \in a \cup b \leftrightarrow x \in a \vee x \in b)$	$simpl$
$\vdash x \in a \cup b \leftrightarrow x \in a \vee x \in b$	$use(\cup(a,b))$
$\vdash x \in \bigcup_{x \in \{a,b\}} x \leftrightarrow x \in a \vee x \in b$	$use(reunion(x, \{a,b\}))$
$\vdash \exists t (t \in \{a,b\} \wedge x \in t) \leftrightarrow x \in a \vee x \in b$	$use(paire(t,a,b))$
$\vdash \exists t ((t = a \vee t = b) \wedge x \in t) \leftrightarrow x \in a \vee x \in b$	$use(Morgan)$
$\vdash \exists t ((t = a \wedge x \in t) \vee (t = b \wedge x \in t)) \leftrightarrow x \in a \vee x \in b$	$use(proppage\ 39)$
$\vdash \exists t (t = a \wedge x \in t) \vee \exists (t = b \wedge x \in t) \leftrightarrow x \in a \vee x \in b$	$use(un\,point\,\exists)(2x)$

Ok

Grace au shéma de compréhension, on construit l'intersection et la différence de deux ensembles :

$$a \cap b \stackrel{\text{def}}{=} \{x | x \in a \wedge x \in b\}$$

$$a - b \stackrel{\text{def}}{=} \{x | x \in a \wedge x \notin b\}$$

L'intersection et la différence vérifient (trivial) :

$(\cap) : \forall a \forall b (a \cap b = \{x | x \in a \wedge x \in b\})$.
$(\cap_1) : \forall x \forall a \forall b (x \in a \cup b \leftrightarrow x \in a \wedge x \in b)$.
$(dif) : \forall a \forall b (a - b = \{x | x \in a \wedge x \notin b\})$.
$(dif_1) : \forall x \forall a \forall b (x \in a - b \leftrightarrow x \in a \wedge x \notin b)$

6.2.3 Couples et Produit cartésien

Un *couple* (a, b) est une paire ordonnée, c'est à dire un ensemble dont le *premier* élément est a et le *second* b. On peut exprimer cette propriété par :

$$(a, b) = (c, d) \leftrightarrow a = c \wedge b = d$$

La paire $\{a, b\}$ ne vérifie pas cette propriété. On a en effet $\{a, b\} = \{b, a\}$. On ne peut donc pas l'utiliser pour définir le couple. En revanche, l'ensemble $\{\{a\}, \{a, b\}\}$ la vérifie. On définit donc :

$$(a, b) \stackrel{\text{def}}{=} \{\{a\}, \{a, b\}\}$$

On peut ensuite définir le *produit cartésien* de deux ensembles a et b comme l'ensemble des couples dont le premier élement est dans a et le second dans b:

$$a \times b \stackrel{\text{def}}{=} \{(x, y) | (x, y) \in \mathcal{P}(\mathcal{P}(a \cup b)) \wedge x \in a \wedge y \in b\}$$

Pour se convaincre que $(x, y) \in \mathcal{P}(\mathcal{P}(a \cup b))$ prenons un exemple :
Posons $a = \{u, v\}$ et $b = \{w\}$. $(u, w) \in a \times b$. Or $(u, w) \stackrel{\text{def}}{=} \{\{u\}, \{u, w\}\}$.
$\{u, w\} \subset a \cup b$ donc $\{u, w\} \in \mathcal{P}(a \cup b)$
$\{u\} \subset a \cup b$ donc $\{u\} \in \mathcal{P}(a \cup b)$ donc
$\{\{u\}, \{u, w\}\} \subset \mathcal{P}(a \cup b)$ donc
$\{\{u\}, \{u, w\}\} \in \mathcal{P}(\mathcal{P}(a \cup b))$

Ces objets vérifient :

$(couple) : \forall a \forall b((a, b) = \{\{a\}, \{a, b\}\})$
$(\times) : \forall a \forall b(a \times b = \{(x, y) | (x, y) \in \mathcal{P}(\mathcal{P}(a \cup b)) \wedge x \in a \wedge y \in b\})$
$(\times_1) : \forall x \forall y \forall a \forall b(x \in a \times b \leftrightarrow (x, y) \in \mathcal{P}(\mathcal{P}(a \cup b)) \wedge x \in a \wedge y \in b)$

6.2.4 quelques propriétés supplémentaires

commutativité	$(com_\cup) : \forall a \forall b (a \cup b = b \cup a)$ $(com_\cap) : \forall a \forall b (a \cap b = b \cap a)$
associativité	$(ass_\cup) : \forall a \forall b ((a \cup b) \cup c = a \cup (b \cup c))$ $(ass_\cap) : \forall a \forall b \forall c ((a \cap b) \cap c = a \cap (b \cap c))$
distributivité	$(dist_1) : \forall a \forall b \forall c ((a \cup b) \cap c = (a \cap c) \cup (b \cap c))$ $(dist_2) : \forall a \forall b \forall c ((a \cap b) \cup c = (a \cup c) \cap (b \cup c))$ $(dist_3) : \forall a \forall b \forall c ((a \cup b) - c = (a - c) \cup (b - c))$ $(dist_4) : \forall a \forall b \forall c ((a \cap b) - c = (a - c) \cap b)$ $(dist_5) : \forall a \forall b \forall c (c - (a \cup b) = (c - a) \cap (c - b))$ $(dist_6) : \forall a \forall b \forall c (c - (a \cap b) = (c - a) \cup (c \cap b))$
	$(dist_7) : \forall a (a \cup a = a)$ $(dist_8) : \forall a (a \cap a = a)$
absorption	$(abs_1) : \forall a \forall b (a \cap (a \cup b) = a)$ $(abs_2) : \forall a \forall b (a \cup (a \cap b) = a)$ $(abs_3) : \forall a (\emptyset \cap a = \emptyset)$ $(abs_4) : \forall a (a \cup \emptyset = a)$
ensemble vide	$\forall a (a - a = \emptyset)$
inclusion	$(\subseteq \cup) : \forall a \forall b (a \subseteq (a \cup b))$ $(\subseteq \cap) : \forall a \forall b ((a \cap b) \subseteq a)$

$\vdash \forall a \forall b \forall c ((a \cup b) \cap c = (a \cap c) \cup (b \cap c))$ *simpl*
$\vdash (a \cup b) \cap c = (a \cap c) \cup (b \cap c)$ $use(ext(a \cup b) \cap c, (a \cap c) \cup (b \cap c))$
$\vdash x \in ((a \cup b) \cap c) \leftrightarrow x \in (a \cap c) \cup (b \cap c)$ $use(\cap_1(a \cup b, c))$
$\vdash x \in a \cup b \wedge x \in c \leftrightarrow x \in (a \cap c) \cup (b \cap c)$ $use(\cup_1(a, b))$
$\vdash (x \in a \vee x \in b) \wedge x \in c \leftrightarrow x \in (a \cap c) \vee x \in (b \cap c)$ $use(Morgan)$
$\vdash (x \in a \wedge x \in c) \vee (x \in b \wedge x \in c) \leftrightarrow x \in (a \cap c) \vee x \in (b \cap c)$ $use(\cap_1[(x, a, c), (x, b, c)])$
Ok

6.3 Les relations binaires

6.3.1 définitions

Une relation de a vers b est un sous-ensemble de $a \times b$.
On définit donc *l'ensemble des relations binaires de a vers b* (noté $a \rightleftharpoons b$) par :

$$a \rightleftharpoons b \stackrel{\text{def}}{=} \mathcal{P}(a \times b)$$

Considérons $R \in a \rightleftharpoons b$: Le *domaine* ($dom$) de R est l'ensemble des premières

composantes des couples de R, et le *codomaine* (ran) l'ensemble des deuxièmes composantes :

$$dom(R) = \{x | x \in a \wedge \exists y(y \in b \wedge (x, y) \in R)\}$$

$$ran(R) = \{y | y \in b \wedge \exists x(x \in a \wedge (x, y) \in R)\}$$

Soit s un sous ensemble de a. La *restriction* de R à s ($s \lhd R$) est l'ensemble des couples de (x, y) de R tels que $x \in a$:

$$s \lhd R \stackrel{\text{def}}{=} \{(x, y) | (x, y) \in R \wedge x \in s\}$$

L' *inverse* de R est définie par :

$$R^{-1} \stackrel{\text{def}}{=} \{(y, x) | (y, x) \in b \times a \wedge (x, y) \in R\}$$

La *composée* de deux relations $R \in a \rightleftharpoons b$ et $T \in b \rightleftharpoons c$ (notée $R; T$) est la relation de a vers c définie par :

$$R; T \stackrel{\text{def}}{=} \{(x, z) | (x, z) \in a \times c \wedge \exists y((x, y) \in R \wedge (y, z) \in T)\}$$

Voici quelques autres constructions sur les relations :
$R \rhd p \stackrel{\text{def}}{=} \{(x, y) | (x, y) \in R \wedge y \in p\}$
$p \lhd\!\!\!\!- R \stackrel{\text{def}}{=} \{(x, y) | (x, y) \in R \wedge x \notin p\}$
$R \rhd\!\!\!\!- p \stackrel{\text{def}}{=} \{(x, y) | (x, y) \in R \wedge y \notin p\}$
$R[p] \stackrel{\text{def}}{=} ran(p \lhd R)$
$R \leftarrow S \stackrel{\text{def}}{=} (dom(S) \lhd\!\!\!\!- R) \cup S$

6.3.2 Propriétés directes

Voici les propriétés nous permettant d'utiliser cs constructions avec *use*:

$(rela) : \forall a \forall b(a \rightleftharpoons b = \mathcal{P}(a \times b))$
$(rela_1) : \forall R \forall a \forall b(R \in a \rightleftharpoons b \leftrightarrow \forall x \forall y((x,y) \in R \rightarrow x \in a \wedge y \in b))$

$(dom) : \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow dom(R) = \{x | x \in a \wedge \exists y(y \in b \wedge (x,y) \in R)\})$
$(dom_1) : \forall x \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow x \in dom(R) \leftrightarrow x \in a \wedge \exists y(y \in b \wedge (x,y) \in R))$

$(ran) : \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow ran(R) = \{y | y \in b \wedge \exists x(x \in a \wedge (x,y) \in R)\})$
$(ran_1) : \forall y \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow y \in ran(R) \leftrightarrow y \in b \wedge \exists x(x \in a \wedge (x,y) \in R))$

$(\lhd) : \forall s \forall R(s \lhd R = \{(x,y) | (x,y) \in R \wedge x \in s\})$
$(\lhd_1) : \forall x, \forall y \forall s \forall R(x \in s \lhd R \leftrightarrow (x,y) \in R \wedge x \in s)$

