Chapitre 7 : Logique et preuves

1 Introduction

1.1 Rappels sur la logique propositionnelle

Formules. Pour $\mathcal{Q}$ un ensemble de variables propositionnelles (non vide), l'ensemble $\mathcal{F}$ des formules de la logique propositionnelle sur $\mathcal{Q}$ est un ensemble de termes défini inductivement à partir
- des constructeurs d'arité 0 : $\top$, $\bot$ (la tautologie, l'antilogie) ;
- du constructeur d'arité 0 paramétré par l'ensemble de variables $\mathcal{Q}$;
- du constructeur d'arité 1 : $\neg$ (la négation) ;
- et des constructeurs d'arité 2 : $\vee$, $\wedge$, et $\rightarrow$ (la disjonction, la conjonction, l'implication).

Ainsi les formules logiques sont des objets **syntaxiques**, on peut d'ailleurs les représenter par leur arbre de syntaxe.

Exercice de cours 1.1

> Donner les arbres de syntaxe des formules $\neg b \vee a$ et $a \rightarrow b$. Ces deux formules sont-elles égales ?

Booléens. L'ensemble des booléens est l'ensemble $\mathbb{B} = \{\mathsf{V}, \mathsf{F}\}$ muni d'une loi unaire interne $\overline{\square}$ et de deux lois binaires internes $+$ et $\cdot$. On rappelle que pour tout $(x, y) \in \mathbb{B}^2$:
- $\overline{x} = \mathsf{V}$ si et seulement si $x = \mathsf{F}$;
- $x + y = \mathsf{V}$ si et seulement si $x = \mathsf{V}$ ou $y = \mathsf{V}$;
- $x + y = \mathsf{V}$ si et seulement si $x = \mathsf{V}$ et $y = \mathsf{V}$).

Ainsi les booléens sont des objets **sémantiques**.

Exercice de cours 1.2

> Donner la valeur des expressions booléennes $\overline{\mathsf{F}} \cdot \mathsf{V}$, $\mathsf{V} + \mathsf{F}$ et $x + \overline{x}$ pour $x \in \mathbb{B}$.
> Ces expressions booléennes ont-elles la même valeur ?

Sémantique. Un environnement propositionnel est une fonction de $\mathbb{B}^{\mathcal{Q}}$, autrement dit elle affecte à chaque variable propositionnelle une valeur de vérité. Pour un environnement propositionnel $\mu : \mathbb{B}^{\mathcal{Q}}$ donné, la sémantique (ou l'interprétation) des formules est une fonction de $\mathcal{Q} \rightarrow \mathbb{B}$ définie inductivement. On note alors $[\![G]\!]^{\mu}$ l'interprétation de la formule G dans l'environnement propositionnel μ. Si $[\![G]\!]^{\mu} = \mathsf{V}$ on dit que μ **satisfait** G, ou que μ est **modèle** de G. On dit qu'une formule G est **satisfiable** s'il existe un environnement qui la satisfait.

Exercice de cours 1.3

> Pour un environnement propositionnel $\mu : \mathbb{B}^{\mathcal{Q}}$ donné, rappeler la définition inductive de $[\![\bullet]\!]^{\mu}$.

La définition d'une sémantique nous permet de définir une notion de conséquence logique[♣] $\models$ et d'équivalence logique $\equiv$ dont on rappelle les définitions ci-dessous.

♣. parfois appelée conséquence sémantique

Définition 1.4

Soient G et H deux formules de $\mathcal{F}$.
*On dit que G est **conséquence logique** de H, noté $H \vDash G$, dès lors que :*

$$\forall \mu \in \mathbb{B}^{\mathscr{P}}, [\![H]\!]^{\mu} = V \Rightarrow [\![G]\!]^{\mu} = V.$$

Exercice de cours 1.5

Démontrer que $((p \vee q) \wedge (p \to r) \wedge (q \to t)) \vDash (r \vee t)$.

Définition 1.6

Soit $\Gamma \subseteq \mathcal{F}$ un ensemble♣ de formules. Soit $G \in \mathcal{F}$.
*On dit que G est **conséquence logique** de Γ, noté $\Gamma \vDash G$, dès lors que :*

$$\forall \mu \in \mathbb{B}^{\mathscr{P}}, (\forall H \in \Gamma [\![H]\!]^{\mu} = V) \Rightarrow [\![G]\!]^{\mu} = V.$$

Exercice de cours 1.7

Démontrer que $\{(p \vee q), (p \to r), (q \to t)\} \vDash (r \vee t)$.

Exercice de cours 1.8

Soit $\Gamma = \{H_1, H_2, \ldots, H_n\}$ un ensemble fini de formules, soit $G \in \mathcal{F}$ une formule.
Démontrer que $\Gamma \vDash G$ si et seulement si $(H_1 \wedge H_2 \wedge \ldots H_n) \vDash G$.

Définition 1.9

Soient G et H deux formules de $\mathcal{F}$.
*On dit que G et H sont **équivalentes**, noté $G \equiv H$, dès lors que :*

$$\forall \mu \in \mathbb{B}^{\mathscr{Q}}, [\![G]\!]^{\mu} = [\![H]\!]^{\mu}$$

Remarque 1.10

Pour G et H deux formules de $\mathcal{F}$, $G \equiv H$ si et seulement si $G \vDash H$ et $H \vDash G$.

Exercice de cours 1.11

Démontrer la remarque précédente.

Définition 1.12

*On dit qu'une formule est **valide** ou que c'est une **tautologie** dès lors qu'elle est interprétée à V dans tout environnement.*
*On dit que c'est une **antilogie** dès lors qu'elle est interprétée à F dans tout environnement.*

Exercice de cours 1.13

Montrer qu'une formule G est une tautologie si et seulement si elle est équivalente à $\top$.
Montrer qu'une formule G est une antilogie si et seulement si elle est équivalente à $\top$.

1.2 Motivation

Les définitions de la section précédente nous permettent de raisonner sur les formules et de démontrer par exemple des conséquences logiques au moyen de :

- la construction d'une table de vérité ;
- un raisonnement équationnel[♣] dans $\mathbb{B}$.

Cependant aucune de ces méthodes ne colle à la manière dont les preuves sont effectivement élaborées. En effet une preuve "mathématique" de l'énoncé : G admet H comme conséquence logique serait plutôt de la forme ci-dessous.

> Supposons que $(p \to q) \wedge (q \to r)$ est vraie, montrons que $p \to r$ aussi.
> - Supposons p et montrons r.
>   - Montrons q.
>   ↪ Montrons p : c'est notre hypothèse
>   ↪ Montrons $p \to q$: on le déduit de $(p \to q) \wedge (q \to r)$
>   On en déduit que q est vrai.
>   - Montrons $q \to r$: on le déduit de $(p \to q) \wedge (q \to r)$.
>
>   On en déduit que r est vrai.
> - On a montré que $p \to r = H$ est vrai.
>
> On a montré que $p \to r$ est vrai dès que $(p \to q) \wedge (q \to r)$ l'est.

Dans ce chapitre nous nous intéressons donc à définir une notion de "preuve", afin de démontrer des conséquences sémantiques.
Par ailleurs la définition formelle, au moyen d'un objet syntaxique, représentable en machine, de la notion de preuve nous permet de nous poser la question de l'automatisation possible de divers problèmes : découverte automatique de preuves, vérification automatique de preuves, ….

2 La déduction naturelle en logique propositionnelle

Raisonnement par conditions suffisantes. Une preuve est un raisonnement par condition suffisante : pour montrer la propriété $P \vee Q$ il **suffit** de montrer la propriété P ou de montrer la propriété Q. Aussi lorsqu'on essaie de former une preuve pour la propriété $P \vee Q$ on peut essayer de former une preuve pour la propriété P. Si cette tentative de preuve échoue on peut alors essayer d'établir une preuve de la propriété Q.

Exercice de cours 2.1

> Donner deux formules de la logique propositionnelle P et Q telles que Q est valide, mais il n'est pas possible d'établir que P est valide et $P \to Q$ est valide.

2.1 Séquents

Reprenons l'exemple introductif et mettons en avant deux composants essentiels du **contexte** de la preuve : les **hypothèses** et l'**objectif** de preuve.

> Hypothèses : $(p \to q)$, $(q \to r)$. Objectif : $p \to r$.
> - Hypothèses : $(p \to q)$, $(q \to r)$, et p. Objectif : r.
>   - Hypothèses : $(p \to q)$, $(q \to r)$, et p. Objectif : q.
>   ↪ Hypothèses : $(p \to q)$, $(q \to r)$, et p. Objectif : p. C'est une hypothèse.
>   ↪ Hypothèses : $(p \to q)$, $(q \to r)$, et p. Objectif : $p \to q$. C'est une hypothèse.
>   - Hypothèses : $(p \to q)$, $(q \to r)$, et p. Objectif : $q \to r$. C'est une hypothèse.

