# Cours Logique et Calculabilité

L3 Informatique 2014/2015

Texte par Séverine Fratani, avec addenda par Luigi Santocanale
Version du 23 mars 2015

# Table des matières

# Chapitre 1

# Calcul des prédicats

## 1.1 Introduction

Le calcul des propositions est bien trop limité pour décrire des situations réelles. En effet il ne permet que de décrire des phrases dont la vérité ne dépend pas des individus (par exemple « Il pleut ») ; il ne peut pas représenter des phrases qui mettent en jeu des individus ou des objets (par exemple « Si x est le père de y et si z est le père de x alors z est un grand-père de y » ou « Tout individu a un père »).

Le calcul des prédicats (ou Logique du Premier Ordre) permet d'exprimer de telles relations entre individus, il est donc bien plus riche que le calcul propositionnel. En premier lieux, il contient des individus (ou entités) (donnés par des symboles de variables $x, y, z, \ldots$). Il contient des fonctions $(f, g, \ldots, s, \ldots)$ permettant de transformer des entités en autres entités (par exemple la fonction qui associe une personne à son père), et des relations $(\ldots, P, Q, R, \ldots)$ permettant de lier les individus entre eux.

Les relations appliquées aux entités (par exemple $R(x, f(y))$) peuvent être évaluées à vrai ou faux (selon les valeurs attribuées aux entités, aux fonctions et aux relations) et servent de briques de base à un langage du premier ordre obtenu à l'aide des connecteurs logiques du calcul propositionnel et de deux autres connecteurs appelés quantificateurs.

Le calcul des prédicats est donc très similaire à celui des propositions. On aura des formules définies inductivement à partir des symboles de prédicats et de fonctions. On les interprétera dans divers mondes possibles et alors elles deviendront vraies ou fausses. On aura également un système formel correct et complet pour démontrer ou réfuter des formules.

Il y a néanmoins une différence algorithmique importante : le calcul des prédicat est indécidable : il est absolument impossible de vérifier qu'une formule est vraie pour toute interprétation. Ceci vient du fait que les interprétations comportent en général une infinité d'individus, il est alors difficile de vérifier que $\forall x \varphi$ puisque $x$ peut prendre une infinité de valeurs différentes.

## 1.2 Préliminaires

On rappelle ici les notions de bases sur les fonctions et relations.

### 1.2.1 Les fonctions

Etant donné un ensemble $E$, et $n$ un entier positif, une fonction $n$-aire (ou d'arité $n$) sur $E$ est une fonction de $E^n$ dans $E$. Une fonction n'est pas une forcément une application : elle peut être non définie pour certains éléments de $E^n$, dans ce cas on dira que c'est une fonction partielle.

**Exemple 1.1.**

1. $E = \{1, 2, 3\}$ et $f$ est la fonction binaire (d'arité 2) définie pour tout couple $(a, b) \in E^2$ par :
   — $f(a, b) = 1$ si $a = 1$ et $b = 2$,
   — $f(a, b) = 2$ si $a = 2$ et $b = 3$,

— $f(a,b) = 3$ si $a = 3$ et $b = 1$,
— $f(a,b)$ est indéfinie sinon (i.e., pour les couples $(1,1),(2,2),(3,3)(3,2),(2,1),(1,3)$).

2. $E = \mathbb{N}$ et $f$ est la fonction d'arité 1 définie pour tout $n \in \mathbb{N}$ par $f(n) = n + 1$.

Une fonction d'arité 0 sur $E$ est une constante $c \in E$.

### 1.2.2 Les relations

Etant donné un ensemble $E$, et $n$ un entier positif, une relation $n$-aire (ou d'arité $n$) sur $E$ est un sous-ensemble de $E^n$.

**Exemple 1.2.**

1. $E = \{1,2,3\}$ et $R$ est la relation binaire (d'arité 2) définie par $R = \{(1,1),(2,2),(3,3)\}$.
2. $E = \mathbb{N}$ et $S$ est la relation d'arité 2 définie par $S = \{(n, n+1) \mid n \in \mathbb{N}\}$.
3. $E = \{1,2,3\}$ et $R$ est la relation unaire (d'arité 1) définie par $R = \{1,2\}$.

Une relation d'arité 0 sur $E$ est un ensemble vide puisque c'est un sous-ensemble de $E^0$.
Si $R$ est une relation d'arité $n$, on note $R(a_1,\ldots,a_n)$ ssi $(a_1,\ldots,a_n) \in R$.

## 1.3 Un exemple

Avant d'en venir aux définitions formelles, considérons les formules suivantes :
$\varphi_G : \forall x \forall y \forall z (P(x,y) \wedge P(y,z)) \Rightarrow G(x,z)$
$\varphi_P : \forall x \exists y P(y,x)$
$\varphi_C : \forall x \exists y G(y,x)$
$\varphi_D : \forall x \forall z P(z, f(x)) \Rightarrow G(z,x)$
$\varphi_F : (\varphi_G \wedge \varphi_P) \Rightarrow \varphi_C$.
Il est ici impossible de donner une valeur de vérité à toutes ces formules, et ce pour différentes raisons :

— on ne sait pas dans quel ensemble $D$ sont prises les valeurs $x, y, z$
— on ne connait pas la valeur de la fonction $f : D \to D$
— on ne connait pas la valeur des relations $P \subseteq D \times D$, et $G \subseteq D \times D$

Il est donc nécessaire de choisir une interprétation pour évaluer ces formules. Toutefois, nous verrons que c'est inutile pour la formule $\varphi_F$ qui est vraie pour toute interprétation (c'est une tautologie, ou un théorème).
Considérons donc diverses interprétations de ces formules.

### 1.3.1 Interprétation 1

Les individus sont les êtres humains. La relation $P(x,y)$ signifie que $x$ est le père de $y$. La relation $G(x,y)$ signifie que $x$ est un grand-père de $y$. La fonction $f$ associe à un individu sa mère.
$\varphi_G$ signifie alors : pour tous êtres humains $x, y, z$, si ($x$ est le père de $y$ et $y$ est le père de $z$) alors ($x$ est un grand-père de $z$).
$\varphi_P$ signifie : « pour tout individu $x$ il existe un individu $y$ tel que $y$ est le père de $x$ » soit, plus simplement, « tout individu a un père ».
$\varphi_C$ signifie que « pour tout individu $x$ il existe un individu $y$ tel que $y$ est le grand-père de $x$ » soit, plus simplement, « tout individu a un grand-père ».
$\varphi_D$ dit que si $z$ est le père de la mère de $x$ alors $z$ est un grand père de $x$.
Ces quatre formules sont vraies dans cette interprétation [1].
L'implication $\varphi_F : ((\varphi_G \wedge \varphi_P) \Rightarrow \varphi_C)$ est donc vraie dans cette interprétation.
On remarquera que les deux formules $\varphi_P$ et $\varphi_G$ sont loin de modéliser toutes les propriétés des relations $G$ et $P$. On n'a pas dit que « le père de chaque individu est unique », ni qu'« un individu $x$ peut-être le grand-père d'un autre $z$ sans qu'il existe un individu dont $x$ soit le père et qui soit le père de $z$ » (le grand-père maternel).

1. Sauf peut-être $\varphi_P$ : soit il y a un premier homme et celui-ci n'a pas de père, soit on se retrouve peu à peu, en remontant l'évolution, à inclure dans le genre humains des singes, des poissons, des bactéries, ...

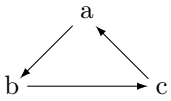
### 1.3.2 Interprétation 2

Les individus sont trois points $a, b, c$. On interprète les prédicats par les relations suivantes :

La relation $P$ est vraie pour les couples $(a, b)$, $(b, c)$ et $(c, a)$ et fausse pour les autres. En d'autres termes, $P(a, b)$, $P(b, c)$, $P(c, a)$ sont vraies, et $P(a, a)$, $P(a, c)$, $P(a, b)$, $P(b, b)$, $P(c, b)$, $P(c, c)$ sont fausses.

La relation $G$ est vraie pour les couples $(b, a)$, $(c, b)$ et $(a, c)$ et fausse pour les autres.

La fonction $f$ est définie par $f(a) = a$, $f(b) = b$ et $f(c) = a$.




En représentant $a$, $b$ et $c$ par les points suivants, $P(x, y)$ signifie « $x$ précède immédiatement $y$ » et $G(x, y)$ signifie « $x$ suit immédiatement $y$ ».

La formule $\varphi_P$ signifie que tout point a un prédécesseur immédiat, et elle vraie.

La formule $\varphi_G$ signifie que pour tous points $x, y, z$, si $x$ précède immédiatement $y$ et si $y$ précède immédiatement $z$, alors $x$ suit immédiatement $z$. Elle est également vraie.

Finalement la formule $\varphi_C$ signifie que tout point a un successeur immédiat ; elle est également vraie.

L'implication $((\varphi_P \wedge \varphi_G) \Rightarrow \varphi_C)$ est donc vraie dans cette interprétation.

La formule $\varphi_D$ signifie que pour tout $z$ et pour tout $x$, si $z$ précède immédiatement $f(x)$, alors $z$ suit immédiatement $x$. Elle est donc fausse : $c$ précède $f(a) = a$ sans que $c$ suive immédiatement $a$.

### 1.3.3 Interprétation 3

Les individus sont les quatre sommets $a, b, c, d$ du graphe suivant :

$$\begin{array}{ccc} a & \rightarrow & b \\ \uparrow & & \downarrow \\ d & \leftarrow & c \end{array}$$

$P(x, y)$ signifie que $x$ précède immédiatement $y$ sur le graphe.

$G(x, y)$ signifie que $x$ suit immédiatement $y$ sur le graphe.

La formule $\varphi_P$ signifie que tout point a un prédécesseur immédiat, et elle vraie.

La formule $\varphi_G$ signifie que pour tous points $x, y, z$, si $x$ précède immédiatement $y$ et si $y$ précède immédiatement $z$, alors $x$ suit immédiatement $z$. Elle est fausse. Finalement la formule $\varphi_C$ signifie que tout point a un successeur immédiat. Elle est également vraie.

L'implication $((\varphi_P \wedge \varphi_G) \Rightarrow \varphi_C)$ est donc vraie dans cette interprétation.

### 1.3.4 Interprétation 4

Les individus sont encore les quatre sommets $a, b, c, d$ du graphe précédent.

$P(x, y)$ signifie que $x$ précède immédiatement $y$. $G(x, y)$ signifie que pour aller de $x$ à $y$ on rencontre exactement un point $z$ différent de $x$ et de $y$.

La formule $\varphi_P$ signifie que tout point a un prédécesseur immédiat, et elle vraie.

La formule $\varphi_G$ signifie que pour tous points $x, y, z$, si $x$ précède immédiatement $y$ et si $y$ précède immédiatement $z$, alors $x$ suit immédiatement $z$. Elle est vraie.

Finalement la formule $\varphi_C$ signifie que tout point a un successeur immédiat. Elle est également vraie.

L'implication $((\varphi_P \wedge \varphi_G) \Rightarrow \varphi_C)$ est donc vraie dans cette interprétation.

### 1.3.5 Interprétation 5

Les individus sont les nombres entiers positifs.

$P(x, y)$ signifie que $x = y + 1$. Par exemple $P(5, 4)$ est vraie, mais $P(4, 5)$ est fausse.

$G(x, y)$ signifie que $x = y + 2$. Par exemple $P(6, 4)$ est vraie, mais $P(4, 6)$ est fausse.

La formule $\varphi_P$ signifie alors que pour tout entier $y$, il existe un entier $x$ tel que $x = y + 1$. Elle est vraie.

La formule $\varphi_G$ signifie que pour tous entiers $x, y, z$, si $x = y + 1$ et $z = y + 1$ alors $z = x + 2$. Elle est vraie.

Finalement la formule $\varphi_C$ signifie que pour tout entier $x$, il existe un entier $z$ tel que $z = x + 2$. Elle est vraie.

L'implication $((\varphi_P \wedge \varphi_G) \Rightarrow C)$ est donc vraie dans cette interprétation.

### 1.3.6 Interprétation 6

Les individus sont les nombres entiers positifs.

$P(x, y)$ signifie que $y = x + 1$. Par exemple $P(4, 5)$ est vraie, mais $P(5, 4)$ est fausse.

$G(x, y)$ signifie que $y = x + 2$. Par exemple $P(4, 6)$ est vraie, mais $P(6, 4)$ est fausse.

La formule $\varphi_P$ signifie alors que pour tout entier $y$, il existe un entier $x$ tel que $y = x + 1$. Elle est fausse : pour $y = 0$ un tel entier positif n'existe pas.

La formule $\varphi_G$ signifie que pour tous entiers $x, y, z$, si $y = x + 1$ et $z = y + 1$ alors $z = x + 2$. Elle est vraie.

Finalement la formule $\varphi_C$ signifie que pour tout entier $x$, il existe un entier $z$ tel que $x = z + 2$. Elle est fausse : pour $x = 0$ il n'existe pas de tel entier positif.

L'implication $((\varphi_P \wedge \varphi_G) \Rightarrow \varphi_C)$ est donc vraie dans cette interprétation.

### 1.3.7 Comparaison des interprétations

On est donc tout à fait libre d'interpréter les formules dans un « monde » de son choix, de sorte que certains énoncés deviennent vrais ou faux. On remarque néanmoins que pour chacune des interprétations considérées, l'implication $((\varphi_P \wedge \varphi_G) \Rightarrow \varphi_C)$ est vraie. Ce n'est pas un hasard, cette formule est une tautologie du calcul des prédicats. Elle est vraie dans toute interprétation.

## 1.4 Expressions et formules

### 1.4.1 Les termes

**Définition 1.3** (Signature)**.** Une *signature* est un ensemble $\mathcal{S}$ de symboles muni d'une application $\rho : \mathcal{S} \to \mathbb{N}$, appelée *arité*.

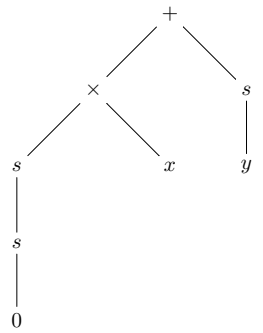
On écrira une signature comme un ensemble liste de couples. Par exemple, $\{ (f, 2), (g, 1), (h, 0) \}$ est la signature dont les symboles sont $f, g, h$, d'arité $2, 1$ et $0$, respectivement. Formellement, on a ici $\mathcal{S} = \{ f, g, h \}$, $\rho(f) = 2$, $\rho(g) = 1$, $\rho(h) = 0$. Un symbole $f \in \mathcal{S}$ tel que $\rho(f) = 0$ est appellé *constante*.

**Définition 1.4** (Termes)**.** Étant donné une signature $\mathcal{S}$ et un ensemble $X$ (de variables individuelles), l'ensemble $\mathcal{T}_{\mathcal{S}}(X)$ des termes est le plus petit ensemble tel que :

- $x \in \mathcal{T}_{\mathcal{S}}(X)$ pour toute variable $x \in X$
- si $f \in \mathcal{S}$ est d'arité $n \geq 0$ et si $t_1, \ldots, t_n \in \mathcal{T}_{\mathcal{S}}(X)$, alors $f(t_1, \ldots, t_n) \in \mathcal{T}_{\mathcal{S}}(X)$.

C'est-à-dire qu'un terme est une expression formée à partir de $X$ en utilisant les symboles de $\mathcal{S}$ de sorte qu'un symbole $f$ soit appliqué à un nombre de termes égal à $\rho(f)$. En particulier, si $f \in \mathcal{S}$ est une constante, alors elle s'applique a une liste vide de termes : $f()$ est un terme ; pour simplifier la notation, on écrit également $f$. Un terme est dit **clos** si il ne contient aucune variable.