$(^{-1}) : \forall R \forall A \forall B(R \in A \rightleftharpoons B \rightarrow R^{-1} = \{(y,x) | (y,x) \in b \times a \wedge (x,y) \in R\})$
$(^{-1}_1) : \forall x \forall y \forall R \forall A \forall B(R \in A \rightleftharpoons B \rightarrow (y,x) \in R^{-1} \leftrightarrow (x,y) \in R)$

$(;) : \forall R \forall T \forall A \forall B \forall C(R \in A \rightleftharpoons B, T \in B \rightleftharpoons C \rightarrow$
$R;T = \{(x,z) | (x,z) \in a \times c \wedge \exists y((x,y) \in R \wedge (y,z) \in T)\})$
$(;_1) : \forall x \forall z \forall R \forall T \forall A \forall B \forall C(R \in A \rightleftharpoons B, T \in B \rightleftharpoons C \rightarrow$
$(x,z) \in R;T \leftrightarrow \exists y((x,y) \in R \wedge (y,z) \in T))$

$(\rhd) : \forall R \forall p(R \rhd p = \{(x,y) | (x,y) \in R \wedge y \in p\})$
$(\rhd_1) : \forall x \forall y \forall R \forall p((x,y) \in R \rhd p \leftrightarrow (x,y) \in R \wedge y \in p)$

$(\lhd) : \forall R \forall p(p \lhd R = \{(x,y) | (x,y) \in R \wedge x \notin p\})$
$(\lhd_1) : \forall x \forall y \forall R \forall p((x,y) \in p \lhd R \leftrightarrow (x,y) \in R \wedge x \notin p)$

$(\rhd) : \forall R \forall p(R \rhd p = \{(x,y) | (x,y) \in R \wedge y \notin p\})$
$(\rhd_1) : \forall x \forall y \forall R \forall p((x,y) \in R \rhd p \leftrightarrow (x,y) \in R \wedge y \notin p)$

$[] : \forall R \forall p(R[p] = ran(p \lhd R))$
$[]_1 : \forall y \forall R \forall p(y \in R[p] \leftrightarrow \exists x(x,y) \in R \wedge x \in p)$

6.3.3 Un exemple de preuve

Nous allons montrer que la composée de deux relations est une relation
$(rela;) \forall f \forall g \forall a \forall b \forall c(f \in a \rightleftharpoons b, g \in b \rightleftharpoons c \rightarrow f;g \in a \rightleftharpoons c)$

$\vdash \forall f \forall g \forall a \forall b \forall c(f \in a \rightleftharpoons b, g \in b \rightleftharpoons c \rightarrow f;g \in a \rightleftharpoons c)$	*simpl*
$f \in a \rightleftharpoons b, g \in b \rightleftharpoons c \vdash f;g \in a \rightleftharpoons c$	$use(\rightleftharpoons_1 (f;g,a,c))$
$\ldots, (t,v) \in f;g \vdash t \in a \wedge v \in b$	$use(;_1 (t,v,f,g,a,b,c))$

$$\begin{array}{ll}
\ldots (t,u) \in f, (u,v) \in g \vdash t \in a \wedge v \in b & use(\rightleftharpoons_1 (f,a,b)) \\
\ldots, (t,u) \in f, (u,v) \in g, (H_5) \forall x \forall y((x,y) \in f \rightarrow x \in a \wedge x \in b) & \\
\qquad \vdash t \in a \wedge v \in b & use(H_5(t,u)) \\
\ldots, (t,u) \in f, (u,v) \in g, t \in a, u \in b \vdash t \in a \wedge v \in b & use(\rightleftharpoons_1 (g,b,c)) \\
\ldots, (t,u) \in f, (u,v) \in g, t \in a, u \in b, (H6) \forall x \forall y((x,y) \in g \rightarrow x \in b \wedge x \in c) & \\
\qquad \vdash t \in a \wedge v \in b & use(H6(g,b,c)) \\
\ldots, (t,u) \in f, (u,v) \in g, t \in a, u \in b, u \in b, v \in c) \vdash t \in a \wedge v \in b & \wedge_d \\
\quad \ldots, (t,u) \in f, (u,v) \in g, t \in a, u \in b, u \in b, v \in c) \vdash t \in ab & ax \\
 & \\
\quad \ldots, (t,u) \in f, (u,v) \in g, t \in a, u \in b), u \in b, v \in c \vdash v \in b & ax
\end{array}$$

6.4 Les fonctions

6.4.1 Définitions

Certaines relations sont très utiles : Celles qui ont comme propriétés que tous les éléments de leur domaine ont une image unique. Ce sont les fonctions. Parmi elles on distingue les injections, c'est à dire les fonctions telles que 2 éléments différents de leur domaine ont des images distinctes, les surjections, dont tous les éléments du codomaine sont atteints, et les bijections qui sont à la fois injectives et surjectives. On distingue les fonctions totales, dont tous les éléments de l'ensemble de départ ont une image, des fonctions partielles.

Si f est une fonction, et si $x \in dom(f)$ alors il existe un unique b tel que $(a,b) \in f$. Nous l'appellerons *l'image de x par f* et le noterons $f(x)$.

D'autres part l'ensemble $\{(x,y)|(x,y) \in s \times t \wedge y = e\}$ où e est un terme ne contenant pas d'autre variable que x et tel que $\forall x(x \in s \rightarrow e \in t)$ désigne une unique fonction notée $\lambda x.(x \in s|e)$ telle que $\forall x(x \in s \rightarrow \lambda x.(x \in s|e)(x) = e$.

formellement :

fonction	$a \nrightarrow b \stackrel{\text{def}}{=}$	$\{r|r \in a \rightleftharpoons b \wedge \forall x(x \in dom(r) \rightarrow \exists! y((x,y) \in r))\}$
fonct. totale	$a \rightarrow b \stackrel{\text{def}}{=}$	$\{f|f \in a \nrightarrow b \wedge dom(f) = a\}$
injection	$a \rightarrowtail b \stackrel{\text{def}}{=}$	$\{f|f \in a \nrightarrow b \wedge$ $\forall x, x'((x \in dom(f) \wedge x' \in dom(f) \wedge x \neq x') \rightarrow f(x) \neq f(x'))\}$
injection totale	$a \rightarrowtail b \stackrel{\text{def}}{=}$	$\{f|f \in a \rightarrow b \wedge f \in a \rightarrowtail b\}$
surjection	$a \twoheadrightarrow b \stackrel{\text{def}}{=}$	$\{f|f \in a \nrightarrow b \wedge ran(f) = b\}$
surj. totale	$a \twoheadrightarrow b \stackrel{\text{def}}{=}$	$\{f|f \in a \rightarrow b \wedge f \in a \twoheadrightarrow b\}$

6.4.2 Propriétés directes

$(fonctpa) : \forall a \forall b (a \mapsto b = \{r | r \in a \rightleftharpoons b \wedge$
$\forall x (x \in dom(r) \rightarrow \exists! y ((x, y) \in r))\})$
$(fonctpa_1) : \forall f \forall a \forall b (f \in a \mapsto b \leftrightarrow$
$f \in a \rightleftharpoons b \wedge \forall x (x \in dom(f) \rightarrow \exists! y ((x, y) \in f)))$

$(fonct) : \forall a \forall b (a \rightarrow b = \{f | f \in a \mapsto b \wedge dom(f) = a\})$
$(fonct_1) : \forall f \forall a \forall b (f \in a \rightarrow b \leftrightarrow f \in a \mapsto b \wedge dom(f) = a)$

$(injpa) : \forall a \forall b (a \rightarrowtail b = \{f | f \in a \mapsto b \wedge$
$\forall x, x' (x \in dom(f), x' \in dom(f), x \neq x' \rightarrow f(x) \neq f(x'))\})$
$(injpa_1) : \forall f \forall a \forall b (f \in a \rightarrowtail b \leftrightarrow f \in a \mapsto b \wedge$
$\forall x, x' (x \in dom(f), x' \in dom(f), x \neq x' \rightarrow f(x) \neq f(x')))$

$(inj) : \forall a \forall b (a \rightarrowtail b \stackrel{\text{def}}{=} \{f | f \in a \rightarrow b \wedge f \in a \rightarrowtail b\})$
$(inj_1) : \forall f \forall a \forall b (f \in a \rightarrowtail b \leftrightarrow f \in a \rightarrow b \wedge f \in a \rightarrowtail b)$

$(surjpa) : \forall a \forall b (a \twoheadrightarrow b = \{f | f \in a \mapsto b \wedge ran(f) = b\})$
$(surjpa_1) : \forall f \forall a \forall b (f \in a \leftrightarrow f \in a \mapsto b \wedge ran(f) = b)$

$(surj) : \forall a \forall b (a \twoheadrightarrow b = \{f | f \in a \rightarrow b \wedge f \in a \twoheadrightarrow b\})$
$(surj_1) : \forall f \forall a \forall b (f \in a \leftrightarrow f \in a \rightarrow b \wedge f \in a \twoheadrightarrow b)$

6.5 Les entiers naturels

6.5.1 Qu'est ce que $\mathcal{N}$?

L'ensemble des entiers naturels est une suite $0, 1, 2 \ldots$.
. Plus précisemment, c'est une suite infinie avec un plus petit élément. On peut atteindre tous les entiers en se donnant 0 et une opération *successeur*. Fondamentalement , $\mathcal{N}$ est le plus petit ensemble contenant 0 et clos par l'opération *successeur*.
Cette structure particulière de $\mathcal{N}$ autorise un type de raisonnement nouveau : le raisonnement par récurrence.
De même, on pourra définir des fonctions par récurrence.

6.5.2 Construction des entiers naturels

Dans le chapitre précédent, nous nous sommes donnés des axiomes permettant de manipuler les entiers. Ce n'esp plus nécéssaire ici. Nous allons*construire* l'ensemble des entiers naturels à l'intérieur de la théorie des ensembles. Il nous

faut donc trouver dans la théorie des ensembles un ensemble qui représentera $\mathcal{N}$.
$0 \stackrel{\text{def}}{=} \emptyset$
$1 \stackrel{\text{def}}{=} \{0\} = \{\emptyset\}$
$2 \stackrel{\text{def}}{=} \{0, 1\} = \{\{\emptyset\}, \emptyset\}$
$\vdots$
La fonction *successeur* ($S(x)$) est définie par :

$$S(x) \stackrel{\text{def}}{=} x \cup \{x\}$$

Bien que ces "entiers" aient des propriétés étranges comme par exemple que $0 \in 1 \in 2 \ldots$, nous verrons qu'ils ont toutes les propriétés désirées des entiers, et pourront donc les modéliser.

