Logique

David Baelde, ENS Rennes, L3 SIF

Notes de cours en chantier permanent, version du 22 février 2024.

Table des matières

Chapitre 1

Préliminaires

Un des objectifs de la formalisation de la logique est d'éviter les raisonnements incorrects. Un exemple célèbre est le paradoxe de Russell : si l'on peut former $R = \{x \mid x \notin x\}$ on déduit $R \in R \Leftrightarrow R \notin R$, puis $\bot$. Même si nous n'avons pas pour objectif de bien poser la théorie des ensembles, cet exemple doit nous inciter à questionner les raisonnements et les définitions que nous nous autorisons à faire.

1.1 Définitions inductives

Considérons la définition inductive d'élément accessible par rapport à l'ordre canonique dans $\mathbb{N}$, puis $\mathbb{Z}$. Comment comprendre et justifier ces définitions ?

Proposition 1.1.1. *Soit E un ensemble. Soit $f : 2^E \to 2^E$ une fonction monotone sur les parties de E, c'est à dire qu'on a, pour tous $X, Y \subseteq E$ tels que $X \subseteq Y$, $f(X) \subseteq f(Y)$. La fonction f admet un* plus petit point fixe *: il existe X tel que $X = f(X)$ et tout autre ensemble ayant cette propriété contient X.*

Démonstration. On définit X comme l'intersection des ensembles Y tels que $f(Y) \subseteq Y$.

- Montrons que $f(X) \subseteq X$. Soit Y tel que $f(Y) \subseteq Y$. On a $X \subseteq Y$ par définition de X, donc $f(X) \subseteq f(Y)$ puis $f(X) \subseteq Y$. Comme on a $f(X) \subseteq Y$ pour tout Y satisfaisant $f(Y) \subseteq Y$, on en déduit $f(X) \subseteq X$ par définition de X.
- On montre ensuite que $X \subseteq f(X)$. En fait, par le point précédent et la monotonicité de f on a $f(f(X)) \subseteq f(X)$, d'où $X \subseteq f(X)$ par définition de X.
- Il est enfin clair que $X \subseteq Y$ pour tout Y tel que $f(Y) \subseteq Y$, a fortiori $X \subseteq Y$ pour tout point fixe Y. □

Ce plus petit point fixe contient les itérées de f à partir de l'ensemble vide : $\emptyset \subseteq X$, puis $f(\emptyset) \subseteq f(X) \subseteq X$, $f^2(\emptyset) \subseteq X$, etc. Mais ces inclusions sont en général strictes – considérer par exemple l'accessibilité sur $\mathbb{N} \cup \{\omega\}$ avec $i < \omega$ pour tout $i \in \mathbb{N}$.

Quand on définit inductivement un prédicat sur E, qu'on peut voir comme un sous-ensemble de E (le sous-ensembles de valeurs pour lesquelles p est vrai), on dit en fait que p est le plus petit point fixe de la fonction associée à la définition, qui doit être monotone. C'est le cas pour l'accessibilité :

$$f(X) = \{n \mid \forall m.\ m < n \Rightarrow m \in X\}$$

Exemple 1.1.1. *On peut tout à fait définir inductivement un prédicat p sur un ensemble E par "$p(x)$ si $p(x)$" : c'est une façon détournée de décrire le prédicat faux, puisque le plus petit point fixe de la fonction $f : X \mapsto X$ est l'ensemble vide.*

Il est important de comprendre ce qui n'est pas une définition inductive, i.e. ce que ne permet pas le théorème précédent.

Exemple 1.1.2. *On ne peut pas définir inductivement p par "$p(x)$ si $\neg p(x)$" – ce qui nous ramènerait au paradoxe de Russell. En effet, la fonction associée est $f : X \mapsto \{x \in E \mid x \notin X\}$ mais n'est pas monotone.*

Pour aller plus loin, on pourrait considérer notre théorème de point fixe dans une structure ordonnée autre que les ensembles : c'est le théorème de Knaster-Tarski. On pourrait aussi explorer comment cette construction des définitions inductives justifie le raisonnement par induction et la construction de fonctions (ou relations) par induction.

Chapitre 2

Logique propositionnelle

2.1 Syntaxe

On se donne un ensemble $\mathcal{P}$ de *variables propositionnelles.*

Définition 2.1.1. *L'ensemble des formules du calcul propositionnel est défini inductivement comme suit :*
- $\bot$ *et* $\top$ *sont des formules ;*
- *les variables propositionnelles sont des formules ;*
- *si* ϕ *est une formule, alors* $\neg\phi$ *aussi ;*
- *si* ϕ *et* ψ *sont des formules, alors* $\phi \wedge \psi$, $\phi \vee \psi$ *et* $\phi \Rightarrow \psi$ *aussi.*

Mais sur quel ensemble de départ se place-t-on quand on crée cette définition ? On peut considérer qu'on se place sur un ensemble d'arbres étiquetés, ou de graphes.

Cette définition peut être vue comme la définition d'un type de données algébrique en OCaml :

```
type form = Bot | Top | Var of var | Not of form | ...
```

C'est une intuition utile, mais la variante OCaml permet des objets récursifs qui sont exclus ici : par exemple, `let rec x = Not x`.

Certains considèrent que les formules sont des suites de symboles, mais il faut alors ajouter des symboles parenthèses dans la définition ci-dessus pour assurer une lecture unique : on préfère une vision de plus haut niveau. Cela n'empêche pas d'écrire nos formules-arbres en utilisant une notation linéaire, avec des règles de priorité pour nos opérateurs et des parenthèses pour désambiguer :
- On considèrera que les trois connecteurs binaires sont associatifs à droite.
- On considèrera que le $\wedge$ lie plus fortement que $\vee$, qui lie plus fortement que $\Rightarrow$.

Ainsi, $A \vee B \wedge C \vee D$ correspond à $A \vee ((B \wedge C) \vee D)$.

2.2 Sémantique

Pour donner un sens à nos formules, on utilise la notion d'interpétation : une interprétation est une fonction $\mathcal{I} : \mathcal{P} \to \{0, 1\}$.

Définition 2.2.1. *La relation de satisfaction entre les interprétations et les formules, notée* $\mathcal{I} \models \phi$*, est définie par induction sur les formules :*
- $\mathcal{I} \models \top$ *et* $\mathcal{I} \not\models \bot$ *;*
- $\mathcal{I} \models A$ *ssi* $\mathcal{I}(A) = 1$ *;*
- $\mathcal{I} \models \phi \wedge \psi$ *ssi* ($\mathcal{I} \models \phi$ *et* $\mathcal{I} \models \psi$) *;*
- $\mathcal{I} \models \phi \vee \psi$ *ssi* ($\mathcal{I} \models \phi$ *ou* $\mathcal{I} \models \psi$) *;*
- $\mathcal{I} \models \phi \Rightarrow \psi$ *ssi* ($\mathcal{I} \models \phi$ *implique* $\mathcal{I} \models \psi$) *;*

— $\mathcal{I} \models \neg\phi$ *ssi* $\mathcal{I} \not\models \phi$.

Il faut bien noter ici que la relation de satisfaction n'est pas définie inductivement.

À partir de la relation de satisfaction on peut définir une fonction d'interprétation, qu'on peut noter sans confusion $\mathcal{I}(\phi)$ et définie par $\mathcal{I}(\phi) = 1$ si $\mathcal{I} \models \phi$ et 0 sinon. Ces deux définitions peuvent être posées dans l'autre sens : si l'on définit d'abord $\mathcal{I}$, on définit ensuite $\mathcal{I} \models \phi$ par $\mathcal{I}(\phi) = 1$.

Dans ce cours, on utilise en priorité la notation $\models$ qui est de toute façon standard, et en fait plus versatile en logique.

On dérive à partir de la notion de satisfaction de nombreuses notions incontournables.

Définition 2.2.2. *L'ensemble des modèles d'une formule ϕ est l'ensemble des interprétations $\mathcal{I}$ qui satisfont ϕ, i.e. $\mathcal{I} \models \phi$.*

Définition 2.2.3. *Une formule ϕ est* valide *si elle est satisfaite par toute interprétation : $\mathcal{I} \models \phi$ pour tout $\mathcal{I}$.*

Définition 2.2.4. *La formule ψ est* conséquence logique *de la formule ϕ, ce qu'on note $\phi \models \psi$, si tous les modèles de ϕ sont aussi des modèles de ψ :*

$$\mathcal{I} \models \phi \text{ implique } \mathcal{I} \models \psi \text{ pour tout } \mathcal{I}$$

Définition 2.2.5. *Deux formules sont* logiquement équivalentes *si chacune est conséquence logique de l'autre, i.e. elles ont les mêmes modèles. L'équivalence logique de ϕ et ψ est notée $\phi \equiv \psi$.*

Plusieurs équivalentes logiques remarquables, quelles que soient ϕ et ψ : $\neg\neg\phi \equiv \phi$, $\neg(\phi \wedge \psi) \equiv \neg\phi \vee \neg\psi$, etc.

2.3 Formes normales

Proposition 2.3.1. *Pour toute formule ϕ il existe une formule logiquement équivalente ψ n'utilisant pas le connecteur $\Rightarrow$ et dans laquelle la négation n'est utilisée que sur des variables.*

Exemple 2.3.1. $\neg(\neg P \Rightarrow Q)$ *sera transformé en* $\neg P \wedge \neg Q$.

Démonstration.

— Première possibilité : par induction sur la taille (ou la hauteur de) la formule. Si la formule n'est pas construite (à toplevel) avec une négation ou une implication, on conclut aisément par hypothèse d'induction. Sinon, on se ramène à une formule équivalente avant d'invoquer l'hypothèse d'induction : par exemple pour ϕ de la forme $\neg(\phi_1 \Rightarrow \phi_2)$ on appliquera l'hypothèse d'induction sur ϕ_1 et $\neg\phi_2$ pour obtenir ϕ'_1 et ϕ'_2 et on conclut avec $\phi'_1 \wedge \phi'_2$.

— Deuxième possibilité, équivalente : on définit une fonction / un algorithme qui calcule récursivement la transformation, on constate sa correction partielle et on vérifie aisément qu'il termine.

— Troisième possibilité : on oriente les équivalences logiques utiles comme des règles de réécriture, qu'on applique autant que possible sans stratégie particulière ; on vérifie qu'une quantité décroit à chaque réécriture et qu'on a la forme normale attendue quand on ne peut plus réécrire.
Ici, une quantité naturelle est la somme des tailles des sous-formules commençant par une négation ou une implication. □

Dans tous les cas il y a unicité [1] de la forme normale obtenue, qui a une taille linéaire en la taille de la formule de départ, et peut être calculée en temps polynomial.

1. Il n'y a pas unicité dans l'absolu : l'énoncé de notre proposition permet de donner $\bot$ comme forme normale pour $\neg\neg P \wedge \neg P$, mais nos transformations donnent $P \wedge \neg P$.

Définition 2.3.1. *Un littéral est une variable ou la négation d'une variable.*

Proposition 2.3.2 (Forme normale conjonctive)**.** *Toute formule est logiquement équivalente à une conjonction de disjonctions de litéraux.*

Proposition 2.3.3 (Forme normale disjonctive)**.** *Toute formule est logiquement équivalente à une disjonction de conjonctions de litéraux.*

2.4 Le problème de la satisfaisabilité

On a déja vu que SAT, le problème général de satisfaisabilité, est NP-complet : c'est le théorème de Cook. Si on rentrait dans les détails de la preuve, on verrait qu'on a en fait (modulo ajustements mineurs) une preuve que CNF-SAT est NP-complet. Il n'est pas utile de le détailler car on va faire mieux plus bas.

Il n'est pas intéressant de restreindre SAT en fixant l'ensemble des variables utilisables — pourquoi ? Par contre, il est intéressant de voir ce qu'il se passe si on restreint la forme syntaxique des formules :

— Il est clair que DNF-SAT se résout en temps polynomial.
— On verra plus tard que 2-SAT est polynomial et même linéaire.
— En fait, 3-SAT est encore NP-complet. Cela dérive du résultat suivant.

Définition 2.4.1. *On dit que deux formules sont equi-satisfaisables quand la satisfaisabilité de l'une est équivalente à la satisfaisabilité de l'autre. Autrement dit, soit les deux sont satisfaisables, soit aucune ne l'est.*

Proposition 2.4.1 (Transformation de Tseitin)**.** *Pour toute formule ϕ on peut calculer en temps polynomial une formule équisatisfaisable ψ (de taille linéaire en $|\phi|$) en forme 3-CNF.*

Démonstration (à compléter). Soit ϕ une formule de départ. On se donne des nouvelles variables P_ψ pour toute sous-formule de ϕ, y compris ϕ.

On considère la formule $T(\phi) = P_\phi \wedge \bigwedge_{\psi \text{ sous-formule}} t(\psi)$ où les formules $t(\psi)$ sont définies comme suit :

— ...

On vérifie qu'on a bien construit une 3-CNF. On constate de plus qu'il existe une borne k sur la taille de tous les $t(\psi)$, donc la taille de notre formule est linéaire en la taille de ϕ.

Vérifions que la formule construite est équisatisfaisable à ϕ :

— Si on a un modèle $\mathcal{I} \models \phi$ on l'étend en $\mathcal{I}'$ tel que $\mathcal{I}'(P_\psi) = 1$ ssi $\mathcal{I} \models \psi$. On vérifie que $\mathcal{I}' \models T(\phi)$.
— Si on a un modèle $\mathcal{J} \models T(\phi)$ on prend $\mathcal{J}'$ sa restriction aux variables de ϕ et on vérifie $\mathcal{J}' \models \phi$.

□

2.5 Compacité

Théorème 2.5.1. *Un ensemble de formules E est insatisfaisable ssi il existe un sous-ensemble fini $F \subseteq_{\mathsf{fin}} E$ insatisfaisable.*

Démonstration (grandes lignes). La direction intéressante est celle où l'on montre qu'un ensemble insatisfaisable admet un sous-ensemble insatisfaisable. On la démontre par la méthode des arbres sémantiques, sous l'hypothèse que $\mathcal{P}$ est dénombrable. On observera d'abord que l'arbre élagué d'un ensemble de formules a une branche infinie ssi l'ensemble est satisfaisable. L'arbre élagué d'un ensemble insatisfaisable est donc sans branche infinie et, par le lemme de König, il est donc fini. Le sous-ensemble fini recherché est obtenu en collectant les formules décorant les feuilles de cet arbre. □

On verra diverses applications de la compacité : pour montrer que certains ensembles de modèles ne peuvent être axiomatisés ; pour montrer que des problèmes encodables dans SAT satisfont une propriété de compacité analogue ; mais surtout, la compacité nous dit qu'on peut toujours établir une conséquence logique à partir d'un sous-ensemble fini, ce qui est nécessaire pour obtenir des systèmes de preuves finies.

2.6 Résolution

La preuve par résolution travaille sur des formules en CNF, ou plutôt sur le résultat de l'éclatement des conjonctions de telles formules. C'est suffisant pour traiter le problème de la satisfaisabilité : pour décider la satisfaisabilité d'un ensemble de formules, on peut mettre chaque formule en CNF, éclater les conjonctions, et décider la satisfaisabilité de l'ensemble résultant.

2.6.1 Définitions

Un littéral est une variable, ou la négation d'une variable. On définit la négation d'un littéral, notée $\overline{L}$, comme suit :

$$\overline{P} = \neg P \qquad \overline{\neg P} = P$$

Une clause est alors une disjonction de littéraux ; la disjonction vide est $\bot$. Plus précisément, on considère les clauses comme des multi-ensembles de littéraux. Ainsi $C \vee C'$ doit être vu comme une union multi-ensembliste, et $\bot$ est le multi-ensemble vide.

Le système de preuve par résolution est défini par les deux règles suivantes, appelées Résolution et Factorisation :

$$\frac{C \vee L \quad \overline{L} \vee C'}{C \vee C'}\ R \qquad \frac{C \vee L \vee L}{C \vee L}\ F$$

On dit qu'une clause C est dérivable par résolution à partir d'un ensemble de clauses E quand il existe une arbre de dérivation avec des feuilles dans E, C en conclusion, et utilisant nos deux règles. On note cela $E \vdash_{RF} C$.

Théorème 2.6.1 (Correction)**.**
Pour tous E, C tels que $E \vdash_{RF} C$, on a $E \models C$.

Démonstration. Il suffit de vérifier que, pour chacune de nos règles, la conclusion est conséquence logique des prémisses. □

La réciproque s'appellerait complétude, mais n'est vraie que quand C est la clause vide : on peut dériver la clause vide à partir de tout ensemble de clauses insatisfaisable, et c'est déja bien utile.

Théorème 2.6.2 (Complétude réfutationnelle)**.**
Pour tout E tel que $E \models \bot$, on a $E \vdash_{RF} \bot$.

Démonstration (grandes lignes). On reprend les outils de la preuve du théorème de compacité. Étant donné un ensemble E, on définit l'ensemble des clauses déductibles à partir de E par résolution :

$$E^* = \{C \mid E \vdash_{RF} C\}$$

On montre aisément que $E^{**} = E^*$.