**Exemple 1.5.** La signature de l'arithmétique contient la constante 0, le symbole $s$ d'arité 1 (qui représente la fonction « successeur »), et les symboles $+$ et $\times$ d'arité 2. On emploie la notation $\mathcal{S} = \{(0, 0), (s, 1), (+, 2), (\times, 2)\}$ pour représenter cette signature. Ainsi, pour $x, y \in X$, l'expression $+(\times(s(s(0)), x), s(y))$ (que nous nous autoriserons à écrire $(s(s(0)) \times x) + s(y)$) est un terme de $\mathcal{T}_{\mathcal{S}}(X)$ qu'on peut représenter par l'arbre suivant :




En fait, tout terme est un arbre à branchements finis. Leur hauteur (et donc leur taille) n'est par contre pas bornée car, par exemple,

$$s^n(0) := \underbrace{s(s(\dots s}_{n\text{-fois}}(0)\dots))$$

est un terme, pour tout $n \geq 1$.

**Exemple 1.6.** Le signature de la théorie des groupes est $\{\,(e,0),(inv,1),(*,2)\,\}$, où $*$ est l'opérateur de composition et $inv$ est l'opérateur « inverse » qui est habituellement noté $x^{-1}$.

### 1.4.2 Le langage

**Définition 1.7** (Langage)**.** Un *langage* (ou *vocabulaire*) du premier ordre est la donnée d'un couple de signatures $\mathcal{S} = (\mathcal{S}_\mathtt{f}, \mathcal{S}_\mathtt{r})$ disjointes (i.e. avec $\mathcal{S}_\mathtt{f} \cap \mathcal{S}_\mathtt{r} = \varnothing$).

On dit que :
— Les elements de $\mathcal{S}_\mathtt{f}$ sont les *symboles de fonction* du langage.
— Les elements de $\mathcal{S}_\mathtt{r}$ sont les *symboles de relation* (ou de prédicat) du langage.

**Remarque 1.8.**

1. Rappelons qu'une constante est un symbole (de fonction) d'arité 0.
2. On considérera (sauf mention contraire) que chaque langage contient le symbole de relation binaire $=$ et un symbole de relation d'arité 0, $\bot$ qui représentera le faux. Un langage contenant le symbole binaire $=$ sera appelé **langage égalitaire**.
3. Le rôle des relations et des fonctions est très différent. Les fonctions et constantes seront utilisés pour construire les termes (i.e., les objets du langage) tandis que les relations serviront à construire des formules (i.e., des propriétés sur ces objets)

   Par exemple $1+2$ est un terme, il désigne un objet, tandis que $1+2=3$ désigne une formule logique.

**Exemple 1.9.**

1. Le langage $\mathcal{L}_1$ de la théorie des groupes contient les symboles
   — de constante : $e$ ;
   — de fonction : $*$ (binaire) et $inv$ (unaire) ;
   — de relation : $=$ (binaire).
2. Le langage $\mathcal{L}_2$ de la théorie des ensembles contient les symboles
   — de constante : $\varnothing$ ;
   — de fonction : $\cap$ et $\cup$ binaires, $C$ unaire (le complément) ;
   — de relation : $=$, $\in$ et $\subset$, tous binaires. [2]

2. En fait, en théorie des ensembles, seulement les symboles $=$ et $\in$ sont considérés élémentaires. Les autres se définissent à partir de ces deux.

### 1.4.3 Les formules du calcul des prédicats

Etant donné un langage $\mathcal{S} = (\mathcal{S}_\mathtt{f}, \mathcal{S}_\mathtt{r})$, on construit les formules de la logique du premier ordre en utilisant les connecteurs de la logique propositionnelle et deux quantificateurs : $\forall$ et $\exists$.

Les formules sont construites à partir des formules atomiques, qui sont elles mêmes construites à partir des termes. Une formule atomique est obtenue en appliquant un symbole de relation à des termes.

On se fixe pour toute la suite un ensemble $X = \{x, x_0, x_1, \ldots, y, y_0, y_1, \ldots, z, z_0, z_1, \ldots\}$ de variables.

**Définition 1.10** (Formules atomiques)**.** Soit $\mathcal{S} = (\mathcal{S}_\mathtt{f}, \mathcal{S}_\mathtt{r})$ un langage, une formule atomique sur $\mathcal{S}$ est une expression de la forme suivante :

— $r(t_1, \ldots, t_n)$ où $r \in \mathcal{S}_\mathtt{r}$ est d'arité $n \geq 0$ et $t_1, t_2, \ldots, t_n \in \mathcal{T}_{\mathcal{S}_\mathtt{f}}(X)$ sont des termes.

**Définition 1.11** (Formules)**.** Etant donné un langage $\mathcal{S}$, l'ensemble $\mathcal{F}_{\mathrm{po}}(\mathcal{S})$ des formules du premier ordre sur $\mathcal{S}$ est le plus petit ensemble tel que :

1. toute formule atomique sur $\mathcal{S}$ appartient à $\mathcal{F}_{\mathrm{po}}(\mathcal{S})$,
2. si $\varphi$, $\psi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$ sont deux formules alors :
   — $\varphi \wedge \psi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$,
   — $\varphi \vee \psi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$,
   — $\varphi \Rightarrow \psi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$,
   — $\neg\varphi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$,
3. si $\varphi \in \mathcal{F}_{\mathrm{po}}$ et $x \in X$ alors
   — $\forall x\varphi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$,
   — $\exists x\varphi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$.

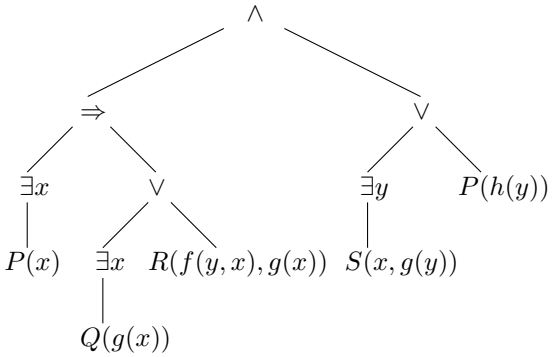
**Exemple 1.12.** Reprenons les langages de l'exemple 1.9 :

1. Dans $\mathcal{L}_1$, $x * y = e$, $y * x = e$, $x = e$, $y = e$ et $y * x = e$ sont des formules atomiques. Par conséquent, $\forall x \exists y((x * y = e) \wedge (\neg(y * x = e)))$ et $\forall x((x = e) \vee \exists y((\neg(y = e)) \wedge (y + x = e)))$ sont des formules.
2. Dans $\mathcal{L}_2$, $\forall x \forall y(x \cap y = \varnothing \Rightarrow (x = \varnothing \wedge y = \varnothing))$ est une formule.

Remarquez que, comme pour les termes, les formules peuvent être vues comme des arbres dont les feuilles sont des formules atomiques et les noeuds sont les connecteurs et quantificateurs. Par exemple, la formule

$$[(\exists x P(x)) \Rightarrow (R(f(y,x), g(x)) \vee [\exists x Q(g(x))])] \wedge [(\exists y S(x, g(y))) \vee P(h(y))] \tag{1.1}$$

sera représentée **de façon unique** par l'arbre suivant :



### 1.4.4 Occurrences libres et liées d'une variable

Lorsqu'une variable $x$ appartient à une sous-formule précédée d'un quantificateur, $\forall x$ ou $\exists x$ , elle est dite **liée** par ce quantificateur. Si une variable n'est liée par aucun quantificateur, elle est **libre**.

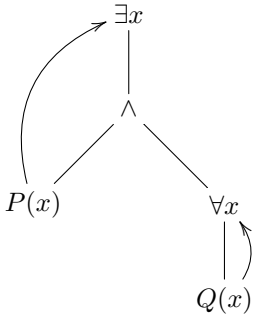
La distinction entre variable libre et variable liée est importante. Une variable liée ne possède pas d'identité propre et peut être remplacée par n'importe quel autre nom de variable qui n'apparaît pas dans la formule. Ainsi, $\exists x(x < y)$ est identique à $\exists z(z < y)$ mais pas à $\exists x(x < z)$ et encore moins à $\exists y(y < y)$.

L'ensemble des variables libres $\mathsf{FV}(\varphi)$ (depuis l'anglais, *free variable*), et l'ensemble des variables liées $\mathsf{BV}(\varphi)$ (depuis l'anglais, *bound variable*) [3] d'une formule $\varphi$ sont définis par induction sur la structure de $\varphi$ :

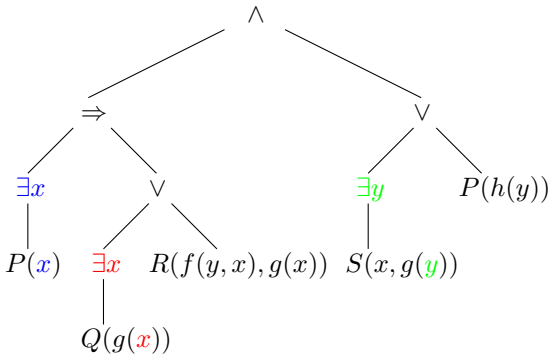
- si $\varphi$ est une formule atomique, alors tout occurrence d'une variable $x$ dans $\varphi$ est libre : $\mathsf{FV}(\varphi) = Var(\varphi)$ et $\mathsf{BV}(\varphi) = \varnothing$.
- si $\varphi$ est $\exists x\psi$ ou $\forall x\psi$, alors $\mathsf{FV}(\varphi) = \mathsf{FV}(\psi) - \{x\}$ ; $\mathsf{BV}(\varphi) = \mathsf{BV}(\varphi) \cup \{x\}$ (et toute occurrence de $x$ **libre dans** $\psi$ devient liée dans $\varphi$ par le quantificateur introduit) ;
- si $\varphi = \varphi_1 \circ \varphi_2$ (où $\circ \in \{\wedge, \vee, \Rightarrow\}$) alors $\mathsf{FV}(\varphi) = \mathsf{FV}(\varphi_1) \cup \mathsf{FV}(\varphi_2)$, $\mathsf{BV}(\varphi) = \mathsf{BV}(\varphi_1) \cup \mathsf{BV}(\varphi_2)$ ;
- si $\varphi = \neg\psi$, alors $\mathsf{FV}(\varphi) = \mathsf{FV}(\psi)$ et $\mathsf{BV}(\varphi) = \mathsf{BV}(\psi)$.

**Définition 1.13** (Formule close)**.** Une formule $\varphi$ est dite **close** ssi $\mathsf{FV}(\varphi) = \varnothing$.

**Exemple 1.14.** Une variable liée est attachée à une et une seule occurence d'un quantificateur dans la formule : celui qui dans l'arbre est son ancêtre le plus proche. Par exemple, considérons la formule $\exists x P(x) \wedge \forall x Q(x)$. Les variables sont liées de la façon suivante :

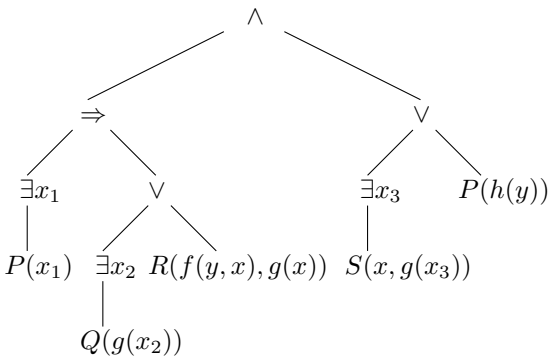



**Exemple 1.15.** Considérons la formule (1.1) et son arbre :




3. Nous ne pouvons pas utiliser un acronyme français, car on obtiendrait le même acronyme pour les variables libres et celles liées.

Une variable est liée si elle a une occurrence dans une feuille qui est descendant d'un noeud étiqueté par un quantificateur avec cette variable. Depuis l'exemple precedent, nous observons qu'il peut bien etre le cas que $\mathsf{FV}(\varphi) \cap \mathsf{BV}(\varphi) \neq \varnothing$. Nous verrons par la suite que les variables liées peuvent être renommée sans modifier la sémantique d'une formule, ceci par exemple pour éviter les ambiguités. Ainsi, nous pourrons toujours supposer que $\mathsf{FV}(\varphi) \cap \mathsf{BV}(\varphi) = \varnothing$. Par exemple, la formule précédente peut-être renommée de la façon suivante :




La nouvelle formule est donc

$$\{\,[\,\exists x_1 P(x_1)\,]\;\Rightarrow\;[\,(\exists x_2 Q(g(x_2)))\;\vee\;R(f(y,x),g(x))\,]\,\}\wedge[\,(\exists x_3 S(x,g(x_3)))\;\vee\;P(h(y))\,]\;.$$

## 1.5 Sémantique

Jusqu'ici nous nous sommes contentés de définir la **syntaxe** des langages. Les formules n'ont encore aucune signification, en partie car nous n'avons pas donné de signification aux symboles des langages. Une signature ne donne qu'un ensemble de symboles, sans en donner d'interprétation.

### 1.5.1 Structures

**Notation 1.16.** Si $D$ est un ensemble, alors $D^n$ dénotera le produit cartésien de $D$ avec lui-même $n$-fois. C'est l'ensemble des tuplets de longueur $n$ dont les éléments sont tous tirés de $D$. Soit :

$$D^n = \underbrace{D \times D \times \ldots \times D}_{n\text{-fois}} := \{\,(d_1, d_2, \ldots, d_n) \mid d_i \in D, \text{pour } i = 1, \ldots, n\,\}\,.$$

**Remarque 1.17.** Remarquez que nous pouvons donner un sens à l'ensemble $D^n$ avec $n = 0$ :

$$D^0 := \{\,()\,\}\,.$$

En particulier, le lecteur observera que les fonctions $f : D^0 \to D$ sont en bijection avec les éléments $d \in D$ (via l'évaluation $f() = d \in D$). Nous allons donc identifier fonctions $f : D^0 \to D$ (elles sont les fonctions constantes, qui ne dépendent pas de paramètres) et éléments $d \in D$.

Pour donner une **sémantique** aux formules, il faut donc commencer par donner une signification aux éléments de la signature du langage. On fait cela en associant une *structure* au langage.

**Définition 1.18.** Soit $\mathcal{S} = (\mathcal{S}_\mathtt{r}, \mathcal{S}_\mathtt{f})$ un langage. Une **$\mathcal{S}$-structure** (ou **$\mathcal{S}$-interprétation**) $\mathcal{M}$ est la donnée d'un ensemble $D_\mathcal{M}$ et

(a) d'une relation $R^\mathcal{M} \subseteq D_\mathcal{M}^{\rho(R)}$, pour chaque symbole de relation $R \in \mathcal{S}_\mathtt{r}$ ;

(b) d'une fonction totale (application) $f^\mathcal{M} : D_\mathcal{M}^{\rho(f)} \to D_\mathcal{M}$, pour chaque symbole de fonction $f \in \mathcal{S}_\mathtt{f}$.

**Remarque 1.19.** A cause de la remarque 1.17, la clause (b) dans la définition d'une $\mathcal{S}$-structure peut se rephraser comme suit :

(b.1) d'un élément $c^{\mathcal{M}}$ de $D_{\mathcal{M}}$, pour chaque symbole de constante (symbole de fonction d'arité 0) $c \in \mathcal{S}_{\mathtt{f}}$,

(b.2) d'une fonction totale (application) $f^{\mathcal{M}} : D_{\mathcal{M}}^{\rho(f)} \to D_{\mathcal{M}}$, pour chaque symbole de fonction $f \in \mathcal{S}_{\mathtt{f}}$, tel que $\rho(f) \geq 1$.