♣. comprendre une suite d'égalités entre expressions booléennes

Définition 2.2

*Un **séquent** est la donnée :*
- *d'un ensemble Γ de formules de la logique propositionnelle ;*
- *d'une formule G de la logique propositionnelle.*

On le note $\Gamma \vdash G$.

Exemple 2.3
Dans le cas de la preuve introductive, l'écriture de la preuve au moyen de séquents conduit à la preuve suivante.

$(p \to q), (q \to r) \vdash p \to r$.
- $(p \to q), (q \to r), p \vdash r$.
  - $(p \to q), (q \to r), p \vdash q$.
    - $\hookrightarrow$ $(p \to q), (q \to r), p \vdash p$. C'est une hypothèse.
    - $\hookrightarrow$ $(p \to q), (q \to r), p \vdash p \to q$. C'est une hypothèse.
  - $(p \to q), (q \to r), p \vdash q \to r$. C'est une hypothèse.

Exemple 2.4
$\emptyset \vdash (n \in 4\mathbb{N} \to n \in 2\mathbb{N}) \wedge (n \in 4\mathbb{N} + 3 \to n \in 2\mathbb{N} + 1)$ est un séquent.
$\{n \in 4\mathbb{N}\} \vdash n \in 2\mathbb{N}$ est un séquent.
$\{n \in 4\mathbb{N} + 3\} \vdash n \in 2\mathbb{N} + 1$ est un séquent.

Preuves arborescentes. Dans la suite, afin de rendre évident le côté arborescent, syntaxique et manipulable par machine des preuves, on les représentera au moyen d'arbres.

Exemple 2.5
On considère l'énoncé : "(Si n est divisible par 4, alors il est divisible par 2) et (Si n est congru à 3 modulo 4 alors n est impair)". On donne ci-dessous deux preuves de cet énoncé : à gauche une preuve typographiée comme à l'accoutumée et à droite sa version arborescente.

$\emptyset \vdash (n \in 4\mathbb{N} \to n \in 2\mathbb{N}) \wedge (n \in 4\mathbb{N}+3 \to n \in 2\mathbb{N}+1)$
- $\emptyset \vdash n \in 4\mathbb{N} \to n \in 2\mathbb{N}$
  - $\{n \in 4\mathbb{N}\} \vdash n \in 2\mathbb{N}$
    - $\hookrightarrow$ ...
- $\emptyset \vdash n \in 4\mathbb{N} + 3 \to n \in 2\mathbb{N} + 1$
  - $\{n \in 4\mathbb{N} + 3\} \vdash n \in 2\mathbb{N} + 1$
    - $\hookrightarrow$ ...

$$\dfrac{\dfrac{\dfrac{\dots}{\{n \in 4\mathbb{N}\} \vdash n \in 2\mathbb{N}}}{\emptyset \vdash n \in 4\mathbb{N} \to n \in 2\mathbb{N}} \quad \dfrac{\dfrac{\dots}{\{n \in 4\mathbb{N} + 3\} \vdash n \in 2\mathbb{N} + 1}}{\emptyset \vdash n \in 4\mathbb{N} + 3 \to n \in 2\mathbb{N} + 1}}{\emptyset \vdash (n \in 4\mathbb{N} \to n \in 2\mathbb{N}) \wedge (n \in 4\mathbb{N} + 3 \to n \in 2\mathbb{N} + 1)}$$

2.2 Règles d'inférence

À ce stade, nous avons une formalisation de ce qu'est l'état courant d'une preuve mathématique : un séquent. Intéressons-nous donc aux règles permettant la manipulation des séquents.

Vocabulaire 2.6

*On appelle **règle d'inférence** une règle "de la forme" :*

$$\frac{\Gamma_1 \vdash \varphi_1 \quad \Gamma_2 \vdash \varphi_2 \quad \dots \quad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}\ \text{nom}$$

*On dit alors que les $\Gamma_i \vdash \varphi_i$ sont les **prémisses** de la règle et que $\Gamma \vdash \varphi$ en est la **conclusion**.*
*De plus si $n = 0$, cette règle est dite **règle de base**.*
*On appelle **système de preuve** un ensemble de règles d'inférence.*

Remarque 2.7

De telles règles de construction énoncent une condition suffisante à la preuve du séquent $\Gamma \vdash \varphi$: il suffit de montrer les n séquents $\Gamma_1 \vdash \varphi_1$, $\Gamma_2 \vdash \varphi_2$, ..., $\Gamma_n \vdash \varphi_n$. Ainsi dans le cas $n = 0$, la règle énonce que le séquent $\Gamma \vdash \varphi$ est trivialement prouvé.

Notation 2.8

Afin d'éviter d'alourdir les notations, on écrira Γ, φ plutôt que $\Gamma \cup \{\varphi\}$.
De même, on écrira $\vdash \varphi$ plutôt que $\emptyset \vdash \varphi$.

Système de preuve jouet. On se munit d'un système de preuve simple qui nous servira d'exemple dans le reste de cette section. Ce système "jouet" contient les règles d'inférences suivantes.

- La règle de base appelée **axiome** :

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

  Cette règle énonce que tout séquent de la forme $\Gamma \vdash \varphi$ et tel que $\varphi \in \Gamma$ ne nécessite pas d'avantage d'étape de démonstration.
- La règle dite d'**introduction de la conjonction** :

$$\frac{\Gamma \vdash \varphi_1 \quad \Gamma \vdash \varphi_2}{\Gamma \vdash \varphi_1 \wedge \varphi_2}\ \wedge_{\text{i}}$$

  Cette règle énonce que pour établir une preuve du séquent $\Gamma \vdash \varphi_1 \wedge \varphi_2$, il suffit d'obtenir une preuve du séquent $\Gamma \vdash \varphi_1$ et une preuve du séquent $\Gamma \vdash \varphi_2$.
- Deux règles d'**introduction du** $\vee$:

$$\frac{\Gamma \vdash \varphi_1}{\Gamma \vdash \varphi_1 \vee \varphi_2}\ \vee_{\text{i,g}} \qquad \frac{\Gamma \vdash \varphi_2}{\Gamma \vdash \varphi_1 \vee \varphi_2}\ \vee_{\text{i,d}}$$

  La première règle (resp. la seconde) énonce, que pour établir une preuve du séquent $\Gamma \vdash \varphi_1 \vee \varphi_2$, il suffit d'obtenir une preuve du séquent $\Gamma \vdash \varphi_1$ (resp. $\Gamma \vdash \varphi_2$).

2.3 Arbres de preuves

Vocabulaire 2.9

*On appelle **arbre de preuve** un arbre étiqueté par des séquents dont les liens parent-enfant sont des liens autorisés par les règles d'inférence, en particulier les feuilles♣sont celles autorisées par les règles de base.*

Exemple 2.10
En utilisant le système de preuve jouet, on peut établir l'arbre de preuve suivant.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\{P,Q,R\}\vdash P}\,\mathsf{ax}}{\{P,Q,R\}\vdash P\vee Q}\,\vee_{\mathsf{i,g}} \quad \dfrac{\dfrac{}{\{P,Q,R\}\vdash Q}\,\mathsf{ax}}{\{P,Q,R\}\vdash Q\vee\neg R}\,\vee_{\mathsf{i,g}}}{\{P,Q,R\}\vdash (P\vee Q)\wedge(Q\vee\neg R)}\,\wedge_{\mathsf{i}}}{\{P,Q,R\}\vdash (P\wedge T)\vee((P\vee Q)\wedge(Q\vee\neg R))}\,\vee_{\mathsf{i,d}}$$

Définition 2.11

*On dira alors qu'**un séquent** $\Gamma\vdash G$ **admet une preuve** dans un système de preuve dès lors qu'il est possible de construire un arbre de preuve du séquent $\Gamma\vdash G$.*
*On dit qu'**une formule** G **admet une preuve** lorsque le séquent $\emptyset\vdash G$ admet une preuve.*

Notation 2.12

Le fait que le séquent $\Gamma\vdash G$ admette une preuve est noté... $\Gamma\vdash G$. Ainsi le symbole $\vdash$ au sein des arbres de preuve est juste une notation pour représenter les couples hypothèses/objectifs de preuve de manière lisible, tandis qu'en dehors le symbole $\vdash$ est un symbole de relation entre ensemble de formules et formule, au même titre que $\models$.

Exemple 2.13
Dans le système de preuve jouet le séquent $\{P,Q,R\}\vdash(P\wedge T)\vee((P\vee Q)\wedge(Q\vee\neg R))$ admet une preuve (c'est l'arbre donné dans l'exemple 2.3).

Exercice de cours 2.14

Donner une preuve dans le système jouet du séquent $\{P,Q\vee R\}\vdash(P\wedge(\bot\vee(Q\vee R)))\vee(P\to\neg P)$.