Supposons maintenant E insatisfaisable. On a aussi E^* insatisfaisable car $E \subseteq E^*$. On montre qu'en plus de cela, l'arbre sémantique de E^* est restreint à la racine, par l'absurde : si l'arbre a un noeud interne, on considère un noeud interne de profondeur maximale, qui a donc pour filles deux feuilles, chacune étant falsifiée par une clause de E^* ; par résolution et factorisation à partir de ces clauses, on obtient une clause de E^* qui falsifie le noeud interne, qui devrait donc être une feuille. L'arbre étant réduit à la racine, on a $\bot \in E^*$, ce qui conclut la démonstration. □

2.6.2 Variantes et stratégies

La résolution fournit un outil pour déterminer si un ensemble de clauses E est satisfaisable [2]. Il suffit de générer toutes les conséquences possibles de E par les règles de résolution et factorisation. Si ce processus finit par engendrer la clause vide, E est insatisfaisable par correction des règles. Sinon, si l'on arrive à générer toutes les conséquences en temps fini et que celles-ci ne comportent pas la clause vide, c'est que E est satisfaisable par complétude réfutationnelle.

Toute la difficulté est d'atteindre en temps fini un ensemble *saturé* par déduction (i.e. un F tel que $F^* = F$). Une utilisation naïve des règles de résolution et factorisation va rarement permettre cela. De plus, même en mettant de côté le problème de la terminaison, les règles standard permettent de nombreuses inférences qu'on peut chercher à limiter (sans pour autant perdre la complétude réfutationnelle) afin d'obtenir la saturation plus rapidement.

Tout ceci motive l'étude des preuves par résolution pour déterminer des restrictions sur les preuves qui vont permettre de limiter les clauses engendrées sans pour autant perdre la possibilité de dériver la clause vide à partir d'un ensemble insatisfaisable. On donne ci-dessous quelques résultats allant dans ce sens ; certains se démontrent par transformation de preuves, d'autres par des observations sémantiques.

Variante ensembliste

Il est possible de considérer les clauses comme des ensembles de littéraux, i.e. silencieusement identifier $C \vee L \vee L$ et $C \vee L$. Dans ce cas, la règle de factorisation est inutile, et la règle de résolution seule est réfutationnellement complète. Cela se démontre aisément en adaptant la preuve de complétude réfutationnelle vue plus haut, où les clauses étaient des multi-ensembles.

Ce résultat nous indique qu'on va travailler dans un ensemble de clauses fini : toute clause engendrée contiendra au plus $2n$ littéraux, où n est le nombre de variables de l'ensemble de clauses de départ. On verra plus bas qu'il n'est pas utile de considérer les clauses contenant un littéral et son opposé, ce qui permet de borner le nombre de littéraux d'une clause par n. Dans les deux cas, on s'assure ainsi de la terminaison de la saturation [3].

Résolution ordonnée

On suppose une énumération $(X_i)_{i\in\mathbb{N}}$ des variables propositionnelles, qui peut être choisie arbitrairement. On l'étend en un ordre strict sur les littéraux : $L < L'$ si L est construit sur la variable X_i et L' sur X_j avec $i < j$. Par exemple, $X_2 < \neg X_4$.

On restreint les règles de résolution et de factorisation à ne s'appliquer que si le littéral concerné est maximal dans la clause :

$$\frac{C \vee L \quad \overline{L} \vee C'}{C \vee C'}\ RO \qquad \frac{C \vee L \vee L}{C \vee L}\ FO$$

à la condition que L soit maximal dans $C \vee L$ et $\overline{L}$ maximal dans $\overline{L} \vee C'$.

En reprenant la preuve précédente on peut montrer que la résolution ordonnée est encore réfutationnellement complète.

Exemple 2.6.1. *La contrainte de maximalité peut drastiquement limiter les résolutions possibles. Sur l'ensemble de clauses* $\neg X_1 \vee X_2$, $\neg X_2 \vee X_3$, $\ldots$, $\neg X_n \vee X_{n+1}$, *on ne pourra faire aucune résolution ordonnée. Cela nous indique tout de suite que cet ensemble est satisfaisable. Si l'on ajoute les clauses* X_1 *et* $\neg X_{n+1}$, *on va pouvoir dériver la clause vide, mais il n'y a essentiellement qu'une façon de le faire : là où on résolution normale on pouvait résoudre* $\neg X_k \vee X_{k+1}$ *et* $\neg X_{k+1} \vee X_{k+2}$ *pour tout* k, *il nous faudra ici commencer par*

2. Cette méthode n'est pas tellement intéressante pour le cas propositionnel, où l'algorithme DPLL (et ses optimisations comme CDCL) fonctionne mieux. L'intérêt de la résolution est qu'elle s'adapte bien au premier ordre, contrairement aux algorithmes de recherche de modèles à la DPLL.

3. Au premier ordre, aucune des observations faites ici ne nous permettra de garantir la terminaison, mais en pratique ces optimisations nous seront bien utiles.

résoudre $\neg X_n \vee X_{n+1}$ et $\neg X_{n+1}$, puis $\neg X_{n-1} \vee X_n$ et $\neg X_n$, etc. de façon linéaire. Il est intéressant de considérer comment les dérivations possibles changent quand on fait un autre choix de numérotation des variables.

Élimination des clauses triviales

On dit qu'une clause est *triviale* quand elle comporte un littéral et son opposé. On a le résultat suivant, pour tous les systèmes de dérivation considérés dans ce chapitre :

Si E est insatisfaisable, alors il existe une dérivation de la clause vide à partir de E dans laquelle aucune clause n'est triviale.

On peut le démontrer, dans le cas de la résolution standard sur des clauses ensemblistes (donc sans factorisation), simplement en étudiant la forme des preuves, et en les transformant :

— On remarque d'abord qu'on peut éliminer les clauses triviales quitte à insérer dans la dérivation des utilisations de la règle d'affaiblissement (voir plus bas).
— On élimine ensuite les affaiblissement, pour obtenir une dérivation d'une clause plus forte : si $E \vdash_{RA} C$ (dérivation par résolution et affaiblissement) alors $E \vdash_R C'$ pour un $C' \subseteq C$.
— Dans le cas où C est vide, le résultat précédent nous donne la complétude réfutationnelle.

Ce résultat signifie que, quand on génère les conséquences d'un ensemble de clauses par déduction pour déterminer si la clause vide est dérivable, on peut ignorer les clauses triviales qui seraient générées. Ignorer les clauses triviales veut aussi dire qu'on ne va jamais résoudre une clause contre elle-même.

Stratégie *set of support*

Soit un ensemble de clauses E de la forme $E' \cup E''$ avec E' satisfaisable. La stratégie *set-of-support* exige qu'on ne considère que des dérivations dans lesquelles toute sous-dérivation (sous-arbre) non restreint à une feuille comporte au moins une feuille dans E''.

Pour comprendre l'intérêt d'une telle stratégie, il faut réaliser qu'on utilise souvent la résolution pour déterminer si un ensemble de clauses E', vu comme un ensemble d'axiomes qu'on sait satisfaisable, a pour conséquence logique une certaine formule ϕ. Pour déterminer cela, on met $\neg\phi$ en CNF et on forme E'' l'ensemble des clauses de cette CNF : on a $E' \models \phi$ ssi $E' \cup E''$ est insatisfaisable.

La stratégie *set of support* permet de profiter du fait que E' est satisfaisable quand on cherche à déterminer si $E \vdash \bot$. Elle est réfutationnellement complète :

— Une dérivation de la clause vide ne peut être obtenue sans faire intervenir au moins une clause de E'' : sinon, par correction, E' serait insatisfaisable.
— Une fois qu'on sait qu'on a une feuille dans E'', on peut transformer la dérivation graduellement (par des "rotations" des règles de résolution) afin d'obtenir un dérivation satisfaisant la contrainte de la stratégie.

Cette stratégie donne d'excellents résultats en pratique, puisqu'elle évite d'énumérer toute les conséquences inutiles des axiomes : au lieu de cela, elle ne fait qu'explorer les conséquences des axiomes conjointement avec les formules E'' qu'on sait nécessaires pour obtenir une contradiction.

Résolution complète

On enrichit la résolution de deux nouvelles règles :

$$\frac{C \vee L \quad \overline{L} \vee C'}{C \vee C'}\ R \qquad \frac{C \vee L \vee L}{C \vee L}\ F \qquad \frac{C}{C \vee C'}\ A \qquad \frac{}{L \vee \overline{L}}\ T$$

Pour C une clause et E un ensemble de clauses, on note $E \vdash_{RFAT} C$ quand C est dérivable à partir des clauses de E au moyen des règles R, F, A et T.

Théorème 2.6.3. *Ce système est correct et complet : pour tous E et C, on a $E \models C$ ssi $E \vdash_{RFAT} C$.*

Idée de preuve. Ce résultat a été démontré en TD, par transformation des preuves de $E, \overline{L_1}, \ldots, \overline{L_k} \vdash_{RF} C$ en preuves de $E \vdash_{RFAT} C \vee L_1 \ldots \vee L_k$. □

2.7 Calcul des séquents

On rappelle les règles de la déduction naturelle (pour le calcul propositionnel) en figure 2.1. Cette présentation du système est dûe au logicien Gerhard Gentzen, qui est plus connu pour avoir introduit le calcul des séquents, dont l'objectif est de faciliter l'étude des dérivations (et de la dérivabilité). On verra en effet plusieurs propriétés agréables du calcul des séquents.

$$\frac{}{\Gamma, \phi \vdash \phi}\ \text{ax} \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash \phi}\ \bot_{\text{E}} \qquad \frac{}{\Gamma \vdash \top}\ \top_{\text{I}}$$

$$\frac{\Gamma \vdash \phi_1 \quad \Gamma \vdash \phi_2}{\Gamma \vdash \phi_1 \wedge \phi_2}\ \wedge_{\text{I}} \qquad \frac{\Gamma \vdash \phi_1 \wedge \phi_2}{\Gamma \vdash \phi_i}\ \wedge^i_{\text{E}}$$

$$\frac{\Gamma \vdash \phi_i}{\Gamma \vdash \phi_1 \vee \phi_2}\ \vee^i_{\text{I}} \qquad \frac{\Gamma \vdash \phi_1 \vee \phi_2 \quad \Gamma, \phi_1 \vdash \psi \quad \Gamma, \phi_2 \vdash \psi}{\Gamma \vdash \psi}\ \vee_{\text{E}}$$

$$\frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \Rightarrow \psi}\ \Rightarrow_{\text{I}} \qquad \frac{\Gamma \vdash \phi \Rightarrow \psi \quad \Gamma \vdash \phi}{\Gamma \vdash \psi}\ \Rightarrow_{\text{E}}$$

$$\frac{\Gamma, \phi \vdash \bot}{\Gamma \vdash \neg\phi}\ \neg_{\text{I}} \qquad \frac{\Gamma \vdash \neg\neg\phi}{\Gamma \vdash \phi}\ \text{RAA} \qquad \frac{\Gamma \vdash \neg\phi \quad \Gamma \vdash \phi}{\Gamma \vdash \bot}\ \neg_{\text{E}}$$

FIGURE 2.1 – Système $\mathbf{NK}_0$: déduction naturelle pour la logique propositionnelle classique.

Définition 2.7.1. *Un séquent classique s'écrit $\Gamma \vdash \Delta$ et est composé de deux multi-ensembles de formules.*

Un séquent $\Gamma \vdash \Delta$ peut être lu comme "la *conjonction* des formules de Γ implique la *disjonction* des formules de Δ". On formalise cela dans la notion suivante de validité.

Définition 2.7.2. *Le séquent $\Gamma \vdash \Delta$ est* valide *quand toute interprétation satisfaisant* toutes *les formules de Γ va satisfaire* une *des formules de Δ.*

Le séquent $\Gamma \vdash \Delta$ n'est *pas* valide ssi il existe une interprétation I qui satisfait toutes les formules de Γ mais aucune de Δ ; une telle interprétation est appelée un *contre-modèle* du séquent.

Les règles du calcul des séquents classique propositionnel LK_0 sont données en figure 2.2. Ce système de preuve est structuré différemment de la déduction naturelle. Là où la déduction naturelle est structurée en règles d'introduction et d'élimination, on va trouver en calcul des séquents des règles gauche et droite :

— La règle droite permet de dériver un séquent ayant, dans sa partie droite, une formule construite au moyen du connecteur logique concerné. Ces règles sont très proches des règles d'introduction de la déduction naturelle.

— La règle gauche permet de dériver un séquent ayant, dans sa partie gauche, une formule construite au moyen du connecteur logique concerné. Comme les règles d'élimination, elles permettent de tirer des conclusion d'une formule mais, là où cette formule était

à droite d'un séquent en prémisse de la règle d'élimination, elle va maintenant être à gauche du séquent conclusion. On illustre cela sur le cas de la conjonction :

$$\frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \phi} \qquad \frac{\Gamma, \phi, \psi \vdash \Delta}{\Gamma, \phi \wedge \psi \vdash \Delta}$$

Dans le cas présent, le changement de style permet de déduire d'un coup les deux sous-formules ϕ et ψ de leur conjonction, là où la règle d'élimination ne permet de déduire qu'une sous-formule à la fois.

Une autre différence majeure du calcul des séquents est la présence de multiples conclusions dans les séquents. C'est ce qui permet d'avoir une règle $\vee_R$ qui n'oblige pas à choisir quelle sous-formule on va démontrer. Plus généralement, c'est ce qui apporte une très grande symétrie dans le système. Enfin, on remarquera que le système LK_0 ne comporte pas de règle de raisonnement par l'absurde (ou tiers-exclu) ; ces principes sont en effet dérivables avec les règles données, en exploitant les multiples conclusions du séquent classique.

Exemple 2.7.1.

$$\dfrac{\dfrac{\dfrac{}{\phi \vdash \phi}\,\mathrm{ax}}{\vdash \neg\phi, \phi}\,\neg_R}{\neg\neg\phi \vdash \phi}\,\neg_L \qquad \dfrac{\dfrac{\dfrac{}{\phi \vdash \phi}\,\mathrm{ax}}{\vdash \phi, \neg\phi}\,\neg_R}{\vdash \phi \vee \neg\phi}\,\vee_R$$

Théorème 2.7.1. *Un séquent est dérivable en* LK_0 *ssi il est valide.*

Démonstration. Pour démontrer la correction de la dérivabilité : on montre que chaque règle du système est *correcte*, i.e. si les prémisses sont valides alors la conclusion aussi.

Pour démontrer la complétude, on montre que les règles *logiques* ont la propriété réciproque : les prémisses sont valides si la conclusion l'est. On appelle parfois cela l'*inversibilité* : cette propriété garantit que si l'on applique une règle logique sur une conclusion dont on espère trouver une dérivation, on ne peut pas se tromper, au sens où les prémisses seront encore valides (et donc, in fine, dérivable). L'inversibilité des règles logiques n'est pas démontrée en détail ici mais c'est un exercice à faire. Il est plus aisé d'aborder cette preuve par la contraposée : on montre qu'un contre-modèle d'une prémisse est aussi un contre-modèle de la conclusion.

Une fois qu'on a l'inversibilité, il suffit de remarquer que les prémisses des règles logiques sont plus petites (au sens du nombre de connecteurs logiques) que la conclusion. À partir d'un séquent valide, on peut donc appliquer de façon gourmande les règles logiques pour construire une dérivation, et ce processus termine. On obtient une dérivation avec des séquents valides non justifiés aux feuilles : ces séquents ne contiennent que des atomes, et leur validité implique que l'axiome sera appliquable sur ceux-ci, ce qui permet de terminer la dérivation. □

La simplicité du résultat précédent, et l'algorithme de recherche de preuve sous-jacents, sont déja très différents de ce qu'on pourrait faire en déduction naturelle.

La preuve de complétude qu'on a faite nous donne en réalité plus que la complétude de LK_0 : on a montré que les règles structurelles et la coupure ne sont pas nécessaires à la complétude. Le système privé de la règle de coupure est particulièrement simple, et a notamment deux propriétés importantes, dont les analogues en déduction naturelle ne sont pas du tout évidents :

— Toutes les formules apparaissant dans une dérivation sans coupure sont sous-formules des formules apparaissant dans la conclusion.

— Il n'y a pas de dérivation sans coupure du séquent vide, ou de $\vdash \bot$.

2.7.1 Lien avec la déduction naturelle

Le résultat précédent nous permet déja de conclure que LK_0 et NK_0 permettent de dériver les mêmes séquent $\Gamma \vdash \phi$. Il est néanmoins intéressant de prouver ce résultat directement, en travaillant sur la structure des preuves.