**Exemple 1.20.** Supposons que le langage soit composé de la constante $o$ et de la fonction unaire $s$. On peut choisir, par exemple, les structures suivantes :

1. $D_{\mathcal{M}}$ est l'ensemble des entiers naturels, $o^{\mathcal{M}}$ est le nombre 0 et $s^{\mathcal{M}}$ est la fonction donnant le successeur, c'est-à-dire $s^{\mathcal{M}}(n) = n + 1$ ;
2. $D_{\mathcal{M}}$ est un ensemble de personnes, $o^{\mathcal{M}}$ c'est Paul et $s^{\mathcal{M}}$ est la fonction donnant le père d'une personne.

**Exemple 1.21.** Supposons que $\mathcal{S}_{\mathtt{f}} = \{ (o, 0), (s, 1) \}$ et $\mathcal{S}_{\mathtt{r}} = \{ (\mathtt{Even}, 1) \}$. Comme auparavant, on peut choisir $D_{\mathcal{M}}$ est l'ensemble des entiers naturels, $o^{\mathcal{M}}$ est le nombre 0 et $s^{\mathcal{M}}$ est la fonction donnant le successeur. Pour compléter la définition de $\mathcal{S}$-structure nous pouvons poser

$$\mathtt{Even}^{\mathcal{M}} := \{ n \in \mathbb{N} \mid n \bmod 2 = 1 \}.$$

Bien que la $\mathcal{S}$-structure ainsi définie puisse apparaître bien drôle (ou inappropriée), la définition de cette structure n'est pas incorrecte : *rien nous oblige à donner à un symbole une interprétation par défaut.*

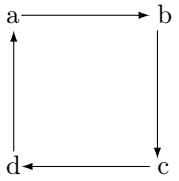
La situation est assez différente quand le symbole d'égalité est parmi les symboles de relation. Dans ce cas, on interprétera, *par défaut*, la relation d'égalité sur l'image de la fonction diagonale :

$$=^{\mathcal{M}} := \{ (d, d) \mid d \in D_{\mathcal{M}} \}.$$

**Exemple 1.22.** Supposons que le langage est composé du symbole de relation $B$ d'arité 2.

On peut choisir par exemple les interprétations suivantes :

1. $D_{\mathcal{M}}$, le domaine, est l'ensemble des sommets du graphe




et $B$ est interprété comme la relation "flèche" : $B^{\mathcal{M}} = \{(a, b), (b, c), (c, d), (d, a)\}$

2. $D_{\mathcal{M}}$, le domaine, est le même qu'aupravant, mais maintenant $B$ est interprété comme la relation "ne pas être voisin direct" : $B^{\mathcal{M}} = \{(b, d), (d, b), (a, c), (c, a)\}$. Ce modèle, même s'il partage avec le précédent le domaine, est différent du précédent.
3. Le domaine est l'ensemble des triangles et $B$ est la relation "avoir la même aire"

**Exemple 1.23.** Supposons que $\rho(R) \leq 2$ pour tout $R \in \mathcal{S}_{\mathtt{r}}$, et que $\rho(f) \leq 1$ pour tout $f \in \mathcal{S}_{\mathtt{f}}$. On peut représenter une $\mathcal{S}$-structure $\mathcal{M}$ comme un (sorte de) graphe orienté étiqueté :

— les noeuds du graphe sont les éléments du domaine $D_{\mathcal{M}}$ ;
— un noeud $d \in D_{\mathcal{M}}$ est étiqueté par $P \in \mathcal{S}_{\mathtt{r}}$ si $d \in P^{\mathcal{M}}$ ;
— un noeud $d \in D_{\mathcal{M}}$ est étiqueté par un symbole de constante $c \in \mathcal{S}_{\mathtt{f}}$ si $d = c^{\mathcal{M}}$ ;
— on met une arête de $d$ vers $d'$ si $(d, d') \in R^{\mathcal{M}}$, pour un quelque symbole de relation $R \in \mathcal{S}_{\mathtt{r}}$ ; une arête $d \to d'$ est étiqueté par les symboles $R \in \mathcal{S}_{\mathtt{r}}$ tels quel $(d, d') \in R$ ;
— on met une arête pointillée de $d$ vers $d'$ si $f(d) = d'$ ; une arête pointillée $d \to d'$ est étiqueté par les symboles $f \in \mathcal{S}_{\mathtt{f}}$ tels quel $f(d) = d'$ ;

Par exemple, si $\mathcal{S}_{\mathtt{r}} = \{ (P, 2) \}$ et $\mathcal{S}_{\mathtt{f}} = \{ (f, 1) \}$, alors la structure $\mathcal{M} = \langle D_{\mathcal{M}}, P^{\mathcal{M}}, f^{\mathcal{M}} \rangle$ avec
— $D_{\mathcal{M}} = \{a, b, c\}$,

— $P^{\mathcal{M}} = \{(a,b),(b,c),(c,a)\}$,
— $f^{\mathcal{M}} : a \mapsto a, b \mapsto b, c \mapsto a$ ;
peut se représenter comme le graphe étiqueté de la figure 1.1.

**Exemple 1.24.** Voici quelques exemples de structures importantes :

| Domaine | Fonctions | Relations | Nom |
|---|---|---|---|
| $\mathbb{N}$ | $0, s, +$ | $=, \leq$ | Arithmétique de Presburger |
| $\mathbb{N}$ | $0, s, +, \times$ | $=, \leq$ | Arithmétique de Peano |
| $\mathbb{R}$ | $0, s, +, \times$ | $=, \leq$ | Théorie des réels |
| $\{0\}$ | $\varnothing$ | $\{p_0, \ldots, p_n, \ldots\}$ | Structure propositionnelle |
| $\mathcal{T}_{\mathcal{S}_\mathtt{f}}(\varnothing)$ | $\hat{\mathcal{S}}_\mathtt{f}$ | $\hat{\mathcal{S}}_\mathtt{r}$ | Modèle de Herbrand |

Dans le cas des modèle de Herbrand, le domaine est l'ensemble des termes définis sur un ensemble de variables vide. Une fonction $\hat{f} \in \hat{\mathcal{S}}_\mathtt{f}$ d'arité $n$ associe aux termes $t_1, \ldots, t_n$ le nouveau terme $f(t_1, \ldots, t_n)$, et les relations sont quelconques.

### 1.5.2 Evaluation (des termes et) des formules

Comme pour le calcul des propositions, on va définir l'interprétation d'une formule en fonction de l'interprétation des formules élémentaires (ici les formules atomiques). Pour qu'une formule soit évaluable à vrai ou faux (1 ou 0), il faut non seulement dire comment s'interprètent les prédicats et les symboles de fonctions, (ce qui est l'analogue d'interpréter les propositions pour le calcul propositionnel) mais aussi ce que valent les variables : en effet les formules peuvent contenir des variables libres, et on a besoin de connaître leur valeur pour que la formule soit évaluable. Bien sûr la valeur des variables liées n'intervient en rien dans le calcul de la valeur d'une formule.

Par exemple pour connaître la valeur de $P(x,y)$ il faut non seulement connaître la signification de $P$ (donnée par la structure), mais aussi la valeur de $x$ et de $y$ qui sera donnée par une **valuation**. Dans la formule $\exists x P(x,y)$ il faut connaître la valeur $P$ et celle de $y$ (mais celle de $x$ n'a aucune importance). Cependant, comme la valeur de $\exists x P(x,y)$ va être définie en fonction de la valeur de $P(x,y)$, on fera aussi intervenir la valeur de $x$, ou plus précisément les valeurs de $P(x,y)$ pour toutes les valeurs de $x$.

Nous allons donc définir la valeur d'une formule $\varphi$ d'un langage $(\mathcal{S}, X)$ en fonction d'une $\mathcal{S}$-structure $\mathcal{M}$ et d'une valuation des variables.

**Définition 1.25.** Une **valuation** de l'ensemble $X$ des variables individuelles dans une structure $\mathcal{M}$ est une fonction $\mathcal{V}$ de l'ensemble $X$ vers le domaine de $\mathcal{M}$, soit $\mathcal{V} : X \to D_{\mathcal{M}}$.

On définit maintenant la valeur $[\![\, \varphi \,]\!]_{\mathcal{M},\mathcal{V}}$ d'une formule $\varphi$ en fonction d'une structure $\mathcal{M}$ et d'une valuation $\mathcal{V}$. On commence naturellement par donner la valeur des termes.

**Définition 1.26.** Soit $\mathcal{M}$ une $\mathcal{S}$-structure et $\mathcal{V}$ une valuation de $X$ dans $D_{\mathcal{M}}$ ; la valeur d'un terme $t \in \mathcal{T}_{\mathcal{S}_\mathtt{f}}(X)$, notée $[\![\, t \,]\!]_{\mathcal{M},\mathcal{V}}$ est un élément de $D_{\mathcal{M}}$ défini (par induction) par :
— pour toute variable $x \in X$, $[\![\, x \,]\!]_{\mathcal{M},\mathcal{V}} = \mathcal{V}(x)$ ;
— pour tout $f \in \mathcal{S}_\mathtt{f}$ d'arité $n \geq 0$, pour tous termes $t_1, \ldots t_n$,

$$[\![\, f(t_1, \ldots, t_n) \,]\!]_{\mathcal{M},\mathcal{V}} := f^{\mathcal{M}}([\![\, t_1 \,]\!]_{\mathcal{M},\mathcal{V}}, \ldots, [\![\, t_n \,]\!]_{\mathcal{M},\mathcal{V}}) \,.$$

**Remarque 1.27.** La définition de l'évaluation pour un terme construit via un symbole de fonction peut se séparer en deux cas, selon l'arité du symbole :
— pour toute constante $c \in \mathcal{S}_\mathtt{f}$ (c'est-à-dire symbole de fonction d'arité 0), $[\![\, c() \,]\!]_{\mathcal{M},\mathcal{V}} = [\![\, c \,]\!]_{\mathcal{M},\mathcal{V}} := c^{\mathcal{M}}$ ;
— pour tout $f \in \mathcal{S}_\mathtt{f}$ d'arité $n \geq 1$, pour tous termes $t_1, \ldots t_n$, $[\![\, f(t_1, \ldots, t_n) \,]\!]_{\mathcal{M},\mathcal{V}} := f^{\mathcal{M}}([\![\, t_1 \,]\!]_{\mathcal{M},\mathcal{V}}, \ldots, [\![\, t_n \,]\!]_{\mathcal{M},\mathcal{V}})$.

Si $\mathcal{M}$ et $\mathcal{V}$ sont fixées, toute formule atomique prend la valeur 0 ou 1, selon la définition formelle suivante, qui suit l'intuition :

**Définition 1.28.** La valeur $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}}$ d'une formule atomique $\varphi$ est définie par :
— pour tout $R \in \mathcal{S}_{\mathtt{r}}$ d'arité $n$, pour tous termes $t_1, \ldots, t_n$,

$$[\![ R(t_1, \ldots, t_n) ]\!]_{\mathcal{M},\mathcal{V}} = 1 \qquad \text{ssi} \qquad ([\![ t_1 ]\!]_{\mathcal{M},\mathcal{V}}, \ldots, [\![ t_n ]\!]_{\mathcal{M},\mathcal{V}}) \in R^{\mathcal{M}} .$$

Pour évaluer une formule contenant des quantificateurs, il nous faudra la notion de variante d'une valuation.

**Notation 1.29.** Nous allons noter $\mathcal{V}[x := a]$ la valuation $\mathcal{V}'$ telle que $\mathcal{V}'(x) = a$ et $\mathcal{V}(y)' = \mathcal{V}(y)$ pour tout $y \in X \setminus \{ x \}$. Autrement, pour tout $y \in X$ :

$$\mathcal{V}[x := a](y) = \begin{cases} a , & \text{si } y = x , \\ \mathcal{V}(y) , & \text{sinon.} \end{cases}$$

On dira alors que $\mathcal{V}[x := a]$ est une **variante** en $x$ de $\mathcal{V}$.

Enfin, la valeur d'une formule est définie par induction sur la structure de la formule :

**Définition 1.30** (Valeur d'une formule)**.** Etant donné un langage $\mathcal{S}$ et une formule $\varphi$ de $\mathcal{F}_{\mathrm{po}}(\mathcal{S})$, la valeur de $\varphi$ pour la $\mathcal{S}$-structure $\mathcal{M}$ et la valuation $\mathcal{V}$ est notée $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}}$ et est définie de la façon suivante :
— $[\![ \varphi \wedge \psi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ ssi $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ et $[\![ \psi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ ;
— $[\![ \varphi \vee \psi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ ssi $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ ou $[\![ \psi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ ;
— $[\![ \neg\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ ssi $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}} = 0$ ;
— $[\![ \varphi \Rightarrow \psi ]\!]_{\mathcal{M},\mathcal{V}} = 0$ ssi $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ et $[\![ \psi ]\!]_{\mathcal{M},\mathcal{V}} = 0$ ;
— $[\![ \forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ si et seulement si pour tout $a \in D_{\mathcal{M}}$, $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 1$ ;
— $[\![ \exists x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ si et seulement s'il existe $a \in D_{\mathcal{M}}$ tel que $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 1$.

**Remarque 1.31.** Le quantificateur universel peut se considérer comme une sorte de grande conjonction. En fait, on a

$$[\![ \forall x.\varphi ]\!]_{\mathcal{M},\mathcal{V}} = \min\{ [\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} \mid a \in D_{\mathcal{M}} \} .$$

De façon similaire, le quantificateur existentiel est une sorte disjonction :

$$[\![ \exists x.\varphi ]\!]_{\mathcal{M},\mathcal{V}} = \max\{ [\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} \mid a \in D_{\mathcal{M}} \} .$$

Par ailleurs, il devient possible remplacer un quantificateur universel par une conjonction (de façon à simuler la logique du premier ordre par la logique propositionnelle) seulement si le domaine $D_{\mathcal{M}}$ est fini, et en plus il est fixé. Que dire si $D_{\mathcal{M}}$ est $\mathbb{N}$ (ici, le domaine est infini) ou si on se pose la question si $\forall x(x = x)$ est vraie dans n'importe quelle structure (ici, on ne peut pas fixer le domaine) ?

**On notera souvent $\mathcal{M}, \mathcal{V} \models \varphi$ à la place de $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$.**

**Remarque 1.32.** La valeur de $[\![ \forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}}$ ou $[\![ \exists x\varphi ]\!]_{\mathcal{M},\mathcal{V}}$ ne dépend pas de $\mathcal{V}(x)$. Par suite, la valeur de $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}}$, où $\varphi$ est une formule quelconque ne dépend pas de $\mathcal{V}(x)$ lorsque $x$ n'est pas une variable libre. En particulier, si $\varphi$ est une formule close (sans variables libres), alors $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}}$ ne dépend pas de $\mathcal{V}$ et par conséquent on écrira $[\![ \varphi ]\!]_{\mathcal{M}}$ à la place de $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}}$ et $\mathcal{M} \models \varphi$ à la place de $\mathcal{M}, \mathcal{V} \models \varphi$.