2.4 Correction

Les arbres de preuves opèrent sur les formules logiques en tant qu'objets syntaxiques. Cependant, notre motivation étant de démontrer des relations de conséquence sémantique, les règles d'inférence considérées ont été introduites pour représenter fidèlement notre raisonnement mathématique (qui a habituellement lieu du côté de la sémantique). La notion de correction définie ci-après traduit cette fidélité.

Définition 2.15

*Un système de preuve est dit **correct** dès lors que pour tout ensemble Γ de formules et pour toute formule G, si $\Gamma\vdash G$ admet une preuve alors $\Gamma\models G$, autrement dit : $\Gamma\vdash G\Rightarrow\Gamma\models G$.*

Exemple 2.16
Le système de preuve jouet utilisé jusqu'à maintenant est correct. Il suffit de le montrer par induction sur l'arbre de preuve.
Démonstration : Pour $\mathcal{T}$ un arbre de preuve, notons $P(\mathcal{T})$ le prédicat suivant.

$$P(\mathcal{T}) : \text{le séquent en racine de } \mathcal{T}\text{, noté } \Gamma\vdash G\text{, induit une conséquence logique, } \textit{i.e.}\ \Gamma\models G$$

Démontrons par induction que pour tout arbre de preuve $\mathcal{T}$, $P(\mathcal{T})$ est vrai.

♣. On note que dans les arbres de preuves les feuilles sont en haut (et la racine en bas).

- Soit $\mathcal{T}$ un arbre réduit à la règle de l'axiome. Il est de la forme $\dfrac{}{\Gamma, G \vdash G}\ \mathsf{ax}$. Or un modèle de toute formule de $\Gamma \cup \{G\}$ est en particulier un modèle de G, donc $\Gamma, G \vDash G$. Ainsi $P(\mathcal{T})$ est vrai.

- Soit $\mathcal{T}$ un arbre dont la racine est la règle $\vee_{i,g}$. Supposons que son arbre fils $\mathcal{T}_1$ vérifie P.
  Par définition de la règle $\vee_{i,g}$, $\mathcal{T}$ est de la forme suivante.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash G_1}}{\Gamma \vdash G_1 \vee G_2}\ \vee_{\mathsf{i,g}}$$

  Montrons que $\Gamma \vDash G_1 \vee G_2$. Soit μ un modèle de toutes les formules de Γ. Par $P(\mathcal{T}_1)$, $\Gamma \vDash G_1$, donc μ est aussi un modèle de G_1, autrement dit $[\![G_1]\!]^\mu = \mathsf{V}$. On en déduit que $[\![G_1]\!]^\mu + [\![G_2]\!]^\mu = \mathsf{V}$, soit $[\![G_1 \vee G_2]\!]^\mu = \mathsf{V}$. Finalement $\Gamma \vDash G_1 \vee G_2$. Ainsi $P(\mathcal{T})$ est vrai.

- Si $\mathcal{T}$ un arbre dont la racine est la règle $\vee_{i,d}$ et dont le fils vérifie P, on montre de même que $P(\mathcal{T})$ est vrai.

- Soit $\mathcal{T}$ un arbre dont la racine est la règle $\wedge_i$. Supposons que ses fils $\mathcal{T}_1$ et $\mathcal{T}_2$ vérifient P.
  Par définition de la règle $\wedge_i$, $\mathcal{T}$ est de la forme suivante.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash G_1} \qquad \dfrac{\vdots}{\Gamma \vdash G_2}}{\Gamma \vdash G_1 \wedge G_2}\ \wedge_{\mathsf{i}}$$

  Montrons que $\Gamma \vDash G_1 \wedge G_2$. Soit μ un modèle de toutes les formules de Γ. Par $P(\mathcal{T}_1)$, $\Gamma \vDash G_1$, donc μ est un modèle de G_1. De même, par $P(\mathcal{T}_2)$, μ est un modèle de G_2, donc μ est un modèle de $G_1 \wedge G_2$. Finalement $\Gamma \vDash G_1 \wedge G_2$. Ainsi $P(\mathcal{T})$ est vrai.

Par principe d'induction on en déduit que tout arbre de preuve du système jouet (*i.e.* construit à partir des règles ax, $\vee_{i,g}$, $\vee_{i,d}$ et $\wedge_i$) vérifie P. Ainsi le système jouet est un système de preuve correct. □

Vocabulaire 2.17

On dira qu'une règle de la forme $\dfrac{\Gamma_1 \vdash \varphi_1 \qquad \Gamma_2 \vdash \varphi_2 \qquad \dots \qquad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}\ \mathsf{nom}$ *est une* ***règle correcte*** *dès lors que, si* $\Gamma_1 \vDash \varphi_1$ *et* $\Gamma_2 \vDash \varphi_2$ *et* $\dots$ *et* $\Gamma_n \vDash \varphi_n$, *alors* $\Gamma \vDash \varphi$.

Exercice de cours 2.18

Les règles ci-dessous sont-elles correctes ? Justifier.

- $\dfrac{\Gamma \vdash G_1 \vee G_2}{\Gamma \vdash G_1}$
- $\dfrac{\Gamma \vdash G_1 \to \neg G_1}{\Gamma \vdash \neg G_1}$
- $\dfrac{\Gamma \vdash G_1 \qquad \Gamma \vdash G_2}{\Gamma \vdash G_1 \to G_2}$
- $\dfrac{\Gamma \vdash G_1 \to G_2}{\Gamma \vdash G_2}$

Proposition 2.19

Un système de preuve constitué de règles correctes est correct.

Le pendant de la correction : la complétude. Bien que la propriété de correction soit la propriété la plus importante d'un système de preuve, ce n'est pas la seule. En effet, il suffirait de considérer des systèmes de preuves très pauvres qui ne permettent de rien prouver (ou seulement des trivialités), on assurerait ainsi la correction, au prix de ne plus rien pouvoir prouver. La notion de complétude ♣ définie ci-après traduit la capacité d'un système de preuve à pouvoir prouver toutes les conséquences sémantiques.

♣. notion hors-programme

Définition 2.20

Un système de preuve est dit ***complet*** *dès lors que pour tout ensemble* Γ *de formules et toute formule* G*, si* $\Gamma \vDash G$ *alors il existe une preuve du séquent* $\Gamma \vdash G$*, autrement dit :* $\Gamma \vDash G \Rightarrow \Gamma \vdash G$*.*

Exemple 2.21

Considérons par exemple le système de preuve jouet que nous avons manipulé jusqu'à maintenant. Nous avons établi sa correction, il ne permet toutefois pas de manipuler des énoncés de la forme $P \to Q$ et serait donc bien incapable d'établir une preuve du séquent tautologique $\emptyset \vdash P \to P$. Ainsi ce système n'est pas complet.

2.5 La déduction naturelle

Dans cette section on s'intéresse aux deux systèmes de preuve au programme pour la logique propositionnelle appelés déduction naturelle classique et déduction naturelle intuitionniste. Afin de définir ces système on introduit différentes règles d'inférence dans le tableau ci-dessous.

Symbole	Règle d'introduction	Règle d'élimination
$\top/\bot$	$\dfrac{}{\Gamma \vdash \top}\ \top_{\mathsf{i}}$	$\dfrac{\Gamma \vdash \bot}{\Gamma \vdash \varphi}\ \bot_{\mathsf{é}}$
$\neg$	$\dfrac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \neg\varphi}\ \neg_{\mathsf{i}}$	$\dfrac{\Gamma \vdash \varphi \quad \Gamma \vdash \neg\varphi}{\Gamma \vdash \bot}\ \neg_{\mathsf{é}}$
$\wedge$	$\dfrac{\Gamma \vdash \varphi_1 \quad \Gamma \vdash \varphi_2}{\Gamma \vdash \varphi_1 \wedge \varphi_2}\ \wedge_{\mathsf{i}}$	$\dfrac{\Gamma \vdash \varphi_1 \wedge \varphi_2}{\Gamma \vdash \varphi_2}\ \wedge_{\mathsf{é,g}}$ $\dfrac{\Gamma \vdash \varphi_1 \wedge \varphi_2}{\Gamma \vdash \varphi_1}\ \wedge_{\mathsf{é,d}}$
$\vee$	$\dfrac{\Gamma \vdash \varphi_2}{\Gamma \vdash \varphi_1 \vee \varphi_2}\ \vee_{\mathsf{i,g}}$ $\dfrac{\Gamma \vdash \varphi_1}{\Gamma \vdash \varphi_1 \vee \varphi_2}\ \vee_{\mathsf{i,d}}$	$\dfrac{\Gamma \vdash \varphi_1 \vee \varphi_2 \quad \Gamma, \varphi_1 \vdash \psi \quad \Gamma, \varphi_2 \vdash \psi}{\Gamma \vdash \psi}\ \vee_{\mathsf{é}}$
$\to$	$\dfrac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \to \psi}\ \to_{\mathsf{i}}$	$\dfrac{\Gamma \vdash \varphi \to \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi}\ \to_{\mathsf{é}}$