Groupe identité

$$\frac{}{\Gamma, \phi \vdash \phi, \Delta}\ \text{axiom} \qquad \frac{\Gamma \vdash \Delta, \phi \quad \phi, \Gamma \vdash \Delta}{\Gamma \vdash \Delta}\ \text{cut}$$

Groupe structurel

$$\frac{\Gamma, \phi, \phi \vdash \Delta}{\Gamma, \phi \vdash \Delta}\ \mathrm{c}_L \qquad \frac{\Gamma \vdash \phi, \phi, \Delta}{\Gamma \vdash \phi, \Delta}\ \mathrm{c}_R$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \phi \vdash \Delta}\ \mathrm{w}_L \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \phi, \Delta}\ \mathrm{w}_R$$

Groupe logique

$$\frac{}{\Gamma, \bot \vdash \Delta}\ \bot_L \qquad \frac{}{\Gamma \vdash \top, \Delta}\ \top_R$$

$$\frac{\Gamma, \phi_1, \phi_2 \vdash \Delta}{\Gamma, \phi_1 \wedge \phi_2 \vdash \Delta}\ \wedge_L \qquad \frac{\Gamma \vdash \phi_1, \Delta \quad \Gamma \vdash \phi_2, \Delta}{\Gamma \vdash \phi_1 \wedge \phi_2, \Delta}\ \wedge_R$$

$$\frac{\Gamma, \phi_1 \vdash \Delta \quad \Gamma, \phi_2 \vdash \Delta}{\Gamma, \phi_1 \vee \phi_2 \vdash \Delta}\ \vee_L \qquad \frac{\Gamma \vdash \phi_1, \phi_2, \Delta}{\Gamma \vdash \phi_1 \vee \phi_2, \Delta}\ \vee_R$$

$$\frac{\Gamma \vdash \phi_1, \Delta \quad \Gamma, \phi_2 \vdash \Delta}{\Gamma, \phi_1 \Rightarrow \phi_2 \vdash \Delta}\ \Rightarrow_L \qquad \frac{\Gamma, \phi_1 \vdash \phi_2, \Delta}{\Gamma \vdash \phi_1 \Rightarrow \phi_2, \Delta}\ \Rightarrow_R$$

$$\frac{\Gamma \vdash \phi, \Delta}{\Gamma, \neg\phi \vdash \Delta}\ \neg_L \qquad \frac{\Gamma, \phi \vdash \Delta}{\Gamma \vdash \neg\phi, \Delta}\ \neg_R$$

FIGURE 2.2 – Système LK_0 : calcul des séquents pour la logique propositionnelle classique.

Lemme 2.7.1. *Si* $\Gamma \vdash \phi$ *est dérivable dans* NK_0 *alors il l'est aussi dans* LK_0.

Démonstration. On peut en fait traduire chaque règle de NK_0 en une succession de règles de LK_0. Pour cela, les règles de coupure et structurelles sont précieuses. □

Pour l'autre sens, il nous faut trouver un moyen de traiter les multiples formules en conclusion d'un séquent classique : on va pour cela les passer en hypothèse, moyennant une négation. Si Δ est un multi-ensemble de formules, on note $\neg\Delta$ le multi-ensemble des négations de ces formules.

Théorème 2.7.2. *Pour tous multi-ensembles* Γ *et* Δ*, le séquent* $\Gamma \vdash \Delta$ *est dérivable en* LK_0 *ssi le séquent* $\Gamma, \neg\Delta \vdash \bot$ *est dérivable en* NK_0.

Démonstration. Le passage de NK_0 à LK_0 est un corollaire du résultat précédent : si $\Gamma, \neg\Delta \vdash \bot$ est dérivable en NK_0, il l'est aussi en LK_0, on dérive ensuite $\Gamma \vdash \Delta$ par une succession de coupures sur les négations des formules de Δ.

Pour la transformation d'une dérivation LK_0 en NK_0, on traduit chaque règle du calcul des séquents avec $\Gamma \vdash \Delta$ en conclusion en un succession d'applications de règles de la déduction naturelle avec $\Gamma, \neg\Delta \vdash \bot$ en conclusion, et des prémisses de la même forme. □

Chapitre 3

Logique constructive

Nous évoquons rapidement dans ce chapitre la logique constructive, aussi appelée logique *intuitionniste* d'après le nom du courant de pensée initié par le mathématicien Brouwer. On se restreindra ici au cas propositionnel – mais il est tout à fait possible de transposer tout ce qu'on dira au premier ordre, sans grand changement dans les résultats ni les méthodes.

Il y a deux motivations principales pour inclure ce chapitre dans un cours de logique *informatique* :

— Une fois formalisée, la notion de preuve constructive coïncide avec la notion de programme en λ-calcul typé.

— On verra par ailleurs qu'on peut donner une sémantique aux preuves constructives (pour laquelle on aura correction et complétude) via la notion de structure de Kripke. Ce type de sémantique est très utilisé en informatique pour les logiques modales (e.g. temporelles) permettant de spécifier le comportement des systèmes, la forme des données, etc.

3.1 Systèmes de preuve

On discute tout d'abord de comment adapter les systèmes de preuve vus précédemment pour la logique constructive.

3.1.1 Déduction naturelle

La déduction naturelle pour la logique constructive propositionnelle, appelée NJ_0, est simplement obtenue à partir de NK_0 en supprimant la règle du raisonnement par l'absurde : il ne reste que l'axiome et, pour chaque connecteur logique, les règles d'introduction et d'élimination.

Une remarque très importante à propos de ce système est qu'il entretient un lien fort avec le λ-calcul simplement typé : c'est l'*isomorphisme de Curry-Howard*. On retrouve en effet les règles de NJ_0, pour le fragment purement implicatif, en prenant les règles du λ-calcul simplement typé et en effaçant les programmes (i.e. les λ-termes, et les variables dans les environnements de typage). Par exemple :

$$\frac{\Gamma, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x.M : \tau \to \tau'} \quad \text{devient} \quad \frac{\Gamma, \phi_\tau \vdash \phi_{\tau'}}{\Gamma \vdash \phi_\tau \Rightarrow \phi_{\tau'}}$$

La correspondance n'est pas restreinte au fragment implicatif : la conjonction correspond à l'ajout du produit cartésien (type des couples) en λ-calcul, la disjonction est une formulation des types sommes (utilisés dans les types algébriques), les constantes $\top$ et $\bot$ correspondent aux types unité et vide.

Le traitement des hypothèses à gauche des séquents comme un multi-ensemble prend son sens à travers la correspondance preuve-programme : il permet par exemple de distinguer deux preuves de $P \Rightarrow P \Rightarrow P$, correspondant aux deux termes $\lambda x.\lambda y.x$ et $\lambda x.\lambda y.y$.

La correspondance de Curry-Howard ne porte pas uniquement sur la structure des règles, mais avant tout sur les opérations sur les preuves (resp. les programmes) : la β-réduction des λ-termes correspond à l'élimination des détours en déduction naturelle. Il s'agit dune opération importante de simplification de preuve par élimination des raisonnements indirects, qui va jouer un rôle analogue à l'élimination des coupures en calcul des séquents

Définition 3.1.1. *Un détour est l'utilisation d'une règle d'introduction pour dériver la première prémisse d'une règle d'élimination.*

La forme de nos règles fait que les deux règles en question sont forcément l'introduction et l'élimination d'un même connecteur logique.

Exemple 3.1.1.

$$\dfrac{\dfrac{\Gamma \vdash \phi_1 \quad \Gamma \vdash \phi_2}{\Gamma \vdash \phi_1 \wedge \phi_2}\,\wedge_{\mathrm{I}}}{\Gamma \vdash \phi_1}\,\wedge_{\mathrm{E}} \qquad \dfrac{\dfrac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \Rightarrow \psi}\,\Rightarrow_{\mathrm{I}} \quad \Gamma \vdash \phi}{\Gamma \vdash \psi}\,\Rightarrow_{\mathrm{E}}$$

On peut vérifier que tout détour peut être éliminé. Dans certains cas cette élimination réduit la taille de la preuve, et n'introduit pas de nouveaux détours. Dans le cas de l'implication, la situation est plus compliquée : il va en fait falloir faire une β-réduction au niveau des preuves ; cela devrait vous rappeler le théorème de préservation du typage.

On obtient enfin l'analogue du théorème de normalisation forte du λ-calcul simplement typé : l'élimination systématique des détours est possible.

Théorème 3.1.1. *Tout séquent dérivable dans* NJ_0 *admet une dérivation sans détour.*

L'importance de ce résultat se mesure notamment à ses corollaires, qui s'obtiennent simplement en remarquant qu'une dérivation sans détour d'un séquent sans antécédent débute forcément par une règle d'introduction.

Corollaire 3.1.1. *Le séquent* $\vdash \bot$ *n'est pas dérivable en* NJ_0.

Corollaire 3.1.2. *Si un séquent* $\vdash \phi_1 \vee \phi_2$ *est dérivable en* NJ_0*, alors il existe* $i \in \{1, 2\}$ *tel que* $\vdash \phi_i$ *est aussi dérivable.*

Corollaire 3.1.3. *Le tiers exclu n'est pas prouvable en logique constructive.*

3.1.2 Calcul des séquents

Le calcul des séquents s'adapte quant à lui en imposant d'avoir toujours exactement une formule à droite des séquents [1]. Le système obtenu est donné en figure 3.1 : en bref, on a supprimé les règles structurelles à droite ; la règle $\vee_R$ a été remplacée, essentiellement, par les deux règles d'introduction de la déduction naturelle ; enfin, dans les règles $\Rightarrow_L$ et $\neg_L$, on ne peut pas garder dans la première prémisse la formule ψ à droite du séquent conclusion. Ces trois règles modifiées ne sont plus inversibles : il est possible qu'une des prémisses ne soit pas prouvable même si la conclusion l'état. En fait, comme l'exemple précédent l'illustre, il est parfois nécessaire d'utiliser la contraction avant d'appliquer $\Rightarrow_L$ ou $\neg_L$, alors que la contraction était toujours éliminable en LK_0.

Exemple 3.1.2.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{axiom}}{A \vdash A \vee \neg A}\,\vee_R}{A, \neg(A \vee \neg A) \vdash \bot}\,\neg_L}{\neg(A \vee \neg A) \vdash \neg A}\,\neg_R}{\neg(A \vee \neg A) \vdash A \vee \neg A}\,\vee_R}{\neg(A \vee \neg A), \neg(A \vee \neg A) \vdash \bot}\,\neg_L}{\neg(A \vee \neg A) \vdash \bot}\,C_L}{\vdash \neg\neg(A \vee \neg A)}\,\neg_R$$

1. On pourrait aussi demander d'avoir *au plus* une formule à droite, ce qui ne change rien d'essentiel : n'avoir aucune formule à droite revient à avoir uniquement la formule $\bot$.

Groupe identité

$$\frac{}{\Gamma, \phi \vdash \phi}\ \text{axiom} \qquad \frac{\Gamma \vdash \phi \quad \phi, \Gamma \vdash \psi}{\Gamma \vdash \psi}\ \text{cut}$$

Groupe structurel

$$\frac{\Gamma, \phi, \phi \vdash \psi}{\Gamma, \phi \vdash \psi}\ \mathrm{c}_L \qquad \frac{\Gamma \vdash \psi}{\Gamma, \phi \vdash \psi}\ \mathrm{w}_L$$

Groupe logique

$$\frac{}{\Gamma, \bot \vdash \psi}\ \bot_L \qquad \frac{}{\Gamma \vdash \top}\ \top_R$$

$$\frac{\Gamma, \phi_1, \phi_2 \vdash \psi}{\Gamma, \phi_1 \wedge \phi_2 \vdash \psi}\ \wedge_L \qquad \frac{\Gamma \vdash \phi_1 \quad \Gamma \vdash \phi_2}{\Gamma \vdash \phi_1 \wedge \phi_2}\ \wedge_R$$

$$\frac{\Gamma, \phi_1 \vdash \psi \quad \Gamma, \phi_2 \vdash \psi}{\Gamma, \phi_1 \vee \phi_2 \vdash \psi}\ \vee_L \qquad \frac{\Gamma \vdash \phi_i}{\Gamma \vdash \phi_1 \vee \phi_2}\ \vee^i_R$$

$$\frac{\Gamma \vdash \phi_1 \quad \Gamma, \phi_2 \vdash \psi}{\Gamma, \phi_1 \Rightarrow \phi_2 \vdash \psi}\ \Rightarrow_L \qquad \frac{\Gamma, \phi_1 \vdash \phi_2}{\Gamma \vdash \phi_1 \Rightarrow \phi_2}\ \Rightarrow_R$$

$$\frac{\Gamma \vdash \phi}{\Gamma, \neg\phi \vdash \psi}\ \neg_L \qquad \frac{\Gamma, \phi \vdash \bot}{\Gamma \vdash \neg\phi}\ \neg_R$$

FIGURE 3.1 – Système LJ_0 : calcul des séquents pour la logique propositionnelle constructive.

On peut prouver, comme pour la logique classique, l'équivalence entre déduction naturelle et calcul des séquents (en utilisant la coupure).

Théorème 3.1.2. *Les systèmes* NJ_0 *et* LJ_0 *dérivent les mêmes séquents.*

On a vu, à l'occasion du résultat de complétude pour la logique classique, que la coupure peut être éliminée. Cette preuve là ne se transposera pas. Néanmoins, l'élimination des coupures se démontre aussi directement par transformations successives des preuves (l'analogue de la simplification des détours en déduction naturelle) ce qui se fait aussi bien dans les deux logiques.

Théorème 3.1.3. *Les séquents dérivables en* LJ_0 *admettent aussi des dérivations sans coupure.*

Les conséquences de ce théorème sont les mêmes que pour l'élimination des coupures : propriété de la disjonction, absence de preuve du tiers-exclu, etc.

3.1.3 Résolution

La résolution ne s'adapte pas au cas intuitionniste, en effet la mise en forme clausale n'est déja plus possible, car elle s'appuyait sur des équivalences qui ne sont pas toutes constructives – on verra par exemple que $\neg\neg A$ et A ne sont pas équivalents.

3.2 Sémantique de Kripke

On définit d'abord la sémantique de Kripke et, après quelques remarques basiques, on démontre que la déduction naturelle intuitionniste est correcte et complète pour cette sémantique. Ce résultat peut être adapté sans difficulté au calcul des séquents intuitionniste, en utilisant néanmoins la coupure.

3.2.1 Définitions

Les interprétations $I : \mathcal{P} \to \{0,1\}$ de la logique classique représentent des mondes où chaque variable est déterminée : vrai ou faux. Dans la sémantique de Kripke, les interprétations sont généralisées en des structures qui représentent plusieurs situations possibles.

Définition 3.2.1 (Structure de Kripke)**.** *Une* structure de Kripke *est la donnée de :*
- *un ensemble de mondes $\mathcal{W}$;*
- *un ordre partiel $\leq$ sur les mondes ;*
- *pour chaque monde $w \in \mathcal{W}$, une interprétation $I_w : \mathcal{P} \to \{0,1\}$ telle que pour tous $w \leq w'$ et $P \in \mathcal{P}$, $I_w(P) \leq I_{w'}(P)$.*

On note $\mathcal{W}(\mathcal{K})$ l'ensemble des mondes d'une structure de Kripke $\mathcal{K}$.

Quand $w \leq w'$, on dira que w' est *accessible* à partir de w, ou que w' est un *futur possible* de w.

Définition 3.2.2 (Satisfaction)**.** *Étant donnés une structure de Kripke $\mathcal{K}$, un monde $w \in \mathcal{W}(\mathcal{K})$ et une formule $\phi \in \mathcal{F}_0(\mathcal{P})$, on définit la relation de* satisfaction *par induction sur ϕ :*
- *$\mathcal{K}, w \models P$ ssi $I_w(P) = 1$;*
- *$\mathcal{K}, w \models \top$;*
- *$\mathcal{K}, w \not\models \bot$;*
- *$\mathcal{K}, w \models \phi \wedge \psi$ ssi $\mathcal{K}, w \models \phi$ et $\mathcal{K}, w \models \psi$;*
- *$\mathcal{K}, w \models \phi \vee \psi$ ssi $\mathcal{K}, w \models \phi$ ou $\mathcal{K}, w \models \psi$;*
- *$\mathcal{K}, w \models \phi \Rightarrow \psi$ ssi pour tout $w' \geq w$, $\mathcal{K}, w' \models \phi$ implique $\mathcal{K}, w' \models \psi$;*
- *$\mathcal{K}, w \models \neg\psi$ ssi pour tout $w' \geq w$, $\mathcal{K}, w' \not\models \phi$.*

On omettra souvent de spécifier $\mathcal{K}$ quand il est évident ou inutile, en écrivant simplement $w \models \phi$.

On dit qu'un ensemble de formules E est *satisfait* par $w \in \mathcal{W}(\mathcal{K})$ quand $\mathcal{K}, w \models \phi$ pour tout $\phi \in E$. Une formule ϕ est dite *valide* quand elle est satisfaite dans tout monde de toute structure de Kripke. Un ensemble de formules E a pour *conséquence logique* la formule ϕ quand, pour tous $w \in \mathcal{W}(\mathcal{K})$ satisfaisant E, on a aussi a $\mathcal{K}, w \models \phi$. Deux formules sont logiquement équivalentes si elles sont satisfaites dans les mêmes mondes.

Remarque 3.2.1. *Les formules $\neg\phi$ et $\phi \Rightarrow \bot$ sont logiquement équivalentes. Par contre, $w \models \neg\phi$ n'est pas équivalent à $w \not\models \phi$: ainsi le tiers exclu n'est pas valide, et ϕ et $\neg\neg\phi$ ne sont pas logiquement équivalentes.*

La monotonicité des interprétations dans une structure de Kripke a la conséquence suivante, qui se démontre par induction sur la formule.

Proposition 3.2.1 (Monotonicité de la satisfaction)**.** *Si $w \models \phi$ et $w \leq w'$, alors $w' \models \phi$.*

On peut enfin remarquer que les formules valides au sens de la sémantique de Kripke sont classiquement valides, puisque les interprétations de la logique classique peuvent être vues comme des structures de Kripke réduites à un seul monde possible.

3.2.2 Correction et complétude

Un séquent $\Gamma \vdash \phi$ est valide quand $\Gamma \models \phi$ au sens de sous-section précédente. On peut constater que la déduction naturelle intuitionniste est correcte pour la sémantique de Kripe, en vérifiant que chaque règle préserve cette notion de validité.