L'ordre d'apparition des quantificateurs dans une formule est important. Il détermine le sens de la formule. Si nous donnons au prédicat $p(x, y)$ la signification "$x$ aime $y$", alors, nous obtenons des significations différentes pour la formule $Q_1xQ_2yp(x, y)$ (où $Q_1$ et $Q_2$ sont des quantificateurs).
— $\forall x\forall yp(x, y)$ tout le monde aime tout le monde
— $\exists x\forall yp(x, y)$ il existe des personnes qui aiment tout le monde
— $\exists y\forall xp(x, y)$ il existe des personnes aimées de tous
— $\forall x\exists yp(x, y)$ toute personne aime quelqu'un

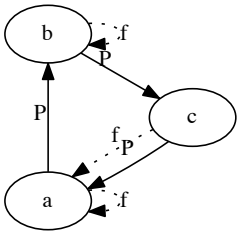



FIGURE 1.1 – Structure en forme de graphe étiqueté

— $\forall y \exists x p(x, y)$ toute personne est aimée par quelqu'un
— $\exists x \exists y p(x, y)$ il y a une personne qui aime quelqu'un.

On peut remarquer que les seuls cas où on peut échanger l'ordre des quantificateurs sans modifier le sens de la formule sont ceux où les quantificateurs sont identiques : $\exists x \exists y \varphi \equiv \exists y \exists x \varphi$ et $\forall x \forall y \varphi \equiv \forall y \forall x \varphi$. C'est pourquoi en mathématique on écrit souvent $\exists xy \varphi$ ou $\forall xy \varphi$.

**Exemple 1.33.** Considérons la formule

$$\varphi := \forall x \exists y (\, P(x, f(y)) \vee P(y, f(x)) \,) \,,$$

à évaluer dans $\mathcal{M} = \langle D_{\mathcal{M}}, P^{\mathcal{M}}, f^{\mathcal{M}} \rangle$ où :

— $D_{\mathcal{M}} = \{a, b, c\}$,
— $P^{\mathcal{M}} = \{(a, b), (b, c), (c, a)\}$,
— $f^{\mathcal{M}} : a \mapsto a, b \mapsto b, c \mapsto a$.

Cette structure, étant sur un langage relationnel d'arité au plus 2 et sur un langage fonctionnel d'arité au plus 1, est représentée en forme de graphe étiqueté en figure 1.1.

Pour calculer $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}}$ on peut d'abord considérer toutes les valuations possibles de $x$ et de $y$, soient 9 valuations. Pour chaque valuation, nous pouvons évaluer la formule $P(x, f(y)) \vee P(y, f(x))$, en évaluant d'abord les termes $x, f(x), y, f(y)$, pour ensuite évaluer la disjonction selon les règles usuelles de la logique propositionnelle. Nous avons donc :

$\mathcal{V}_{aa}$ : $x = a, y = a$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} = \max([\![ P(x, f(y)) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} , [\![ P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{aa}}) = 0$$

car $[\![ f(y) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} = [\![ f(x) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} = a$ et donc

$$[\![ P(x, f(y)) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} = [\![ P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} = 0 \,, \quad \text{car} \quad (a, a) \notin P^{\mathcal{M}} \,.$$

Dans la suite, nous allons abréger l'exposition, tous ces calculs seront sous-entendus. Avec un abus de notation, nous allons simplement écrire :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{aa}} = [\![ P(a, a) \vee P(a, a) ]\!]_{\mathcal{M}} = 0 \,.$$

$\mathcal{V}_{ab}$ : $x = a, y = b$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{ab}} = [\![ P(a, b) \vee P(b, a) ]\!]_{\mathcal{M}} = 1;$$

$\mathcal{V}_{ac}$ : $x = a, y = c$ :

$$[\![ (P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}_{ac}} = [\![ P(a, a) \vee P(c, a) ]\!]_{\mathcal{M}} = 1;$$

$\mathcal{V}_{ba}$ : $x = b, y = a$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{ba}} = [\![ P(b, a) \vee P(a, b) ]\!]_{\mathcal{M}} = 1 \,;$$

$\mathcal{V}_{bb}$ : $x = b, y = b$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{bb}} = [\![ P(b, b) \vee P(b, b) ]\!]_{\mathcal{M}} = 0\,;$$

$\mathcal{V}_{bc}$ : $x = b, y = c$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{bc}} = [\![ P(b, a) \vee P(c, b) ]\!]_{\mathcal{M}} = 0\,;$$

$\mathcal{V}_{ca}$ : $x = c, y = a$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{ca}} = [\![ P(c, a) \vee P(a, a) ]\!]_{\mathcal{M}} = 1\,;$$

$\mathcal{V}_{cb}$ : $x = c, y = b$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{cb}} = [\![ P(c, b) \vee P(b, a) ]\!]_{\mathcal{M}} = 0\,;$$

$\mathcal{V}_{cc}$ : $x = c, y = c$ :

$$[\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{cc}} = [\![ P(c, a) \vee P(c, a) ]\!]_{\mathcal{M}} = 1\,.$$

Commençons par étudier les valeurs possibles de la sous-formule $\exists y(P(x, f(y)) \vee P(y, f(x)))$ :

$\mathcal{V}_a : x := a$ :

$$[\![ \exists y(P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}_a} = 1\,, \qquad \text{car } [\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{ab}} = 1.$$

Donc pour toute valuation $\mathcal{V}$ telle que $\mathcal{V}(x) = a$, $[\![ \exists y(P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}} = 1$.

$\mathcal{V}_b : x := b$ :

$$[\![ \exists y(P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}_b} = 1\,, \qquad \text{car } [\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{ba}} = 1.$$

Donc pour toute valuation $\mathcal{V}$ telle que $\mathcal{V}(x) = b$, $[\![ \exists y(P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}} = 1$.

$\mathcal{V}_c : x := c$ :

$$[\![ \exists y(P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}_c} = 1\,, \qquad \text{car } [\![ P(x, f(y)) \vee P(y, f(x)) ]\!]_{\mathcal{M},\mathcal{V}_{ca}} = 1.$$

Donc pour toute valuation $\mathcal{V}$ telle que $\mathcal{V}(x) = c$, $[\![ \exists y(P(x, f(y)) \vee P(y, f(x))) ]\!]_{\mathcal{M},\mathcal{V}} = 1$.

Donc pour toute valuation $\mathcal{V}$, nous avons $\mathcal{M}, \mathcal{V} \models \exists y(P(x, f(y)) \vee P(y, f(x)))$, i.e.,

$$\mathcal{M} \models \forall x \exists y(P(x, f(y)) \vee P(y, f(x)))\,.$$

**Remarque 1.34.** Observons que la notation $P(a, a)$ et les notations similaires sont impropres, mais très utiles en pratique, et aussi très utilisés par les logiciens!!! En fait, $P(a, a)$ n'est pas une formule (atomique) du premier ordre, car ici $a$ n'est pas un terme, mais plutôt un élément du domaine $D_{\mathcal{M}}$. Pour pouvoir justifier cette notation il faut ajouter au langage un symbole de constante $c_a$, pour tout élément du domaine $a \in D_{\mathcal{M}}$ ; en plus, nous devons étendre l'interprétation, de façon que $c_a^{\mathcal{M}} = a$.

## Vocabulaire

**Définition 1.35** (Modèle)**.** Soit $\varphi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$ une formule close (i.e., qui ne contient pas de variable libre) et $\mathcal{M}$ une $\mathcal{S}$-structure. La structure $\mathcal{M}$ est un **modèle** de $\varphi$ si $\mathcal{M} \models \varphi$.

Soit $\Gamma \subseteq \mathcal{F}_{\mathrm{po}}(\mathcal{S})$ un ensemble de formules closes (i.e., qui ne contient pas de variable libre) et $\mathcal{M}$ une $\mathcal{S}$-structure. La structure $\mathcal{M}$ est un **modèle** de $\Gamma$ si $\mathcal{M} \models \varphi$ pour tout $\varphi \in \Gamma$.

**Définition 1.36** (Tautologie)**.** Une formule close $\varphi \in \mathcal{F}_{\mathrm{po}}(\mathcal{S})$ est une **tautologie** si $\mathcal{M} \models \varphi$ pour toute $\mathcal{S}$-structure $\mathcal{M}$.

**Définition 1.37** (Formule insatisfaisable)**.** Une formule close $\varphi \in \mathcal{F}_{\text{po}}(\mathcal{S})$ est **insatisfaisable** si elle n'a pas de modèle.

**Définition 1.38** (Conséquence logique)**.** Une formule close $\varphi$ est **conséquence logique** d'un ensemble de formules closes $\Gamma$ si tout modèle de $\Gamma$ est un modèle de $\varphi$. On écrit alors $\Gamma \models \varphi$.

**Définition 1.39** (Théorie)**.** Une **théorie** est l'ensemble des conséquences logiques d'un ensemble de formules closes.

Par exemple, la théorie des groupes est l'ensemble des formules logiques qui sont vraies dans tous les groupes.

**Définition 1.40** (Equivalence)**.** Deux formules $\varphi$ et $\psi$ de $\mathcal{F}_{\text{po}}(\mathcal{S})$ sont **équivalentes** (noté $\varphi \equiv \psi$) si pour toute $\mathcal{S}$-structure $\mathcal{M}$ et toute valuation $\mathcal{V} : X \to D_{\mathcal{M}}$, on $\mathcal{M}, \mathcal{V} \models \varphi$ ssi $\mathcal{M}, \mathcal{V} \models \psi$.

Attention : on parle d'équivalence de deux formules aussi quand elle ne sont pas de formules closes.

On peut par ailleurs noter que $\varphi$ et $\psi$ sont équivalentes lorsque $\mathcal{M}, \mathcal{V} \models \varphi \Leftrightarrow \psi$ pour tout $\mathcal{M}$ et $\mathcal{V}$, de façon qu'elles sot équivalentes ssi $\overline{\forall}(\varphi \Leftrightarrow \psi)$ est une tautologie. Ici, $\overline{\forall}(\varphi)$ est la clôture universelle de la formule $\varphi$, obtenue de $\varphi$ en lui ajoutant une suite de quantificateurs universels $\forall x_1 \forall x_2 \ldots \forall x_n$, où $x_1, \ldots, x_n$ est la liste des variables libres de $\varphi$.

## 1.6 Manipulation de formules

### 1.6.1 Substitution de variables

Dans la suite, nous allons préciser ce que veut dire qu'une formule $\psi \in \mathcal{F}_{\text{po}}(\mathcal{S})$ est obtenue d'une formule $\varphi \in \mathcal{F}_{\text{po}}(\mathcal{S})$ en substituant toute occurrence d'une variable libre $x$ par un terme (ce qui sera noté par $\psi = \varphi_{\{x \to t\}}$).

**Remarque 1.41.** Bien que la notion soit intuitive, il ne faut pas que des variables libres deviennent liées par cette substitution. Considérons ce qu'il se passe si l'on substitue de façon naïve $x$ par le terme $y$ dans $\exists y R(x, y)$. La formule $\exists y R(y, y)$ n'est pas le résultat souhaité. On souhaite plutôt avoir comme résultat la formule suivante : $\exists z R(y, z)$, d'où on devine la nécessite de renommer les variables de façon qu'une substitution ne crée pas des nouvelle variables liées.

Si $\sigma$ est la substitution $\{x_1 \to t_1, \ldots, x_n \to t_n\}$, on note $\varphi_\sigma$ la formule $\varphi$ dans laquelle toutes les occurrences libres de $x_1, \ldots, x_n$ ont été remplacées respectivement par $t_1, \ldots, t_n$. Pour définir formellement la notion de substitution dans une formule, on commence par la définir pour les termes. Soit $t$ un terme de $\mathcal{T}_{\mathcal{S}}(X)$ et $\sigma$ une substitution :

— si $t = x$ et $x \in Dom(\sigma)$ alors $t_\sigma = \sigma(x)$, (et $t_\sigma = x$ si $x \notin Dom(\sigma)$) ;
— si $t = c$ alors $t_\sigma = c$ ;
— si $t = f(t_1, \ldots, t_n)$ alors $t_\sigma = f(t_{1\sigma}, \ldots, t_{n\sigma})$.

On définit maintenant la substitution par récurrence sur la structure de la formule :

$$
\begin{aligned}
R(t_1, \ldots, t_n)_\sigma &:= R(t_{1\sigma}, \ldots, t_{n\sigma})\,, \\
(\varphi \circ \psi)_\sigma &:= (\varphi_\sigma) \circ (\psi_\sigma), \text{ pour tout connecteur binaire } \circ\,, \\
(Qx.\varphi)_\sigma &:= \begin{cases} Qx.(\varphi_{\sigma[x \to x]})\,, & \text{si } x \notin Im(\sigma) \\ Qy.(\varphi_{\{x \to y\}})_\sigma\,, & \text{si } x \in Im(\sigma), \text{ où } y \notin Var(\varphi) \cup Dom(\sigma) \cup Im(\sigma)\,, \end{cases}
\end{aligned}
$$

où $Q \in \{\exists, \forall\}$, et $\sigma[x \to x]$ est la substitution $\sigma'$ telle que $\sigma'(x) = x$, et $\sigma'(y) = \sigma(y)$ pour $y \neq x$.

**Proposition 1.42.** *Si $y \notin Var(\varphi)$, alors $\exists x \varphi \equiv \exists y(\varphi_{\{x \to y\}})$.*

Grâce à cette proposition, on pourra supposer désormais, sans perte de généralité, que les différentes occurrences liées d'une variables sont toutes liées au même quantificateur et qu'aucune variable n'admet à la fois des occurrences libres et des occurrences liées.

### 1.6.2 Equivalences classiques

Les équivalences données dans le cadre du calcul propositionnel restent vraies. Nous en donnons d'autres ici, les preuves sont laissées en exercice.

— Lois de **conversion** des quantificateurs :

$$\neg\forall x\varphi \equiv \exists x\neg\varphi\,, \qquad \neg\exists x\varphi \equiv \forall x\neg\varphi\,.$$

— Lois de **distribution** des quantificateurs :

$$\forall x(\varphi \wedge \psi) \equiv (\forall x\varphi \wedge \forall x\psi)\,, \qquad \exists x(\varphi \vee \psi) \equiv (\exists x\varphi \vee \exists x\psi)\,.$$

— Lois de **permutation** des quantificateurs de même sorte :

$$\forall x\forall y\varphi \equiv \forall y\forall x\varphi\,, \qquad \exists x\exists y\varphi \equiv \exists y\exists x\varphi\,.$$

— Lois de **passage** : *si $x$ ne figure pas à titre d'occurence libre dans $\psi$*, on a les lois suivantes :

$$
\begin{aligned}
\forall x(\varphi \wedge \psi) &\equiv (\forall x\varphi) \wedge \psi & \exists x(\varphi \wedge \psi) &\equiv (\exists x\varphi) \wedge \psi \\
\forall x(\varphi \vee \psi) &\equiv (\forall x\varphi) \vee \psi & \exists x(\varphi \vee \psi) &\equiv (\exists x\varphi) \vee \psi \\
\forall x(\varphi \Rightarrow \psi) &\equiv (\exists x\varphi) \Rightarrow \psi & \exists x(\varphi \Rightarrow \psi) &\equiv (\forall x\varphi) \Rightarrow \psi \\
\forall x(\psi \Rightarrow \varphi) &\equiv \psi \Rightarrow (\forall x\varphi) & \exists x(\psi \Rightarrow \varphi) &\equiv \psi \Rightarrow (\exists x\varphi)
\end{aligned}
$$

*Remarquez le changement de quantificateur dans les équivalences* $\forall x(\varphi \Rightarrow \psi) \equiv (\exists x\varphi) \Rightarrow \psi$ et $\exists x(\varphi \Rightarrow \psi) \equiv (\forall x\varphi) \Rightarrow \psi$.
La règle qui établit l'équivalence entre $\exists x(\varphi \wedge \psi)$ et $(\exists x\varphi) \wedge \psi$ (lorsque $x$ n'est pas une variable libre de $\psi$) est aussi connue sous le nom de **loi de Frobenius**.