Vocabulaire 2.22

La règle $\to_{\mathsf{é}}$ *est aussi appelée modus ponens. La règle* $\vee_{\mathsf{é}}$ *est aussi appelée disjonction de cas.*

À ces règles on ajoute parfois une règle de base en plus de l'axiome, la règle d'affaiblissement :

$$\frac{\Gamma \vdash \varphi}{\Gamma, \Delta \vdash \varphi}\ \mathsf{aff}$$

Définition 2.23

On appelle ***déduction naturelle intuitionniste*** *le système de preuve induit par les règles d'axiome, d'affaiblissement et par les douze du tableau ci-dessus, autrement dit par l'ensemble de règles suivant.*

$$\mathsf{DNI} = \{\mathsf{ax}, \mathsf{aff},\ \top_{\mathsf{i}}, \bot_{\mathsf{é}},\ \neg_{\mathsf{i}}, \neg_{\mathsf{é}},\ \wedge_{\mathsf{i}}, \wedge_{\mathsf{é,g}}, \wedge_{\mathsf{é,d}},\ \vee_{\mathsf{i,g}}, \vee_{\mathsf{i,d}}, \vee_{\mathsf{é}},\ \to_{\mathsf{i}}, \to_{\mathsf{é}}\ .\}$$

Exemple 2.24
Notons $H = (p \to q) \wedge (q \to r)$. Le séquent $H \vdash p \to r$, déjà étudié dans l'introduction, admet l'arbre de preuve suivant.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{H, p \vdash H}\,\mathsf{ax}}{H, p \vdash q \to r}\,\wedge_{\text{é,d}} \quad \dfrac{\dfrac{\dfrac{}{H, p \vdash H}\,\mathsf{ax}}{H, p \vdash p \to q}\,\wedge_{\text{é,g}} \quad \dfrac{}{H, p \vdash p}\,\mathsf{ax}}{H, p \vdash q}\,\to_{\text{é}}}{H, p \vdash r}\,\to_{\text{é}}}{H \vdash p \to r}\,\to_{\text{i}}$$

Exemple 2.25
La déduction naturelle intuitionniste permet de prouver $G \wedge H \vdash H \wedge G$, par exemple avec l'arbre de preuve suivant.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\{H \wedge G\} \vdash H \wedge G}\,\mathsf{ax}}{\{H \wedge G\} \vdash H}\,\wedge_{\text{é,d}} \quad \dfrac{\dfrac{}{\{H \wedge G\} \vdash H \wedge G}\,\mathsf{ax}}{\{H \wedge G\} \vdash G}\,\wedge_{\text{é,d}}}{\{H \wedge G\} \vdash H \wedge G}\,\wedge_{\text{i}}}{\emptyset \vdash H \wedge G \to H \wedge G}\,\to_{\text{i}}$$

Remarque 2.26

La déduction naturelle intuitionniste n'est pas un système de preuve complet♣.

Au vu de la remarque précédente, on introduit les trois règles suivantes.

Tiers exclu.
$$\dfrac{}{\Gamma \vdash \varphi \vee \neg\varphi}\,\mathsf{te}$$

Élimination de la double négation.
$$\dfrac{\Gamma \vdash \neg\neg\varphi}{\Gamma \vdash \varphi}\,\neg\neg_{\text{é}}$$

Raisonnement par l'absurde♡.
$$\dfrac{\Gamma, \neg\varphi \vdash \bot}{\Gamma \vdash \varphi}\,\mathsf{abs}$$

Remarque 2.27

Ces trois règles sont équivalentes pour la déduction naturelle intuitionniste. Cela signifie que les séquents admettant une preuve sont les mêmes peu importe laquelle de ces trois règles est ajoutée à DNI (*Cf.* exercice de cours 2.29).

Vocabulaire 2.28

On dit qu'une règle $\dfrac{\Gamma_1 \vdash \varphi_1 \quad \Gamma_2 \vdash \varphi_2 \quad \dots \quad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}$ *nom* ***dérive*** *d'un système de preuve* S *s'il existe une preuve du séquent* $\Gamma \vdash \varphi$ *utilisant les règles de* S *ainsi que les règles de bases supplémentaires correspondant aux prémisses :* $\dfrac{}{\Gamma_1 \vdash \varphi_1}\,\mathit{prem}_1$, …, $\dfrac{}{\Gamma_n \vdash \varphi_n}\,\mathit{prem}_n$.

Exercice de cours 2.29

Montrer que :
- $a)$ $\neg\neg_{\text{é}}$ dérive de DNI $\cup \{\text{te}\}$;
- $b)$ abs dérive de DNI $\cup \{\text{te}\}$;
- $c)$ $\neg\neg_{\text{é}}$ dérive de DNI $\cup \{\text{abs}\}$;
- $d)$ abs dérive de DNI $\cup \{\neg\neg_{\text{é}}\}$;
- $e)$ te dérive de DNI $\cup \{\text{abs}\}$;
- $f)$ te dérive de DNI $\cup \{\neg\neg_{\text{é}}\}$.

♣. La démonstration de ce résultat n'est pas au programme.
♡. Règle qu'on veillera à ne pas confondre avec $\bot_{\text{é}}$.

Définition 2.30

*On appelle **déduction naturelle classique** le système de preuve induit par les règles de la déduction naturelle intuitionniste ainsi que le tiers-exclu, le raisonnement par l'absurde et l'élimination de la double négation, i.e. par* $\mathsf{DNC} = \mathsf{DNC} \cup \{\mathsf{te}, \mathsf{abs}, \neg\neg_{\mathsf{é}}\}$.

Théorème 2.31

La déduction naturelle intuitionniste et la déduction naturelle classique sont deux systèmes de preuve corrects.

Exercice de cours 2.32

Démontrer le théorème précédent. D'après la proposition 2.19, il suffit de montrer que chaque règle est correcte.

Théorème 2.33 (*Admis, hors-programme*)

La déduction naturelle classique est un système de preuve complet pour la logique propositionnelle.

Exemple 2.34
Le séquent $\neg(P \wedge Q) \vdash \neg P \vee \neg Q$ admet une preuve en déduction naturelle classique, par exemple celle qui suit.

$$\dfrac{\dfrac{}{\neg(P\wedge Q)\vdash P\vee\neg P}\,\mathsf{te} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma\vdash P}\,\mathsf{ax} \quad \dfrac{}{\Gamma\vdash Q}\,\mathsf{ax}}{\Gamma\vdash P\wedge Q}\,\wedge_{\mathsf{i}} \quad \dfrac{}{\Gamma\vdash\neg(P\wedge Q)}\,\mathsf{ax}}{\underbrace{\neg(P\wedge Q),P,Q}_{\Gamma}\vdash\bot}\,\neg_{\mathsf{é}}}{\neg(P\wedge Q),P\vdash\neg Q}\,\neg_{\mathsf{i}}}{\neg(P\wedge Q),P\vdash\neg P\vee\neg Q}\,\vee_{\mathsf{i,g}} \qquad \dfrac{\dfrac{}{\neg(P\wedge Q),\neg P\vdash\neg P}\,\mathsf{ax}}{\neg(P\wedge Q),\neg P\vdash\neg P\vee\neg Q}\,\vee_{\mathsf{i,d}}}{\neg(P\wedge Q)\vdash\neg P\vee\neg Q}\,\vee_{\mathsf{é}}$$

3 La logique du premier ordre

Bien que de nombreux problèmes soient encodables au moyen d'un nombre borné de variables propositionnelles (le problème du sudoku, le problème de l'existence d'une clique, …), certains problèmes ne peuvent être modélisés au moyen de la logique propositionnelle. En effet, si X est un ensemble fini, par exemple $X = \{x_1, x_2, \ldots, x_n\}$, la conjonction nous permet d'énoncer $\forall x \in X, P(x)$ au moyen de $P(x_1) \wedge P(x_2) \wedge \cdots \wedge P(x_n)$, toutefois si X est infini la conjonction ne permet pas d'énoncer $\forall x \in X, P(x)$ dans une formule de la logique propositionnelle.

Ces remarques nous invitent à étendre la logique propositionnelle, afin de pouvoir énoncer des quantifications portant sur un monde non borné. Afin de nous faire une idée claire des différences entre la logique du premier ordre et la logique propositionnelle, inspectons une formule de la logique "de tous les jours", par exemple $G \overset{\text{déf}}{=} \forall x, ((x > 0) \wedge (\exists y, x = y + 1)) \vee (x = 0)$. On représente ci-dessous à gauche l'arbre de syntaxe de cette formule G, ci-dessous à droite son "squelette propositionnelle".