Théorème 3.2.1. *Les séquents dérivables en* NJ_0 *sont valides.*

Pour montrer la réciproque, il va nous falloir construire et étudier une structure de Kripke particulière. Dans la suite, on suppose l'ensemble $\mathcal{P}$ dénombrable, ce qui nous permet de se donner une bijection $r : \mathcal{F}_0 \to \mathbb{N}$.

Définition 3.2.3 ($E \vdash_{\mathrm{NJ}} \phi$)**.** *Soit E un ensemble de formules (fini ou infini). On écrit $E \vdash_{\mathrm{NJ}} \phi$ quand il existe un sous-ensemble fini $\Gamma \subseteq E$ tel que $\Gamma \vdash \phi$ est dérivable dans* NJ_0.

Définition 3.2.4 (Ensemble saturé)**.** *Un ensemble de formules E est* saturé *quand, pour tout ϕ tel que $E \vdash_{\mathrm{NJ}} \phi$, on a $\phi \in E$.*

Proposition 3.2.2. *L'ensemble $E^* = \{\ \phi\ :\ E \vdash_{\mathrm{NJ}} \phi\ \}$ est toujours saturé.*

Définition 3.2.5 (Ensemble-monde)**.** *On dit qu'un ensemble de formules E est* cohérent *quand il ne contient pas $\bot$. On dit qu'il satisfait la* propriété de la disjonction *quand, pour tout $\phi_1 \vee \phi_2 \in E$, on a $\phi_i \in E$ pour un $i \in \{1, 2\}$. On dit enfin qu'un ensemble de formule E est un* ensemble-monde *quand il est saturé, cohérent, et satisfait la propriété de la disjonction.*

Définition 3.2.6. *La* structure de Kripke universelle $\mathcal{U}$ *est définie par :*
- *l'ensemble de mondes $\mathcal{W}(\mathcal{U}) = \{\ w_E \mid E$ est un ensemble-monde $\}$;*
- *$w_E \leq w_{E'}$ ssi $E \subseteq E'$;*
- *$I_{w_E}(P) = 1$ ssi $P \in E$.*

Lemme 3.2.1 (Lemme de Lindenbaum)**.** *Soit E un ensemble de formules quelconque et ϕ une formule, tels que $E \nvdash_{\mathrm{NJ}} \phi$. Il existe un ensemble-monde E' tel que $E \subseteq E'$ et $E' \nvdash_{\mathrm{NJ}} \phi$.*

Démonstration. On définit une séquence d'ensembles saturés croissante, dont la limite est un ensemble-monde. Pour cela, on va s'appuyer sur l'énumération des formules induite par r. □

Lemme 3.2.2. *Soit E un ensemble-monde et ϕ une formule. On a $\mathcal{U}, w_E \models \phi$ ssi $\phi \in E$.*

Démonstration. Ce résultat se démontre par induction sur ϕ, en utilisant la définition de $\models$ et $\vdash_{\mathrm{NJ}}$. Dans le cas de l'implication, on utilisera le lemme de Lindenbaum. □

Théorème 3.2.2. $\Gamma \models \phi$ *implique* $\Gamma \vdash_{\mathrm{NJ}} \phi$.

Démonstration. Supposons $\Gamma \models \phi$ et $\Gamma \nvdash_{\mathrm{NJ}} \phi$. Par le lemme 3.2.1 il existe un ensemble-monde E tel que $\Gamma \subseteq E$ et $\phi \notin E$. On a $w_E \models \Gamma$, et puisque $\Gamma \models \phi$ on en conclut $w_E \models \phi$. Par le lemme 3.2.2, cela implique $\phi \in E$: contradiction. □

Chapitre 4

Déduction naturelle

Ce chapitre correspond au cours de l'année 2022–2023. À partir de l'année 2023–2024 la déduction naturelle est supposée connue car au programme de MPI. Le chapitre est laissé ici pour référence, mais attention : il est beaucoup plus détaillé que les pré-requis du cours correspondant au programme de MPI. En particulier, la MPI ne traite que de déduction naturelle classique, et on ne s'y intéresse pas autant aux propriétés et transformations des dérivations.

Les preuves par déduction naturelle correspondent assez bien aux preuves qu'on écrit en mathématiques, et elles ne nécessitent notamment pas de passer par une mise en CNF c'est le cas pour la résolution. Néanmoins, la déduction naturelle ne vous paraitra pas tout de suite naturelle. En particulier, il n'est pas toujours facile de formuler en déduction naturelle une démonstration mathématique informelle, et il est encore plus difficile d'utiliser la déduction naturelle pour découvrir des démonstrations. Néanmoins, c'est un formalisme important, construit de façon modulaire et jouissant de propriétés riches, dont des connexions avec la programmation fonctionnelle typée pure.

La déduction naturelle est dûe à Gerhard Gentzen en 1934, elle pré-date donc largement la résolution, inventée par Robinson en 1965.

4.1 Définitions

La déduction naturelle est un formalisme de preuve arborescent, où l'on construit des dérivations par applications successives de règles d'inférence, comme en résolution. Contrairement à la résolution, on ne va pas dériver des clauses, ni même des formules, mais des *séquents*.

Définition 4.1.1. *Un séquent est une paire* (Γ, ϕ) *formée d'un multi-ensemble*[1] *de formules* Γ *et d'une formule* ϕ*. On note cet objet* $\Gamma \vdash \phi$*.*

Les formules de Γ sont vues comme des hypothèses, et le séquent énonce intuitivement que la conjonction de ces hypothèses implique ϕ. Il serait naturel d'appeler ϕ la *conclusion* du séquent, mais cela créerait une confusion avec la notion de conclusion d'un arbre de dérivation (dont les noeuds seraient décorés par des séquents). On parle ainsi parfois des antécédents Γ et du succédent ϕ d'un séquent $\Gamma \vdash \phi$.

Définition 4.1.2. *La lecture logique d'un séquent* $\psi_1, \ldots, \psi_n \vdash \phi$ *est la formule* $\psi_1 \Rightarrow \ldots \Rightarrow \psi_n \Rightarrow \phi$*. On dit que le séquent est valide quand cette formule est valide.*

1. La plupart du temps, on peut considérer qu'il s'agit en fait d'un ensemble. C'est le cas quand on cherche une preuve. Cependant, certains résultats sont plus clairs avec des multi-ensembles.

$$\frac{}{\Gamma, \phi \vdash \phi}\ \text{ax}$$

$$\frac{\Gamma \vdash \phi_1 \quad \Gamma \vdash \phi_2}{\Gamma \vdash \phi_1 \wedge \phi_2}\ \wedge_{\text{I}} \qquad \frac{\Gamma \vdash \phi_1 \wedge \phi_2}{\Gamma \vdash \phi_i}\ \wedge_{\text{E}}$$

$$\frac{\Gamma \vdash \phi_i}{\Gamma \vdash \phi_1 \vee \phi_2}\ \vee_{\text{I}} \qquad \frac{\Gamma \vdash \phi_1 \vee \phi_2 \quad \Gamma, \phi_1 \vdash \psi \quad \Gamma, \phi_2 \vdash \psi}{\Gamma \vdash \psi}\ \vee_{\text{E}}$$

$$\frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \Rightarrow \psi}\ \Rightarrow_{\text{I}} \qquad \frac{\Gamma \vdash \phi \Rightarrow \psi \quad \Gamma \vdash \phi}{\Gamma \vdash \psi}\ \Rightarrow_{\text{E}}$$

FIGURE 4.1 – Règles de la déduction naturelle minimale.

4.2 Déduction naturelle minimale

La déduction naturelle minimale, notée NM_0 quand elle porte sur des formules propositionnelles, est le système de preuve composé des règles données en Figure 4.1.

La première règle, l'axiome ax, a un statut particulier. C'est la seule règle qui exige que quelquechose soit présent à gauche du séquent ; ce sera donc la seule façon d'utiliser une hypothèse. Ensuite, pour chaque connecteur logique on a deux (schémas de) règle(s) :

— La règle d'introduction permet de dériver un séquent ayant comme succédent une formule construite avec ce connecteur.

— La règle d'introduction permet de dériver quelquechose à partir d'un séquent ayant comme succédent une formule construite avec ce connecteur.

On peut remarquer que les règles d'introduction et d'élimination d'un connecteur ne parlent que de ce connecteur logique.

Exemple 4.2.1. On peut dériver les séquents suivants :

— $\phi \Rightarrow \psi, \phi \vdash \psi$

— $(\phi \Rightarrow \phi' \Rightarrow \phi'') \vdash (\phi' \Rightarrow \phi \Rightarrow \phi'')$

— $\phi \wedge \phi' \vdash \phi' \wedge \phi$

Les règles d'introduction et d'élimination disent "tout ce qu'il y a à dire" sur chaque connecteur logique, au sens où elles définissent complètement le connecteur. Par exemple, la règle d'introduction dit qu'il suffit d'avoir ϕ_1 et ϕ_2 pour avoir leur conjonction, et la règle d'introduction énonce qu'on a nécessairement chaque ϕ_i dès lors qu'on a $\phi_1 \wedge \phi_2$. On peut voir cela en jouant à définir un nouveau connecteur, défini par les mêmes règles qu'un connecteur existant, et à vérifier en déduction naturelle que les deux sont équivalents. Par exemple, on peut démontrer $\phi_1 \star \phi_2 \vdash \phi_1 \wedge \phi_2$ et vice versa si l'on ajoute un connecteur logique $\star$ binaire à la syntaxe de nos formules, et que l'on se dote des règles suivantes :

$$\frac{\Gamma \vdash \phi_1 \quad \Gamma \vdash \phi_2}{\Gamma \vdash \phi_1 \star \phi_2}\ \star_{\text{I}} \qquad \frac{\Gamma \vdash \phi_1 \star \phi_2}{\Gamma \vdash \phi_i}\ \star_{\text{E}}$$

4.2.1 Propriétés simples

Voyons quelques propriétés simples et utiles de ce système de preuve.

Proposition 4.2.1 (Affaiblissement)**.**
Si $\Gamma \vdash \phi$ *est dérivable dans* NM_0*, alors* $\Gamma, \psi \vdash \phi$ *aussi.*

Proposition 4.2.2 (Renforcement)**.**
Si $\Gamma, \psi \vdash \phi$ *a une dérivation dans* NM_0 *sans règle axiome portant sur* ψ*, alors* $\Gamma \vdash \phi$ *est dérivable dans* NM_0*.*

Proposition 4.2.3 (Coupure)**.**
Si $\Gamma, \phi \vdash \psi$ *et* $\Gamma \vdash \phi$ *sont dérivables dans* NM_0*,*
alors $\Gamma \vdash \psi$ *l'est aussi.*

Toutes ces propositions peuvent s'énoncer comme l'admissibilité de nouvelles règles, que l'on peut distinguer par une double ligne pour insister sur le fait que ce ne sont pas des règles données dans le système de déduction mais des règles qui ne permettent de dériver que des choses qui étaient déja dérivables dans le système de départ :

$$\frac{\Gamma \vdash \phi}{\Gamma, \psi \vdash \phi} \qquad \frac{\Gamma \vdash \phi \quad \Gamma, \phi \vdash \psi}{\Gamma \vdash \psi}$$

On peut remarquer que ces deux règles admissibles ont des statuts un peu différents : la seconde peut être obtenue comme un *widget*, une composition de règles de NM_0, tandis que l'admissibilité de la première nécessite de raisonner sur la dérivation de sa prémisse.

4.2.2 Détours

La structuration des preuves en déduction naturelle fait apparaitre la notion de détour. Comme le nom l'indique, ces détours peuvent être évités pour obtenir des preuves plus directes – mais potentiellement plus grosses.

Définition 4.2.1. *Un détour est l'utilisation d'une règle d'introduction pour dériver la première prémisse d'une règle d'élimination.*

La forme de nos règles fait que les deux règles en question sont forcément l'introduction et l'élimination d'un même connecteur logique.

Exemple 4.2.2.

$$\dfrac{\dfrac{\Gamma \vdash \phi_1 \quad \Gamma \vdash \phi_2}{\Gamma \vdash \phi_1 \wedge \phi_2}\,\wedge_{\mathrm{I}}}{\Gamma \vdash \phi_1}\,\wedge_{\mathrm{E}} \qquad \dfrac{\dfrac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \Rightarrow \psi}\,\Rightarrow_{\mathrm{I}} \quad \Gamma \vdash \phi}{\Gamma \vdash \psi}\,\Rightarrow_{\mathrm{E}}$$

On peut vérifier que tout détour peut être éliminé. Dans certains cas cette élimination réduit la taille de la preuve, et n'introduit pas de nouveaux détours. Dans le cas de l'implication, la situation est plus compliquée. Néanmoins, l'élimination systématique des détours est possible. Mieux, toute stratégie d'élimination des détours (utilisant les simplifications officielles, détaillées en cours) va aboutir, en temps fini, sur une preuve sans détour.

Théorème 4.2.1. *Tout séquent dérivable dans* NM_0 *admet une dérivation sans détour.*

L'importance de ce résultat se mesure notamment à ses corollaires, qui s'obtiennent simplement en remarquant qu'une dérivation sans détour d'un séquent sans antécédent débute forcément par une règle d'introduction.

Corollaire 4.2.1. *Le séquent* $\vdash \bot$ *n'est pas dérivable en* NM_0*.*

Corollaire 4.2.2. *Si un séquent* $\vdash \phi_1 \vee \phi_2$ *est dérivable en* NM_0*, alors il existe* $i \in \{1, 2\}$ *tel que* $\vdash \phi_i$ *est aussi dérivable.*

Corollaire 4.2.3. *Le tiers exclu n'est pas prouvable en logique minimale.*

4.3 Isomorphisme de Curry-Howard

Interlude : tout ceci ressemble fort à du λ*-calcul !*

Pour faire simple, ne gardons que l'implication comme seul connecteur logique. Alors un type simple du λ-calcul peut être vu comme une formule : les types de base deviennent

des variables propositionnelles, et la flèche devient l'implication. Un jugement de typage $\Gamma \vdash M : T$ induit un séquent, en traduisant les types en formules, et en effaçant le programme M ainsi que les variables de Γ.

Les règles de NM_0 ne sont alors rien d'autre que les règles de typage, auxquelles on a fait subir la traduction précédente. Plus fort : les détours sont les β-redexes, et le Théorème 4.2.1 découle du résultat de normalisation (faible ou forte) pour le λ-calcul simplement typé !

Tout ceci est surprenant, mais aussi fructueux : cette observation est à la base d'une vision des preuves comme des programmes, donnant corps à l'intuition mathématique de "preuve constructive". Quand je prouve $\phi, \phi \Rightarrow \psi \vdash \psi$ en déduction naturelle minimale, je construis en fait un programme qui permet de transformer une preuve de ϕ et preuve de $\phi \Rightarrow \psi$ en une preuve de ψ. Tout ceci s'étend à la conjonction (type produit) et à la disjonction (type somme) et l'on comprend que la preuve de la commutativité de la conjonction n'est autre que le programme qui déconstruit une paire et la reconstruit dans l'autre sens.

4.4 Déduction naturelle intuitionniste

On obtient la déductibles naturelle intuitionniste NJ_0 en ajoutant à NM_0 les règles donnant leur sens aux constantes logiques. Le faux n'a qu'une règle d'élimination (puisqu'on ne peut pas le prouver) et le vrai n'a qu'une règle d'introduction (puisqu'on ne peut rien en déduire) :

$$\frac{}{\Gamma \vdash \top}\ \top_{\mathrm{I}} \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash \phi}\ \bot_{\mathrm{E}}$$

On peut définir dans ce système la négation $\neg\phi$ comme $\phi \Rightarrow \bot$. De façon équivalente, on peut se donner des règles définissant la négation :

$$\frac{\Gamma, \phi \vdash \bot}{\Gamma \vdash \neg\phi}\ \neg_{\mathrm{I}} \qquad \frac{\Gamma \vdash \neg\phi \quad \Gamma \vdash \phi}{\Gamma \vdash \bot}\ \neg_{\mathrm{E}}$$

Proposition 4.4.1. *Les règles de* NJ_0 *sont correctes en logique propositionnelle classique : tout séquent dérivable en* NJ_0 *est valide.*

Tout ce qu'on a dit sur la logique minimale reste vrai en logique intuitionniste, notamment la possibilité de transformer toute preuve en preuve sans détour, et l'impossibilité de dériver le tiers-exclu. La logique intuitionniste est parfois appelée logique constructive.

Exemple 4.4.1. On a $\phi \Rightarrow \psi \vdash_{\mathbf{NJ}} \neg\psi \Rightarrow \neg\phi$, mais on ne pourra pas dériver la réciproque. Intuitivement, montrer une formule en établissant sa contraposée n'est pas un argument constructif.

4.5 Déduction naturelle classique

La déduction naturelle classique NK_0 est enfin obtenue en ajoutant aux règles précédentes l'unique règle suivante, appelée *reductio ad absurdo* :

$$\frac{\Gamma \vdash \neg\neg\phi}{\Gamma \vdash \phi}\ \mathrm{RAA}$$

Théorème 4.5.1. NK_0 *est correcte et complète : un séquent est dérivable dans* NK_0 *ssi il est valide.*

On peut vérifier qu'on sait dériver le tiers-exclu grâce à cette nouvelle règle. Inversement, si on s'était donné le tiers-exclu comme règle, on aurait pu dériver la règle du raisonnement par l'absurde.