L'axiome de l'infini nous assure qu'il existe un ensemble vérifiant :

$$\vdash \exists x \emptyset \in x \wedge \forall y (y \in x \rightarrow S(y) \in x)$$

donc qui contient bien tous nos entiers. en effet : il contient nécessairement $\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}, \ldots$ donc tous nos "entiers". Mais rien n'assure qu'il ne contienne que ceux là. Il nous faut pouvoir exprimer la propriété d'etre le *plus petit ensemble qui contenant zéro et clos par successeur*. Pour cela il suffit de construire l'intersection de tous les ensembles ayant cette propriété. Appelons A l'ensemble dont l'existence est postulée par l'axiome de l'infini, et I(x) la propriété d'etre le plus petit ensemmble contenant 0 et clos par *successeur*.
$I(x) \stackrel{\text{def}}{=} \emptyset \in \wedge (y \in x \rightarrow S(y) \in x)$.
On peut alors définir $\mathcal{N}$ par :
$\mathcal{N} \stackrel{\text{def}}{=} \{b | b \in A \wedge \forall y (I(y) \rightarrow b \in y\}$

Le shéma de compréhension nous assure que $\mathcal{N}$ existe.

$\mathcal{N}$ vérifie $I(x)$: $\emptyset \in \mathcal{N}$ car $\emptyset$ est dans tout ensemble vérifiant I.
Si $a \in \mathcal{N}$ alors par définition de $\mathcal{N}$, a est dans tout ensemble vérifiant I, et donc par définition de I, $S(a)$ aussi, donc $S(a) \in \mathcal{N}$.

$\mathcal{N}$ vérifie $\forall y (I(y) \rightarrow B \subseteq y)$ par définition de $\mathcal{N}$.

6.5.3 Les axiomes de péano

Maintenant que nous avons construit $\mathcal{N}$, les axiomes de péano énoncés dans le chapitre précédent sont démontrables. Nous ne les démontrerons pas, sauf le shéma de récurrence. Nous pourrons les utiliser dans les preuves. Nous avons aussi :

Proposition 6.5.1 $\forall x(x \in \mathcal{N} \rightarrow (x = 0 \vee \exists! y(y \in \mathcal{N} \wedge x = S(y))))$

Nous pouvons aussi montrer que la relation d'appartenance $\in$ est une relation d'ordre stricte (transitive et anti reflexive) et totale sur $\mathcal{N}$. Elle a aussi un plus petit élément : 0. La relation $<$ sera donc représentée par l'appartenance :

$$x < y \stackrel{\text{def}}{=} x \in y$$

Le raisonnement par récurrence

Le raisonnement par récurrence à la forme suivante : Pour montrer qu'une propriété F vaut pour tout entier, il suffit de montrer que F est vraie de 0, et que si l'on suppose qu'elle est vraie d'un entier n quelconque, alors elle l'est aussi de son successeur. Sa correction est assurée par le fait que tout entier est atteignable par itération de S sur 0.

Formellement le principe de récurrence s'énonce comme suit : Soit F une formule quelconque,

$$Rec \stackrel{\text{def}}{=} (F(0) \wedge \forall y(y \in \mathcal{N} \wedge F(y) \rightarrow F(S(y)))) \rightarrow \forall x(x \in \mathcal{N} \rightarrow F(x))$$

Cette formule est démontrable en théorie des ensembles :

preuve :
Supposons $F(0)$, $(HR) = \forall y(y \in \mathcal{N} \wedge F(y) \rightarrow F(S(y))$,
et montrons $\forall x(x \in \mathcal{N} \rightarrow F(x)$
Supposons que cela ne soit pas le cas: Il y aurait donc un entier n tel que $\neg F(n)$. Considérons le plus petit entier m tel que $\neg F(m)$. Cela ne peut être 0 car on a en hypothèse $F(0)$. m est donc le successeur d'un entier m_1. $m_1 < m$ donc m_1 vérifie F. donc par HR m aussi, ce qui contredit $\neg F(m)$.

Comme à notre habitude, on peut à partir de la proposition précédente engendrer une règle dérivée :

$$\frac{\Gamma \vdash F(0) \qquad \Gamma, y \in \mathcal{N}, F(y) \vdash F(S(y))}{\Gamma, x \in \mathcal{N} \vdash F(x)}\ Rec(*)$$

(*) y non libre dans $\forall x F, \Gamma$

6.5.4 Définition de fonctions par récurrence

Nous avons défini un ensemble particulier, l'ensemble des entiers naturels, à l'intérieur de la théorie des ensembles. Nous avons ensuite montré qu'il était possible de raisonner par récurrence pour montrer des propriétés universelles sur cet ensemble. Puisque nous avons aussi défini la notion de fonctions d'ensembles vers d'autres ensembles dans la théorie, nous savons qu'il existe des fonctions sur les entiers. Nous pouvons montrer par exemple que $S \in \mathcal{N} \rightarrow \mathcal{N}$. Nous pouvons appliquer tous les opérateurs sur les fonctions existantes sur les entiers

pour définir de nouvelles fonctions. Par exemple $(S;S) \in \mathcal{N} \to \mathcal{N}$ et désigne intuitivement la fonction qui a x associe $x+2$. Pour définir de nouvelles fonctions sur les entiers, on a envie de pouvoir procéder de la façon suivante : Supposons que l'on ait à l'esprit une fonction h de $\mathcal{N} \to A$ (A est un ensemble quelconque) donnée par les deux clauses suivantes :
$h(0) = a$ (pour $a \in A$)
$h(S(n)) = F(h(n))$ (pour une fonction donnée $F \in A \to A$).

Intuitivement, ces deux clauses permettent de calculer la valeur de h en tout point x. En effet : $h(n) = F(h(n-1)) = F(F(h(n-2))) = \ldots = F^n(a)$.
Ce principe de définition de fonction (définition par récurrence) est admissible en théorie des ensembles.

Proposition 6.5.2 *Soit A un ensemble, $F \in A \to A$. Il existe une unique fonction $h \in \mathcal{N} \to A$ telle que :*
$h(0) = a$ (pour $a \in A$) et
$\forall x(x \in \mathcal{N} \to h(S(x)) = F(h(x)))$

On peut généraliser le principe de définition par récurrence, pour les fonctions à n arguments. Voici le principe pour les fonctions à deux arguments:
Soit A un ensemble, $F_1 \in B \to A$, $F_2 \in A \times N \times B \to A$ $F_3 \in B \to B$. Il existe une unique fonction $h \in \mathcal{N} \times B \to A$ telle que :
$h(0, y) = F_1(y)$ et
$\forall x, y(x \in \mathcal{N} \wedge y \in B \to h(S(x), y) = F_2(h(x, F_3(y)), x, y))$

6.5.5 Quelques fonctions sur les entiers

addition	$(+_0) : \forall y(N(y) \to 0 + y = y)$ $(+_1) : \forall x \forall y(N(x), N(y) \to s(x) + y = s(x + y))$
multiplication	$(*_0) : \forall y(N(y) \to 0 * y = 0)$ $(*_1) : \forall x \forall y(N(x), N(y) \to s(x) * y = (x * y) + y)$
exponentiel	$(exp_0) : \forall m(N(n) \to m^0 = 1$ $(exp_1) : \forall m \forall n(m^{S(n)} = m \times (m^n)$
soustraction	$\{(m, n, a) \mid m, n, a \in \mathcal{N} \wedge n \leq m \wedge n + a = m\}$
division	$\{(m, n, a) \mid m, n, a \in \mathcal{N} \wedge m \neq 0 \wedge m \times a \leq n \wedge m < m \times S(a)\}$

6.5.6 Quelques propriétés

Voici quelques propriétés sur les fonctions précédentes :

$(commut_+) : \forall m \forall n (N(n), N(m) \rightarrow m + n = n + m$
$(assoc_+) : \forall m \forall n \forall p (N(n), N(m), N(p) \rightarrow m + (n + p) = (m + n) + p$
$(commut_*) : \forall m \forall n (N(n), N(m) \rightarrow m \times n = n \times m$
$(assoc_*) : \forall m \forall n \forall p (N(n), N(m), N(p) \rightarrow m \times (n \times p) = (m \times n) \times p$
$(dist1) : \forall m \forall n \forall p (N(n), N(m), N(p) \rightarrow m \times (n + p) = (m \times n) + (m \times p)$
$(dist2) : \forall m \forall n \forall p (N(n), N(m), N(p) \rightarrow m^{n+p} = m^n \times m^p$
$(assoc_e xp) : \forall m \forall n \forall p (N(n), N(m), N(p) \rightarrow m^{n \times p} = m^{np}$

Etant donné que les axiomes de péano sont démontrables en théorie des ensembles, les preuves de ces propriétés se font exactement de la même façon que dans le chapitre précédent.

Exercice 6.5.1 *Montrer que la soustraction et la division sont des fonctions*

Exercice 6.5.2 *Montrer les propriétés énoncées sur les fonctions*

6.6 Les suites

6.6.1 Définition des suites

Soit A un ensemble. Une suite d'éléments de A de longueur n est une fonction totale de l'intervalle $1..n$ vers A.
Définissons donc d'abord la notion d'intervalle de 1 à n :

$$1..n \stackrel{\text{def}}{=} \{x | x \in \mathcal{N} \wedge 1 \leq x \leq n\}$$

On peut maintenant construire l'ensemble des suites d'éléments de A, noté $seq(A)$ comme l'ensemble des fonctions totales des intervalles d'entiers vers A :

$$\{f | f \in \mathcal{N} \rightarrow A \wedge \exists n (n \in \mathcal{N} \wedge f \in 1..n \rightarrow A)\}$$

$\emptyset \in seq(A)$ car c'est une fonction totale de $1..0$ vers A. $\emptyset$ désignera dans ce contexte la suite vide et le noterons $[\,]$.

On peut aussi définir la fonction : qui étant donné un élément a de A et une suite $s \in seq(A)$, insère a en tête de s. Il suffit d'augmenter de 1 les indices de s puis d'ajouter le couple $1 \mapsto a$. Pour augmenter de 1 les indices de s, il suffit de composer la fonction prédécesseur ($pred$) avec s comme l'indique la figure suivante :

$$\begin{array}{cccccc} & pred & & s & \\ 2 & \mapsto & 1 & \mapsto & a_1 \\ 3 & \mapsto & 2 & \mapsto & a_2 \\ & & \vdots & & \\ n+1 & \mapsto & n & \mapsto & a_n \end{array}$$

Soit $a \in A$ et $s \in seq(A)$ On définit donc :

$$a : s \stackrel{\text{def}}{=} \{1 \mapsto a\} \cup (pred; s)$$

Proposition 6.6.1 $:\in A \times s \in seq(A) \rightarrow\ s \in seq(A)$

6.6.2 Raisonnement par récurrence sur les suites

On se rend compte que l'ensemble des suites est exactement le plus petit ensemble qui contient la suite vide et est clos par la fonction $cons$:

Proposition 6.6.2

$$s \in seq(A) \leftrightarrow s = [\,] \vee (\exists y \exists a(y \in seq(A) \wedge a \in A \wedge s = y : a))$$

Preuve :
$\leftarrow$: On a déja montré que $[\,] \in seq(A)$ et le fait que étant donnés $y \in seq(A)$ et $a \in A$ quelconque $a : y \in seq(A)$ découle du fait que : est une fonction totale.