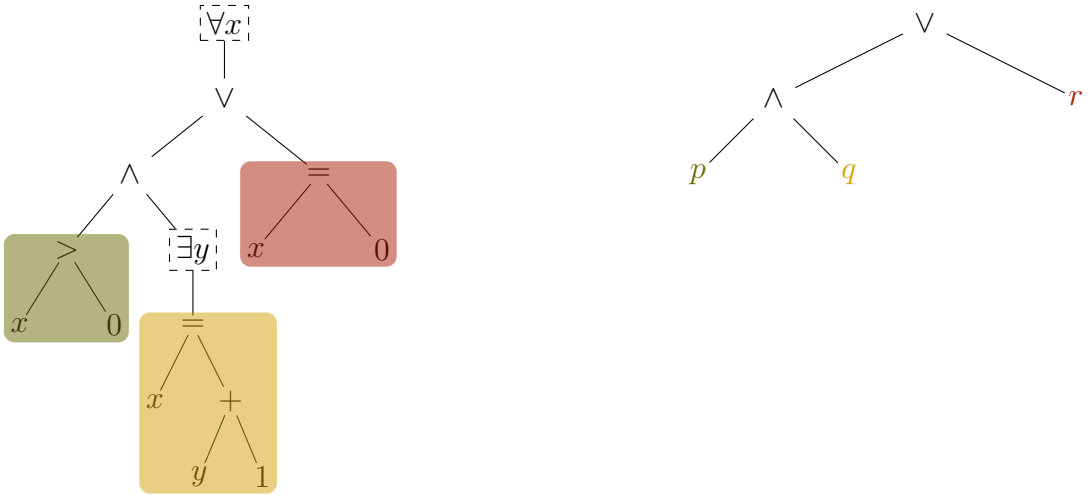

FIGURE 1 – Arbre de syntaxe et squelette propositionnel de G

Cette mise en parallèle met en avant plusieurs différences :
- deux nouveaux constructeurs unaires (étiquetés par une variable) permettent de former des formules : $\forall$ et $\exists$;
- on peut construire des termes contenant des variables (*e.g.* x, y), des constantes (*e.g.* 0, 1) et des symboles de fonctions (*e.g.* $+$) ;
- au lieu des variables propositionnelles on a des formules atomiques (*e.g.* $x > 0$), c'est-à-dire des prédicats sur des termes.

3.1 Syntaxe de la logique du premier ordre

Rappel 3.1

*Un opérateur qui a un seul opérande est dit **unaire**, celui qui en a deux **binaire**, celui qui en a trois **ternaire** … De façon plus générale, un opérateur qui a n opérandes est dit **d'arité** n. Ce vocabulaire s'étend à d'autres objets, par exemple :*
- *aux relations (une relation n-aire sur X est un sous-ensemble de X^n) ;*
- *aux fonctions (une fonction n-aire prend n arguments) ;*
- *aux nœuds et par extension aux arbres (dans un arbre n-aire tous les nœuds internes ont n fils).*

Définition 3.2

*On appelle **signature** du premier ordre la donnée de deux ensembles disjoints :*
- *$\mathcal{S}$ un ensemble de symboles munis d'une arité, appelés **symboles de fonctions** ;*
- *$\mathcal{P}$ un ensemble de symboles munis d'une arité, appelés **prédicats**.*

Vocabulaire 3.3

*Les symboles de fonction d'arité nulle sont appelés **symboles de constante**, voire simplement **constantes**.*

Définition 3.4

*Étant donné un ensemble de symboles de fonctions $\mathcal{S}$ et un ensemble de variables $\mathcal{V}$ tels que $\mathcal{S} \cap \mathcal{V} = \emptyset$, l'ensemble des **termes** construit sur $\mathcal{S}$ et $\mathcal{V}$, noté $\mathcal{T}(\mathcal{S}, \mathcal{V})$, est défini inductivement par les règles suivantes.*
- *Si $x \in \mathcal{V}$, alors x est un terme de $\mathcal{T}(\mathcal{S}, \mathcal{V})$.*
- *Si $f \in \mathcal{S}$ est un symbole de fonction d'arité n, et $t_1, t_2, \ldots, t_n$ sont n termes de $\mathcal{T}(\mathcal{S}, \mathcal{V})$, alors $f(t_1, t_2, \ldots, t_n)$ est un terme de $\mathcal{T}(\mathcal{S}, \mathcal{V})$.*

Remarque 3.5

Le deuxième point de la définition ci-dessus décrit autant de règles qu'il y a de symboles de fonctions. Il décrit notamment à la fois des règles de base (dans le cas d'un symbole de fonction d'arité 0), et des règles d'induction (dans les autres cas). Ainsi on peut retenir que les termes sont des arbres dont les feuilles peuvent être des variables ou des constantes, et dont les nœuds internes sont des symboles de fonctions, l'arité du symbole de fonction fixant l'arité du nœud.

Exemple 3.6

Pour les exemples, on considère l'ensemble de symboles de fonctions $\mathcal{S}_{\mathsf{e}} \overset{\text{déf}}{=} \{+, -, \mathbf{0}\}$ où $+$ est d'arité 2, $-$ est d'arité 1 et $\mathbf{0}$ d'arité 0, et l'ensemble de variables $\mathcal{V}_{\mathsf{e}} \overset{\text{déf}}{=} \{x, y, z\}$. Ainsi x, y et z sont des termes de $\mathcal{T}(\mathcal{S}_{\mathsf{e}}, \mathcal{V}_{\mathsf{e}})$, mais aussi $x + y$, $x + (-y)$, $\mathbf{0}$, $\mathbf{0} + (\mathbf{0} + \mathbf{0})$ ….

Exercice de cours 3.7

Représenter les arbres de syntaxe des termes de l'exemple précédent.
Donner d'autres exemples de termes sur $\mathcal{S}_{\mathsf{e}}$ et $\mathcal{V}_{\mathsf{e}}$.

Définition 3.8

Soit $(\mathcal{S}, \mathcal{P})$ une signature du premier ordre. Soit $\mathcal{V}$ un ensemble de variables.
*La **logique du premier ordre sur** $\mathcal{S}$, $\mathcal{P}$ **et** $\mathcal{V}$, notée $\mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$, est le plus petit ensemble tel que :*

- *$\bot$ et $\top$ sont des formules de $\mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$;*
- *si $P \in \mathcal{P}$ est d'arité n et si $(t_1, t_2, \ldots, t_n) \in \mathcal{T}(S, \mathcal{V})^n$, alors $P(t_1, t_2, \ldots, t_n) \in \mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$;*
- *si H et G sont des formules de $\mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$, alors $\{\neg H; H \wedge G; H \vee G; H \to G\} \subseteq \mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$;*
- *si H est une formule de $\mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$ et $x \in \mathcal{V}$, alors $\{\forall x, H; \exists x, H\} \subseteq \mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$.*

Remarque 3.9

Au vu de la définition ci-dessus, on peut voir une formule du premier ordre comme un arbre organisé en plusieurs couches :

- en bas se trouvent les termes, donc des nœuds qui sont des symboles de fonctions et en feuille des constantes ou des variables ;
- au-dessus se trouve une couche de nœuds prédicats (un seul par branche) ;
- au-dessus encore se trouve une couche d'opérateurs logiques ($\neg$, $\vee$, $\to$ ou les quantificateurs $\forall$ ou $\exists$[♣]).

♣. On a listé ici les constructeurs d'arité > 0, mais il est aussi possible d'avoir des feuilles $\top$ et $\bot$.

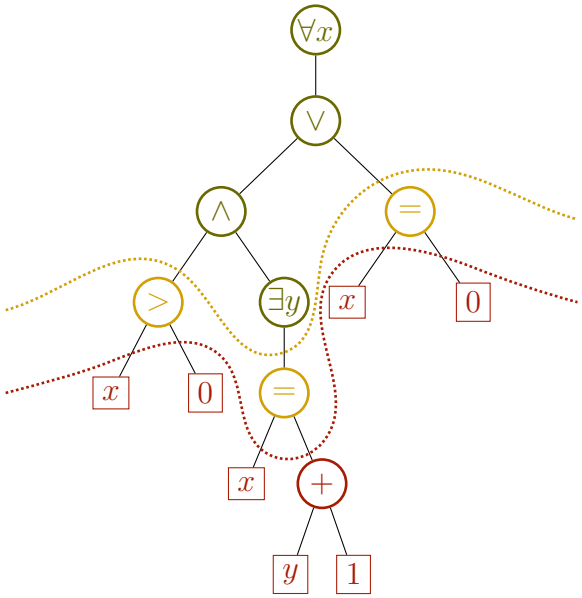

FIGURE 2 – Structure de la formule du premier ordre
$G = \forall x, ((x > 0) \wedge (\exists y, x = y + 1)) \vee (x = 0)$

Exercice de cours 3.10

Pour cet exercice, on utilise $\mathcal{S}_e = \{+, -, \mathbf{0}\}$ défini précédemment et $\mathcal{P}_e \overset{\text{déf}}{=} \{=, <, >\}$ où ces trois prédicats sont binaires. Pour la logique du premier ordre de signature $(\mathcal{S}_e, \mathcal{P}_e)$,
- donner un exemple de formule de taille 3 qui ne contient aucun terme ;
- donner un exemple de formule de taille 5 qui ne contient aucun opérateur logique.