Attention, l'introduction de ces nouvelles règles invalide tout ce qu'on a dit sur la notion de détour et les résultats afférents !

4.6 Extension au premier ordre

La déduction naturelle s'étend très bien au delà du calcul propositionnel. En particulier, nous allons l'étudier dans le cadre du calcul des prédicats, aussi appelé logique du premier ordre. Ce langage logique est celui dans lequel la plupart des mathématiques se fait.

4.6.1 Syntaxe

On définit ici la syntaxe des formules du premier ordre. La grande nouveauté est que les variables propositionnelles sont maintenant des prédicats qui énoncent des propriétés à propos d'objets représentés par des termes. Il y a une construction à deux étages, avec les termes en dessous et les formules au dessus, qu'il faut bien comprendre et respecter quand on fait de la logique du premier ordre !

Définition 4.6.1. *Étant donné une signature $\mathcal{F}$ spécifiant un ensemble de symboles de fonctions munis d'une arité, et un ensemble de variables $\mathcal{X}$, l'ensemble $\mathcal{T}(\mathcal{F}, \mathcal{X})$ des termes est défini inductivement :*

— $\mathcal{X} \subseteq \mathcal{T}(\mathcal{F}, \mathcal{X})$;

— *pour tout $f \in \mathcal{F}$ d'arité k et $t_1, \ldots, t_k \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, on a $f(t_1, \ldots, t_k) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$.*

Les termes seront toujours notés avec les lettres s, t, u, v, w. Les variables seront notées avec les noms x, y, z.

Exemple 4.6.1. Sur $\mathcal{X} = \{x, y, z\}$ et $\mathcal{F} = \{f, c\}$ avec f d'arité 2 et c d'arité 0, on peut former les termes c, $f(x, c)$, ou encore $f(f(c, c), x)$. Par contre, ni $c(x)$ ni $f(c)$ ne sont des termes.

Définition 4.6.2. *Étant donnés une signature $\mathcal{F}$ ainsi qu'un ensemble de variables $\mathcal{X}$, et un ensemble de symboles de prédicats $\mathcal{P}$ munis d'une arité, on définit inductivement les formules de la logique du premier ordre :*

— *$p(t_1, \ldots, t_k)$ est une formule si $p \in \mathcal{P}$ d'arité k et que les $t_1, \ldots, t_k \in \mathcal{T}(\mathcal{F}, \mathcal{X})$;*

— *si ϕ et ψ sont des formules, alors $\top$, $\bot$, $\neg\phi$, $\phi \wedge \psi$, $\phi \vee \psi$ et $\phi \Rightarrow \psi$ sont des formules ;*

— *si ϕ est une formule est $x \in \mathcal{X}$, alors $\forall x.\phi$ et $\exists x.\phi$ sont des formules.*

On dit qu'une variable x est libre dans une formule ϕ quand elle apparait dans ϕ à une position qui n'est pas "sous" une quantification utilisant la même variable. On note $\mathsf{fv}(\phi)$ l'ensemble des variables libres de ϕ. Les termes et formules sont munis d'une opération de substitution, notée $u[x := v]$ ou $\phi[x := v]$. On considèrera les formules modulo renommage des variables liées (ou muettes). Nous reviendrons sur tout cela plus formellement dans la suite...

Dans la suite on suppose $\mathcal{P}$, $\mathcal{F}$ et $\mathcal{X}$ fixés, avec $\mathcal{X}$ infini.

Nous n'avons pas défini de sémantique pour pouvoir définir ce qu'est une formule valide, et donc une règle de déduction valide. Néanmoins, nous pouvons avancer un peu avec notre connaissance intuitive de la logique du premier ordre, basée sur notre pratique informelle des mathématiques.

4.6.2 Systèmes de preuve

Les séquents manipulés jusque là, contenant des formules de la logique propositionnelle, s'adaptent naturellement à la logique du premier ordre, en considérant simplement des formules du premier ordre. Les règles de la Figure 4.1 peuvent alors être vues comme des règles portant sur des séquents du premier ordre, et elles sont intuitivement raisonnables pour cette logique là.

La déduction naturelle minimale au premier ordre, NM_1, est obtenue en ajoutant les règles de la Figure 4.2 aux règles de la Figure 4.1.

Ensuite, NJ_1 est obtenue en ajoutant à NM_1 les règles du faux et du vrai, puis NK_1 est obtenue en ajoutant la règle du raisonnement par l'absurde. Autrement dit, pour tout

$$\frac{\Gamma \vdash \phi[x := t]}{\Gamma \vdash \exists x.\phi}\ \exists_{\mathrm{I}} \qquad \frac{\Gamma \vdash \exists x.\phi \quad \Gamma, \phi \vdash \psi}{\Gamma \vdash \psi}\ \exists_{\mathrm{E}}\ (x \notin \mathsf{fv}(\Gamma, \psi))$$

$$\frac{\Gamma \vdash \phi}{\Gamma \vdash \forall x.\phi}\ \forall_{\mathrm{I}}\ (x \notin \mathsf{fv}(\Gamma)) \qquad \frac{\Gamma \vdash \forall x.\phi}{\Gamma \vdash \phi[x := t]}\ \forall_{\mathrm{E}}$$

FIGURE 4.2 – Règles de déduction naturelle pour le premier ordre

$\mathrm{X} \in \{\mathrm{M}, \mathrm{J}, \mathrm{K}\}$, le système NX_1 est obtenu en ajoutant les règles de la Figure 4.2 à celles de NX_0 (vu comme un système de déduction pour des séquents du premier ordre).

Exemple 4.6.2. Si p est un symbole de prédicat unaire et f un symbole de fonction unaire, on peut dériver $\forall x.\ p(x) \vdash \forall x.\ p(f(x))$ dans NM_1.

L'enjeu est de sentir ici l'intérêt des conditions sur les variables libres des formules impliquées dans les règles $\forall_{\mathrm{I}}$ et $\exists_{\mathrm{E}}$. Sans ces conditions, on pourrait dériver des âneries, par exemple $p(x) \vdash \forall x.\ p(x)$.

Nous verrons dans la suite du cours comment munir la logique du premier ordre d'une sémantique, et justifier ainsi les règles. En attendant, une propriété purement preuve-théorique peut déja nous donner confiance dans nos règles :

Proposition 4.6.1. *Si un séquent $\Gamma \vdash \phi$ est dérivable dans l'un des systèmes de déduction naturelle au premier ordre, alors $\Gamma[x := t] \vdash \phi[x := t]$ est dérivable aussi dans le même système.*

Considérons une instance de $\forall_{\mathrm{I}}$, avec une conclusion $\Gamma \vdash \forall x.\phi$ telle que $x \notin \mathsf{fv}(\Gamma)$, et une prémisse $\Gamma \vdash \phi$. Si l'on prouve la prémisse, la proposition précédente nous garantit déja qu'on a aussi des preuves de $\Gamma \vdash \phi[x := t]$ pour tout t (la condition $x \notin \mathsf{fv}(\Gamma)$ nous assure que la substitution n'a pas d'effet sur Γ). On a donc bien vérifié que ϕ était vrai pour toute valeur de x... ou du moins, toute valeur qu'on peut représenter par un terme.

Chapitre 5

Logique du premier ordre

Dans ce chapitre nous introduisons la syntaxe et la sémantique de la *logique du premier ordre*, aussi appelée *calcul des prédicats*. C'est la logique dans laquelle on fait la plupart des mathématiques, elle sert notamment de cadre à la théorie des ensembles, à l'arithmétique, etc. On la retrouve aussi largement en informatique, en preuve automatique (e.g. solveurs SMT), preuve de programmes (cf. logique de Hoare), bases de données, etc.

Les notes de cours de ce chapitre sont fortement inspirées des notes du MOOC "Introduction à la logique informatique (partie 2)", réalisé par David Baelde, Hubert Comon et Étienne Lozes en 2015.

5.1 Syntaxe

La syntaxe du premier ordre est stratifiée entre termes et formules, qu'il faut bien distinguer. Les termes représentent des individues (entiers naturels, listes, etc.) tandis que les formules représentent des propositions, i.e. des énoncés à propos de ces individus. On utilisera des variables pour représenter des termes arbitraires ; les variables ne pourront représenter des formules. On pourra enfin quantifier sur des termes en représentant un terme arbitraire par une variable. On ne pourra *pas* quantifier sur les formules : cela serait de la logique du *second* ordre.

5.1.1 Termes

Une *signature* est un ensemble $\mathcal{F}$ dont les éléments seront appelés *symboles de fonction*. Chaque symbole $f \in \mathcal{F}$ est muni d'une *arité* $a(f) \in \mathbb{N}$ qui fixe le nombre d'arguments. On se donne de plus un ensemble infini $\mathcal{X}$ de *symboles de variables*, disjoint de $\mathcal{F}$.

Définition 5.1.1. *L'ensemble $T(\mathcal{F}, \mathcal{X})$ des* termes *sur la signature $\mathcal{F}$ et les variables $\mathcal{X}$ est le plus petit ensemble tel que :*

- *$X \subseteq T(\mathcal{F}, \mathcal{X})$;*
- *si $f \in \mathcal{F}, a(f) = n$ et $t_1, \ldots, t_n \in T(\mathcal{F}, \mathcal{X})$, alors $f(t_1, \ldots, t_n) \in T(\mathcal{F}, \mathcal{X})$.*

Les termes clos*, i.e. sans variables, sont les éléments de $\mathcal{T}(\mathcal{F}, \emptyset)$, qu'on notera simplement $T(\mathcal{F})$.*

Un terme doit être vu comme un arbre fini étiqueté par $\mathcal{F}$ et $\mathcal{X}$.

On note $\mathsf{fv}(t)$ l'ensemble des variables apparaissant dans le terme t. C'est aussi le plus petit ensemble S tel que $t \in \mathcal{T}(\mathcal{F}, S)$.

Exemple 5.1.1. Si l'on suppose que $\mathcal{F}$ est composé des symboles $+, 0, s$ d'arités respectives $2, 0, 1$, et $x \in \mathcal{X}$, alors

$$+(x, x) \qquad \text{et} \qquad +(+(0, +(x, x)), x)$$

sont des termes de $T(\mathcal{F}, \mathcal{X})$. On utilise parfois l'écriture en notation infixée pour certains symboles usuels. Par exemple, $+(0, s(0))$ s'écrira aussi $0 + s(0)$.

Dans les exemples, $\mathcal{F}$ est donné en listant ses éléments avec, entre parenthèses, l'arité du symbole correspondant.

Exemple 5.1.2. Si $\mathcal{F} = \{\mathsf{nil}(0), \mathsf{cons}(2), @(2)\}$ et $x, y, z \in \mathcal{X}$, $@(\mathsf{cons}(x, y), z) \in T(\mathcal{F}, \mathcal{X})$.

Exemple 5.1.3. Si $\mathcal{F} = \{0(0), s(1), +(2), \times(2)\}$ et $x, y \in \mathcal{X}$, $\times(s(x), y) \in T(\mathcal{F}, \mathcal{X})$. On écrit aussi $\times(s(x), y)$ en notation infixe : $s(x) \times y$.

On notera que $\mathcal{F}$ peut être vide, auquel cas $T(\mathcal{F})$ est aussi vide. La réciproque n'est pas vraie. Bien sûr, $T(\mathcal{F}, \mathcal{X})$ n'est jamais vide.

5.1.2 Formules atomiques

On se donne un ensemble $\mathcal{P}$ dont les éléments seront appelés *symboles de prédicat*. Chacun de ces symboles est à nouveau muni d'une arité. On suppose $\mathcal{P}$ disjoint de $\mathcal{F}$ et de $\mathcal{X}$.

Les termes $P(t_1, \ldots, t_n)$ où $t_1, \ldots, t_n \in T(\mathcal{F}, \mathcal{X})$ et $P \in \mathcal{P}$ est d'arité n sont appelés *formules atomiques*.

5.1.3 Formules du premier ordre

Définition 5.1.2. *L'ensemble $CP_1(\mathcal{P}, \mathcal{F}, \mathcal{X})$ des formules du premier ordre sur les symboles de prédicat $\mathcal{P}$, les symboles de fonction $\mathcal{F}$ et les variables $\mathcal{X}$ est le plus petit ensemble tel que :*

- *les formules atomiques sont dans $CP_1(\mathcal{P}, \mathcal{F}, \mathcal{X})$;*
- *Si $\phi, \psi \in CP_1(\mathcal{P}, \mathcal{F}, \mathcal{X})$ et $x \in \mathcal{X}$ alors les formules suivantes sont toutes dans $CP_1(\mathcal{P}, \mathcal{F}, \mathcal{X})$:*

$$\bot,\ \top,\ \phi \wedge \psi,\ \phi \vee \psi,\ \neg\phi,\ \phi \Rightarrow \psi,\ \forall x.\phi,\ \exists x.\phi.$$

Remarquons que, lorsque tous les symboles de $\mathcal{P}$ sont d'arité 0, les formules sans quantificateur de $CP_1(\mathcal{P}, \mathcal{F}, \mathcal{X})$ sont aussi des formules de la logique propositionnelle.

On précisera rarement l'ensemble $\mathcal{X}$ utilisé, car il n'a (dans le cadre de ce cours) par grande importance : on supposera simplement qu'il est infini. Les symboles de $\mathcal{X}$ sont notés avec les lettres x, y, z.

Exemple 5.1.4. Si $\mathcal{P} = \{B(1)\}$ et $\mathcal{X} = \{x, y, z, \ldots\}$,

$$\exists x.\ (B(x) \Rightarrow\ (\forall y.\ B(y)))$$

est une formule de $CP_1(\mathcal{P}, \mathcal{F}, \mathcal{X})$. On l'appelle la formule du buveur.

Exemple 5.1.5. La formule suivante n'est pas une formule du premier ordre :

$$\forall P.(P(0) \wedge \forall x.P(x) \Rightarrow P(s(x))) \Rightarrow \forall x.P(x)$$

5.1.4 Variables libres et variables liées

Les quantificateurs *lient* les variables. On définit ainsi $\mathsf{fv}(\phi)$, l'ensemble des variables libres (*free variables*) d'une formule ϕ, par récurrence sur la formule :

$$\begin{array}{rcl}
\mathsf{fv}(P(t_1, \ldots, t_n)) & = & \mathsf{fv}(t_1) \cup \ldots \cup \mathsf{fv}(t_n) \quad \text{pour tout } P \in \mathcal{P} \text{ d'arité } n \\
\mathsf{fv}(\bot) = \mathsf{fv}(\top) & = & \emptyset \\
\mathsf{fv}(\phi \wedge \psi) = \mathsf{fv}(\phi \vee \psi) & = & \mathsf{fv}(\phi) \cup \mathsf{fv}(\psi) \\
\mathsf{fv}(\phi \Rightarrow \psi) & = & \mathsf{fv}(\phi) \cup \mathsf{fv}(\psi) \\
\mathsf{fv}(\neg\phi) & = & \mathsf{fv}(\phi) \\
\mathsf{fv}(\exists x.\phi) = \mathsf{fv}(\forall x.\phi) & = & \mathsf{fv}(\phi) \setminus \{x\}
\end{array}$$

Quand $\mathsf{fv}(\phi) = \emptyset$, on dit que ϕ est une formule *close*.

On définit ensuite $\mathsf{bv}(\phi)$, l'ensemble des variables liées (*bound variables*) de ϕ, comme l'ensemble des variables x tel que ϕ contient une sous-formule de la forme $\exists x.\psi$ ou $\forall x.\psi$.

Exemple 5.1.6. Si ϕ est la formule

$$P(x) \wedge \exists x.Q(f(x)) \wedge \exists x.\exists z.Q(g(x,y,z))$$

alors $\mathsf{fv}(\phi) = \{x, y\}$ et $\mathsf{bv}(\phi) = \{x, z\}$.

5.2 Sémantique

5.2.1 $\mathcal{F}$-algèbres

Etant donné une signature $\mathcal{F}$ une *$\mathcal{F}$-algèbre* $\mathcal{A}$ est constituée d'un ensemble non vide $D_{\mathcal{A}}$ appelé son *domaine* et, pour chaque symbole de fonction $f \in \mathcal{F}$ d'arité n, d'une fonction $f_{\mathcal{A}} : {D_{\mathcal{A}}}^n \to D_{\mathcal{A}}$.

Exemple 5.2.1. $T(\mathcal{F})$ et $T(\mathcal{F}, \mathcal{X})$ sont des $\mathcal{F}$-algèbres, avec $f_{T(\mathcal{F})}(t_1, t_2) = f(t_1, t_2)$ et de même pour $f_{T(\mathcal{F},\mathcal{X})}$.

Exemple 5.2.2. Soit $\mathcal{F} = \{0(0), s(1), +(2)\}$. La $\mathcal{F}$-algèbre canonique pour cette signature est :

$$\big(\mathbb{N}, 0, (n \mapsto n+1), (x, y \mapsto x+y)\big)$$

Une autre $\mathcal{F}$-algèbre, construite sur l'ensemble des rationnels strictement positifs, est :

$$\big(\mathbb{Q}_+, 1, (x \mapsto x \div 2), (x, y \mapsto x \div y)\big)$$

Définition 5.2.1 (Affectation)**.** *Si $\mathcal{A}$ est une $\mathcal{F}$-algèbre, une* $\mathcal{A}$-affectation *est une application σ de $\mathcal{X}$ dans $\mathcal{A}$.*

Si $a_1, \ldots, a_n \in \mathcal{A}$, et $\mathcal{X} = \{x_1, \ldots, x_n\}$, on note $\{x_1 \mapsto a_1, \ldots, x_n \mapsto a_n\}$ l'affectation σ telle que $\sigma(x_i) = a_i$ pour tout i. Cette notation suppose que $x_1, \ldots, x_n$ sont des variables distinctes.