$\rightarrow$: Soit $s \in seq(A)$ montrons :
$s = [\,] \vee (\exists y \exists a(y \in seq(A) \wedge a \in A \wedge s = y : a))$
On a par définition de $seq(A)$:
$s \in \mathcal{N} \nrightarrow A$, $\exists n(n \in \mathcal{N} \wedge s \in 1..n \rightarrow\ A)$
Soit n_0 tel que : $n_0 \in \mathcal{N}$ et $s \in 1..n_0 \rightarrow\ A$
Puisque n_0 est un entier nous pouvons raisonner par cas :
1. $n_0 = 0$: dans ce cas $s = [\,]$ car $1..0 = \emptyset$ et nous concluons en choisissant la première partie de la disjonction qui est notre but.
2. $n_0 = S(n_1)$ pour un certain entier n_1 : en ce cas, puisque $s \in 1..S(n_1) \rightarrow\ A$, on peut construire $s_1 = S; (1 \leftarrowtail s)$ telle que $s_1 =\in 1..n_1 \rightarrow\ A$
Il suffit alors de choisir dans notre but la deuxième partie de l'alternative en prenant s_1 comme témoin pour y , et $s(1)$ comme témoin pour a .
On a en effet $s(1) : s_1 = s$.

Gràce au résultat précédent, on déduit qu'il est possible de raisonner par récurrence sur les suites :

pour montrer qu'une propriété F est vraie de toutes les suites, il suffit de montrer que F est vraie de la suite vide, et que si l'on suppose F vraie d'une suite s, alors on sait montrer qu'elle reste vraie de la suite $a : x$ pour a l'un quelconque des éléments de A.
Ceci signifie que la formule suivante est démontrable dans la théorie des ensembles:

$$(F([\,]) \wedge \forall a \forall y (a \in A \wedge y \in seq(A) \wedge F(y) \rightarrow F(a : y))) \rightarrow \forall s (s \in seq(A) \rightarrow F(s))$$

La preuve est identique au cas des entiers naturels. Elle utilise donc aussi une notion de relation d'ordre sur les suites. On prendra bien sur comme relation d'ordre la longueur des suites. On peut aussi définir l'ordre lexicographique sur les suites. On peut aussi définir la fonction d'insertion en queue d'une suite. On aura aussi que $seq(A)$ est le plus petit ensemble contenant la suite vide et clos par insertion en queue. On aura donc un principe de récurrence fondé sur ce résultat. Contrairement aux entiers, où le raisonnement par récurrence est systématique ou presque, de nombreux résultats sur les suites ne se démontrent pas par récurrence mais en utilisant la définition et des résultats sur les fonctions.

6.6.3 Principe de définition par récurrence sur les suites

Soit A un ensemble, $F \in B \times A \rightarrow B$,
Il existe une unique fonction $h \in seq(A) \rightarrow B$ telle que :
$h([\,]) = b$
$h(a : x) = F(h(x), a)$

Soit A un ensemble, $F_1 \in B \rightarrow C$,
$F_2 \in C \times seq(A) \times A \times B \rightarrow C$ et
$F_3 \in B \rightarrow B$.
Il existe une unique fonction $h \in seq(A) \times B \rightarrow C$ telle que :
$h([\,], y) = F_1(y)$ et
$\forall x, y (x \in seq(A) \wedge y \in A \rightarrow h(x : a, y) = F_2(h(x, F_3(y)), x, a, y))$

6.6.4 Quelques fonctions sur les suites

En plus de celles définies dans le chapitre précédent , on peut définir par exemple :

$reverse$	$(rev_0) : reverse([]) = []$ $(rev_1) : \forall a \forall x (N(a), N(x) \rightarrow$ $\quad reverse(a : x) = append(reverse(x), a : []))$
$t \uparrow n$	$t \uparrow n \stackrel{\text{def}}{=} (1..n) \lhd t$

6.6.5 le cardinal d'un ensemble fini

Comment exprimer le fait qu'un ensemble est fini ? Il suffit de pouvoir le mettre en bijection avec un sous ensemble strict de $\mathcal{N}$:

$$Fini(x) \stackrel{\text{def}}{=} \exists f \exists n(n \in \mathcal{N} \wedge f \in 1..n \nrightarrow x \wedge f \in 1..n \rightarrowtail x)$$

Soit maintenant A un ensemble et x un sous ensemble de A.
L'ensemble $card(x) \stackrel{\text{def}}{=} \{(x,m)|x \subseteq A \wedge \exists n \exists f(n \in \mathcal{N} \wedge f \in 1..n \nrightarrow x \wedge f \in 1..n \rightarrowtail x \wedge m = max(dom(f)))\}$ définit une fonction qui associe à chaque sous ensemble fini de A le nombre de ses éléments.
Pour montrer que c'est une fonction, il faut montrer que x a une unique image. Ceci est assuré par le fait que s'il existe n et une bijection de $1..n$ vers x alors ceux ci sont uniques. Le fait qu'elle soit totale sur les sous ensembles finis de A est déduit de la définition de $Fini$.

Exemple 6.6.1 *Soit s et t des suites d'éléments de A.*
Ecrire des formules exprimant les faits suivants :

1. *s est ordonnée*
2. *s est une permutation de t*
3. *s est sans répétition*

6.7 Exercices

Exercice 6.7.1 *Montrer que :* $\forall a \forall b \forall c((a \cup b) - c = (a - c) \cup (b - c))$

$\vdash \forall a \forall b \forall c((a \cup b) - c = (a - c) \cup (b - c))$ *simpl*
$\vdash (a \cup b) - c = (a - c) \cup (b - c)$ $use(ext((a \cup b) - c, (a - c) \cup (b - c)))$
$\vdash x \in (a \cup b) - c \leftrightarrow x \in (a - c) \cup (b - c)$ $use(dif_1(x, a \cup b, c))$
$\vdash x \in (a \cup b) \wedge x \notin c \leftrightarrow x \in (a - c) \cup (b - c)$ $use(\cup_1(x, a, b))$
$\vdash (x \in a \vee x \in b) \wedge x \notin c \leftrightarrow x \in (a - c) \cup (b - c)$ $Morgan$
$\vdash (x \in a \wedge x \notin c) \vee (x \in b \wedge x \notin c) \leftrightarrow x \in (a - c) \cup (b - c)$ $use(\cup_1(a - c, b - c))$
$\vdash (x \in a \wedge x \notin c) \vee (x \in b \wedge x \notin c) \leftrightarrow x \in (a - c) \vee x \in (b - c)$ $use(_1[(x, a, c), (x, b, c)])$
OK

Exercice 6.7.2 *Montrer :* $\forall r \forall A \forall B \forall u \forall v u \lhd (v \lhd r) = (u \cap v) \lhd r$

$\vdash \forall r \forall u \forall v(u \lhd (v \lhd r) = (u \cap v) \lhd r)$ *simpl*
$\vdash u \lhd (v \lhd r) = (u \cap v) \lhd r$ $use(ext(u \lhd (v \lhd r), (u \cap v) \lhd r))$
$\vdash (x, y) \in u \lhd (v \lhd r) \leftrightarrow (x, y) \in (u \cap v) \lhd r$ $use(\lhd_1(u, v \lhd r))$
$\vdash (x, y) \in (v \lhd r) \wedge x \in u \leftrightarrow (x, y) \in (u \cap v) \lhd r)$ $use(\lhd_1(x, y, v, r)$

$\vdash ((x,y) \in r \wedge x \in v) \wedge x \in u) \leftrightarrow (x,y) \in (u \cap v) \lhd r)$ $use(assoc_\wedge)$
$\vdash (x,y) \in r \wedge (x \in u \wedge x \in v) \leftrightarrow (x,y) \in (u \cap v) \lhd r)$ $use(\cap_1(x,u,v))$
$\vdash \forall(x,y)(((x,y) \in r \wedge (x \in u \cap v)) \leftrightarrow (x,y) \in (u \cap v) \lhd r)$ $use(\lhd_1(x,y,u \cap v,r))$
OK

Exercice 6.7.3 *Donnons nous la fonction som qui prend comme argument une suite l et qui rend la somme des éléments de l et définie par les équations suivantes :*
$som([]) = 0$
$som(a : l) = a + som(l)$

Démontrer que :
$\forall l_1 \forall l_2 (l \in seq(\mathcal{N}) \wedge l_2 \in s \in seq(\mathcal{N})) \rightarrow som(l_1@l_2) = som(l_1) + som(l_2))$.

Exercice 6.7.4 *On veut modéliser le fonctionnement d'une bibliothèque : On observe les règles suivantes :*

1. *Un exemplaire est toujours associé à un livre. Celui ci est unique.*
2. *Un même exemplaire de livre ne peut etre emprunté par différents abonnés.*
3. *Un même abonné ne peut emprunter plus d'un exemplaire d'un même livre*

- *formaliser les règles en Calcul des prédicats On se donne :*
  $Ex(\_)$(etre un exemplaire),
  $L(\_)$ (etre un livre)
  $A(\_)$ (etre un abonné)
  $Emp(a,e)$ (a emprunte l'exemplaire e) et
  $Exde(e,l)$ (e est un exemplaire du livre l)

  1. $\forall e(Ex(e) \rightarrow \exists! y(L(y) \wedge Exde(e,l)))$
  2. $\forall e, aa, ab(Ex(e), A(aa), A(ab), Emp(aa,e), Emp(ab,e) \rightarrow aa = ab)$
  3. *Si deux exemplaire sont empruntes par un même abonnes, ils concernent des livres differents :*
     $\forall ea, eb(Ex(ea), Ex(eb), ea \neq eb, \exists a(A(a) \wedge Emp(a,ea) \wedge Emp(a,eb))$
     $\rightarrow \forall la, lb(Exde(ea,la), Exde(eb,lb), L(la)\, L(lb) \rightarrow la \neq lb))$

- *Formaliser les règles en théorie des ensembles On se donne :*
  3 ensembles :
  Ex (exemplaires),
  L (livres)
  A (abonnés)
  et 2 relations :
  Emp entre A et EX
  Exde entre Ex et L

1. $Exde \in Ex \rightarrow L$
2. $Emp \in Ex \nrightarrow A$
3. *Construisons la relation (e, (a,l)) reliant un exemplaire emprunté à son emprunteur et au livre qu'il référence. :*
   $f \stackrel{\text{def}}{=} \{(e,(a,l))/e \in E \wedge a \in A \wedge l \in L \wedge (e,a) \in Emp \wedge (e,l) \in Exde\}$
   . Cette relation est une fonction car un exemplaire ne peut etre relié à deux couples (aa,la) (ab,lb) différents du fait que Emp et Exde sont des fonctions de domaine Ex. Il suffit d'imposer que notre fonction soit de plus injective : 2 exemplaires differents ne peuvent etre relies a un même couple (a,l) ie ne peuvent etre des exemplaires du même livre emprunté par le même abonné :
   $f \in Ex \rightarrowtail\!\!\!\!\!\nrightarrow (A \times L)$

Exercice 6.7.5 *Pour formaliser des relations de parenté, on se donne les entités suivantes :*

- *3 ensembles : Humain, Homme, Femme.*
- *2 relations : epoux_de reliant reliant des femmes à leurs époux :*
  $epoux\_de \in Femme \rightleftharpoons Homme$
  et mere_de reliant des humains à leur mère :
  $mere\_de \in Humain \rightleftharpoons Femme$

1. *Formaliser les contraintes suivantes :*
   (a) *Personne ne peut être en même temps une femme et un homme*
   (b) *Cependant on est femme ou on est homme.*
   (c) *Les femmes n'ont qu'un seul époux.*
   (d) *Les mères sont des femmes mariées.*
2. *Définir les objets suivants en utilisant les entités précédentes :*
   (a) *epouse_de : l'épouse d'un homme est une femme dont cet homme est l'époux.*
   (b) *pere_de : Le père de quelqu'un est l'epoux de la mère.*
   (c) *conjoint_de : Le conjoint d'une femme est son époux, celui d'un homme est son épouse.*
   (d) *femmes_celibataires : une femme célibataire est une femme non mariée.*

Corrigé

1. (a) *Personne ne peut être en même temps une femme et un homme:*
   $\forall x(x \in Humain \rightarrow (x \in Homme \leftrightarrow x \notin Femme))$.
   ou : $Homme \cap Femme = \emptyset$.