Vocabulaire 3.11

Une formule qui ne contient aucun opérateur logique, qui est donc réduite à un prédicat sur des termes, est appelée une ***formule atomique****.*

Définition 3.12

On définit la fonction vars, à valeurs dans $\mathcal{P}(\mathcal{V})$, par induction sur $\mathcal{T}(\mathcal{S}, \mathcal{V})$ comme suit.

$$\begin{array}{ll} \mathit{vars}(v) & = \{v\} \\ \mathit{vars}(f(t_1, t_2, \ldots, t_m)) & = \bigcup_{i=1}^{m} \mathit{vars}(t_i) \end{array}$$

Ainsi si t est un terme, vars(t) est l'ensemble des variables apparaissant dans ce terme.

Exercice de cours 3.13

Donner l'ensemble des variables des termes de $\mathcal{T}(\mathcal{S}_e, \mathcal{V}_e)$ suivants.
- $\mathbf{0 + 0}$
- $x + \mathbf{0}$
- $x + x$
- $x + (-y)$

Définition 3.14

On définit les fonctions FV et BV, à valeurs dans $\mathcal{P}(\mathcal{V})$, par induction sur $\mathcal{F}(\mathcal{S},\mathcal{P},\mathcal{V})$ comme suit.

$$\begin{array}{ll|ll}
FV(\bot\|\top) & = \emptyset & BV(\bot\|\top) & = \emptyset \\
FV(P(t_1,t_2,\dots t_m)) & = \bigcup_{i=1}^{m} vars(t_i) & BV(P(t_1,t_2,\dots t_m)) & = \emptyset \\
FV(\neg H) & = FV(H) & BV(\neg H) & = BV(H) \\
FV(H_1 \odot H_2) & = FV(H_1) \cup FV(H_1) & BV(H_1 \odot H_2) & = BV(H_1) \cup BV(H_1) \\
FV(\exists x, H\|\forall x, H) & = FV(H) \setminus \{x\} & BV(\exists x, H\|\forall x, H) & = BV(H) \cup \{x\}
\end{array}$$

*Pour une formule $H \in \mathcal{F}(\mathcal{S},\mathcal{P},\mathcal{V})$, $FV(H)$ (resp. $BV(H)$) est l'ensemble de ses **variables libres** (resp. **liées**)*♣*. Une formule sans variable libre est dite **close**.*

Remarque 3.15

Une variable pouvant apparaître plusieurs fois dans une formule, elle peut être à la fois libre et liée. Par exemple dans la formule $Q(x) \vee (\forall x, P(x))$, la première occurrence de x est libre tandis que la seconde est liée (liée au quantificateur $\forall$).

Exemple 3.16

Pour la formule $G = \forall x, ((x > 0) \wedge (\exists y, x = y + 1)) \vee (x = 0)$ déjà étudiée, $\mathsf{FV}(G) = \emptyset$ et $\mathsf{BV} = \{x, y\}$.
Pour sa sous-formule $G' = ((x > 0) \wedge (\exists y, x = y + 1)) \vee (x = 0)$, $\mathsf{FV}(G') = \{x\}$ et $\mathsf{BV}(G') = \{y\}$.
Enfin pour $G'' = \exists y, x = y + 1$, $\mathsf{FV}(G'') = \{x\}$ et $\mathsf{BV}(G'') = \{y\}$.

Exercice de cours 3.17

Donner les variables libres et les variables liées des formules de $\mathcal{F}(\mathcal{S}_e,\mathcal{P}_e,\mathcal{V}_e)$ suivantes.

- $\forall x, \exists z, (P(x, y) \vee P(y, z))$
- $\forall x, \forall y, \forall z, ((P(x, y) \wedge P(y, z)) \to P(y, z))$
- $P(x, y) \vee P(y, z)$
- $(\exists x, P(x, y)) \vee (\exists x, P(y, x))$

3.2 Interlude : syntaxe de la logique du premier ordre en OCaml.

Afin de fixer les idées sur ce qu'est la syntaxe de la logique du premier ordre, on fournit ci-dessus une proposition de définition des objets de la logique du premier ordre en OCaml.

```
type variable = string
type fonction = string
type predicat = string

type term =
  | V of variable                 (* V(x)              : variable x           *)
  | S of fonction * term list     (* S(f, t1, ..., tm) : terme f(t1, ..., tm) *)

type foform =
  | Top | Bot                       (* Les constantes ⊤ et ⊥                 *)
  | Pred   of predicat * term list  (* Pred(P, t1, ..., tm) : P(t1, ... tm)  *)
  | Not    of foform                (* ¬                                     *)
  | Exists of variable * foform     (* Exists(x, H) : ∃ x, H                 *)
  | Forall of variable * foform     (* Forall(x, H) : ∀ x, H                 *)
  | And    of foform * foform       (* ∧                                     *)
  | Or     of foform * foform       (* ∨                                     *)
  | Imply  of foform * foform       (* →                                     *)
```

♣. FV comme *free variables*, BV comme *bound variables*

La formule G qu'on a utilisé en exemple dans la partie précédente est représentée en OCaml par l'objet ex défini comme suit.

```
let ex =
  Forall("x", Or(
      And(Pred(">", [V("x"); S("0", [])]),
          Exists("y", Pred("=", [V("x"); S("+", [V("y"); S("1", [])])]))),
      Pred("=", [V("x"); S("0", [])])))
```

Il est clair que ces définitions inductives se prêtent particulièrement bien aux définitions inductives de fonctions. À titre d'exemple, en supposant défini un module `S` permettant les opérations ensemblistes usuelles sur des ensembles d'objets de type `variable`, on peut définir la fonction `free_vars : foform -> S.t` calculant l'ensemble des variables libres d'une formule comme suit.

```
let rec vars (t: term): S.t =
  match t with
  | V(x)          -> S.singleton x
  | S(s, tl)      -> List.fold_left (fun u t -> S.union u (vars t)) S.empty tl
```

```
let rec free_vars (f: foform): S.t =
  match f with
  | Top | Bot     -> S.empty
  | Not(f)        -> free_vars f
  | Exists(x, f)
  | Forall(x, f)  -> S.remove x (free_vars f)
  | Pred(s, tl)   -> List.fold_left (fun u t -> S.union u (vars t)) S.empty tl
  | And(f1, f2)
  | Or(f1, f2)
  | Imply(f1, f2) -> S.union (free_vars f1) (free_vars f2)
```

Exercice de cours 3.18

Définir en OCaml la fonction `bounded_vars : foform -> S.t` calculant l'ensemble des variables liées d'une formule.

3.3 Substitution et α-renommage

Afin de définir des règles d'inférence de la déduction naturelle pour la logique du premier ordre dans la section suivante, il nous faut définir deux opérations syntaxiques :

- la substitution, qui permet par exemple de transformer $P(x,x)$ en $P(0,0)$, et $P(x,x)\wedge(\exists x, Q(x,x))$ en $P(0,0)\wedge(\exists x, Q(x,x))$;
- le α-renommage, qui permet par exemple de transformer $\forall x, P(x,x)$ en $\forall y, P(y,y)$, et $\exists y, Q(x,y)$ en $\exists z, Q(x,z)$[♣].

Dans cette section et dans la suivante, on fixe $(\mathcal{S},\mathcal{P})$ une signature de la logique du premier ordre, et $\mathcal{V}$ un ensemble de variables. Implicitement, on travaillera avec des variables de $\mathcal{V}$, des termes sur $\mathcal{S}$ et $\mathcal{V}$, et avec des formules de $\mathcal{F}(\mathcal{S},\mathcal{P},\mathcal{V})$.

Définition 3.19

*Une **substitution** est une fonction $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{S},\mathcal{V})$ qui est l'identité partout sauf sur un ensemble fini de variables, que l'on appelle le **support** de la substitution.*

♣. mais surtout pas en $\exists z, Q(z,z)$

Notation 3.20

Pour σ une substitution, x_1, x_2, ... x_n des variables distinctes et t_1, t_2, ... t_n des termes, on note $\sigma[x_1 \mapsto t_1, \dots, x_n \mapsto t_n]$ la substitution σ' définie par $\sigma'(x) = t_i$ si $x = x_i$ et $\sigma'(x) = \sigma(x)$ sinon. Autrement dit, les associations antécédent-image entre crochets "écrasent" celles définies par σ.