Définition 5.2.2 (Interprétation)**.** *Si $t \in T(\mathcal{F}, \{x_1, \ldots, x_n\})$ et si σ est une affectation dans la $\mathcal{F}$-algèbre $\mathcal{A}$ telle que $\{x_1, \ldots, x_n\} \subseteq \mathrm{Dom}(\sigma)$, on définit $[\![t]\!]_{\sigma,\mathcal{A}}$ par induction structurelle sur t :*

$$\begin{aligned} [\![x]\!]_{\sigma,\mathcal{A}} &= \sigma(x) \\ [\![f(t_1, \ldots, t_n)]\!]_{\sigma,\mathcal{A}} &= f_{\mathcal{A}}([\![t_1]\!]_{\sigma,\mathcal{A}}, \ldots, [\![t_n]\!]_{\sigma,\mathcal{A}}) \end{aligned}$$

Une substitution θ est une affectation de domaine $\mathcal{X}$ et à images dans la $\mathcal{F}$-algèbre des termes. L'application d'une substitution θ à un terme t, notée $t\theta$, est simplement définie comme $[\![t]\!]_{\theta,T(\mathcal{F},\mathcal{X})}$.

On définit $\mathrm{Dom}(\theta) = \{x \in \mathcal{X} \mid x \neq \theta(x)\}$, c'est l'ensemble des variables sur lesquelles θ "fait quelquechose", qui sera bien souvent fini. On notera parfois une substitution $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$, il est alors implicite que la substitution se comporte comme la fonction identité sur les autres variables.

Exemple 5.2.3. $[\![x + x]\!]_{\{x\mapsto 1\},\mathbb{N}} = 2$

Exemple 5.2.4. $[\![x + x]\!]_{\{x\mapsto s(0)\},T(\mathcal{F},\mathcal{X})} = s(0) + s(0) = (x + x)\{x \mapsto s(0)\}$

Le résultat suivant explicite le lien entre substitution et interprétation, et se prouve facilement par induction sur t.

Proposition 5.2.1 (substitution)**.** *Soit $\mathcal{A}$ une $\mathcal{F}$-algèbre, $t \in T(\mathcal{F}, \mathcal{X})$, $\theta : \mathcal{X} \to T(\mathcal{F}, \mathcal{X})$ une substitution, et σ une affectation de domaine $\mathcal{X}$. On a $[\![t\theta]\!]_{\sigma,\mathcal{A}} = [\![t]\!]_{\theta\sigma,\mathcal{A}}$, où $\theta\sigma = x \mapsto [\![\theta(x)]\!]_{\sigma,\mathcal{A}}$.*

Un autre résultat élémentaire bien utile exprime que $[\![t]\!]_{\sigma,\mathcal{A}}$ ne dépend que des valeurs $\sigma(x)$ pour $x \in \mathsf{fv}(t)$.

Proposition 5.2.2. *Soit $\mathcal{A}$ une $\mathcal{F}$-algèbre, $t \in T(\mathcal{F}, \mathcal{X})$, σ et σ' des affectations. Si $\sigma|_{\mathsf{fv}(t)} = \sigma'|_{\mathsf{fv}(t)}$, alors $[\![t]\!]_{\sigma,\mathcal{A}} = [\![t]\!]_{\sigma',\mathcal{A}}$.*

5.2.2 $\mathcal{F}, \mathcal{P}$-structures

Définition 5.2.3. *Une $\mathcal{F}, \mathcal{P}$-*structure *$\mathcal{S}$ est donnée par une $\mathcal{F}$-algèbre $\mathcal{A}$ et, pour chaque symbole de prédicat $P \in \mathcal{P}$ d'arité n une relation $P_\mathcal{S} \subseteq D_\mathcal{A}^n$, où $D_\mathcal{A}$ est le domaine de $\mathcal{A}$.*

On confondra parfois une structure et la $\mathcal{F}$-algèbre sous-jacente.

Soit ϕ une formule, $\mathcal{S}$ une $\mathcal{F}, \mathcal{P}$-structure d'algèbre sous-jacente $\mathcal{A}$ et σ une $\mathcal{A}$-affectation σ telle que $\mathsf{fv}(\phi) \subseteq \mathrm{Dom}(\sigma)$. On définit la relation de satisfaction $\mathcal{S}, \sigma \models \phi$ par récurrence sur ϕ :

- $\mathcal{S}, \sigma \models P(t_1, \ldots, t_n)$ si et seulement si $([\![t_1]\!]_{\sigma,\mathcal{A}}, \ldots, [\![t_n]\!]_{\sigma,\mathcal{A}}) \in P_\mathcal{S}$;
- $\mathcal{S}, \sigma \models \phi * \psi$ où $*$ est l'un des connecteurs logiques binaires est défini, comme en calcul propositionnel, à partir des modèles de ϕ et des modèles de ψ, par exemple $\mathcal{S}, \sigma \models \phi \vee \psi$ si et seulement si ($\mathcal{S}, \sigma \models \phi$ ou $\mathcal{S}, \sigma \models \psi$) ;
- $\mathcal{S}, \sigma \models \neg\phi$ ssi $\mathcal{S}, \sigma \not\models \phi$;
- $\mathcal{S}, \sigma \models \exists x.\phi$ ssi il existe $a \in D_\mathcal{A}$ tel que $\mathcal{S}, \sigma\{x \mapsto a\} \models \phi$;
- $\mathcal{S}, \sigma \models \forall x.\phi$ ssi pour tout $a \in D_\mathcal{A}$ on a $\mathcal{S}, \sigma\{x \mapsto a\} \models \phi$.

Ici $\sigma\{x \mapsto a\}$ désigne l'affectation σ' qui coïncide avec σ sur $\mathcal{X} \setminus \{x\}$ et telle que $\sigma'(x) = a$.

Proposition 5.2.3. *Soit ϕ une formule, $\mathcal{S}$ une structure, et σ,σ' des affectations coïncidant sur $\mathsf{fv}(\phi)$. On a $\mathcal{S}, \sigma \models \phi$ ssi $S, \sigma' \models \phi$.*

Quand ϕ est une formule close (i.e. sans variable libre) on s'autorisera à écrire simplement $\mathcal{S} \models \phi$ pour signifier que $\mathcal{S}, \sigma \models \phi$ pour un σ quelconque.

5.2.3 Modèle, validité, conséquence logique

Une structure $\mathcal{S}$ est un *modèle* d'une formule close ϕ si $\mathcal{S} \models \phi$. Un modèle d'un ensemble de formules closes est une structure qui satisfait toutes les formules de l'ensemble. Une formule close est valide quand elle est satisfaite dans tout modèle.

Plus généralement, un modèle d'une formule ϕ telle que $\mathsf{fv}(\phi) = \{x_1, \ldots, x_n\}$ est donné par une structure S et une affectation σ tel que $\mathcal{S}, \sigma \models \phi$, et ϕ est valide quand sa clôture universelle $\forall x_1 \ldots \forall x_n.\ \phi$ est valide.

Ces notions se généralisent à des ensembles de formules, comme suit.

Définition 5.2.4. *Si $\mathcal{E}$ est un ensemble de formules sans variable libre et ϕ est une formule sans variable libre, alors ϕ* est une conséquence logique de *$\mathcal{E}$, ce que l'on note $\mathcal{E} \models \phi$, si, pour toute structure $\mathcal{S}$, $\mathcal{S} \models \mathcal{E}$ entraîne $\mathcal{S} \models \phi$.*

Deux ensembles de formules sans variable libre $\mathcal{E}_1$ et $\mathcal{E}_2$ sont logiquement équivalents *si toute formule de $\mathcal{E}_2$ est conséquence logique de $\mathcal{E}_1$ et, réciproquement, toute formule de $\mathcal{E}_1$ est conséquence logique de $\mathcal{E}_2$.*

Exemple 5.2.5. Soit $\phi \stackrel{\text{def}}{=} \exists x.\forall y.P(x, y)$ et $\psi \stackrel{\text{def}}{=} \forall y.\exists x.P(x, y)$. La formule ϕ a pour conséquence logique ψ, mais la réciproque n'est pas vraie. Autrement dit, $\phi \Rightarrow \psi$ est valide mais il existe une structure ne satisfaisant pas $\psi \Rightarrow \phi$.

Exemple 5.2.6. La formule du buveur (cf. exemple 5.1.4) est valide.

5.3 Substitution

L'application d'une substitution à une formule pose deux problèmes : il ne faut pas remplacer des variables liées $((\forall x.p(x))\{x \mapsto t\}$ ne doit pas être $(\forall x.p(t)))$; il ne faut pas que les lieurs de la formule "capturent" des variables des termes insérés par la substitution $((\forall x.p(y))\{y \mapsto x\}$ ne doit pas être $(\forall x.p(x)))$. Pour éviter ces problème on aura recours à l'α-équivalence, qui identifie par exemple $\forall x.p(x)$ et $\forall z.p(z)$, ou encore $\forall x.p(y)$ et $\forall z.p(y)$.

Définition 5.3.1. *Si θ est une substitution, on pose*

$$\mathsf{vars}(\theta) = \mathrm{Dom}(\theta) \cup \bigcup_{x \in \mathrm{Dom}(\theta)} \mathsf{fv}(\theta(x)).$$

On dit que θ est applicable à une formule ϕ quand $\mathsf{bv}(t) \cap \mathsf{vars}(\theta) = \emptyset$, et l'on définit alors $\phi\theta$ par induction sur ϕ :

$$\begin{array}{rcll} \bot\theta & \stackrel{def}{=} & \bot & \\ \top\theta & \stackrel{def}{=} & \top & \\ (P(t_1,\ldots,t_n))\theta & \stackrel{def}{=} & P(t_1\theta,\ldots,t_n\theta) & \\ (\neg\phi)\theta & \stackrel{def}{=} & \neg(\phi\theta) & \\ (\phi * \psi)\theta & \stackrel{def}{=} & \phi\theta * \psi\theta & \textit{pour } * \in \{\wedge,\vee,\Rightarrow\} \\ (\mathcal{Q}x.\phi)\theta & \stackrel{def}{=} & \mathcal{Q}x.(\phi\theta) & \textit{pour } \mathcal{Q} \in \{\exists,\forall\} \end{array}$$

Autrement dit, $\phi\theta$ est obtenue à partir de t en remplaçant chaque terme t apparaissant dans ϕ (nécessairement en argument d'un prédicat) par $t\theta$.

Proposition 5.3.1 (substitution)**.** *Soit ϕ une formule, θ une substitution applicable à ϕ, $\mathcal{S}$ une structure, et σ une affectation de domaine $\mathcal{X}$. On a $\mathcal{S},\sigma \models \phi\theta$ ssi $\mathcal{S},\theta\sigma \models \phi$.*

Démonstration. On procède par induction sur ϕ, les cas intéressants étant ceux des quantificateurs. Considérons le cas où ϕ est de la forme $\forall x.\psi$. On a les équivalences suivantes :

$\mathcal{S},\sigma \models \forall x.\psi\theta$

ssi pour tout $a \in \mathcal{S}$, $\mathcal{S},\sigma\{x \mapsto a\} \models \psi\theta$ par définition de la satisfaction

ssi pour tout $a \in \mathcal{S}$, $\mathcal{S},\theta(\sigma\{x \mapsto a\}) \models \psi$ par hypothèse d'induction sur ψ

On remarque alors que $\theta(\sigma\{x \mapsto a\}) = (\theta\sigma)\{x \mapsto a\}$: les deux affectations valent a en x car $x \in \mathsf{bv}(\phi)$ donc $x \notin \mathrm{Dom}(\theta)$, i.e. $\theta(x) = x$; elles coïncident sur toute autre variable y car $[\![\theta(y)]\!]_{\sigma\{x\mapsto a\}} = [\![\theta(y)]\!]_\sigma$ puisque $x \in \mathsf{bv}(\phi)$ entraîne $x \notin \mathsf{fv}(\theta(y))$. On conclut alors aisément par les équivalences suivantes :

$\mathcal{S},\sigma \models \forall x.\psi\theta$

ssi pour tout $a \in \mathcal{S}$, $\mathcal{S},\theta(\sigma\{x \mapsto a\}) \models \psi$

ssi pour tout $a \in \mathcal{S}$, $\mathcal{S},(\theta\sigma)\{x \mapsto a\} \models \psi$

ssi $\mathcal{S},\theta\sigma \models \forall x.\psi$

□

On définit maintenant l'α-équivalence qui va nous permettre, pour tout θ et ϕ, de toujours nous ramener à une formule α-equivalente (et logiquement équivalente) ψ pour laquelle θ est applicable.

Définition 5.3.2. *La relation d'α-renommage est définie par*

$$\mathcal{Q}x.\phi \to_\alpha \mathcal{Q}y.\phi\{x \mapsto y\}$$

où $\mathcal{Q}$ est un quantificateur, ϕ une formule, x et y sont des variables telles que $y \notin \mathsf{fv}(\phi)$ et que la substitution $\{x \mapsto y\}$ s'applique à ϕ.

L'α-équivalence $\equiv_\alpha$ est la plus petite congruence[1] *contenant l'α-renommage.*

1. Une congruence est une relation d'équivalence qui passe au contexte : ici cela signifie que $\phi \equiv_\alpha \psi$ entraîne $\phi \wedge \phi' \equiv_\alpha \psi \wedge \phi'$, $\forall x.\phi \equiv_\alpha \forall x.\psi$, etc.

$$\forall x. \quad x = x$$

$$\forall x, y. \quad x = y \Leftrightarrow y = x$$

$$\forall x, y, z. \quad (x = y \wedge y = z) \Rightarrow x = z$$

Pour tout $n \in \mathbb{N}$ et tout symbole $f \in \mathcal{F}$ d'arité n :

$$\forall x_1, \ldots, x_n, y_1, \ldots, y_n. \quad (\bigwedge_{1 \leq i \leq n} x_i = y_i) \Rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$

Pour tout $n \in \mathbb{N}$ et tout symbole $P \in \mathcal{P}$ d'arité n :

$$\forall x_1, \ldots, x_n, y_1, \ldots, y_n. \quad (\bigwedge_{1 \leq i \leq n} x_i = y_i) \Rightarrow P(x_1, \ldots, x_n) \Rightarrow P(y_1, \ldots, y_n)$$

FIGURE 5.1 – Axiomes de l'égalité : $\mathcal{A}_{eq}$

Proposition 5.3.2. *Pour tous ϕ et θ il existe $\psi \equiv_\alpha \phi$ tel que θ s'applique à ψ.*

Idée de preuve. On peut toujours changer, par α-renommage, les variables liées de ϕ (en traitant les quantificateurs du plus interne au plus externe) pour éviter l'ensemble fini $\mathsf{vars}(\theta)$. □

Proposition 5.3.3. *Deux formules α-équivalentes sont logiquement équivalentes : pour tous $\mathcal{S}$, σ, ϕ et ψ tels que $\phi \equiv_\alpha \psi$, on a $\mathcal{S}, \sigma \models \phi$ ssi $\mathcal{S}, \sigma \models \psi$.*

Démonstration. Il suffit de le montrer pour l'α-renommage, cela passe ensuite directement à la congruence. On détaille seulement le cas d'une quantification existentielle. Soit $\exists x.\phi \to_\alpha \exists y.\phi\{x \mapsto y\}$ avec $y \notin \mathsf{fv}(\phi)$ et $\{x \mapsto y\}$ s'appliquant à ϕ. On vérifie :

$\mathcal{S}, \sigma \models \exists x.\phi$

ssi il existe a tel que $\mathcal{S}, \sigma\{x \mapsto a\} \models \phi$

ssi il existe a tel que $\mathcal{S}, \{x \mapsto y\}(\sigma\{y \mapsto a\}) \models \phi$

ssi il existe a tel que $\mathcal{S}, \sigma\{y \mapsto a\} \models \phi\{x \mapsto y\}$

ssi $\mathcal{S}, \sigma \models \exists y.\phi\{x \mapsto y\}$

On notera bien l'utilisation de $y \notin \mathsf{fv}(\phi)$ pour la deuxième étape, et de l'applicabilité de $\{x \mapsto y\}$ à ϕ pour l'étape suivante. □

5.4 Exemples de théories

Exemple 5.4.1. L'ensemble des formules données dans la figure 5.1 est connu sous le nom d'*axiomes de l'égalité*. C'est un ensemble fini de formules si $\mathcal{F}$ et $\mathcal{P}$ sont finis. On note, comme nous en avons l'habitude, $u = v$ au lieu de $=(u, v)$.

Toute structure $\mathcal{S}$ dans laquelle $=$ est interprété par l'égalité sur $D_\mathcal{S}$ est un modèle de $\mathcal{A}_{eq}$.