(b) *Cependant on est femme ou on est homme:*
$\forall x(x \in Humain \rightarrow (x \in Homme \vee x \in Femme))$.
ou : $Homme \cup Femme = Humain$.

(c) *Les femmes n'ont qu'un seul époux :*
$epoux\_de \in Femme \nrightarrow Homme$

(d) *Les mères sont des femmes mariées:*
$ran(mere\_de) \subseteq dom(epoux\_de)$

2. *Définir les objets suivants en utilisant les entités précédentes :*

(a) *epouse_de : l'épouse d'un homme est une femme dont cet homme est l'époux :*
$epouse\_de \stackrel{\text{def}}{=} epoux\_de^{-1}$

(b) *pere_de : Le père de quelqu'un est l'epoux de la mère :*
$pere\_de \stackrel{\text{def}}{=} mere\_de; epoux\_de$

(c) *conjoint_de : Le conjoint d'une femme est son époux, celui d'un homme est son épouse :*
$conjoint\_de \stackrel{\text{def}}{=} epoux\_de \cup epouse\_de$

(d) *femmes_celibataires : une femme célibataire est une femme non mariée:*
$femme\_celibataire \stackrel{\text{def}}{=} femme - dom(epoux\_de)$

Exercice 6.7.6 *Elaborer un modèle mathématique permettant de formaliser le fonctionnement d'un hopital décrit par les contraintes suivantes :*

1. *Un hopital est constitué d'un certain nombre de chambre numérotées de 1 à n.*
2. *Dans chaque chambre on trouve 1 ou plusieurs lits.*
3. *Les chambres ne disposent pas toutes du même nombre de lits.*
4. *Une chambre peut etre dans l'un des états suivants : Vide, Pleine ou partiellement occuppée.*
5. *Les malades sont répertoriés selon leur catégorie : enfant, adulte homme, adulte femme.*
6. *Une chambre ne comporte que des malades d'une même catégorie*

Corrigé :
Les contraintes 2 et 3 expriment le fait que les chambres ont une capacité fixe pour chacune d'elles mais variable de l'une à l'autre.
De la description sommaire qui nous est fournie, il ressort qu'une chambre est déterminée par son numéro, sa capacité, son état, et qu'un malade est déterminé par sa catégorie et la chambre qu'il occuppe.
Il apparait que l'état d'une chambre est une notion qui peut se définir à partir des autres notions. Les entités de base que nous choisissons sont donc :

- *3 ensembles : Chambre , Malade, Categorie* $= \{H, F, E\}$.
- *4 relations :*
  $numero \in Chambre \rightleftharpoons 1..n$
  $capacite \in Chambre \rightleftharpoons \mathcal{N}$
  $chambre\_de \in Malade \rightleftharpoons Chambre$
  $categ \in Malade \rightleftharpoons Categorie$

Les contraintes s'expriment alors de la façon suivante :

1. *Un hopital est constitué d'un certain nombre de chambre numérotées de 1 à n :*
   numero est une bijection de Chambre vers $1..n$
2. *Dans chaque chambre on trouve 1 ou plusieurs lits et les chambres ne disposent pas toutes du même nombre de lits:*
   $capacite \in Chambre \rightarrow \mathcal{N}$
3. *Les malades sont répertoriés selon leur catégorie : enfant, adulte homme, adulte femme :*
   $categ \in Malade \rightarrow Categorie$
4. *contrainte implicite : les malades n'occuppent qu'une seule chambre :*
   $chambre\_de \in Malade \rightarrow Chambre$
5. *Une chambre peut être dans l'un des états suivants : Vide, Pleine ou partiellement occuppée.*
   On définit les 4 ensembles suivants :
   $occupant\_de \stackrel{\text{def}}{=} chambre\_de^{-1}$
   $chambre\_vide \stackrel{\text{def}}{=} \{c/c \in Chambre \wedge card(occupant\_de[\{c\}]) = 0\}$
   $chambre\_pleine \stackrel{\text{def}}{=} \{c/c \in Chambre \wedge card(occupant\_de[\{c\}]) = capacite(c)\}$
   $chambre\_part\_occuppee \stackrel{\text{def}}{=} \{c/c \in Chambre \wedge 0 < card(occupant\_de[\{c\}]) < capacite(c)\}$
   La contrainte devient :
   $Chambre = chambre\_vide \cup chambre\_pleine \cup chambre\_part\_occuppee$
6. *Une chambre ne comporte que des malades d'une même catégorie :*
   $occupant\_de; categ \in Chambre \nrightarrow categorie$

Exercice 6.7.7 *Il s'agit de construire un modèle partiel du fonctionnement d'une banque. Considérons les règles informelles suivantes.:*

- *Une banque gère pour ses clients deux types de comptes : les comptes* courant *et les comptes* épargne.
- *Chaque compte appartient à un unique client.*

- *Un client peut posséder plusieurs comptes courants mais un seul compte épargne.*

1. *Formaliser les règles précédentes en* Calcul des Prédicats
   Il s'agit donc de se donner des symboles de prédicats et d'énoncer les règles au moyen de ceux ci. l'utilisation du connecteur $\exists!$ *est autorisée.*

   - $\forall x(C(x) \rightarrow Courant(x) \vee Epargne(x))$
     $\forall x(Courant(x) \rightarrow C(x) \wedge \neg Epargne(x))$
     $\forall x(Epargne(x) \rightarrow C(x) \wedge \neg Courant(x))$

   - $\forall x(C(x) \rightarrow \exists! y(Client(y) \wedge possede(y, x)))$
   - $\forall x \forall y(Client(x), Epargne(y), possede(x, y) \rightarrow \exists! y(Epargne(y) \wedge possede(x, y)))$

2. *Formaliser les règles précédentes en* Théorie des Ensembles
   Il s'agit donc de se donner des symboles représentant des ensembles et d'énoncer les règles au moyen de ceux ci.

   - $Compte = Courant \cup Epargne.$
     $Courant \cap Epargne = \emptyset$
   - $possede \in Compte \nrightarrow Client$
   - $Epargne \lhd possede \in Compte \rightarrowtail Client$

Exercice 6.7.8 *Construire un modèle mathématique permettant de rendre compte des contraintes suivantes sur le fonctionnement d'une caisse de retraite.*

- *Il existe deux types de pensions : les pensions de droit propre et les pensions de droit dérivé.*
- *Tout assuré possède une et une seule pension de droit propre.*
- *Pour posséder une pension de droit dérivé, il faut que le conjoint de l'assuré possède une pension de droit propre et soit décédé.*
- *Um même assuré peut posséder plusieurs pensions de droit dérivé.*

Exercice 6.7.9 *Soit* $t \in 1..n \rightarrow A$. *Définir les ensembles suivants :*

1. *L'ensemble des indices des éléments de t qui sont supérieurs à leurs voisins de gauche.*
2. *L'ensemble des éléments de t qui apparaissent plus d'une fois dans t.*
3. *L'ensemble des éléments de t qui sont supérieurs à un objet* $a \in A$ *donné.*

Corrigé :

1. $\{i/i \in 2..n \wedge t(i-1) < t(i)\}$
2. $\{a/a \in A \wedge \exists i,j(i \neq j \wedge i \in 1..n \wedge j \in 1..n \wedge t(i) = a \wedge t(j) = a\}$
3. $\{b/b \in A \wedge \exists i(i \in 1..n \wedge t(i) > a)\}$

Exercice 6.7.10 *Soit $t \in 1..n \rightarrow A$, $t' \in seq(A)$ et P un prédicat unaire. Exprimer les propriétés suivantes :*

1. *Tous les éléments de t vérifient P.*
2. *au moins un élément de t vérifient P.*
3. *x est un élément de t vérifiant P.*
4. *t' est une suite formée des éléments de t vérifiant P.*
5. *t est ordonnée.*
6. *t' est une permutation de t*
7. *t' est un tri de t.*

Corrigé :

1. $P_1(t) \stackrel{\text{def}}{=} \forall i(i \in 1..n \rightarrow P(t(i)))$
2. $P_2(t) \stackrel{\text{def}}{=} \exists i(i \in 1..n \wedge P(t(i)))$
3. $P_3(t,x) \stackrel{\text{def}}{=} \exists i(i \in 1..n \wedge t(i) = x \wedge P(x))$
4. $P_4(t,t') \stackrel{\text{def}}{=} ran(t') \subseteq ran(t) \wedge \forall x(x \in ran(t') \rightarrow P(x))$ *ou :*
   $P_4(t,t') \stackrel{\text{def}}{=} ran(t') = ran(t) \rhd \{x/x \in A \wedge P(x)\}$
5. $trie(t) \stackrel{\text{def}}{=} \forall i,\forall j(i \in dom(t) \wedge j \in dom(t) \wedge i < j \rightarrow t(i) \leq t(j))$
6. $Permut(t,t') \stackrel{\text{def}}{=} \exists s(s \in 1..n \rightarrowtail 1..n \wedge s \in 1..n \twoheadrightarrow 1..n \wedge t' = s;t)$
7. $tri(t,t') \stackrel{\text{def}}{=} permut(t,t') \wedge trie(t')$