— Lois de **réalphabétisation** (renommage) des variables.
On peut toujours renommer une variable liée et la variable du quantificateur au sein d'une formule. Cependant, le nouveau nom ne doit pas être un nom déjà utilisé pour une variable libre ou liée de la formule.
Par exemple, dans $\exists x(\forall xF(x,y) \Rightarrow (G(x) \vee q))$, on peut opérer deux renommages : on peut renommer d'abord l'occurrence de $x$ dans $F(x,y)$ à $t$, en obtenant ainsi $\exists x(\forall tF(t,y) \Rightarrow (G(x) \vee q))$, et ensuite renommer le $x$ restant à $z$ ; on obtient $\exists z(\forall tF(t,y) \Rightarrow (G(z) \vee q))$.
Soit $\{ x \to t \}$ la substitution qui remplace la variable $x$ par la variable $t$. La loi de réalphabétisation peut se déduire de la regle élémentaire de réalphabétisation suivante :

$$Qx\varphi \equiv Qt(\varphi_{\{ x \to t \}}) \qquad \text{où } Q \in \{ \forall, \exists \}$$

pourvu que $t$ *n'est pas une variable libre de* $\varphi$ (en particulier, quand $t$ n'a aucune occurrence dans $t$) et *toutes les occurrences de* $x$ *sont libres dans* $\varphi$.

**Exemple 1.43.** Nous pouvons démontrer l'équivalence entre les formules $\neg\forall x\varphi(x)$ et $\exists x\neg\varphi(x)$ de la façon suivante. Soient $\mathcal{M}$ une $\mathcal{S}$-structure et $\mathcal{V}$ une valuation fixés.

(a) Supposons que $[\![ \exists x\neg\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ et montrons que $[\![ \neg\forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$. De $\max_{a \in D_{\mathcal{M}}} [\![ \neg\varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 1$, nous déduisons que $[\![ \neg\varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 1$ pour un quelque $a \in D_{\mathcal{M}}$, donc $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 0$ et $[\![ \forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = \min_{a \in D_{\mathcal{M}}} [\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 0$, donc $[\![ \neg\forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$.

(b) Supposons, par contre, que $[\![ \neg\forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$ et montrons que $[\![ \exists x\neg\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$. On a bien que $[\![ \forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 0$ ; depuis

$$0 = [\![ \forall x\varphi ]\!]_{\mathcal{M},\mathcal{V}} = \min_{a \in D_{\mathcal{M}}} [\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 0$$

nous déduisons que ce minimum est réalisé : donc $[\![ \varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 0$ pour un quelque $a \in D_{\mathcal{M}}$, d'où $[\![ \neg\varphi ]\!]_{\mathcal{M},\mathcal{V}[x:=a]} = 1$, et $[\![ \exists x\neg\varphi ]\!]_{\mathcal{M},\mathcal{V}} = 1$.

Les argumentaires ci-dessus sont acceptés dans un cadre classique. L'argumentaire (b) se révèle par contre insatisfaisante, si depuis une preuve d'existence d'un objet avec une certaine propriété, nous souhaitons construire aussi un tel objet. Supposons, par exemple, que nous avons réussi à montrer que « non pour tout $n$, si $n$ est premier, alors $n = 2^k - 1$ pour un nombre $k$ ». L'argumentaire (b) prétend qu'il est possible déduire l'existance d'un nombre $n$ qui n'est pas de la forme $2^k - 1$, mais ne spécifie d'aucune façon comment le construire. Pour cette raison, en *logique intuitionniste* (du premier ordre), qui se construit autour de l'idée qu'une preuve logique d'existence doit donner aussi un moyen de construire un objet témoignant l'existence, seule la formule $\exists x\neg\varphi \Rightarrow \neg\forall x\varphi$ est considérée une tautologie, la formule $\neg\forall x\varphi \Rightarrow \exists x\neg\varphi$ n'est pas considérée une tautologie. Voir par exemple [Miq05].

**Exercice 1.44.** Montrez que les formules $\forall x(\varphi \vee \psi)$ et $\forall x\varphi \vee \forall x\psi$ ne sont pas, en général, équivalentes. Argumentez de façon similaire pour $\exists x(\varphi \wedge \psi)$ et $\exists x\varphi \wedge \exists x\psi$.

### 1.6.3 Formes Normales

Établir la consistance d'une formule du calcul des prédicats est un problème difficile. On peut alors essayer de travailler sur une version de forme plus simple mais de consistance équivalente à la formule initiale. Nous introduisons dans cette section la forme clausale pour la logique des prédicats. Toute formule de la logique des prédicats du premier ordre admet une représentation sous forme de clause qui préserve sa satisfiabilité. À la différence des formes normales, une clause n'est pas logiquement équivalente à la formule dont elle dérive. Nous décrivons, à présent, les différentes étapes qui mènent à une représentation sous forme clausale.

#### 1.6.3.1 Forme prénexe

**Définition 1.45** (Forme prénexe)**.** Une formule $\varphi$ est sous **forme prénexe** lorsqu'elle a la forme :

$$Q_1x_1Q_2x_2\dots Q_nx_n\psi$$

où chaque $Q_i$ est un quantificateur (existentiel ou universel) et $\psi$ est sans quantificateurs. la partie $Q_1x_1Q_2x_2\dots Q_nx_n$ est appelée le **préfixe** et $\psi$ étant qualifiée de **matrice**.

Mettre une formule sous forme prénexe consiste donc à renvoyer tous les quantificateurs au début de la formule.

**Théorème 1.46** (Forme prénexe équivalente)**.** *Pour toute formule $\varphi \in \mathcal{F}_{po}$ il existe une formule équivalente $\psi \in \mathcal{F}_{po}$ en forme prénexe.*

L'algorithme de mise sous forme prénexe suit les étapes suivantes :

**Etape 1 :** Renommer les variables de façon à ce qu'aucune variable n'ait d'occurrence libre et liée et d'occurrences liées à des quantificateurs différents ;

**Etape 2 :** Appliquer tant que possible les substitutions suivantes : substitution du membre droit par le membre gauche pour toutes les lois de passage et de conversion des quantificateurs.

**Exemple 1.47.** Considérez la formule suivante :

$$\neg\exists x\forall y R(x,y) \wedge \forall x(\exists y R(x,y) \Rightarrow R(x,x))$$

La première étape, renommage des variables, donne la formule suivante :

$$\neg\exists z_0\forall z_1 R(z_0,z_1) \wedge \forall z_2(\exists z_3 R(z_2,z_3) \Rightarrow R(z_2,z_3))\,.$$

Nous appliquons ensuite la deuxième étape :

$$\begin{aligned}
&\neg\exists z_0\forall z_1 R(z_0,z_1) \wedge \forall z_2(\exists z_3 R(z_2,z_3) \Rightarrow R(z_2,z_2)) \\
&\qquad\rightsquigarrow\quad \forall z_0\neg\forall z_1 R(z_0,z_1) \wedge \forall z_2(\exists z_3 R(z_2,z_3) \Rightarrow R(z_2,z_2)) \\
&\qquad\rightsquigarrow\quad \underline{\forall z_0}\,\underbrace{\exists z_1\neg R(z_0,z_1)}_{\varphi} \wedge \underbrace{\forall z_2(\exists z_3 R(z_2,z_3) \Rightarrow R(z_2,z_2))}_{\psi} \\
&\qquad\rightsquigarrow\quad \forall z_0(\underline{\exists z_1}\,\underbrace{\neg R(z_0,z_1)}_{\varphi} \wedge \underbrace{\forall z_2(\exists z_3 R(z_2,z_3) \Rightarrow R(z_2,z_2))}_{\psi}) \\
&\qquad\rightsquigarrow\quad \forall z_0\exists z_1(\underbrace{\neg R(z_0,z_1)}_{\psi} \wedge \underline{\forall z_2}(\underbrace{\exists z_3 R(z_2,z_3) \Rightarrow R(z_2,z_2)}_{\varphi})) \\
&\qquad\rightsquigarrow\quad \forall z_0\exists z_1\forall z_2(\neg R(z_0,z_1) \wedge (\underline{\exists z_3}\,\underbrace{R(z_2,z_3)}_{\varphi} \Rightarrow \underbrace{R(z_2,z_2)}_{\psi})) \\
&\qquad\rightsquigarrow\quad \forall z_0\exists z_1\forall z_2(\underbrace{\neg R(z_0,z_1)}_{\psi} \wedge \underline{\forall z_3}(\underbrace{R(z_2,z_3) \Rightarrow R(z_2,z_2)}_{\varphi})) \\
&\qquad\rightsquigarrow\quad \forall z_0\exists z_1\forall z_2\forall z_3(\neg R(z_0,z_1) \wedge (R(z_2,z_3) \Rightarrow R(z_2,z_2)))\,.
\end{aligned}$$

La nécessite de renommer les variables peut se comprendre si on essaie d'appliquer les règles de passage sans un renommage préalable. Ainsi :

$$\begin{aligned}
&\neg\exists x\forall y R(x,y) \wedge \forall x(\exists y R(x,y) \Rightarrow R(x,x)) \\
&\qquad\rightsquigarrow\quad \forall x\exists y\neg R(x,y) \wedge \forall x(\exists y R(x,y) \Rightarrow R(x,x)) \\
&\qquad\rightsquigarrow\quad \forall x\exists y(\underbrace{\neg R(x,y)}_{\psi} \wedge \underline{\forall x}(\underbrace{\exists y R(x,y) \Rightarrow R(x,x)}_{\varphi}))
\end{aligned}$$

Ici on ne peut pas appliquer la loi de passage, donc on renomme :

$$\rightsquigarrow \quad \forall x \exists y (\underbrace{\neg R(x,y)}_{\psi} \wedge \underline{\forall t} (\underbrace{\exists y R(t,y) \Rightarrow R(t,t)}_{\varphi}))$$

On ne peut ainsi appliquer la loi de passage :

$$\rightsquigarrow \quad \forall x \exists y \forall t (\neg R(x,y) \wedge (\exists y R(t,y) \Rightarrow R(t,t))) \ldots$$

#### 1.6.3.2 Forme de Skolem

**Définition 1.48.** Une formule est sous **forme de Skolem** lorsqu'elle est sous forme prenexe et qu'elle ne contient que des quantifications universelles.

**Exemple 1.49.** La formule $\forall x R(x, f(x))$ est sous forme de Skolem. La formule $\forall x \exists y R(x,y)$ est en forme prénexe, mais elle n'est pas sous forme de Skolem, car on trouve un quantificateur existentiel dans le préfixe.

Pour effectuer une *skolémisation*, on part donc d'une formule sous forme prenexe et on « supprime » les quantificateurs existentiels, en appliquant de façon itérée la règle suivante.

**Règle de Skolémisation.** Elle est de la forme

$$\frac{\exists y \psi}{\psi_{\{y \to f(z_1,\ldots,z_m)\}}}$$

où

— $z_1, ..., z_m$ sont les variables libres de la formule $\exists y.\psi$, et
— $f$ un *nouveau* symbole de fonction (dit de Skolem) d'arité $m$.

*Rappel* (cf. Définition 1.13). Une formule est *close* si elle ne contient pas de variables libres.

**Exemple 1.50.** Considérons une formule close $\varphi = \exists x \psi$ écrite sur un langage $\mathcal{S}$. La règle de skolémisation donne la formule $\psi_{\{x \to c\}}$, où $c$ est un nouveau symbole de constante.

Une $\mathcal{S}$-structure $\mathcal{M}$ est un modèle de $\varphi$ ssi il existe $a \in D_{\mathcal{M}}$ tel que $[\![ \psi ]\!]_{\mathcal{M},\mathcal{V}}$ où $\mathcal{V}$ est telle que $\mathcal{V}(x) = a$. Considérons maintenant le langage $\mathcal{S}' := \mathcal{S} \cup \{ (c, 0) \}$ (où $c$ est le nouveau symbole de constante) et la $\mathcal{S}'$-structure $\mathcal{M}'$ obtenue depuis $\mathcal{M}$ en interprétant la constante $c$ par $c^{\mathcal{M}'} := a$. Clairement, $\mathcal{M}'$ est un modèle de la formule $\psi_{\{x \to c\}}$.

Nous avons donc argumenté que si $\varphi$ a un modèle, alors $\psi_{\{x \to c\}}$ a un modèle ; l'implication inverse est d'ailleurs aussi vraie. Donc $\varphi$ est satisfaisable ssi $\psi_{\{x \to c\}}$ est satisfaisable.

**Exemple 1.51.** Considérons maintenant la formule $\varphi = \forall y \exists x \psi$ écrite sur un langage $\mathcal{S}$. Une $\mathcal{S}$-structure $\mathcal{M}$ est un modèle de $\varphi$ ssi pour chaque $b \in D_{\mathcal{M}}$ il existe $a \in D_{\mathcal{M}}$ tel que $[\![ \psi ]\!]_{\mathcal{M},[y:=b,x:=a]}$ ssi il existe une fonction $f : D_{\mathcal{M}} \to D_{\mathcal{M}}$ telle que, pour tout $b \in D_{\mathcal{M}}$, $[\![ \psi ]\!]_{\mathcal{M},[y:=b,x:=f(b)]}$.

Donc $\varphi$ admet un modèle ssi $\forall y \psi_{\{x \to f(y)\}}$ a un modèle. Remarquez que la formule $\forall y \psi_{\{x \to f(y)\}}$ est construite à partir du langage $\mathcal{S} \cup \{ (f, 1) \}$.
C'est ce principe qui va être généralisé pour mettre une formule sous forme de Skolem.

**Définition 1.52.** Une formule universelle (c'est-à-dire une formule close contenant seulement des quantificateurs existentiels), obtenue par
— mise en forme prenexe, et ensuite
— application itérée de la règle de skolémisation
est appelée **forme de skolem** ou **skolémisée** de $\varphi$.

**Exemple 1.53.**

1. Soit $\varphi = \forall x \exists y P(x,y) \Rightarrow \forall x \exists y P(y,x)$.
   La mise sous forme prénexe donne la formule équivalente $\varphi' = \exists x \forall y \forall x' \exists y' ((P(x,y) \Rightarrow P(y',x'))$.

La skolémisation donne :

$$\frac{\exists x \forall y \forall x' \exists y' ((P(x,y) \Rightarrow P(y',x'))}{\frac{\forall y \forall x' \exists y' ((P(c,y) \Rightarrow P(y',x'))}{\forall x \forall x' ((P(c,y) \Rightarrow P(f(y,x'),x'))}}$$

2. Considérons la formule $\exists x_1 \forall x_2 \forall x_3 \exists x_4 \forall x_5 \exists x_6 P(x_1, x_2, x_3, x_4, x_5, x_6)$. On obtient sa formule de Skolem de la façon suivante :

$$\frac{\frac{\frac{\exists x_1 \forall x_2 \forall x_3 \exists x_4 \forall x_5 \exists x_6 P(x_1, x_2, x_3, x_4, x_5, x_6)}{\forall x_2 \forall x_3 \exists x_4 \forall x_5 \exists x_6 P(f_1, x_2, x_3, x_4, x_5, x_6)}}{\forall x_2 \forall x_3 \forall x_5 \exists x_6 P(f_1, x_2, x_3, f_4(x_2, x_3), x_5, x_6)}}{\forall x_2 \forall x_3 \forall x_5 P(f_1, x_2, x_3, f_4(x_2, x_3), x_5, f_6(x_2, x_3, x_5))}$$

Dans la formule précédente, $\exists x_1$ n'est précédé par aucun quantificateur universel. C'est pourquoi on introduit une nouvelle constante $f_1$.