Proposition 5.4.1. *Soit $\mathcal{S}$ une structure satisfaisant $\mathcal{A}_{eq}$. Il existe une structure $\mathcal{S}'$ dans laquelle $=$ est interprété comme l'égalité (sur $D_{\mathcal{S}'}$) telle que pour toute formule close ϕ, on a $\mathcal{S} \models \phi$ ssi $\mathcal{S}' \models \phi$.*

Exemple 5.4.2. On considère ici $\mathcal{F} = \{0(0), s(1)\}$ et $\mathcal{P} = \{\geq(2)\}$. L'algèbre des entiers

$$
\begin{array}{lrcl}
\forall x. & 0 + x = x & & \\
\forall x, y. & s(x) + y = s(x + y) & & \\
\forall x. & 0 \times x = 0 & & \\
\forall x, y. & s(x) \times y = (x \times y) + y & & \\
\forall x. & s(x) \neq 0 & & \\
\forall x, y. & s(x) = s(y) & \Rightarrow & x = y \\
\forall x. \exists y. & x \neq 0 & \Rightarrow & x = s(y)
\end{array}
$$

FIGURE 5.2 – Arithmétique élémentaire : $\mathcal{A}_{EL}$

naturels (avec l'ordre habituel sur les entiers) satisfait la formule suivante :

$$
\begin{array}{ll}
& \forall x. \geq(x, 0) \\
\wedge & \forall x. \geq(x, x) \\
\wedge & \forall x, y. \geq(x, y) \Rightarrow \geq(s(x), s(y))
\end{array}
$$

Cette formule est aussi satisfaite par d'autres structures sur l'algèbre des entiers naturels, par exemple la structure où $\geq$ est toujours vrai.

Exercice 5.4.1. *On considère cette fois* $\mathcal{F} = \{@(2), \mathsf{cons}(2), \mathsf{nil}(0)\}$ *et* $\mathcal{P} = \{=(2)\}$ *et la formule suivante, censée définir* $@$ *:*

$$\forall x, y, z. \quad @(\mathsf{nil}, x) = x \ \wedge \ @(\mathsf{cons}(x, y), z) = \mathsf{cons}(x, @(y, z))$$

Donner un exemple de structure $\mathcal{S}$ *qui satisfait ces formules ainsi que les axiomes de l'égalité, mais dans laquelle* $\mathcal{S} \not\models \forall x, y, z.\ @(x, @(y, z)) = @(@(x, y), z)$.

Exemple 5.4.3. On considère ici $\mathcal{F} = \{0(0), s(1), +(2), \times(2)\}$ et $\mathcal{P} = \{= (2)\}$. Comme nous en avons l'habitude, nous notons $u \neq v$ au lieu de $\neg(u = v)$, $u + v$ au lieu de $+(u, v)$, $u \times v$ au lieu de $\times(u, v)$. L'ensemble des sept formules de la figure 5.2, auquel on ajoute les axiomes de l'égalité, est connu sous le nom d'*arithmétique élémentaire*. Il s'agit de la plus simple des tentatives pour axiomatiser les entiers naturels : la structure construite sur les entiers naturels, dans laquelle tous ces symboles ont leur interprétation usuelle, est un modèle de l'arithmétique élémentaire. En revanche, il existe des modèles de $\mathcal{A}_{EL}$ qui ne satisfont pas la commutativité de l'addition $\forall x, y.\ x + y = y + x$.

Chapitre 6

Théorie des modèles

Nous abordons deux résultats centraux : le théorème de Skolem et celui de Herbrand, qui ont notamment pour conséquence le résultat de compacité pour la logique du premier ordre, mais aussi des résultats sur la cardinalité des modèles, ainsi qu'une porte ouverte vers des systèmes de preuve complets.

6.1 Mise en forme prénexe

Définition 6.1.1 (Forme prénexe)**.** *Une formule est* en forme prénexe *si elle est de la forme*

$$Q_1 x_1 \dots Q_n x_n.\ \varphi$$

où $Q_i \in \{\forall, \exists\}$ *pour tout* $i = 1, \dots, n$ *et* φ *est sans quantificateurs.*

On considère les règles de transformation suivantes.

$$\begin{array}{rcll}
(Qx.\ \varphi) * \psi & \rightsquigarrow & Qx.\ (\varphi * \psi) & \text{si } x \notin \mathsf{fv}(\psi),\ \text{et } * \in \{\wedge, \vee\} \\
\psi * (Qx.\ \varphi) & \rightsquigarrow & Qx.\ (\psi * \varphi) & \text{si } x \notin \mathsf{fv}(\psi),\ \text{et } * \in \{\wedge, \vee, \Rightarrow\} \\
\neg\exists x.\ \varphi & \rightsquigarrow & \forall x.\ \neg\varphi & \\
\neg\forall x.\ \varphi & \rightsquigarrow & \exists x.\ \neg\varphi & \\
(\forall x.\ \varphi) \Rightarrow \psi & \rightsquigarrow & \exists x.\ (\varphi \Rightarrow \psi) & \text{si } x \notin \mathsf{fv}(\psi) \\
(\exists x.\ \varphi) \Rightarrow \psi & \rightsquigarrow & \forall x.\ (\varphi \Rightarrow \psi) & \text{si } x \notin \mathsf{fv}(\psi)
\end{array}$$

Exemple 6.1.1. On peut appliquer ces transformations de deux façons différentes à la formule ci-dessous.

$$\begin{array}{ccc}
\big(\forall x.\ B(x)\big) \Rightarrow \big(\forall x.\ B(x)\big) & & \\
\equiv_\alpha & & \\
\big(\forall x.\ B(x)\big) \Rightarrow \big(\forall y.\ B(y)\big) & & \\
\swarrow & \searrow & \\
\exists x.\ \Big(B(x) \Rightarrow \big(\forall y.\ B(y)\big)\Big) & & \forall y.\ \Big(\big(\forall x.\ B(x)\big) \Rightarrow B(y)\Big) \\
\downarrow & & \downarrow \\
\exists x. \forall y.\ \big(B(x) \Rightarrow B(y)\big) & & \forall y. \exists x.\ \big(B(x) \Rightarrow B(y)\big)
\end{array}$$

On étend la notion d'équivalence logique aux formules ayant des variables libres : deux formules φ, ψ sont logiquement équivalentes si pour toute structure $\mathcal{S}$, pour toute $\mathcal{S}$-affectation σ interprétant les variables libres de ces formules, $\mathcal{S}, \sigma \models \varphi$ ssi $\mathcal{S}, \sigma \models \psi$.

Lemme 6.1.1 (Correction des règles)**.** *Si* $\varphi \rightsquigarrow \psi$*, alors* φ *et* ψ *sont logiquement équivalentes.*

Démonstration. Comme l'équivalence logique est une congruence [1], il suffit de prouver ce lemme lorsque la réécriture a lieu à la racine de la formule. On raisonne par analyse de cas. Considérons la règle

$$\underbrace{(\exists x.\ \varphi_1) \vee \varphi_2}_{=\phi} \ \rightsquigarrow\ \underbrace{\exists x.\ (\varphi_1 \vee \varphi_2)}_{=\psi}.$$

On veut montrer que pour tout $\mathcal{S}, \sigma$, on a $\mathcal{S}, \sigma \models \phi$ ssi $\mathcal{S}, \sigma \models \psi$. Soit $\mathcal{S}, \sigma$ fixés.

— Supposons que $\mathcal{S}, \sigma \models \varphi$. Alors $\mathcal{S}, \sigma \models \exists x.\ \varphi_1$ ou $\mathcal{S}, \sigma \models \varphi_2$.

Si	$\mathcal{S}, \sigma \models \exists x.\ \varphi_1$	(1^{er} cas)
alors	il existe $a \in D_{\mathcal{S}}$ tel que $\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_1$	(def. de $\models$)
donc	il existe $a \in D_{\mathcal{S}}$ tel que $\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_1 \vee \varphi_2$	(def. de $\models$)
donc	$\mathcal{S}, \sigma \models \exists x.\ (\varphi_1 \vee \varphi_2)$	(def. de $\models$)
Si	$\mathcal{S}, \sigma \models \varphi_2$	($2^{\text{ème}}$ cas)
alors	il existe $a \in D_{\mathcal{S}}.\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_2$	($D_{\mathcal{S}} \neq \emptyset$)
donc	il existe $a \in D_{\mathcal{S}}$ tel que $\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_1 \vee \varphi_2$	(def. de $\models$)
donc	$\mathcal{S}, \sigma \models \exists x.\ (\varphi_1 \vee \varphi_2)$	(def. de $\models$)

Dans les deux cas, $\mathcal{S}, \sigma \models \psi$.

— Supposons maintenant que $\mathcal{S}, \sigma \models \psi$. Alors il existe $a \in D_{\mathcal{S}}$ tel que $\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_1 \vee \varphi_2$ (par def de $\models$).

Si	$\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_1$	(premier cas)
alors	$\mathcal{S}, \sigma \models \exists x.\ \varphi_1$	(par def. de $\models$)
donc	$\mathcal{S}, \sigma \models (\exists x.\ \varphi_1) \vee \varphi_2$	(par def. de $\models$)
Si	$\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \varphi_2$	(deuxième cas)
alors	$\mathcal{S}, \sigma \models \varphi_2$	($x \notin \mathsf{fv}(\varphi_2)$)
donc	$\mathcal{S}, \sigma \models (\exists x.\ \varphi_1) \vee \varphi_2$	(par def. de $\models$)

Dans les deux cas, $\mathcal{S}, \sigma \models \varphi$.

Les autres règles se traitent de façon similaire, voire parfois plus simplement car certaines règles ne reposent pas sur l'hypothèse que $D_{\mathcal{S}} \neq \emptyset$. □

Théorème 6.1.1. *Pour toute formule φ, on peut calculer une formule φ' logiquement équivalente à φ et en forme prénexe.*

Démonstration. Un algorithme consiste à appliquer les transformations précédentes (et, si besoin, l'α-renommage) jusqu'à obtenir une forme prénexe. On a vu que ces transformations sont correctes. L'argument de terminaison est laissé en exercice. □

6.2 Forme normale négative

Définition 6.2.1 (Littéral)**.** *Un* littéral *est une formule de la forme $P(t_1, \ldots t_n)$ ou sa négation.*

Définition 6.2.2 (Forme normale négative)**.** *Une formule φ est en* forme normale négative *si*

1. *φ ne contient aucune implication, et*
2. *les seules sous-formules de φ de la forme $\neg\psi$ sont les littéraux.*

Exemple 6.2.1. La formule $\exists x.\big((\neg P(x)) \vee \exists y.Q(x,y)\big)$ est en forme normale négative, mais les formules $\exists x.\neg\big(P(x) \vee \exists y.Q(x,y)\big)$ et $\exists x.P(x) \Rightarrow \exists y.Q(x,y)$ ne le sont pas.

Proposition 6.2.1. *Pour toute formule φ, on peut calculer une formule φ' en forme normale négative logiquement équivalente à φ.*

1. Si on remplace dans une formule Φ une sous-formule ϕ par une formule ψ logiquement équivalente, alors on obtient une formule logiquement équivalente à Φ.

Démonstration. On considère les règles de réécriture suivantes.

$$\begin{array}{rcl} \varphi \Rightarrow \psi & \rightsquigarrow & \neg\varphi \vee \psi \\ \neg(\forall x.\ \varphi) & \rightsquigarrow & \exists x.\neg\varphi \\ \neg(\exists x.\ \varphi) & \rightsquigarrow & \forall x.\neg\varphi \\ \neg(\varphi \vee \psi) & \rightsquigarrow & (\neg\varphi) \wedge (\neg\psi) \\ \neg(\varphi \wedge \psi) & \rightsquigarrow & (\neg\varphi) \vee (\neg\psi) \\ \neg\neg\varphi & \rightsquigarrow & \varphi \end{array}$$

On laisse en exercice la preuve que toute suite de réécritures $\varphi_0 \rightsquigarrow \varphi_1 \rightsquigarrow \dots$ est finie. On observe aisément que, si $\varphi \not\rightsquigarrow$, alors φ est en forme normale négative. Enfin, si $\varphi \rightsquigarrow \psi$, alors φ et ψ sont logiquement équivalentes.

Ainsi, si φ est une formule arbitraire et si φ' est obtenue à partir de φ par une suite maximale de réécritures, alors φ' est en forme normale négative et logiquement équivalente à φ. □

6.3 Skolémisation

Nous abordons maintenant une transformation nettement plus surprenante, qui va permettre d'éliminer les quantificateurs existentiels : c'est la skolémisation.

Naturellement, nous allons utiliser ici la possibilité de travailler sur des formules en forme normale négative — sans cela, la distinction entre quantificateurs existentiels et universels n'aurait de sens que si on comptait le nombre de négations (et d'implications) au dessus des quantificateurs.

Attention ! la skolémisation d'une formule ne sera pas logiquement équivalente à la formule de départ, mais seulement équisatisfaisable, i.e. l'une est satisfaisable ssi l'autre est satisfaisable.

6.3.1 La transformation

On se donne un ensemble de symboles de fonctions $\mathcal{F}$ et un ensemble de symboles de prédicats $\mathcal{P}$. Pour chaque $(\mathcal{F},\mathcal{P})$-formule φ et pour chaque variable x, on se donne un nouveau symbole de fonction $f_{\varphi,x}$ d'arité $|\mathsf{fv}(\exists x.\varphi)|$. On note $\overline{\mathcal{F}}$ l'ensemble de symboles de fonction $\mathcal{F}$ augmenté de ces nouveaux symboles de fonction, autrement dit

$$\overline{\mathcal{F}} \ = \ \mathcal{F} \uplus \{f_{\varphi,x} \mid \varphi \text{ est une } (\mathcal{F},\mathcal{P})\text{-formule}, x \in X\}.$$

Enfin, pour tout φ, x, on se fixe une énumération $\vec{y}_{\varphi,x} = (y_1,\dots,y_n)$ de $\mathsf{fv}(\exists x.\varphi)$.

Définition 6.3.1 (skolémisation)**.** *Soit φ une $(\mathcal{F},\mathcal{P})$-formule. La skolémisée de φ, notée $\mathsf{Sk}(\varphi)$, est la $(\overline{\mathcal{F}},\mathcal{P})$-formule définie par récurrence sur la taille de φ comme suit.*

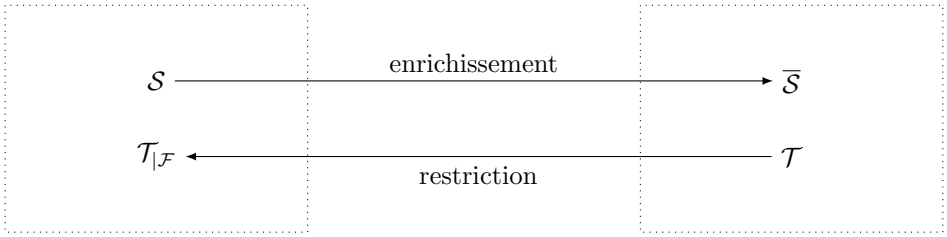
$$\begin{array}{lll} \mathsf{Sk}(P(t_1,\dots,t_n)) & = P(t_1,\dots,t_n) & \\ \mathsf{Sk}(\neg\varphi) & = \neg\mathsf{Sk}(\varphi) & \\ \mathsf{Sk}(\varphi * \psi) & = \mathsf{Sk}(\varphi) * \mathsf{Sk}(\psi) & (* \in \{\vee,\wedge,\Rightarrow\}) \\ \mathsf{Sk}(\forall x.\varphi) & = \forall x.\mathsf{Sk}(\varphi) & \\ \mathsf{Sk}(\exists x.\varphi) & = \mathsf{Sk}(\varphi)\{x \mapsto f_{\varphi,x}(\vec{y}_{\varphi,x})\} & \end{array}$$

En anticipant un peu sur la suite on remarquera que, même si $\mathsf{Sk}(\phi)$ est définie pour tout ϕ, il faudra supposer ϕ en forme normale négative pour obtenir le théorème de Skolem, qui dit que ϕ et $\mathsf{Sk}(\phi)$ sont équisatisfaisables.

Exemple 6.3.1. Soit φ la formule $\exists x\forall y\exists z.\ z * (x * y) \neq z$. On note c le symbole de Skolem d'arité 0 associé à x et $\forall y\exists z.z * (x * y) \neq z$, et f le symbole de Skolem d'arité 1 associé à z et $z * (x * y) \neq z$. Alors $\mathsf{Sk}(\varphi) = \forall y.\ f(c,y) * (c * y) \neq f(c,y)$.

6.3.2 Interprétation des symboles de Skolem

Il nous faut définir et étudier les deux foncteurs de structures suivants.



Soit $\mathcal{S}$ une $(\mathcal{F},\mathcal{P})$-structure. On veut construire à partir de $\mathcal{S}$ une $(\overline{\mathcal{F}},\mathcal{P})$-structure $\overline{\mathcal{S}}$. Pour être tout à fait constructif, on a besoin de se donner une fonction $\mathsf{choix} : 2^{D_\mathcal{S}} \to D_\mathcal{S}$ telle que pour toute partie $T \neq \emptyset$ de $D_\mathcal{S}$, $\mathsf{choix}(T) \in T$. On peut supposer donnée une telle fonction en invoquant l'axiome du choix. De plus, comme $D_\mathcal{S} \neq \emptyset$, on peut se donner un élément $a_0 \in D_\mathcal{S}$ tel que $\mathsf{choix}(\emptyset) = a_0$.