Chapter 7

index

7.1 Calcul des prédicats avec égalité

7.1.1 Règles primitives et dérivées

$\dfrac{}{\Gamma, \boxed{A} \vdash \boxed{A}, \Delta}\ axiome$	$\dfrac{}{\Gamma, A, \neg A \vdash \Delta}\ contr$	
$\dfrac{}{\Gamma \vdash A \leftrightarrow A, \Delta}\ reflequiv$	$\dfrac{}{\Gamma \vdash a = a, \Delta}\ refl =$	
$\dfrac{\Gamma \vdash \boxed{A}, \Delta \qquad \Gamma \vdash \boxed{B}, \Delta}{\Gamma \vdash \boxed{A \wedge B}, \Delta}\ \wedge_d$	$\dfrac{\Gamma, \boxed{A}, \boxed{B} \vdash \Delta}{\Gamma, \boxed{A \wedge B} \vdash \Delta}\ \wedge_g$	
$\dfrac{\Gamma \vdash \boxed{A}, \boxed{B}, \Delta}{\Gamma \vdash \boxed{A \vee B}, \Delta}\ \vee_d$	$\dfrac{\Gamma, \boxed{A} \vdash \Delta \qquad \Gamma, \boxed{B} \vdash \Delta}{\Gamma, \boxed{A \vee B} \vdash \Delta}\ \vee_g$	
$\dfrac{\Gamma, \boxed{A} \vdash \Delta}{\Gamma \vdash \boxed{\neg A}, \Delta}\ \neg_d$	$\dfrac{\Gamma \vdash \boxed{A}, \Delta}{\Gamma, \boxed{\neg A} \vdash \Delta}\ \neg_g$	$\dfrac{\Gamma, \neg A \vdash \Delta}{\Gamma \vdash A, \Delta}\ abs$
$\dfrac{\Gamma, \boxed{A} \vdash \boxed{B}, \Delta}{\Gamma \vdash \boxed{A \to B}, \Delta}\ \to_d$	$\dfrac{\Gamma \vdash \boxed{A} \qquad \Gamma, \boxed{B} \vdash \Delta}{\Gamma, \boxed{A \to B} \vdash \Delta}\ \to_g$	$\dfrac{\Gamma A, A \to B, B \vdash \Delta}{\Gamma, A, A \to B \vdash \Delta}\ MP$
$\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \boxed{A}, \Delta}\ aff_d$	$\dfrac{\Gamma \vdash \Delta}{\Gamma, \boxed{A}, \vdash \Delta}\ aff_g$	
$\dfrac{\Gamma \vdash \boxed{A}, \boxed{A}, \Delta}{\Gamma \vdash \boxed{A}, \Delta}\ contr_d$	$\dfrac{\Gamma \boxed{A}, \boxed{A}, \vdash \Delta}{\Gamma, \boxed{A}, \vdash \Delta}\ contr_g$	
$\dfrac{\Gamma \vdash \boxed{A}, \Delta \qquad \Gamma, \boxed{A} \vdash \Delta}{\Gamma \vdash \Delta}\ cut$	$\dfrac{\Gamma, th \vdash \Delta}{\Gamma \vdash \Delta}\ cut_1(th)*$	(*) th est déja prouvé
$\dfrac{\Gamma, \boxed{P[t]} \vdash \Delta}{\Gamma, \boxed{\forall x P[x]} \vdash \Delta}\ \forall_g$	$\dfrac{\Gamma \vdash \boxed{P[t]}, \Delta}{\Gamma \vdash \boxed{\exists x P[x]}, \Delta}\ \exists_d$	
$\dfrac{\Gamma \vdash \boxed{P[y]}, \Delta}{\Gamma \vdash \boxed{\forall x P[x]}, \Delta}\ \forall_d(*)$	$\dfrac{\Gamma, \boxed{P[y]} \vdash \Delta}{\Gamma, \boxed{\exists x P[x]} \vdash \Delta}\ \exists_g(*)$	(*) y nouvelle variable

$\dfrac{(\Gamma \vdash \Delta)[O//E]}{\Gamma \vdash \Delta} \stackrel{\text{def}}{=} (O)*$ **(*) :** si $O \stackrel{\text{def}}{=} E$ est présente dans la bibliothèque de définitions.	$\dfrac{(\Gamma \vdash \Delta)[A//B]}{\Gamma \vdash \Delta} \leftrightarrow (A \leftrightarrow B)*$ (*) si $A \leftrightarrow B$ est un théorème ou hyp	$\dfrac{(\Gamma \vdash \Delta)[a//b]}{\Gamma \vdash \Delta} = (a = b)*$ (*) si $a = b$ est un théorème ou hyp

7.1.2 propriétés

double négation	$\vdash \neg\neg A \leftrightarrow A$
tiers exclu	$\vdash A \vee \neg A$
distributivité	$\vdash ((A \wedge B) \vee C) \leftrightarrow ((A \vee C) \wedge (B \vee C))$ $\vdash (A \vee B) \wedge C \leftrightarrow (A \wedge C) \vee (B \wedge C)$ $\vdash (A \rightarrow (B \wedge C)) \leftrightarrow (A \rightarrow B) \wedge (A \rightarrow C)$
associativité	$\vdash (A \vee (B \vee C)) \leftrightarrow ((A \vee B) \vee C)$ $\vdash (A \wedge (B \wedge C)) \leftrightarrow ((A \wedge B) \wedge C)$
commutativité	$\vdash (A \vee B) \leftrightarrow (B \vee A)$ $\vdash (A \wedge B) \leftrightarrow (B \wedge A)$
Loi de Morgan	$\vdash \neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$ $\vdash \neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$ $\vdash \neg(A \rightarrow B) \leftrightarrow A \wedge \neg B$
transitivité	$A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$
	$\vdash A \leftrightarrow A$ $\vdash \neg\neg P \leftrightarrow P$ $(P \rightarrow Q) \leftrightarrow (\neg P \vee Q)$ $\vdash (A \vee B) \leftrightarrow \neg(\neg A \wedge \neg B)$ $\vdash ((A \rightarrow B) \wedge A) \rightarrow B$ $\vdash ((A \wedge B) \rightarrow C) \rightarrow (A \rightarrow (B \rightarrow C))$ $\Gamma, A, \neg A \vdash \Delta$

$\vdash \forall x(P(x) \wedge Q(x)) \leftrightarrow \forall x P(x) \wedge \forall x Q(x)$
$\vdash \exists x(P(x) \vee Q(x)) \leftrightarrow \exists x P(x) \vee \exists x Q(x)$
$\vdash \forall x(P(x) \rightarrow Q(x)) \rightarrow (\forall x P(x) \rightarrow \forall x Q(x))$
$\vdash \forall x(\exists y Q(x,y) \rightarrow P(x)) \rightarrow \forall x \forall y(Q(x,y) \rightarrow P(x))$
$\vdash \neg\forall x P(x) \leftrightarrow \exists x \neg P(x)$
$\vdash \neg\exists x P(x) \leftrightarrow \forall x \neg P(x)$

symétrie	$\vdash \forall x \forall y(x = y \rightarrow y = x)$
transitivité	$\vdash \forall x \forall y \forall z(x = y \wedge y = z \rightarrow x = z)$
Un point$_\forall$	$\vdash \forall e(\forall x(x = e \rightarrow P(x)) \leftrightarrow P(e))$
Un point$_\exists$	$\vdash \forall e(\exists x(x = e \wedge P(x)) \leftrightarrow P(e))$

7.2 Théorie des listes et des entiers

7.2.1 axiomes

- 0 est un entier : $(pea_1) : N(0)$
- Si x est un entier, son successeur aussi : $(pea_2) : \forall x(N(x) \rightarrow N(s(x)))$
- 0 n'est le successeur de personne : (pea_3):
  $\forall x(N(x) \rightarrow \neg(s(x) = 0))$
- 2 entiers ayant même successeur sont égaux : (pea_4) :
  $\forall x \forall y(N(x), N(y), s(x) = s(y) \rightarrow x = y)$

- Le schéma de récurrence : (pea_5) : pour tout formule F :
  $(F(0) \wedge \forall y((N(y), F(y)) \rightarrow F(s(y)))) \rightarrow \forall x(N(x) \rightarrow F(x))$
- $[]$ est une liste : $(li_1) : L([])$
- $(li_2) : \forall x \forall l(N(x), L(l) \rightarrow L(x : l))$
- (li_3):
  $\forall x \forall l(N(x), L(l) \rightarrow \neg(x : l = []))$
- égalité de 2 listes : (li_4) :
  $\forall x \forall y \forall l_1 \forall l_2(N(x), N(y), L(l_1), L(l_2), x : l_1 = y : l_2 \rightarrow x = y \wedge l_1 = l_2)$
- Le schéma de récurrence : (rec) : pour tout formule F :
  $(F([]) \wedge \forall l((L(l) \wedge F(l)) \rightarrow \forall x(N(x) \rightarrow F(x : l)))) \rightarrow \forall l(L(l) \rightarrow F(l))$

$(+_0) : \forall y(N(y) \rightarrow 0 + y = y)$
$(+_1) : \forall x \forall y(N(x), N(y) \rightarrow s(x) + y = s(x + y))$
$(*_0) : \forall y(N(y) \rightarrow 0 * y = 0)$
$(*_1) : \forall x \forall y(N(x), N(y) \rightarrow s(x) * y = (x * y) + y)$
$(inf =) : \forall a \forall b(N(a), N(b) \rightarrow (a \leq b \leftrightarrow \exists n(b = n + a)))$ $(le_0) : le([] = 0)$
$(le_1) : \forall a \forall l(N(a), L(l) \rightarrow le(a : l) = s(le(l)))$

$(@_0) : \forall l(L(l) \rightarrow []@l = l)$
$(@_1) : \forall a \forall l \forall m(N(a), L(l), L(m) \rightarrow (a : l)@m = a : (l@m))$

$(pos_0) : \forall n(N(n) \rightarrow pos(n, []) = 0)$
$(pos_1) : \forall a \forall b \forall l(N(a), N(b), L(l) \rightarrow (a = b \rightarrow pos(a, b : l) = 1))$
$(pos_2) : \forall a \forall b \forall l(N(a), N(b), L(l) \rightarrow (a \neq b \rightarrow pos(a, b : l) = s(pos(a, l))))$
$(\in_1) : \forall x(x \notin [])$
$(\in_2) : \forall x \forall a \forall l(N(x), N(a), L(l) \rightarrow x \in a : l \leftrightarrow x = a \vee x \in l)$

7.2.2 règles dérivées

$$\frac{\Gamma \vdash F(0/x), \Delta \qquad \Gamma, N(y), F(y/x) \vdash F(s(y)/x), \Delta}{\Gamma, N(x) \vdash F(x), \Delta} \; Rec(x)*$$

(*) : y nouvelle variable

$$\frac{\Gamma \vdash F([]/x), \Delta \qquad \Gamma, L(y), F(y/x), N(a) \vdash F(a : y/x), \Delta}{\Gamma, L(x) \vdash F(x), \Delta} \; Rec(x)*$$