De plus, si σ est l'identité, on pourra l'omettre dans la notation, ainsi $[x_1 \mapsto t_1, \dots, x_n \mapsto t_n]$ définit une substitution de support inclus dans $\{x_1, x_2 \dots, x_n\}$.

Définition 3.21

*Soit σ une substitution. Lorsque t est un terme, on note $t[\sigma]$ le terme obtenu par **application de la substitution** σ à t. La fonction $t \mapsto t[\sigma]$ est définie par induction sur $\mathcal{T}(\mathcal{S}, \mathcal{V})$ comme suit.*

$$\begin{aligned} v[\sigma] &= \sigma(v) \\ f(t_1, t_2, \dots, t_m)[\sigma] &= f(t_1[\sigma], t_2[\sigma], \dots, t_m[\sigma]) \end{aligned}$$

Exemple 3.22

Par exemple, si $\sigma = [x \mapsto z, y \mapsto z]$ et $\sigma' = [x \mapsto y, y \mapsto z]$, alors :

- le terme $(\mathbf{0} + \mathbf{0})[\sigma]$ et le terme $(\mathbf{0} + \mathbf{0})[\sigma']$ sont tous les deux $\mathbf{0} + \mathbf{0}$;
- le terme $(x + x)[\sigma]$ est $z + z$ tandis que le terme $x + x[\sigma']$ est $y + y$;
- le terme $(x + y)[\sigma]$ est $z + z$ tandis que le terme $(x + y)[\sigma']$ est $y + z$.

Notation 3.23

Lorsque la substitution σ est décrite par $[x_1 \mapsto t_1, \dots, x_n \mapsto t_n]$, on pourra omettre une paire de crochets, et noter seulement $t[x_1 \mapsto t_1, \dots, x_n \mapsto t_n]$ ♣.

Exercice de cours 3.24

Expliciter les termes suivants.

- $(\mathbf{0} + \mathbf{0})[x \mapsto 0]$
- $(x + \mathbf{0})[x \mapsto 0]$
- $(x + x)[x \mapsto y]$
- $(x + x)[y \mapsto x]$
- $(x + (-y))[x \mapsto y]$
- $(x + (-y))[x \mapsto y, y \mapsto x]$

Notation 3.25

Soit σ une substitution. On note $IV(\sigma)$ l'ensemble des variables qui apparaissent dans les termes image du support de σ.

$$IV(\sigma) = \bigcup_{x \in \mathit{support}(\sigma)} \mathit{vars}(\sigma(x))$$

Exemple 3.26

Si $\sigma = [x \mapsto f(y)]$, alors $\mathsf{IV}(\sigma) = \{y\}$ et si $\sigma' = [x \mapsto f(y + x)]$, alors $\mathsf{IV}(\sigma') = \{x, y\}$.

♣. au lieu de $t[[x_1 \mapsto t_1, \dots, x_n \mapsto t_n]]$

Définition 3.27

Soit σ une substitution. Lorsque H est une formule n'ayant aucune variable liée dans $IV(\sigma)$, on note $H[\sigma]$ la formule obtenue par ***application de la substitution*** *σ à H. La fonction partielle $H \mapsto H[\sigma]$ est définie par induction sur $\mathcal{F}(\mathcal{S}, \mathcal{P}, \mathcal{V})$ comme suit.*

$$\begin{array}{ll}
\top[\sigma] & = \top \\
\bot[\sigma] & = \bot \\
P(t_1, t_2, \ldots, t_m)[\sigma] & = P(t_1[\sigma], t_2[\sigma], \ldots, t_m[\sigma]) \\
(\neg G)[\sigma] & = \neg(G[\sigma]) \\
(G \odot H)[\sigma] & = G[\sigma] \odot H[\sigma] \\
(\forall x, G)[\sigma] & = \forall x, (G[\sigma[x \mapsto x]]) \\
(\exists x, G)[\sigma] & = \exists x, (G[\sigma[x \mapsto x]])
\end{array}$$

Remarque 3.28

La définition précédente appelle quelques remarques.

- La contrainte d'application de substitution dans la définition vient de ce que l'on ne souhaite pas que des variables liées par des quantificateurs soient introduites par des substitutions. Par exemple, dans la formule $\exists y, P(x, y)$, on ne souhaite pas pouvoir appliquer la substitution $\sigma = [x \mapsto f(y)]$ qui aurait l'effet de lier la nouvelle variable y au quantificateur existant. En effet, on ne veut pas pouvoir déduire de $\forall x, \exists y, P(x, y)$ que $\exists y, P(f(y), y)$.
- Par définition de l'application d'une substitution à une formule, il n'est pas possible de substituer des variables liées dans une formule. Par exemple, l'application de la substitution $[x \mapsto f(y)]$ sur la formule $(\forall x, P(x)) \wedge (Q(x))$ produit la formule $(\forall x, P(x)) \wedge Q(f(y))$.

Exemple 3.29

Par exemple, si $\sigma = [x \mapsto z, y \mapsto z]$ et $\sigma' = [x \mapsto y, y \mapsto z]$, alors :
- la formule $((a \to b) \to c)[\sigma]$ et la formule $((a \to b) \to c)[\sigma']$ sont toutes les deux $(a \to b) \to c$;
- la formule $(\forall y, P(y) \vee Q(y))[\sigma]$ et la formule $(\forall y, P(y) \vee Q(y))[\sigma']$ sont toutes les deux $\forall y, P(y) \vee Q(y)$;
- la formule $(P(y) \vee Q(y))[\sigma]$ et la formule $(P(y) \vee Q(y))[\sigma']$ sont toutes les deux $P(z) \vee Q(z)$;
- la formule $(\exists x, P(x, y))[\sigma]$ et la formule $(\exists x, P(x, y))[\sigma]$ sont toutes les deux $\exists x, P(x, z)$;
- la formule $(\exists y, P(x, y))[\sigma]$ est $\exists y, P(z, y)$ tandis que σ' ne peut s'appliquer à $\exists y, P(x, y)$ à cause de la variable liée y qui est aussi dans $IV(\sigma')$;
- la formule $(\exists t, P(x, t))[\sigma]$ est $\exists t, P(z, t)$ tandis que la formule $(\exists t, P(x, t))[\sigma']$ est $\exists t, P(y, t)$.

Exercice de cours 3.30

Expliciter les formules suivantes (ou justifier que la substitution ne peut s'appliquer).

- $(\forall x, P(x))[x \mapsto x + y]$
- $(\forall x, P(x))[x \mapsto y + z]$
- $(P(x) \vee (\forall x, Q(y, x)))[x \mapsto y, y \mapsto x]$
- $(P(x) \vee Q(x, y))[x \mapsto y, y \mapsto z]$
- $(\forall x, P(x) \vee Q(x, y))[x \mapsto y, y \mapsto z]$
- $(Q(x, y) \vee Q(y, x))[x \mapsto y, y \mapsto x]$

Définition 3.31

*On appelle α-**renommage** l'opération consistant à changer la variable étiquetant un quantificateur en une nouvelle♣variable et à modifier en conséquence toutes les occurrences de variables liées à ce quantificateur.*

♣. Il est prudent de choisir d'α-renommer en une variable nouvelle au sens où elle n'apparaît pas dans la formule. En effet il serait malvenu de renommer z en x dans la formule $\forall x, \exists z, P(y) \vee Q(x)$.

Remarque 3.32

Afin de ne pas alourdir la définition ci-dessous, on a omis de définir ce que sont les occurrences d'une variable liées à un quantificateur. Si cette notion n'est pas intuitive, on pourra se reporter aux exemples suivants. En dessinant l'arbre de syntaxe de la formule à laquelle on applique le α-renommage, on peut constater que les occurrences de variables liées à un quantificateur sur x sont les occurrences de x qui figurent dans le sous-arbre issu du nœud correspondant à ce quantificateur, à l'exclusion de celles qui se trouvent sous un autre nœud quantificateur sur x.

Exemple 3.33

La formule $(\forall x, P(x)) \wedge (\forall x, \forall y, Q(x, x+y))$ peut par exemple être α-renommée en :

- $(\forall \boldsymbol{z}, P(\boldsymbol{z})) \wedge (\forall x, \forall y, Q(x, x+y))$;
- $(\forall z, P(z)) \wedge (\forall \boldsymbol{z}, \forall y, Q(\boldsymbol{z}, \boldsymbol{z}+y))$;
- $(\forall z, P(z)) \wedge (\forall \boldsymbol{t}, \forall y, Q(\boldsymbol{t}, \boldsymbol{t}+y))$;
- $(\forall z, P(z)) \wedge (\forall t, \forall \boldsymbol{u}, Q(t, t+\boldsymbol{u}))$.

La formule $\forall x, (P(x) \vee \exists y, \forall x, Q(x, y))$ peut par exemple être α-renommée en :

- $\forall \boldsymbol{z}, (P(\boldsymbol{z}) \vee \exists y, \forall x, Q(x, y))$;
- $\forall z, (P(z) \vee \exists \boldsymbol{t}, \forall x, Q(x, \boldsymbol{t}))$.