Soit x, φ fixés, et soient $y_1, \ldots, y_n$ les variables libres de $\exists x.\varphi$. Pour tout $b_1, \ldots b_n \in D_\mathcal{S}$, on pose

$$T_{x,\varphi}(b_1, \ldots, b_n) := \left\{ a \in D_\mathcal{S} \mid \mathcal{S}, \{\vec{y} \mapsto \vec{b}, x \mapsto a\} \models \varphi \right\}$$

l'ensemble des témoins pour $\exists x.\varphi$ dans $\mathcal{S}$ avec l'affectation $\{\vec{y} \mapsto \vec{b}\}$.

On peut maintenant définir $\overline{\mathcal{S}}$:

— le domaine de $\overline{\mathcal{S}}$ est celui de $\mathcal{S}$, i.e. $D_{\overline{\mathcal{S}}} = D_\mathcal{S}$;
— $P_{\overline{\mathcal{S}}} = P_\mathcal{S}$ pour tout symbole de prédicat $P \in \mathcal{P}$
— $f_{\overline{\mathcal{S}}} = f_\mathcal{S}$ pour tout symbole de fonction "original" $f \in \mathcal{F}$
— enfin, pour $f \in \overline{\mathcal{F}} \setminus \mathcal{F}$ un symbole de Skolem associé à x et φ, on pose $f_{\overline{\mathcal{S}}}(\vec{b}) = \mathsf{choix}(T_{x,\varphi}(\vec{b}))$.

Lemme 6.3.1. *Soit $\mathcal{S}$ une $(\mathcal{F},\mathcal{P})$-structure. Pour toute $(\mathcal{F},\mathcal{P})$-formule φ et pour toute affectation σ telle que* $\mathrm{Dom}(\sigma) \supseteq \mathsf{fv}(\varphi)$*,*

$$\mathcal{S}, \sigma \models \varphi \qquad \textit{ssi} \qquad \overline{\mathcal{S}}, \sigma \models \mathsf{Sk}(\varphi).$$

Démonstration. On raisonne par induction sur φ. Si φ est une formule atomique, c'est trivial. Si φ commence par un connecteur $\neg, \vee, \wedge, \Rightarrow$, c'est aussi trivial (pour la négation, noter que l'on montre une équivalence). Reste le cas des quantificateurs : le cas du $\forall$ est aussi trivial, mais ne fera pas de mal d'être détaillé pour s'en persuader, et le cas du $\exists$ utilise tout ce que l'on vient de définir.

— Supposons φ de la forme $\forall x.\psi$. Alors

	$\mathcal{S}, \sigma \models \varphi$	
ssi	pour tout a dans $D_\mathcal{S}$, $\mathcal{S}, \sigma \uplus \{x \mapsto a\} \models \psi$	par def. de $\models$
ssi	pour tout a dans $D_\mathcal{S}$, $\overline{\mathcal{S}}, \sigma \uplus \{x \mapsto a\} \models \mathsf{Sk}(\psi)$	par induction
ssi	$\overline{\mathcal{S}}, \sigma \models \forall x.\mathsf{Sk}(\psi)$	par def. de $\models$
ssi	$\overline{\mathcal{S}}, \sigma \models \mathsf{Sk}(\varphi)$	par def. de $\mathsf{Sk}(.)$

— Supposons φ de la forme $\exists x.\psi$, et notons $b_i = \sigma(y_i)$. On a

	$\mathcal{S}, \sigma \models \varphi$	
ssi	$T_{x,\psi}(\vec{b}) \neq \emptyset$	par def. de $T_{x,\psi}$
ssi	$f_{\overline{\mathcal{S}}}(\vec{b}) \in T_{x,\psi}(\vec{b})$	par def. de $f_{\overline{\mathcal{S}}}$
ssi	$\mathcal{S}, \sigma \uplus \{x \mapsto f_{\overline{\mathcal{S}}}(\vec{b})\} \models \psi$	par def. de $T_{x,\psi}$
ssi	$\overline{\mathcal{S}}, \sigma \uplus \{x \mapsto f_{\overline{\mathcal{S}}}(\vec{b})\} \models \mathsf{Sk}(\psi)$	par induction
ssi	$\overline{\mathcal{S}}, \sigma \models \big(\mathsf{Sk}(\psi)\big)\{x \mapsto f(\vec{y})\}$	par le lemme de substitution
ssi	$\overline{\mathcal{S}}, \sigma \models \mathsf{Sk}(\varphi)$	par def de $\mathsf{Sk}(.)$

□

6.3.3 Restriction

Contrairement au foncteur d'enrichissement, le foncteur de restriction ne préserve la satisfaction que pour certaines formules. C'est le cas notamment pour les formules en forme normale négative.

Lemme 6.3.2. *Soit $\mathcal{S}'$ une $(\overline{\mathcal{F}}, \mathcal{P})$-structure. Pour toute $(\mathcal{F}, \mathcal{P})$-formule φ en forme normale négative, et pour toute affectation σ telle que $\mathsf{fv}(\varphi) \subseteq \mathrm{Dom}(\sigma)$,*

$$\textit{si} \qquad \mathcal{S}', \sigma \models \mathsf{Sk}(\varphi) \qquad \textit{alors} \qquad \mathcal{S}'_{|\mathcal{F}}, \sigma \models \varphi.$$

Démonstration. On raisonne par induction sur φ. Au contraire du lemme précédent dans lequel on prouvait une équivalence, nous ne prouvons ici qu'une implication. Le cas de base est maintenant celui des littéraux, puisque nous avons supposé la formule en forme normale négative.

La récurrence est facile pour les connecteurs $\wedge, \vee$ et les quantifications universelles. Noter qu'elle ne marcherait pas pour des négations ou des implications, pour lesquelles nous aurions besoin d'une hypothèse d'induction plus forte (une équivalence et pas seulement une implication).

A nouveau le seul cas intéressant est celui du $\exists$, mais on détaillera aussi celui du $\forall$ pour s'en convaincre. Soit φ fixée, et supposons $\mathcal{S}', \sigma \models \varphi$.

— Si $\varphi = \forall x.\psi$, alors

	$\mathcal{S}', \sigma \models \forall x.\mathsf{Sk}(\psi)$	par def. de $\mathsf{Sk}(.)$
donc	pour tout $a \in D_{\mathcal{S}'}$, $\mathcal{S}', \sigma \uplus \{x \mapsto a\} \models \mathsf{Sk}(\psi)$	par def. de $\models$
donc	pour tout $a \in D_{\mathcal{S}'}$, $\mathcal{S}'_{\|\mathcal{F}}, \sigma \uplus \{x \mapsto a\} \models \psi$	par induction
donc	$\mathcal{S}'_{\|\mathcal{F}}, \sigma \models \forall x.\psi$	par def de $\models$

— Si $\varphi = \exists x.\psi$, et si $f = f_{\psi,x,\vec{y}}$, alors

	$\mathcal{S}', \sigma \models \mathsf{Sk}(\psi)\{x \mapsto f(\vec{y}_{\varphi,x})\}$	par def. de $\mathsf{Sk}(.)$
donc	$\mathcal{S}', \sigma \uplus \{x \mapsto f_{\mathcal{S}'}(\vec{y}_{\varphi,x}\sigma)\} \models \mathsf{Sk}(\psi)$	par le lemme de substitution
donc	$\mathcal{S}'_{\|\mathcal{F}}, \sigma \uplus \{x \mapsto f_{\mathcal{S}'}(\vec{y}_{\varphi,x}\sigma)\} \models \psi$	par induction
donc	$\mathcal{S}'_{\|\mathcal{F}}, \sigma \models \exists x.\psi$	par def de $\models$

□

On peut aussi montrer ce résultat en remplaçant l'hypothèse "en forme normale négative" par l'hypothèse "en forme prénexe", ce qui est souvent fait dans la littérature. Le point clé dans le lemme précédent est que les quantificateurs existentiels ne doivent pas apparaitre sous des négations, ce qui est vrai à la fois pour la forme normale négative et pour la forme prénexe.

Étant donné un ensemble de formules E, on note $\mathsf{Sk}(E) = \{\mathsf{Sk}(\phi) \mid \phi \in E\}$. On dit qu'un ensemble de formules (ayant potentiellement des variables libres) est satisfaisable quand il existe $\mathcal{S}, \sigma$ tel que $\mathcal{S}, \sigma \models \phi$ pour tout $\phi \in E$.

Théorème 6.3.1. *Pour tout ensemble de formules E en forme normale négative, E et $\mathsf{Sk}(E)$ sont équisatisfaisables.*

6.4 Théorème de Herbrand

Nous avons déjà réduit le cas général à la satisfaisabilité d'un ensemble de formules purement universel. Nous allons plus loin ici en montrant qu'un ensemble de formules purement universel est satisfaisable si et seulement si il a un modèle dans une classe bien particulière : les structures de Herbrand. Ceci permettra ensuite de se ramener au calcul propositionnel (au prix de passer d'un ensemble fini de formules à un ensemble infini de formules) et de donner une procédure pour l'insatisfaisabilité (ou la validité).

Définition 6.4.1 (Structure et base de Herbrand)**.** *Soit $\mathcal{F}$ un ensemble de symboles de fonction qui contient une constante, et $\mathcal{P}$ un ensemble de symboles de prédicats.*

1. *Une $\mathcal{F},\mathcal{P}$-*structure de Herbrand *$\mathcal{H}$ est une $\mathcal{F},\mathcal{P}$-structure de domaine l'ensemble des termes clos, noté $T(\mathcal{F})$, et dans laquelle les symboles de fonctions ont leur interprétation canonique sur le domaine des termes, i.e., $f_{\mathcal{H}}(t_1,\ldots,t_n) = f(t_1,\ldots,t_n)$ pour tout $f \in \mathcal{F}$ d'arité n et pour tous $t_1,\ldots,t_n \in T(\mathcal{F})$.*
2. *La $\mathcal{F},\mathcal{P}$-*base de Herbrand *est l'ensemble des atomes clos sur $\mathcal{F},\mathcal{P}$: $\mathcal{B} = \{P(t_1,\ldots,t_n\} \mid t_i \in T(\mathcal{F}), P \in \mathcal{P}\}$.*

Étant donnés $\mathcal{F},\mathcal{P}$, toutes les structures de Herbrand ont le même domaine et interprètent les symboles de fonction de la même façon. La seule liberté qui reste dans le choix d'une structure de Herbrand est le choix des interprétations des symboles de prédicat.

On peut donc identifier chaque $\mathcal{F},\mathcal{P}$-structure de Herbrand avec une partie de la $\mathcal{F},\mathcal{P}$-base de Herbrand : On identifie une $\mathcal{F},\mathcal{P}$-structure de Herbrand $\mathcal{H}$ avec la partie de la $\mathcal{F},\mathcal{P}$-base de Herbrand qui consiste en tous les atomes clos qui sont vrais en $\mathcal{H}$. Ainsi, la structure de Herbrand où tous les prédicats sont toujours faux correspond à l'ensemble vide, et la structure de Herbrand où tous les prédicats sont toujours vrais correspond à la base de Herbrand elle-même.

Dans la suite nous supposons que $\mathcal{F}$ contient toujours une constante, pour assurer que $T(\mathcal{F})$ est non vide. Remarquons que, si S est un ensemble de $\mathcal{F},\mathcal{P}$-formules et $\mathcal{F}$ ne contient pas de constante, alors S possède un $\mathcal{F},\mathcal{P}$-modèle si et seulement si S possède un $\mathcal{F}\cup\{a\},\mathcal{P}$-modèle où a est un symbole de constante.

Une formule est *universelle* si elle est en forme prénexe et toutes ses variables sont quantifiées universellement.

Théorème 6.4.1 (Herbrand)**.** *Un ensemble E de $\mathcal{F},\mathcal{P}$-formules universelles a un modèle ssi il a un modèle qui est une $\mathcal{F},\mathcal{P}$-structure de Herbrand.*

Démonstration. Un seul sens de l'implication demande une preuve : supposons que $\mathcal{S} \models E$. On considère alors, pour chaque $P \in \mathcal{P}$ l'interprétation $P_{\mathcal{H}}$ dans l'univers de Herbrand :

$$P_{\mathcal{H}} = \{(t_1,\ldots,t_n) \mid (\llbracket t_1 \rrbracket_{\mathcal{S}},\ldots,\llbracket t_n \rrbracket_{\mathcal{S}}) \in P_{\mathcal{S}}\}$$

Montrons que la structure de Herbrand $\mathcal{H}$ satisfait E : pour $\forall x_1,\ldots\forall x_m.\phi \in E$ avec ϕ est sans quantificateur et $t_1,\ldots,t_m \in T(\mathcal{F})$, on montre, par récurrence sur ϕ, qu'on a

$$\mathcal{H},\theta \models \phi \quad \text{ssi} \quad \mathcal{S},\sigma \models \phi$$

où $\theta = \{x_1 \mapsto t_1,\ldots,x_n \mapsto t_n\}$ et $\sigma = \{x_1 \mapsto \llbracket t_1 \rrbracket_{\mathcal{S}},\ldots,x_n \mapsto \llbracket t_n \rrbracket_{\mathcal{S}}\}$. (Noter ici qu'on utilise la substitution close θ comme une assignation quand on considère la structure $\mathcal{H}$, ce qui est justifié puisque cette structure a pour domaine l'ensemble des termes clos.)

La formule ϕ étant sans quantificateur, et les cas des connecteurs et constantes propositionnels étant immédiats, il n'y a qu'à considérer le cas où ϕ est une formule atomique :

	$\mathcal{H},\theta \models P(u_1,\ldots,u_m)$	
ssi	$(u_1\theta,\ldots,u_m\theta) \in P_{\mathcal{H}}$	par définition de $\models$
ssi	$(\llbracket u_1\theta \rrbracket_{\mathcal{S}},\ldots,\llbracket u_m\theta \rrbracket_{\mathcal{S}}) \in P_{\mathcal{S}}$	par définition de $P_{\mathcal{H}}$
ssi	$(\llbracket u_1 \rrbracket_{\sigma,\mathcal{S}},\ldots \llbracket u_m \rrbracket_{\sigma,\mathcal{S}}) \in P_{\mathcal{S}}$	par le lemme de substitution
ssi	$\mathcal{S},\sigma \models P(u_1,\ldots,u_m)$	par définition de $\models$

□

Une conséquence du théorème de Herbrand est qu'on peut maintenant transférer certains théorèmes de la logique propositionnelle vers la logique du premier ordre. L'idée est, étant donnée une formule universelle, de construire ses *instances de Herbrand* :

$$H(\forall x_1,\ldots,x_n\ P) := \{P[x_1 \mapsto t_1,\ldots,x_n \mapsto t_n] \mid t_1,\ldots,t_n \in T(\mathcal{F})\}$$

et, pour un ensemble S, $H(S) := \{H(s) \mid s \in S\}$.

Théorème 6.4.2. *Soit S un ensemble de $\mathcal{F},\mathcal{P}$-formules universelles, et soit $\mathcal{P}'$ la base de Herbrand sur $\mathcal{F},\mathcal{P}$. S a un $\mathcal{F},\mathcal{P}$-modèle si et seulement si l'ensemble de formules propositionnelles $H(S)$ a un modèle propositionnel, où $\mathcal{P}'$ est considéré comme l'ensemble des variables propositionnelles.*

Démonstration. Par Théorème 6.4.1. Remarquez que, pour toute structure de Herbrand $\mathcal{H}$ et toute formule universelle close ϕ, $\mathcal{H} \models \phi$ ssi $\mathcal{H} \models H(\phi)$. □

6.5 Compacité

Théorème 6.5.1. *Un ensemble de formules est satisfaisable ssi tous ses sous-ensembles finis sont satisfaisables.*

Démonstration. Il suffit de montrer, pour tout ensemble de formule E insatisfaisable, il existe une partie finie $F \subseteq E$ qui est déja insatisfaisable.

On peut supposer sans perte de généralité que E est composé de formules purement universelles : sinon, on transforme chaque formule en une formule purement universelle par mise en forme prénexe et skolémisation ; on obtient un ensemble de formules E' équisatisfaisable à E (donc insatisfaisable) et tel que si E' admet un sous-ensemble fini F' insatisfaisable alors E aussi (c'est l'ensemble des formules de E dont les transformées sont dans F').

Considérons maintenant l'ensemble des instances closes de E :

$$H(E) = \{\phi\theta \mid \forall x_1, \ldots, x_n.\phi \in E,\ \phi \text{ sans quantificateur},\ \theta : \mathsf{fv}(\phi) \to \mathcal{T}(\mathcal{F}, \emptyset)\}$$

Puisque E n'a pas de modèle, il n'a pas de modèle de Herbrand. Mais la satisfaction d'une formule $\psi \in E$ dans un modèle de Herbrand est équivalente à la satisfaction de ses instances closes. Ainsi, $H(E)$ n'a pas de modèle de Herbrand. On peut ensuite voir les formules de $H(E)$ comme un ensemble E^0 de formules propositionnelles sur l'ensemble de variables propositionnelles composé des formules atomiques closes de notre langage. Si E^0, était satisfait dans une interprétation de la logique propositionnelle, alors $H(E)$ aurait un modèle de Herbrand. Ce n'est donc pas le cas : E^0 est insatisfaisable, donc par le théorème de compacité de la logique propositionnelle E^0 a une partie finie F^0 insatisfaisable. On prend F le plus petit sous-ensemble de E tel que $F^0 \subseteq H(F)$: on tient un sous-ensemble insatisfaisable et nécessairement fini. □