(*) : y et a nouvelles variables

7.2.3 Propriétés

$(0_o u_s) : \forall x(N(x) \to x = 0 \vee \exists n(N(n) \wedge x = s(n)))$
$(tot+) : \forall x \forall y(N(x), N(y) \to N(x + y))$
$(tot*) : \forall x \forall y(N(x), N(y) \to N(x * y))$
$(assoc_+) :\vdash \forall m \forall n \forall p(N(m), \mathcal{N}(n), \mathcal{N}(p) \to m + (n + p) = (m + n) + p)$
$(commut_+) : \forall x \forall y(N(x), N(y) \to x + y = y + x)$
$(assoc_*) :\vdash \forall m \forall n \forall p(N(m), \mathcal{N}(n), \mathcal{N}(p) \to m * (n * p) = (m * n) * p)$
$(commut_*) : \forall x \forall y(N(x), N(y) \to x * y = y * x)$
$(\leq_0) : \forall x(N(x) \to 0 \leq x)$
$(\leq_1) : \forall x \forall y(N(x), N(y), s(x) \leq s(y) \to x \leq y)$
$(trans_{\leq}) : \forall x \forall y \forall z(N(x), N(y), N(z), x \leq y, y \leq z \to x \leq z)$
$(totle) : \forall x(L(x) \to N(le(x)))$
$(tot@) : \forall x \forall y(L(x), L(y) \to L(x@y))$
$(totpos) : \forall x \forall y(N(x), L(y) \to N(pos(x, y)))$

$(assoc_@) :\vdash \forall m \forall n \forall p(L(m), L(n), L(p) \to m@(n@p) = (m@n)@p)$
$(p_4) : \forall x \forall y(L(x), L(y) \to le(x@y) = le(x) + le(y))$
$(p_5) :\vdash \forall x \forall l_1 \forall l_2(N(x), L(l_1), L(l_2), x \notin l_1 \to pos(x, l_1@l_2) = le(l_1) + pos(x, l_2))$

7.3 Théorie des Ensembles

7.3.1 axiomes

$$(ext) : \forall s \forall t(\forall x(x \in s \leftrightarrow x \in t) \to s = t)$$

$$(paire) : \forall a \forall s \forall t(a \in \{s, t\} \leftrightarrow a = s \vee a = t)$$

$$(reunion) : \forall a \forall s(a \in \bigcup_{x \in s} x \leftrightarrow \exists t(t \in s \wedge a \in t))$$

$$(parties) : \forall a \forall s(a \in \mathcal{P}(s) \leftrightarrow \forall x(x \in a \to x \in s))$$

$$(comp) : \forall a \forall s(a \in \{x | x \in s \wedge F(x)\} \leftrightarrow a \in s \wedge F(a))$$

$$(infini) : \exists x(\emptyset \in x \wedge \forall y(y \in x \to y \cup \{y\} \in x))$$

7.3.2 règles dérivées

$$\frac{\Gamma \vdash F(0) \qquad \Gamma, y \in \mathcal{N}, F(y) \vdash F(S(y))}{\Gamma, x \in \mathcal{N} \vdash F(x)} \; Rec(*)$$

(*) y non libre dans $\forall x F, \Gamma$

7.3.3 définitions

$$a \subseteq b \stackrel{\text{def}}{=} a \in \mathcal{P}(b)$$

$$a \cup b \stackrel{\text{def}}{=} \bigcup_{x \in \{a,b\}} x$$

$$a \cap b \stackrel{\text{def}}{=} \{x | x \in a \wedge x \in b\}$$

$$a - b \stackrel{\text{def}}{=} \{x | x \in a \wedge x \notin b\}$$

$$a \times b \stackrel{\text{def}}{=} \{(x,y) | (x,y) \in \mathcal{P}(\mathcal{P}(a \cup b)) \wedge x \in a \wedge y \in b\}$$

$$a \rightleftharpoons b \stackrel{\text{def}}{=} \mathcal{P}(a \times b)$$

$$dom(R) = \{x | x \in a \wedge \exists y (y \in b \wedge (x,y) \in R)\}$$

$$ran(R) = \{y | y \in b \wedge \exists x (x \in a \wedge (x,y) \in R)\}$$

$$s \lhd R \stackrel{\text{def}}{=} \{(x,y) | (x,y) \in R \wedge x \in s\}$$

$$R^{-1} \stackrel{\text{def}}{=} \{(y,x) | (y,x) \in b \times a \wedge (x,y) \in R\}$$

$$R;T \stackrel{\text{def}}{=} \{(x,z) | (x,z) \in a \times c \wedge \exists y ((x,y) \in R \wedge (y,z) \in T)\}$$

$$R \rhd p \stackrel{\text{def}}{=} \{(x,y) | (x,y) \in R \wedge y \in p\}$$

$$p \lhd R \stackrel{\text{def}}{=} \{(x,y) | (x,y) \in R \wedge x \notin p\}$$

$$R \rhd p \stackrel{\text{def}}{=} \{(x,y) | (x,y) \in R \wedge y \notin p\}$$

$$R[p] \stackrel{\text{def}}{=} ran(p \lhd R)$$

$$R \leftarrow S \stackrel{\text{def}}{=} (dom(S) \lhd R) \cup S$$

fonction	$a \mathrel{+\!\!\!\rightarrow} b \stackrel{\text{def}}{=}$	$\{r \mid r \in a \rightleftharpoons b \wedge \forall x (x \in dom(r) \rightarrow \exists! y ((x,y) \in r))\}$
fonct. totale	$a \rightarrow b \stackrel{\text{def}}{=}$	$\{f \mid f \in a \mathrel{+\!\!\!\rightarrow} b \wedge dom(f) = a\}$
injection	$a \rightarrowtail b \stackrel{\text{def}}{=}$	$\{f \mid f \in a \mathrel{+\!\!\!\rightarrow} b \wedge$ $\forall x, x' ((x \in dom(f) \wedge x' \in dom(f) \wedge x \neq x') \rightarrow f(x) \neq f(x'))\}$
injection totale	$a \rightarrowtail b \stackrel{\text{def}}{=}$	$\{f \mid f \in a \rightarrow b \wedge f \in a \rightarrowtail b\}$
surjection	$a \mathrel{+\!\!\!\rightarrow\!\!\!\!\rightarrow} b \stackrel{\text{def}}{=}$	$\{f \mid f \in a \mathrel{+\!\!\!\rightarrow} b \wedge ran(f) = b\}$
surj. totale	$a \twoheadrightarrow b \stackrel{\text{def}}{=}$	$\{f \mid f \in a \rightarrow b \wedge f \in a \mathrel{+\!\!\!\rightarrow\!\!\!\!\rightarrow} b\}$

$$0 \stackrel{\text{def}}{=} \emptyset$$

$$1 \stackrel{\text{def}}{=} \{0\} = \{\emptyset\}$$

$$2 \stackrel{\text{def}}{=} \{0,1\} = \{\{\emptyset\},\emptyset\}$$

$$S(x) \stackrel{\text{def}}{=} x \cup \{x\}$$

$$x < y \stackrel{\text{def}}{=} x \in y$$

$$Minimum(s,x) \stackrel{\text{def}}{=} \forall y(x \leq y)$$

$$Maximum(s,x) \stackrel{\text{def}}{=} \forall y(y \leq x)$$

$$min \stackrel{\text{def}}{=} \{(s,x)|s \subset \mathcal{N} \wedge x \in s \wedge Minimum(s,x)\}$$

$$max \stackrel{\text{def}}{=} \{(s,x)|s \subset \mathcal{N} \wedge x \in s \wedge Maximum(s,x)\}$$

addition	$(+_0) : \forall y(N(y) \rightarrow 0 + y = y)$ $(+_1) : \forall x \forall y(N(x), N(y) \rightarrow s(x) + y = s(x + y))$
multiplication	$(*_0) : \forall y(N(y) \rightarrow 0 * y = 0)$ $(*_1) : \forall x \forall y(N(x), N(y) \rightarrow s(x) * y = (x * y) + y)$
exponentiel	$(exp_0) : \forall m(N(n) \rightarrow m^0 = 1$ $(exp_1) : \forall m \forall n(m^{S(n)} = m \times (m^n)$
soustraction	$\{(m,n,a)|m,n,a \in \mathcal{N} \wedge n \leq m \wedge n + a = m\}$
division	$\{(m,n,a)|m,n,a \in \mathcal{N} \wedge m \neq 0 \wedge m \times a \leq n \wedge m < m \times S(a)\}$

$$1..n \stackrel{\text{def}}{=} \{x|x \in \mathcal{N} \wedge 1 \leq x \leq n\}$$

$$seq(A) \stackrel{\text{def}}{=} \{f|f \in \mathcal{N} \rightarrow\ A \wedge \exists n(n \in \mathcal{N} \wedge f \in 1..n \rightarrow\ A)\}$$

$$a : s \stackrel{\text{def}}{=} \{1 \mapsto a\} \cup (pred; s)$$

$$Fini(x) \stackrel{\text{def}}{=} \exists f \exists n(n \in \mathcal{N} \wedge f \in 1..n \twoheadrightarrow\ x \wedge f \in 1..n \rightarrowtail x)$$

$card(x) \stackrel{\text{def}}{=} \{(x,m)|x \subseteq A \wedge \exists n \exists f(n \in \mathcal{N} \wedge f \in 1..n \twoheadrightarrow\ x \wedge f \in 1..n \rightarrowtail x \wedge m = max(dom(f)))\}$ (A ensemble fini)

7.3.4 Propriétés

$(\subseteq) : \forall a \forall b(a \subseteq b \leftrightarrow \forall x(x \in a \rightarrow x \in b))$

reflexivité ($refl_{\subseteq}$)	$\forall s(s \subseteq s)$
transitivité ($trans_{\subseteq}$)	$\forall s,t,u(s \subseteq t, t \subseteq u \rightarrow s \subseteq u)$
anti symétrie ($asym_{\subseteq}$)	$\forall s,t(s \subseteq t, t \subseteq s \rightarrow s = t)$

$(\cup) : \forall a \forall b(a \cup b = \bigcup_{x \in \{a,b\}} x)$.
$(\cup_1) : \forall x \forall a \forall b(x \in a \cup b \leftrightarrow x \in a \vee x \in b)$

$(\cap) : \forall a \forall b (a \cap b = \{x | x \in a \wedge x \in b\})$.
$(\cap_1) : \forall x \forall a \forall b (x \in a \cup b \leftrightarrow x \in a \wedge x \in b)$.
$(dif) : \forall a \forall b (a - b = \{x | x \in a \wedge x \notin b\})$.
$(dif_1) : \forall x \forall a \forall b (x \in a - b \leftrightarrow x \in a \wedge x \notin b)$