On préférera d'ailleurs une de ces formes renommées où l'on évite que deux quantificateurs sur la même variable soient sur la même branche.

Remarque 3.34

Le α-renommage est particulièrement important dans le contexte de l'application de substitutions. Lorsqu'une substitution σ n'est pas applicable à une formule G puisque $\mathsf{BV}(G) \cap \mathsf{IV}(\sigma) \neq \emptyset$, alors on peut α-renommer les variables de $\mathsf{BV}(G)$ de sorte qu'elles n'intersectent pas les variables de $\mathsf{IV}(\sigma)$.

Exemple 3.35

On reprend l'exemple ci-dessus. Si l'on veut pouvoir appliquer la substitution $\sigma = [x \mapsto f(y)]$ sur la formule $G = \forall y, P(x, y)$, on α-renomme G en une formule H équivalente[♣] : $H = \forall z, P(x, z)$. On peut alors appliquer la substitution sur H : $H[\sigma] = \forall z, P(f(y), z)$.

3.4 La déduction naturelle en logique du premier ordre

On ajoute à la déduction naturelle (classique ou intuitionniste) définie plus haut les règles d'introduction et d'élimination des quantificateurs universels et existentiels données dans le tableau suivant.

Symbole	Règle d'introduction	Règle d'élimination
$\forall$	Si $x \notin \mathsf{FV}(\Gamma)$: $\dfrac{\Gamma \vdash \varphi}{\Gamma \vdash \forall x, \varphi}\ \forall_{\mathsf{i}}$	Si $\mathsf{vars}(t) \cap \mathsf{BV}(\varphi) = \emptyset$: $\dfrac{\Gamma \vdash \forall x, \varphi}{\Gamma \vdash \varphi[x \mapsto t]}\ \forall_{\mathsf{é}}$
$\exists$	Sans condition : $\dfrac{\Gamma \vdash \varphi[x \mapsto t]}{\Gamma \vdash \exists x, \varphi}\ \exists_{\mathsf{i}}$	Si $x \notin \mathsf{FV}(\Gamma) \cup \mathsf{FV}(\psi)$: $\dfrac{\Gamma \vdash \exists x, \varphi \qquad \Gamma, \varphi \vdash \psi}{\Gamma \vdash \psi}\ \exists_{\mathsf{é}}$

♣. La sémantique de la logique du premier ordre n'a pas été définie dans ce chapitre, mais n'importe quelle définition de cette sémantique devrait assurer cette équivalence.

Remarque 3.36

Afin de se convaincre que les hypothèses sont nécessaires à la correction des règles proposées, considérons les faux arbres de preuves suivants (dont les séquents résultants sont clairement faux).

- Si on omet la condition $x \notin \mathsf{FV}(\varphi)$ pour la règle $\forall_{\mathsf{i}}$, on risque de former l'arbre suivant où H est la formule $z = 0$.

$$\dfrac{\dfrac{}{z = 0 \vdash z = 0}\ \mathsf{ax}}{z = 0 \vdash \forall z, z = 0}\ \forall_{\mathsf{i}}$$

  L'erreur dans cet arbre vient de ce que l'on introduit un quantificateur universel sur la variable z qui est déjà fixée dans les hypothèses puisque que $z \in \mathsf{FV}(H)$.

- Si on omet la condition $\mathsf{vars}(t) \cap \mathsf{BV}(\varphi) = \emptyset$ pour la règle $\forall_{\mathsf{é}}$, on risque de former l'arbre suivant où $H = \forall x, \exists y, x = y + 1$.

$$\dfrac{\dfrac{}{H \vdash \forall x, \exists y, x = y + 1}\ \mathsf{ax}}{H \vdash \exists y, y = y + 1}\ \forall_{\mathsf{é}}$$

  L'erreur dans cet arbre vient de ce que l'on applique la substitution $[x \mapsto y]$ qui admet y dans son image à la formule $\exists y, x = y + 1$ qui contient y comme variable liée (ce qui d'ailleurs n'est pas autorisé d'après la définition 3.27).

- Si on omet la condition $x \notin \mathsf{FV}(\Gamma) \cup \mathsf{FV}(\psi)$ pour la règle $\exists$é, on risque de former l'arbre suivant où $\Gamma = \{z = 0, \exists x, x + 1 = 0\}$.

$$\dfrac{\dfrac{}{\Gamma \vdash \exists x, x + 1 = 0}\ \mathsf{ax} \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, z + 1 = 0 \vdash z + 1 = 0}\ \mathsf{ax} \qquad \dfrac{}{\Gamma, z = 0 \vdash z + 1 = 0}\ \mathsf{ax}}{\Gamma, z + 1 = 0 \vdash z + 1 = 0 \wedge t = 0}\ \wedge_{\mathsf{i}}}{\Gamma, z + 1 = 0 \vdash \exists t, t + 1 = 0 \wedge t = 0}\ \exists_{\mathsf{i}}}{\Gamma, \vdash \exists t, t + 1 = 0 \wedge t = 0}\ \exists_{\mathsf{é}}$$

  L'erreur dans cet arbre vient de ce que l'on élimine la quantification existentielle de $\exists x, x + 1 = 0$ en nommant ce témoin d'existence z alors que la variable x est liée dans Γ (du fait de la formule $z = 0$), *i.e.* $z \in \mathsf{FV}(\Gamma)$.
  On pourrait aussi former l'arbre suivant où $\Gamma = \{H_1, H_2\}$ avec $H_1 = \exists y, y = 0$ et $H_2 = \exists x, x + 1 = 0$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \exists x, x + 1 = 0}\ \mathsf{ax} \qquad \dfrac{}{\Gamma, z + 1 = 0 \vdash z + 1 = 0}\ \mathsf{ax}}{\Gamma \vdash z + 1 = 0}\ \exists_{\mathsf{é}} \qquad \dfrac{\dfrac{}{\Gamma \vdash \exists y, y = 0}\ \mathsf{ax} \qquad \dfrac{}{\Gamma, z = 0 \vdash z = 0}\ \mathsf{ax}}{\Gamma \vdash z = 0}\ \exists_{\mathsf{é}}}{\Gamma, \vdash z + 1 = 0 \wedge z = 0}\ \wedge_{\mathsf{i}}}{\Gamma, \vdash \exists t, t + 1 = 0 \wedge t = 0}\ \exists_{\mathsf{i}}$$

  L'erreur dans cet arbre vient de ce que l'on élimine la quantification existentielle de $\exists x, x + 1 = 0$ en nommant ce témoin d'existence z alors que la variable z est liée dans la formule que l'on cherche à démontrer, *i.e.* $z \in \mathsf{FV}(\psi)$ où ψ est l'objectif de preuve $z + 1 = 0$ (et cette erreur est répétée lorsqu'on élimine la quantification existentielle de $\exists y, y = 0$).

Notation 3.37

Bien que le α-renommage ne constitue pas vraiment une règle de déduction, et bien que ce ne soit pas classique, on le notera parfois comme tel, par exemple on pourra écrire : $\dfrac{\forall y, P(y)}{\forall x, P(x)}\ \alpha$

Exemple 3.38
Notons $H = (\forall x, P(x)) \wedge (\forall y, Q(y))$. Voici une preuve du séquent $H \vdash \forall x, (P(x) \wedge Q(x))$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{H \vdash (\forall x, P(x)) \wedge (\forall y, Q(y))}\,\mathsf{ax}}{H \vdash \forall x, P(x)}\,\wedge_{\text{é,d}}}{H \vdash P(x)}\,\forall_{\text{é}} \qquad \dfrac{\dfrac{\dfrac{}{H \vdash (\forall x, P(x)) \wedge (\forall y, Q(y))}\,\mathsf{ax}}{H \vdash \forall y, Q(y)}\,\wedge_{\text{é,g}}}{H \vdash Q(x) \text{ soit } Q(y)[y \mapsto x]}\,\forall_{\text{é}}}{H \vdash P(x) \wedge Q(x)}\,\wedge_{\text{i}}}{H \vdash \forall x, (P(x) \wedge Q(x))}\,\forall_{\text{i}}$$

Justifions que les règles d'inférence portant sur les quantificateurs pouvaient bien s'appliquer, de bas en haut.
- la règle $\forall_{\text{i}}$ s'applique sans problème car H n'a aucune variable libre ;
- la règle $\forall_{\text{é}}$ à gauche s'applique sans problème car $\varphi = P(x)$ n'a aucune variable liée ;
- la règle $\forall_{\text{é}}$ à gauche s'applique sans problème car $\varphi = Q(y)$ n'a aucune variable liée.

Exercice de cours 3.39

Donner une preuve du séquent $H \vdash \neg(\forall x, P(x))$ où $H = \exists x, \neg P(x)$.