**Remarque 1.54.** La version skolémisée d'une formule *ne lui est* ***pas***, en général, *équivalente*. Le langage étant étendu, donc différent, les modèles des deux formules sont des structures pour des langages différents. Néanmoins :

- tout modèle de la formule skolémisée est modèle de la formule initiale ;
- tout modèle de la formule initiale peut s'étendre en un modèle de la formule skolémisée, obtenu en conservant les interprétations des symboles de la signature initiale, et en interprétant correctement les nouveaux symboles de fonction introduits pas la skolemisation ;
- une formule close et sa forme de Skolem sont dites *équisatisfaisables* : si l'une possède un modèle, l'autre également et réciproquement.

**Proposition 1.55.** *Si $\varphi_s$ est obtenue par skolémisation à partir de $\varphi$ alors $\varphi_s$ est satisfaisable si et seulement si $\varphi$ est satisfaisable.*

#### 1.6.3.3 Forme clausale

**Définition 1.56** (Forme clausale)**.** Une formule close est sous **forme clausale** si elle est

1. en forme prenexe,
2. elle est universelle (tous ses quantificateurs sont universels), et
3. sa matrice est sous forme normale conjonctive.

En utilisant la mise en forme prénexe, puis la Skolemisation, puis la mise en forme clausale du calcul propositionnel, toute formule peut se transformer dans une formule en forme clausale equisatisfiable.

**Définition 1.57.**

- Un **littéral** est une formule atomique ou la négation d'une formule atomique.
- Soit $\psi$ une formule avec $\mathsf{FV}(\psi) = \{ x_1, \ldots, x_n \}$. Sa **fermeture universelle** est la formule

$$\forall x_1 \ldots \forall x_n \psi \,.$$

- Une **clause universelle** ou **clause de premier ordre** est la fermeture universelle d'une disjonction de littéraux.

Lorsqu'une formule est sous forme clausale, on peut ensuite la décomposer en une conjonction de clauses (du premier ordre) en appliquant la règle $\forall x(\varphi \wedge \psi) \equiv (\forall x \varphi) \wedge (\forall x \psi)$. On obtient ainsi un ensemble de clauses (de premier ordre) equisatisfiable à la formule donnée.

**Théorème 1.58** (Satisfiabilité d'un ensemble de clauses)**.** *Soit $S$ un ensemble de clauses résultant de la mise sous forme clausale d'une formule $\varphi$. Alors $\varphi$ est satisfiable si et seulement si $S$ est satisfiable.*

Ce théorème forme la base de nombreux démonstrateurs automatiques utilisant une représentation des formules sous forme de clauses. Il établit que la recherche de l'insatisfiabilité d'une formule $\varphi$ est équivalente à la recherche d'insatisfiabilité de sa représentation sous forme clausale $S$. Cependant $\varphi$ et $S$ ne sont pas logiquement équivalentes : seule la satisfiabilité est préservée.

**Exemple 1.59.** Soit

$$\varphi_0 := \forall x \exists y((P(x,y) \Rightarrow \forall x \exists y P(y,x)) \wedge Q(x))\,.$$

La mise sous forme prénexe donne la formule

$$\varphi_1 := \exists x \forall y \forall z \exists w((P(x,y) \Rightarrow P(w,z)) \wedge Q(x))\,.$$

Par skolémisation, nous obtenons

$$\varphi_2 := \forall y \forall z((P(c,y) \Rightarrow P(f(y,z),z)) \wedge Q(c))\,.$$

Nous pouvons ensuite mettre la matrice sous forme normale conjonctive :

$$\varphi_3 := \forall y \forall z((\neg P(c,y) \vee P(f(y,z),z)) \wedge Q(c))\,,$$

et obtenir ainsi l'ensemble de clauses :

$$S := \{\ \forall y \forall z(\neg P(c,y) \vee P(f(y,z),z))\,,\ \forall y \forall z Q(c)\ \}\,.$$

Cet ensemble est equisatisfiable avec la formule $\varphi_0$.

## 1.7 Unification

### 1.7.1 Substitions et MGUs

Dans la suite, soient $\mathcal{S}_\mathtt{f}$ une signature composée de symboles de fonctions et $X$ un ensemble de variables. Nous nous intéresserons pas, dans cette section, aux symboles de relation.

**Définition 1.60.** Une **substitution** est une fonction $\sigma : X \to \mathcal{T}_{\mathcal{S}_\mathtt{f}}(X)$ telle que l'ensemble $\{\, x \in X \mid \sigma(x) \neq x \,\}$ est fini.

**Exemple 1.61.** Supposons $\mathcal{S}_\mathtt{r} = \{\,(f,2),(g,1)\,\}$ et $X = \{\, x_0, \ldots, x_n, \ldots \}$. La fonction $\sigma$ telle que $\sigma(x_0) = f(g(x_0), x_1)$, $\sigma(x_1) = x_2$, et $\sigma(x_i) = x_i$ pour $i \geq 2$, est une substitution.

On représente (souvent, dans des implémentation sur ordinateur) et on écrit les substitutions par des listes de couples clef,valeur (en jargon informatique, c'est des *listes associatives*), notées d'habitude par :

$$\{\, x_1 \to t_1, \ldots, x_n \to t_n \,\}\,, \quad \text{ou} \qquad [\, t_1/x_1, \ldots, t_n/x_n \,]\,.$$

La convention est la suivante : étant donnée une telle liste on construit la substitution $\sigma$ en posant $\sigma(x_i) = t_i$ ; si une variable $x$ n'apparaît pas dans la liste, alors elle est fixée par $\sigma$, c'est-à-dire $\sigma(x) = x$. Ainsi, la substitution de l'exemple 1.61, sera notée par

$$[f(g(x_0), x_1)/x_0, x_2/x_1]\,.$$

**Exercice 1.62.** Argumentez que toute substitution peut être représentée par une liste associative. Montrez que cette représentation n'est pas unique, c'est-à-dire qu'ils existent plusieurs (même une infinité) de listes représentant la même substitution.

**Définition 1.63.** Pour tout terme $t$, on définit l'action de $\sigma$ sur $t$ comme suit :

$$x\,\sigma = \sigma(x)\,,$$
$$f(t_1, \ldots, t_n)\,\sigma = f(t_1\sigma, \ldots, t_n\sigma)\,.$$

**Exemple 1.64.** On a

$$f(g(x), y)[z/x, g(y)/y] = f(g(z), g(y)), \quad g(f(x, f(y, x)))[g(w)/x] = g(f(g(w), f(y, g(w)))).$$

**Définition 1.65.** La substitution **identité** (ou vide) est celle qui fixe toutes les variables. Elle est donc notée par []. La **composition** de deux substitutions $\sigma$ et $\tau$, notée $\tau \circ \sigma$, est la substitution définie par :

$$(\tau \circ \sigma)(x) = (x\,\sigma)\,\tau\,. \tag{1.2}$$

Observez que, dans la définition ci-dessus, l'expression à la droite est $(\sigma(x))\,\tau$.

**Exemple 1.66.** Soient

$$\sigma = [\,f(x, y)/x\,], \qquad \tau = [\,g(y)/x,\ f(x, z)/y\,].$$

Calculons $\tau \circ \sigma$ :

| $\sigma$ | | | $\rho$ | |
|---|---|---|---|---|
| $x$ | $\mapsto$ | $f(x, y)$ | $\mapsto$ | $f(g(y), f(x, y))$ |
| $y$ | $\mapsto$ | $y$ | $\mapsto$ | $f(x, z)$ |
| $z$ | $\mapsto$ | $z$ | $\mapsto$ | $z$ |
| | | $\vdots$ | | |

On a donc

$$\tau \circ \sigma = [\,f(g(y), f(x, y))/x,\ f(x, z)/y\,].$$

**Exercice 1.67.** En généralisant l'exemple 1.66, proposez un algorithme qui calcule la composition de deux substitutions $\sigma$ et $\tau$ passées en paramètre. Les substitutions, en entrée et en sortie, seront représentées par des listes associatives.

**Exercice 1.68.** Prouvez les relations suivantes :

$$\begin{aligned} t\,[] &= t\,, \\ t\,(\tau \circ \sigma) &= (t\,\sigma)\,\tau\,. \end{aligned} \tag{1.3}$$

Prouvez ensuite que la composition de substitutions est associative, et que la substitution identité est son un élément neutre.

**Remarque 1.69.** La composition $\tau \circ \sigma$ de deux substitution $\sigma, \tau$ revient à une sorte de composition fonctionnelle—car on applique d'abord $\sigma$ et puis $\tau$. Cela justifie de dénoter cette composition par le symbole usuel (le symbole $\circ$) de la composition de fonctions.

Par ailleurs, il est costume en logique (et il s'avère éclaircissant) d'écrire la substitution à la droite du terme dans l'application d'un substitution à un terme, D'ici les relations (1.2) et (1.3) qui, en renversant gauche et droite, pourraient apparaître à première vue un peu bizarres.

**Définition 1.70.** Soient $\sigma$ et $\tau$ deux substitutions. On dit que $\sigma$ est **plus générale** que $\tau$ (et on écrit $\sigma \leq \tau$), s'il existe une substitution $\rho$ telle que $\tau = \rho \circ \sigma$.

**Exemple 1.71.** Soit

$$\sigma = [f(w, x)/x, z/y], \qquad \tau = [f(g(y), x)/x, c/y, c/z, g(y)/w].$$

On a alors $\sigma \leq \tau$, à cause de

$$\rho = [c/z, g(y)/w].$$

En fait, le calcul de la composition donne :

| $\sigma$ | | | $\rho$ | |
|---|---|---|---|---|
| $x$ | $\mapsto$ | $f(w,x)$ | $\mapsto$ | $f(g(y),x)$ |
| $y$ | $\mapsto$ | $z$ | $\mapsto$ | $c$ |
| $z$ | $\mapsto$ | $z$ | $\mapsto$ | $c$ |
| $w$ | $\mapsto$ | $w$ | $\mapsto$ | $g(y)$ |

**Définition 1.72.** Un **problème d'unification** est une liste $(s_1,t_1),\ldots,(s_n,t_n)$ avec $s_i,t_i \in \mathcal{T}_{\mathcal{S}_t}(X)$. Une solution de ce problème—appelé **unificateur** de $(s_1,t_1),\ldots,(s_n,t_n)$—est une substitution $\sigma$ telle que $s_i\sigma = t_i\sigma$, pour $i = 1,\ldots,n$. On notera $\mathsf{Unif}[\,(s_1,t_1),\ldots,(s_n,t_n)\,]$ l'ensemble des unificateurs $(s_1,t_1),\ldots,(s_n,t_n)$.

**Exemple 1.73.**

1. La substitution
$$\sigma = [g(z)/x, g(z)/y]\,.$$
est un unificateur de $(f(x,g(z)), f(g(z),y))$, car
$$f(x,g(z))[g(z)/x,g(z)/y] = f(g(z),g(z)) = f(g(z),y)[g(z)/x,g(z)/y]\,.$$
2. Nous avons $\mathsf{Unif}[\,(f(x,y),g(z))\,] = \varnothing$. De même, $\mathsf{Unif}[\,(x,g(x))\,] = \varnothing$.

Le but de cette section est de montrer le résultat suivant :

**Proposition 1.74.** *Si* $\mathsf{Unif}[\,(s_1,t_1),\ldots,(s_n,t_n)\,] \neq \varnothing$*, alors il existe* $\sigma \in \mathsf{Unif}[\,(s_1,t_1),\ldots,(s_n,t_n)\,]$ *tel que* $\sigma \leq \tau$ *pour tout* $\tau \in \mathsf{Unif}[\,(s_1,t_1),\ldots,(s_n,t_n)\,]$.

On appelle un tel $\sigma$ un **unificateur plus général** des couples $(s_1,t_1),\ldots,(s_n,t_n)$. On dira aussi que $\sigma$ un MGU des couples $(s_1,t_1),\ldots,(s_n,t_n)$, où MGU est un acronyme de l'anglais « Most General Unifier ».

**Exemple 1.75.** $\tau = [g(f(w))/x, g(f(w))/y] \in \mathsf{Unif}[\,(f(x,g(z)),f(g(z),y))\,]$, mais $\tau$ n'est pas un MGU de ce problème. En fait, $\sigma = [g(z)/x, g(z)/y]$ est un MGU, et on a $\sigma \leq \tau$, car $\tau = \rho \circ \sigma$, avec $\rho = [f(w)/z]$.

## 1.7.2 Algorithme d'unification

L'algorithme d'unification est illustré en figure 1.7.2. Nous donnons dans la suite des exemples de calcul de cet algorithme sur des problèmes d'unification.

**Exemple 1.76.** Considérez le problème suivant :
$$(f(x,g(z)), f(g(z),x))\,,\ (x,g(z))\,.$$
L'algorithme marche de la facon suivante :

| Ligne appel récursif (ou return) | Entrée | Pile des résultats partiels |
|---|---|---|
| | $(\,f(x,g(z)), f(g(z),x)),\ (x,g(z)\,)$ | |
| 6 | $(x,g(z)),\ (g(z),x),\ (x,g(z))$ | |
| 12 | $(g(z),g(z)),\ (g(z),g(z))$ | $[g(z)/x]$ |
| 6 | $(z,z),\ (g(z),g(z))$ | $[g(z)/x]$ |
| 9 | $(g(z),g(z))$ | $[g(z)/x]$ |
| 6 | $(z,z)$ | $[g(z)/x]$ |
| 9 | | $[g(z)/x]$ |
| 1 | | $[] \circ [g(z)/x]$ |

**Exercice 1.77.** Exercez vous maintenant avec les problèmes suivants :

1. $(f(g(k(x)),y), f(y,g(x)))$,
2. $(f(g(x),x), f(y,g(z))), (g(x),y))$,
3. $(f(y,k(y),g(x)), f(k(x),k(y),y))$.

UNIFIER

Entrée : un problème d'unification $(s_1, t_1), \ldots, (s_n, t_n)$

Sortie :

un MGU de $(s_1, t_1), \ldots, (s_n, t_n)$ si $\mathsf{Unif}[\,(s_1, t_1), \ldots, (s_n, t_n)\,] \neq \emptyset$

ECHEC, sinon

```
1  Si n = 0, retourner la substitution identité
2  Sinon, on analyse le couple (s_1, t_1) (la tête de la liste) :
3    si s_1 = f(r_1, ..., r_k) et t_1 = g(r'_1, ..., r'_k') alors
4      si f ≠ g, retourner ECHEC
5      sinon          /* f = g implique k = k' */
6        retourner UNIFIER( (r_1, r'_1), ..., (r_k, r'_k), (s_2, t_2), ..., (s_n, t_n) )
7    si s_1 est la variable x, alors :
8        si t_1 est aussi la variable x,
9          retourner UNIFIER( (s_2, t_2), ..., (s_n, t_n) )
10     si x ∈ VAR(t_1), retourner ECHEC
11     sinon,
12       soit τ le résultat de UNIFIER( (s_2[t_1/x], t_2[t_1/x]), ..., (s_n[t_1/x], t_n[t_1/x]) )
13         si τ = ECHEC, retourner ECHEC
14         sinon retourner τ ∘ [t_1/x]
15   si t_1 est la variable x, alors
16     traitement comme auparavant, avec s_1 à la place de t_1
```

FIGURE 1.2 – Algorithme d'unification

**Terminaison.** Définissons la complexité d'un terme comme suit :

$$\sharp(x) = 1\,,$$
$$\sharp(f(t_1,\ldots,t_n) = 1 + \sum_{i=1,\ldots,n} \sharp t_i\,.$$

La complexité d'un problème (que nous noterons par le même symbole $\sharp$) est un couple de nombres entiers (non-négatifs) qui se définit comme suit :

$$\sharp((s_1,t_1),\ldots,(s_n,t_n)) = (\,\mathrm{card}(\bigcup_{i=1,\ldots,n} Var(s_i)\cup Var(t_i))\,,\ \sum_{i=1,\ldots,n} \sharp(s_i)+\sharp(t_i)\,)\,.$$

Le lecteur notera qu'à chaque appel récursif, le problème en paramètre a complexité strictement plus petite par rapport à l'ordre lexicographique sur $\mathbb{N}\times\mathbb{N}$. Donc L'algorithme ne peut pas faire une suite infinie d'appels récursifs, et il termine.