$(couple) : \forall a \forall b ((a, b) = \{\{a\}, \{a, b\}\})$
$(\times) : \forall a \forall b (a \times b = \{(x, y) | (x, y) \in \mathcal{P}(\mathcal{P}(a \cup b)) \wedge x \in a \wedge y \in b\})$
$(\times_1) : \forall x \forall y \forall a \forall b (x \in a \times b \leftrightarrow (x, y) \in \mathcal{P}(\mathcal{P}(a \cup b)) \wedge x \in a \wedge y \in b)$

commutativité	$(com_\cup) : \forall a \forall b (a \cup b = b \cup a)$ $(com_\cap) : \forall a \forall b (a \cap b = b \cap a)$
associativité	$(ass_\cup) : \forall a \forall b ((a \cup b) \cup c = a \cup (b \cup c))$ $(ass_\cap) : \forall a \forall b \forall c ((a \cap b) \cap c = a \cap (b \cap c))$
distributivité	$(dist_1) : \forall a \forall b \forall c ((a \cup b) \cap c = (a \cap c) \cup (b \cap c))$ $(dist_2) : \forall a \forall b \forall c ((a \cap b) \cup c = (a \cup c) \cap (b \cup c))$ $(dist_3) : \forall a \forall b \forall c ((a \cup b) - c = (a - c) \cup (b - c))$ $(dist_4) : \forall a \forall b \forall c ((a \cap b) - c = (a - c) \cap b)$ $(dist_5) : \forall a \forall b \forall c (c - (a \cup b) = (c - a) \cap (c - b))$ $(dist_6) : \forall a \forall b \forall c (c - (a \cap b) = (c - a) \cup (c \cap b))$
	$(dist_7) : \forall a (a \cup a = a)$ $(dist_8) : \forall a (a \cap a = a)$
absorption	$(abs_1) : \forall a \forall b (a \cap (a \cup b) = a)$ $(abs_2) : \forall a \forall b (a \cup (a \cap b) = a)$ $(abs_3) : \forall a (\emptyset \cap a = \emptyset)$ $(abs_4) : \forall a (a \cup \emptyset = a)$
ensemble vide	$\forall a (a - a = \emptyset)$
inclusion	$(\subseteq \cup) : \forall a \forall b (a \subseteq (a \cup b))$ $(\subseteq \cap) : \forall a \forall b ((a \cap b) \subseteq a)$

$(rela) : \forall a \forall b(a \rightleftharpoons b = \mathcal{P}(a \times b))$
$(rela_1) : \forall R \forall a \forall b(R \in a \rightleftharpoons b \leftrightarrow \forall x \forall y((x,y) \in R \rightarrow x \in a \wedge y \in b))$

$(dom) : \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow dom(R) = \{x | x \in a \wedge \exists y(y \in b \wedge (x,y) \in R)\})$
$(dom_1) : \forall x \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow x \in dom(R) \leftrightarrow x \in a \wedge \exists y(y \in b \wedge (x,y) \in R))$

$(ran) : \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow ran(R) = \{y | y \in b \wedge \exists x(x \in a \wedge (x,y) \in R)\})$
$(ran_1) : \forall y \forall R \forall a \forall b(R \in a \rightleftharpoons b \rightarrow y \in ran(R) \leftrightarrow y \in b \wedge \exists x(x \in a \wedge (x,y) \in R))$

$(\lhd) : \forall s \forall R(s \lhd R = \{(x,y) | (x,y) \in R \wedge x \in s\})$
$(\lhd_1) : \forall x, \forall y \forall s \forall R(x \in s \lhd R \leftrightarrow (x,y) \in R \wedge x \in s)$

$(^{-1}) : \forall R \forall A \forall B(R \in A \rightleftharpoons B \rightarrow R^{-1} = \{(y,x) | (y,x) \in b \times a \wedge (x,y) \in R\})$
$(^{-1}_1) : \forall x \forall y \forall R \forall A \forall B(R \in A \rightleftharpoons B \rightarrow (y,x) \in R^{-1} \leftrightarrow (x,y) \in R)$

$(;) : \forall R \forall T \forall A \forall B \forall C(R \in A \rightleftharpoons B, T \in B \rightleftharpoons C \rightarrow$
$R;T = \{(x,z) | (x,z) \in a \times c \wedge \exists y((x,y) \in R \wedge (y,z) \in T)\})$
$(;_1) : \forall x \forall z \forall R \forall T \forall A \forall B \forall C(R \in A \rightleftharpoons B, T \in B \rightleftharpoons C \rightarrow$
$(x,z) \in R;T \leftrightarrow \exists y((x,y) \in R \wedge (y,z) \in T))$

$(\rhd) : \forall R \forall p(R \rhd p = \{(x,y) | (x,y) \in R \wedge y \in p\})$
$(\rhd_1) : \forall x \forall y \forall R \forall p((x,y) \in R \rhd p \leftrightarrow (x,y) \in R \wedge y \in p)$

$(\lhd\!\!\!\cdot) : \forall R \forall p(p \lhd\!\!\!\cdot R = \{(x,y) | (x,y) \in R \wedge x \notin p\})$
$(\lhd\!\!\!\cdot_1) : \forall x \forall y \forall R \forall p((x,y) \in p \lhd\!\!\!\cdot R \leftrightarrow (x,y) \in R \wedge x \notin p)$

$(\rhd\!\!\!\cdot) : \forall R \forall p(R \rhd\!\!\!\cdot p = \{(x,y) | (x,y) \in R \wedge y \notin p\})$
$(\rhd\!\!\!\cdot_1) : \forall x \forall y \forall R \forall p((x,y) \in R \rhd\!\!\!\cdot p \leftrightarrow (x,y) \in R \wedge y \notin p)$

$[] : \forall R \forall p(R[p] = ran(p \lhd R))$
$[]_1 : \forall y \forall R \forall p(y \in R[p] \leftrightarrow \exists x(x,y) \in R \wedge x \in p)$

$(fonctpa) : \forall a \forall b(a \mathrel{+\!\!\!\rightarrow} b = \{r | r \in a \rightleftharpoons b \wedge$
$\forall x(x \in dom(r) \rightarrow \exists! y((x,y) \in r))\})$
$(fonctpa_1) : \forall f \forall a \forall b(f \in a \mathrel{+\!\!\!\rightarrow} b \leftrightarrow$
$f \in a \rightleftharpoons b \wedge \forall x(x \in dom(f) \rightarrow \exists! y((x,y) \in f)))$

$(fonct) : \forall a \forall b(a \rightarrow b = \{f | f \in a \mathrel{+\!\!\!\rightarrow} b \wedge dom(f) = a\})$
$(fonct_1) : \forall f \forall a \forall b(f \in a \rightarrow b \leftrightarrow f \in a \mathrel{+\!\!\!\rightarrow} b \wedge dom(f) = a)$

$(injpa) : \forall a \forall b(a \rightarrowtail b = \{f | f \in a \mathrel{+\!\!\!\rightarrow} b \wedge$
$\forall x, x'(x \in dom(f), x' \in dom(f), x \neq x' \rightarrow f(x) \neq f(x'))\})$
$(injpa_1) : \forall f \forall a \forall b(f \in a \rightarrowtail b \leftrightarrow f \in a \mathrel{+\!\!\!\rightarrow} b \wedge$
$\forall x, x'(x \in dom(f), x' \in dom(f), x \neq x' \rightarrow f(x) \neq f(x')))$

$(inj) : \forall a \forall b(a \rightarrowtail b \stackrel{\text{def}}{=} \{f | f \in a \rightarrow b \wedge f \in a \rightarrowtail b\})$
$(inj_1) : \forall f \forall a \forall b(f \in a \rightarrowtail b \leftrightarrow f \in a \rightarrow b \wedge f \in a \rightarrowtail b)$

$(surjpa) : \forall a \forall b(a \mathrel{+\!\!\!\rightarrow\!\!\!\!\rightarrow} b = \{f | f \in a \mathrel{+\!\!\!\rightarrow} b \wedge ran(f) = b\})$
$(surjpa_1) : \forall f \forall a \forall b(f \in a \leftrightarrow f \in a \mathrel{+\!\!\!\rightarrow} b \wedge ran(f) = b)$

$(surj) : \forall a \forall b(a \twoheadrightarrow b = \{f | f \in a \rightarrow b \wedge f \in a \mathrel{+\!\!\!\rightarrow\!\!\!\!\rightarrow} b\})$
$(surj_1) : \forall f \forall a \forall b(f \in a \leftrightarrow f \in a \rightarrow b \wedge f \in a \mathrel{+\!\!\!\rightarrow\!\!\!\!\rightarrow} b)$

$(commut_+) : \forall m \forall n(N(n), N(m) \rightarrow$
$m + n = n + m$
$(assoc_+) : \forall m \forall n \forall p(N(n), N(m), N(p) \rightarrow$
$m + (n + p) = (m + n) + p$
$(commut_*) : \forall m \forall n(N(n), N(m) \rightarrow$
$m \times n = n \times m$
$(assoc_*) : \forall m \forall n \forall p(N(n), N(m), N(p) \rightarrow$
$m \times (n \times p) = (m \times n) \times p$
$(dist1) : \forall m \forall n \forall p(N(n), N(m), N(p) \rightarrow$
$m \times (n + p) = (m \times n) + (m \times p)$
$(dist2) : \forall m \forall n \forall p(N(n), N(m), N(p) \rightarrow$
$m^{n+p} = m^n \times m^p$
$(assoc_e xp) : \forall m \forall n \forall p(N(n), N(m), N(p) \rightarrow$
$m^{n \times p} = m^{np}$

$s \in seq(A) \leftrightarrow s = [\,] \vee (\exists y \exists a(y \in seq(A) \wedge a \in A \wedge s = y : a))$

Bibliography

[1] J.R.Abrial *The B Book* Cambridge University Press, 96

La bible de la méthode B, faite par son auteur.

[2] R.C. Backhouse *Construction et vérification de programmes* Masson 89

Introduction assez informelle à la preuve de programmes impératifs.

[3] Jon Barwise
An Introduction to First Order Logic
Handbook of Mathematical Logic pp 5-46, ed J. Barwise, North Holland

[4] René Cori, Daniel Lascar
Logique mathématique Masson 1994

Un cours complet de logique mathématique.

[5] Paul Gochet, Pascal Gribomont
Logique, Méthodes pour l'informatique fondamentale, t.1 Hermes, 90-91

Les chapitres 3 à 7 couvrent le programme. Plus particulièrement les chapitres 3.1, 3.5, 4.1, tout le 5 et le 6.2.

[6] P. Halmos
Naive Set Theory D. Van Nostrand, 1960 traduction francaise chez Gauthier villars

[7] Carroll Morgan *Programming from specifications*
Prentice-Hall, International series on Computer Science, C.A.R.Hoare series editor

Excellente présentation de la théorie du raffinement. Nombreux exemples.

[8] John Rushby *Formal methods and certification of critical systems, Appendix A*
Technical report, www.csl.sri.com/csl-93-7.html

Introduction assez intuitive à la logique vue en cours.