### 1.7.3 Correction et completude

Nous souhaitons prouver les propositions suivantes.

**Proposition 1.78** (Correction)**.** *Soit $\pi$ un problème d'unification. Si* UNIFIER($\pi$) *retourne* ECHEC*, alors* Unif$[\,\pi\,] = \varnothing$ *; si* UNIFIER($\pi$) *retourne une substitution $\sigma$, alors $\sigma \in$* Unif$[\,\pi\,]$ *et, de plus, $\sigma$ est un* MGU *de $\pi$.*

**Proposition 1.79** (Complétude)**.** *Soit $\pi$ un problème d'unification. Si* Unif$[\,\pi\,] = \varnothing$*, alors* UNIFIER($\pi$) *retourne* ECHEC *; si* Unif$[\,\pi\,] \neq \varnothing$*, alors* UNIFIER($\pi$) *retourne un* MGU *de $\pi$.*

Afin de prouver ces deux propositions, introduisons quelques notations qui sera utile, ainsi que la Proposition 1.80, qui est le résultat nécessaire le moins évidant à demontrer.

— $\Delta = \{\,(t,t) \mid t \in \mathcal{T}_{\mathcal{S}_t}(X)\,\}$ et $\Delta^n = \underbrace{\Delta\times\ldots\times\Delta}_{n-\text{fois}}$,

— si $\pi = (s_1,t_1),\ldots,(s_n,t_n)$, alors $\ell(\pi) = n$ et $\pi_\sigma = (s_1\sigma,t_1\sigma),\ldots,(s_n\sigma,t_n\sigma)$.

Avec cette notation remarquez que

$$\sigma \in \mathsf{Unif}[\,\pi\,] \text{ ssi } \pi_\sigma \in \Delta^{\ell(\pi)}\,.$$

**Proposition 1.80.** *Soient $\pi$ et $\psi$ deux problemes d'unification, soit $\sigma$ un* MGU *de $\pi$. Alors*

$$\mathsf{Unif}[\,\pi,\psi\,] = \{\,\rho\circ\sigma \mid \rho \in \mathsf{Unif}[\,\psi_\sigma\,]\,\}\,. \tag{1.4}$$

*Par conséquent, si $\rho$ est un* MGU *de $\psi_\sigma$, alors $\tau = \rho\circ\sigma$ un* MGU *de $\pi,\psi$.*

*Démonstration.* Observons que si $\tau \in \mathsf{Unif}[\,\pi,\psi\,]$ alors $\tau \in \mathsf{Unif}[\,\pi\,]$ et, car $\sigma$ est un MGU de $\pi$, $\sigma \leq \tau$, c'est-à-dire $\tau = \rho\circ\sigma$ pour une substitution $\rho$. Car $\tau \in \mathsf{Unif}[\,\psi\,]$, on remarquera que

$$(\psi_\sigma)_\rho = \psi_{\rho\circ\sigma} = \psi_\tau \in \Delta^{\ell(\psi)}\,,$$

donc $\rho$ est un unificateur de $\psi_\sigma$ et que $\tau \in \{\,\rho\circ\sigma \mid \rho \in \mathsf{Unif}[\,\psi_\sigma\,]\,\}$.

Nous avons montré que l'ensemble sur la gauche de (1.4) est inclus dans celui de droite. Montrons donc l'autre inclusion. Si $\rho \in \mathsf{Unif}[\,\psi_\sigma\,]$, alors

$$\psi_{\rho\circ\sigma} = (\psi_\sigma)_\rho \in \Delta^{\ell(\psi_\sigma)} = \Delta^{\ell(\psi)}\,, \qquad \text{car } \rho \in \mathsf{Unif}[\,\psi_\sigma\,]$$
$$\pi_{\rho\circ\sigma} = (\pi_\sigma)_\rho \in (\Delta^{\ell(\pi)})_\rho \subseteq \Delta^{\ell(\pi)}\,, \qquad \text{car } \sigma \in \mathsf{Unif}[\,\pi\,]$$

donc $\rho\circ\sigma \in \mathsf{Unif}[\,\pi,\psi\,]$. Cela complète la preuve de l'égalité (1.4).

Soient maintenant $\rho$ un MGU de $\psi_\sigma$ et soit $\tau \in \mathsf{Unif}[\,\pi,\psi\,]$. A cause l'égalité (1.4), nous pouvons écrire $\tau = \rho'\circ\sigma$ avec $\rho' \in \mathsf{Unif}[\,\psi_\sigma\,]$ ; on a alors $\rho \leq \rho'$, donc $\rho' = \theta\circ\rho$ pour une substitution $\theta$ et, par conséquent, $\tau = \theta\circ\rho\circ\sigma$, ce qui montre que $\rho\circ\sigma \leq \tau$. Nous avons donc montré que $\rho\circ\sigma$ est un MGU du problème $\pi,\psi$. □

La preuve de correction et complétude de l'algorithme d'unification repose sur les lemmes suivants :

**Lemme 1.81.** *Les faits suivants sont vrais :*

1. *Si* $f \neq g$, *alors* $\mathsf{Unif}[\,(f(r_1,\ldots,r_k), g(r'_1,\ldots,r'_{k'}))\,] = \varnothing$.
2. *Si* $x \in Var(t)$ *et* $t \neq x$, *alors* $\mathsf{Unif}[\,(x,t)\,] = \varnothing$.
3. $\mathsf{Unif}[\,\pi,\psi\,] = \mathsf{Unif}[\,\psi,\pi\,] \subseteq \mathsf{Unif}[\,\psi\,]$.
   *En particulier, si* $\mathsf{Unif}[\,(s,t)\,] = \varnothing$, *alors* $\mathsf{Unif}[\,(s,t),(s_2,t_2),\ldots,(s_n,t_n)\,] = \varnothing$.

**Lemme 1.82.** *On a*

$$\begin{aligned}\mathsf{Unif}[\,(f(r_1,\ldots,r_k), f(r'_1,\ldots,r'_k)),(s_2,t_2),\ldots,(s_n,t_n)\,]\\ = \mathsf{Unif}[\,(r_1,r'_1),\ldots,(r_k,r'_k),(s_2,t_2),\ldots,(s_n,t_n)\,]\,.\end{aligned}$$

**Lemme 1.83.** *Un* MGU *de*

$$(x,x),(s_2,t_2),\ldots,(s_n,t_n)$$

*est* $\rho$, *où* $\rho$ *est un* MGU *de*

$$(s_2,t_2),\ldots,(s_n,t_n)\,.$$

*Si ce dernier problème ne possède pas de solutions, alors il en est de même pour* $(x,x),(s_2,t_2),\ldots,(s_n,t_n)$.

**Lemme 1.84.** *Supposons* $x \notin Var(t)$. *Un* MGU *de*

$$(x,t),(s_2,t_2),\ldots,(s_n,t_n)$$

*est* $\rho \circ [t/x]$, *où* $\rho$ *est un* MGU *de*

$$(s_2,t_2)[t/x],\ldots,(s_n,t_n)[t/x]\,.$$

*Si ce dernier problème ne possède pas de solutions, alors il en est de même pour* $(x,t),(s_2,t_2),\ldots,(s_n,t_n)$.

Les Lemmes 1.83 et 1.84 sont un conséquence de la Proposition 1.80, en raison du fait que
— la substitution identité [] est un MGU du problème $(x,x)$ ;
— $[t/x]$ est évidemment un MGU du problème $(x,t)$ quand $x \notin Var(t)$.

**Exercice 1.85.** A l'aide des Lemmes 1.81-1.84 complétez une preuve formelle de correction et complétude de l'algorithme d'unification.

*Exemple : démonstration de la Proposition 1.78, correction de l'algorithme.* L'algorithme retourne échec, sans appels recursifs, à cause des lignes 4,10 et 13. Pour les lignes 4 et 10, on utilise le Lemme 1.81. Pour la ligne 13, on utilise le Lemme 1.84.

Les appels recursifs se trouvent aux lignes 6 et 9. On justifie la ligne 6 par le Lemme 1.82, et la ligne 9 par le e Lemme 1.83.

Nous argumentons ainsi que si UNIFIER($\pi$) retourne ECHEC, alors $\mathsf{Unif}[\,(\,]\pi) = \varnothing$.

La preuve que si UNIFIER($\pi$) retourne $\sigma$, alors $\sigma$ est un MGU de $\pi$ est similaire. □

## 1.8 Résolution

### 1.8.1 Substitution, sur les formules propostionnelles

L'action d'une substitution s'étend aisément aux formules sans quantificateurs :
— $R(t_1, \ldots, t_n)\sigma = R(t_1\sigma, \ldots, t_n\sigma)$,
— $(\neg\varphi)\sigma = \neg(\varphi\sigma)$,
— $(\varphi \circ \psi)\sigma = \varphi\sigma \circ \psi\sigma$, $\circ \in \{ \vee, \wedge, \Rightarrow \}$.

*Rappel* (cf. Définition 1.57). Un **littéral** est ou bien une formule atomique, ou bien la négation d'une formule atomique. Une **clause universelle** est la fermeture universelle d'une disjonction de littéraux.

Désormais, clause sera un synonyme de clause universelle. Bien que une clause soit une formule de la forme

$$\forall x_1, \ldots, \forall x_n (l_1 \vee \ldots \vee l_k)$$

avec $l_i$ des littéraux et $\{ x_1, \ldots, x_n \} = \mathsf{FV}(\{ l_1, \ldots, l_n \})$, il est habituel de laisser l'écriture des quantificateurs implicite. Par exemple, nous allons considérer l'expression

$$R(x, y) \vee \neg Q(f(x), z)$$

comme un raccourci de sa fermeture universelle :

$$\forall x \forall y \forall z (R(x, y) \vee \neg Q(f(x), z)) .$$

Pour de raison de convenance, nous avons donc décidé d'écrire de la même façon une clause universelle et sa matrice (la sous-formule sans quantificateurs). Au cas nous aurions besoin de distinguer une clause universelle $C$ de sa matrice, nous allons écrire $C_{\mathsf{mat}}$ pour la matrice.

La substitution s'étend, en particulier, aux littéraux et aux clauses :

1. $R(t_1, \ldots, t_n)\sigma = R(t_1\sigma, \ldots, t_n\sigma)$,
2. $(\neg R(t_1, \ldots, t_n))\sigma = \neg(R(t_1, \ldots, t_n)\sigma)$,
3. $(\bigvee_{i=1}^n l_i)\sigma = \bigvee_{i=1}^n (l_i)\sigma$.

**Notation 1.86.** Si $C$ est une clause universelle et $\sigma$ une substitution, nous allons utiliser la notation $C\sigma$ pour la fermeture universelle de $C_{\mathsf{mat}}\sigma$. Évidemment, $C\sigma$ est aussi une clause universelle.

Un **unificateur** de deux litteraux $l_0$ et $l_1$ est une substitution $\sigma$ telle que $l_0\sigma = l_1\sigma$.

**Lemme 1.87.** *$\sigma$ est un unificateur de $l_0$ et $l_1$ ssi*

1. *$l_0 = R(s_1, \ldots, s_n)$, $l_1 = R(t_1, \ldots, t_n)$, et $\sigma \in \mathsf{Unif}[\, (s_1, t_1), \ldots, (s_n, t_n) \,]$, ou bien*
2. *$l_0 = \neg R(s_1, \ldots, s_n)$, $l_1 = \neg R(t_1, \ldots, t_n)$, et $\sigma \in \mathsf{Unif}[\, (s_1, t_1), \ldots, (s_n, t_n) \,]$.*

Du Lemme il en découle tout de suite que l'ensemble des unificateurs de deux littéraux, appelons encore une fois $\mathsf{Unif}[\, l_0, l_1 \,]$, ou bien il est vide, ou bien il possède une **substitution la plus générale**, qui sera appelé un $\mathsf{MGU}$ de $l_0$ et $l_1$.

### 1.8.2 Les règles du calcul de la résolution

On peut considérer le calcul de la résolution comme une généralisation de la méthode de la coupure propositionnelle. Comme dans le cas propositionnel, la méthode de résolution prend en paramètre un ensemble $\Gamma$ de clauses (universelles) et essaye de dériver la clause vide $\bot$ depuis les clauses dans $\Gamma$. Les deux règles pour dériver des nouvelles clauses à partir des clauses déjà construites sont les suivantes :

$$\frac{C \vee A_0 \qquad C' \vee \neg A_1}{(C \vee C')\sigma}\ \text{Résolution}$$

où $\sigma$ est un $\mathsf{MGU}$ des formules atomiques $A_0$ et $A_1$, et

$$\frac{C \vee l_0 \vee l_1}{(C \vee l_0)\sigma}\ \text{Factorisation}$$

où $\sigma$ est un $\mathsf{MGU}$ des littéraux $l_0$ et $l_1$.

**Exemple 1.88** (Règle de Résolution)**.** La suivante est une instance de la règle de résolution :

$$\frac{\neg HabiteCL(x) \vee Tue(x, Agate) \qquad \neg Tue(Charles, y) \vee Hait(x, y)}{\neg HabiteCL(Charles) \vee Hait(Charles, Agate)}$$

Ici $l_0 = Tue(x, Agate)$ et $l_1 = Tue(Charles, y)$, $C = HabiteCL(x)$, $C' = Hait(x, y)$. En fait, $[Charles/x, Agate/y]$ est un MGU de $Tue(x, Agate)$ et $Tue(Charles, y)$.

**Exemple 1.89** (Règle de Factorisation)**.** La suivante est une instance de la règle de factorisation :

$$\frac{\neg HabiteCL(x) \vee Hait(x, y) \vee Tue(x, Agate) \vee Tue(Charles, y)}{\neg HabiteCL(Charles) \vee Hait(Charles, Agate) \vee Tue(Charles, Agate)}$$

Le MGU est encore une fois $[Charles/x, Agate/y]$.

### 1.8.3 Correction du calcul de la résolution

La méthode de résolution est correcte, c'est-à-dire, la conclusion d'une règle est conséquence logique des prémisses de la règle. Explicitement, chaque fois que $\mathcal{M} \models C_i$, où $C_i$, $i = 1, \ldots, n$ avec $n \in \{1, 2\}$ sont les prémisses d'une règle, alors on a $\mathcal{M} \models C_0$, où $C_0$ est la conclusion de la règle.

En fait, on peut penser que les règles du calcul sont obtenues comme synthèse de deux règles, l'une qui porte sur les subsitutions (et les quantificateurs universels), et l'autre étant la règle correspondante propositionnelle :

$$\frac{\dfrac{C \vee A_0}{C\sigma \vee A_0\sigma}\,\sigma \qquad \dfrac{C' \vee \neg A_1}{C'\sigma \vee \neg A_0\sigma}\,\sigma}{C\sigma \vee C'\sigma}\ \text{Résolution propositionnelle}$$

où on a $A_0\sigma = A_1\sigma$. De façon semblable :

$$\frac{\dfrac{C \vee l_0 \vee l_1}{C\sigma \vee l_0\sigma \vee l_0\sigma}\,\sigma}{C\sigma \vee l_0\sigma}\ \text{Factorisation propositionnelle}$$

Nous allons, dans la suite, justifier ces règles "plus élémentaires".

**Lemme 1.90.** *Soient $\varphi$ une formule sans quantificateurs, $\sigma$ une substitution, $\mathcal{M}$ une $\mathcal{S}$-structure et $\mathcal{V}$ une valuation. On a que*

$$\mathcal{M}, \mathcal{V} \models \varphi\sigma \text{ ssi } \mathcal{M}, \mathcal{V}\sigma \models \varphi\,, \tag{1.5}$$

*où $\mathcal{V}\sigma$ est la valuation définie par la règle suivante :*

$$(\mathcal{V}\sigma)(x) = [\![\, \sigma(x) \,]\!]_{\mathcal{M},\mathcal{V}}\ .$$

Notez que si $\sigma = [t_1/x_1, \ldots, t_n/x_n]$, alors $\mathcal{V}\sigma = \mathcal{V}[x_1 := [\![\, t_1 \,]\!]_{\mathcal{M},\mathcal{V}}, \ldots, x_n := [\![\, t_n \,]\!]_{\mathcal{M},\mathcal{V}}]$ est la variante de $\mathcal{V}$ satisfaisant aux lois suivantes :

$$\mathcal{V}[x_1 := [\![\, t_1 \,]\!]_{\mathcal{M},\mathcal{V}}, \ldots, x_n := [\![\, t_n \,]\!]_{\mathcal{M},\mathcal{V}}](y) = \begin{cases} [\![\, t_i \,]\!]_{\mathcal{M},\mathcal{V}}\,, & \text{si } y = x_i, \text{ pour quelques } i, \\ \mathcal{V}(y), & \text{sinon.} \end{cases}$$

*Démonstration du Lemme 1.90.* La preuve de cet énoncé se fait aisément par induction. Nous nous limiterons à illustrer le cas de base. Pour toute variable $y \in X$, nous avons

$$[\![\, y \,]\!]_{\mathcal{M},\mathcal{V}\sigma} = (\mathcal{V}\sigma)(y) = [\![\, \sigma(y) \,]\!]_{\mathcal{M},\mathcal{V}}\,.$$

Ainsi, nous avons

$$\begin{aligned} \mathcal{M}, \mathcal{V} \models R(y_1, \ldots, y_m)\sigma \text{ ssi } & \mathcal{M}, \mathcal{V} \models R(\sigma(y_1), \ldots, \sigma(y_m)) \\ & \text{ssi } ([\![\, \sigma(y_1) \,]\!]_{\mathcal{M},\mathcal{V}}, \ldots, [\![\, \sigma(y_m) \,]\!]_{\mathcal{M},\mathcal{V}}) \in R^{\mathcal{M}} \\ & \text{ssi } ([\![\, y_1 \,]\!]_{\mathcal{M},\mathcal{V}\sigma}, \ldots, [\![\, y_m \,]\!]_{\mathcal{M},\mathcal{V}\sigma}) \in R^{\mathcal{M}} \\ & \text{ssi } \mathcal{M}, \mathcal{V}\sigma \models R(y_1, \ldots, y_m). \end{aligned}$$

□

**Lemme 1.91.** *Pour toute clause universelle $C$ et toute substitution $\sigma$, la règle d'inférence suivante est correcte :*

$$\frac{C}{C\sigma}\,\sigma$$

*C'est-à-dire, si $\mathcal{M} \models C$, alors $\mathcal{M} \models C\sigma$, pour toute $\mathcal{S}$-structure $\mathcal{M}$.*

*Démonstration.* Supposons que $\mathcal{M} \models C$ ; pour montrer que $\mathcal{M} \models C\sigma$, nous devons montrer que $\mathcal{M}, \mathcal{V} \models (C\sigma)_{\mathsf{mat}}$, où $\mathcal{V}$ est une valuation arbitraire. En considération que $(C\sigma)_{\mathsf{mat}} = (C_{\mathsf{mat}})\sigma$ et par le Lemme 1.90 cela revient à vérifier que $\mathcal{M}, \mathcal{V}\sigma \models C_{\mathsf{mat}}$ ; cette dernière relation et en effet vraie à cause de l'assomption $\mathcal{M} \models C$. □

Pour les règle de factorisation et résolution propositionnelles, nous devons les justifier. En fait, ces règle manipulent des clauses universelles et non pas simplement des clauses. Par ailleurs, les démonstrations que ces règles propositionnelles s'étendent au cas des clauses universelles sont assez faciles.

**Lemme 1.92.** *Pour toute couple de clauses universelles de la forme $C_1 \vee l$ et $C_2 \vee \neg l$ (avec $l$ un littéral), la règle d'inférence suivante est correcte :*

$$\frac{C_1 \vee l \qquad C_2 \vee \neg l}{C_1 \vee C_2}\ \text{Résolution propositionnelle}$$

*C'est-à-dire, si $\mathcal{M} \models C_1 \vee l$ et $\mathcal{M} \models C_2 \vee \neg l$, alors $\mathcal{M} \models C_1 \vee C_2$, pour toute $\mathcal{S}$-structure $\mathcal{M}$.*

*Démonstration.* Nous devons montrer que $\mathcal{M}, \mathcal{V} \models (C_1 \vee C_2)_{\mathsf{mat}}$ pour toute valuation $\mathcal{V}$. Cela est une conséquence de $(C_1 \vee C_2)_{\mathsf{mat}} = (C_1)_{\mathsf{mat}} \vee (C_2)_{\mathsf{mat}}$, du fait que $\mathcal{M}, \mathcal{V} \models (C_1 \vee l)_{\mathsf{mat}} (= (C_1)_{\mathsf{mat}} \vee l)$, $\mathcal{M}, \mathcal{V} \models (C_2 \vee \neg l)_{\mathsf{mat}} (= (C_2)_{\mathsf{mat}} \vee \neg l)$, et du fait que la règle de la coupure propositionelle est correcte. □

**Exercice 1.93.** Montrez que la règle de factorisation propositionnelle s'étend aux clauses universelles.

### 1.8.4 Complètude du calcul de la résolution

Soit $\Gamma$ un ensemble de clauses universelles. Nous disons que $\Gamma$ est **saturé** si toute clause dérivable (via résolution et factorisation) de formules dans $\Gamma$ appartient déjà à $\Gamma$ ; nous disons que $\Gamma$ est **cohérent** si $\bot \notin \Gamma$.

**Théorème 1.94.** *Un ensemble saturé et cohérent de clauses admet au moins un modèle.*

En fait, nous allons montrer la proposition suivante :

**Proposition 1.95.** *Si un ensemble saturé de clauses $\Gamma$ n'admet pas un modèle, alors $\bot \in \Gamma$.*

*Démonstration.* Soit $\mathcal{S} = (\mathcal{S}_{\mathfrak{f}}, \mathcal{S}_{\mathfrak{r}})$ le langage ; nous allons nous intéresser au *langage propositionnel* caractérisé par le fait que Prop est l'ensemble de proposition atomiques du langage $\mathcal{S}$.

Remarquons d'abord que, étant donnée une valuation (au sens propositionnel) $v$ : Prop $\to \{0, 1\}$, nous pouvons définir une $\mathcal{S}$-structure $\mathcal{M}_v$ de la façon suivante :

- $D_{\mathcal{M}_v} := \mathcal{T}_{\mathcal{S}_{\mathfrak{f}}}(X)$ ;
- pour tout $f \in \mathcal{S}_{\mathfrak{f}}$, $f^{\mathcal{M}_v}$ est la fonction qui envoie un tuplet $(t_1, \ldots, t_n) \in \mathcal{T}_{\mathcal{S}_{\mathfrak{f}}}(X)$ vers le terme $f(t_1, \ldots, t_n)$ ;
- pour tout $R \in \mathcal{S}_{\mathfrak{r}}$, nous posons

$$R^{\mathcal{M}_v} := \{\, (t_1, \ldots, t_n) \in \mathcal{T}_{\mathcal{S}_{\mathfrak{f}}}(X)^n \mid v(\, R(t_1, \ldots, t_n)\,) = 1 \,\}\,.$$

La structure $\mathcal{M}_v$ ainsi définie a cette propriété importante :

**Lemme 1.96.** *Si $C$ est une clause universelle, alors*

$$\mathcal{M}_v \models C \text{ ssi } v(C\sigma) = 1, \text{ pour toute substitution } \sigma.$$

*Preuve du Lemme.* Une valuation $\mathcal{V}$ de l'ensemble de variables vers $D_{\mathcal{M}_v} = \mathcal{T}_{\mathcal{S}_t}(X)$ n'est rien d'autre qu'une substitution. En tenant compte que $C$ est implicitement quantifiée universellement, la condition $\mathcal{M}_v \models C$ est vraie quand $\mathcal{M}_v, \sigma \models C_{\mathsf{mat}}$, pour toute substitution $\sigma$. Rappelons que $[]$ est la substitution identité ; le Lemme 1.90 montre que $\mathcal{M}_v, \sigma \models C_{\mathsf{mat}}$ est équivalent à $\mathcal{M}_v, [] \models C_{\mathsf{mat}}\sigma$ ; cette condition revient à dire que $v(C\sigma) = 1$ (au sens propositionnel). □

Il en découle que l'ensemble de clauses propositionnelles

$$\Delta = \{\, C\sigma \mid C \in \Gamma, \sigma \text{ une substitution} \,\}$$

n'est pas satisfaisable au sens propositionnel. En fait, si $v$ est une valuation vérifiant toutes les formules de cet ensemble, alors $\mathcal{M}_v$ est un modèle satisfaisant toutes les formules de $\Gamma$.

Pour le Théorème de compacité, il existe un sous-ensemble fini $\Delta_f \subseteq \Delta$ tel que $\Delta_f$ n'est pas satisfaisable. Car la méthode de la coupure est complète, il existe une suite de clauses $D_1, \ldots, D_n$ avec $D_n = \bot$ (c'est-à-dire, $D_n$ est la clause vide), telle que, pour tout $i > 0$ :

1. $D_i \in \Delta_f$, ou
2. $D_i$ est déduite de $D_j$ et $D_k$ (avec $j, k < i$) via la règle de coupure, ou
3. $D_i$ est déduite de $D_j$ (avec $j < i$) via la règle de factorization (propositionnelle).

**Lemme 1.97.** *Pour tout $i = 1, \ldots, n$, ils existent $C_i \in \Gamma$ et une substitution $\rho_i$ telle que $D_i = C_i\rho_i$.*

*Preuve du Lemme.* Par induction (sur $i = 1, \ldots, n$), et par cas.

1. Si $D_i \in \Delta_f \subseteq \Delta$, alors cela est vrai par définition de $\Delta$ : $D_i = C \circ \sigma$ pour une clause $C \in \Gamma$ et une substitution $\sigma$ ; on peut donc poser $C_i := C$ et $\rho_i := \sigma$.
2. (Voir la Figure 1.3.) Supposons que $D_i$ est déduite de $D_j$ et $D_k$ (avec $j, k < i$) via la règle de coupure. Par hypothèse d'induction, ils existent $C_j, C_k \in \Gamma$ et deux substitutions $\rho_j, \rho_k$ tels que $D_j = C_j\rho_j$ et $D_k = C_k\sigma_k$.
   Supposons donc que $D_j = D \vee A$, $D_k = D' \vee \neg A$, et $D_i = D \vee D'$. On a alors $C_j = C \vee A_0$, $C_k = C' \vee \neg A_1$, $D = C\rho_j$, $D' = C'\rho_k$, et $A_0\rho_j = A = A_1\rho_k$. Sans perte de généralité, nous pouvons assumer qu'il n'y a pas des variables en commun entre $C_j$ et $C_k$, que $\rho_j$ fixe les variables de $C_k$, et $\rho_k$ fixe les variables de $C_j$. Par conséquent, si $\rho_j = [t_1/x_1, \ldots, t_n/x_n]$ et $\rho_k = [s_1/y_1, \ldots, s_m/y_m]$, alors $\tau = [t_1/x_1, \ldots, t_n/x_n, s_1/y_1, \ldots, s_m/y_m]$ est un unificateur de $A_0$ et $A_1$, $D_j = C_j\tau$ et $D_k = C_k\tau$. Soit $\sigma$ un $\mathsf{MGU}$ de $A_0$ et $A_1$, on a alors $\tau = \rho \circ \sigma$, et

   $$D_i = D \vee D' = C\tau \vee C'\tau = (C \vee C')\tau = [(C \vee C')\sigma]\rho\,.$$

   Nous pouvons alors poser $C_i := (C \vee C')\sigma$ et $\rho_i := \rho$. $C_i \in \Gamma$ car elle est déduite de $C_i$ et $C_j$ via la règle de résolution depuis $C_j, C_k \in \Gamma$, et en plus $\Gamma$ est saturé.
3. Supposons enfin que la clause $D_i$ est déduite de $D_j$ (avec $j < i$) via la règle de factorisation propositionnelle. Par hypothèse d'induction, il existe une clause $C_j \in \Gamma$ et une substitution $\rho_j$ telle que $D_j = C_j\rho_j$.
   Si $D_j = D \vee \ell \vee \ell$, alors $C_j = C \vee \ell_0 \vee \ell_1$ avec $D = C\rho_j$, et $\ell = \ell_0\rho_j = \ell_1\rho_j$. La substitution $\rho_j$ est donc un unificateur de $\ell_0$ et $\ell_1$ ; si $\sigma$ est un $\mathsf{MGU}$ de $\ell_0$ et $\ell_1$, alors il existe une substitution $\rho$ telle que $\rho_j = \rho \circ \sigma$.
   Nous pouvons donc poser $C_i := (C \vee \ell_0)\sigma$ et $\rho_i := \rho$ ; $C_i \in \Gamma$ car il a été déduit depuis $C_j \in \Gamma$ via la règle de factorisation (du premier ordre) et $\Gamma$ saturé ; par ailleurs

   $$C_i\rho_i = [(C \vee \ell_0)\sigma]\rho_i = (C \vee \ell_0)(\rho_i \circ \sigma) = (C \vee \ell_0)\rho_j = C\rho_j \vee \ell_0\rho_j = D \vee \ell = D_j. \qquad \square$$

En particulier, le Lemme dit que $C_n \in \Gamma$ et que $C_n\rho_n = \bot$. Il est facile à voir que si $C_n \neq \bot$, alors $C_n\rho \neq \bot$ (pour n'importe quel $\rho$) ; par conséquent, nous avons $C_n = \bot \in \Gamma$. Cela complète la démonstration de la Proposition 1.95. □

Une analyse fine de la preuve de la Proposition 1.95 amène à une preuve du théorème suivant.

**Théorème 1.98.** *Si un ensemble de clauses $\Gamma$ n'admet pas un modèle, alors il existe une preuve de $\bot$ à partir de $\Gamma$ dans le calcul de la résolution.*

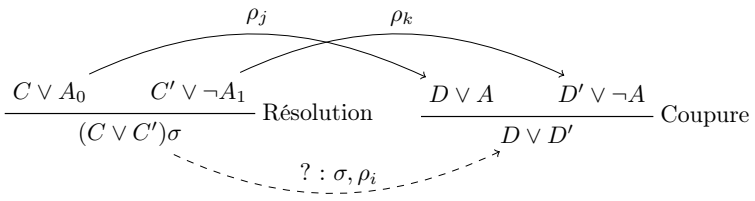



FIGURE 1.3 – Simulation de la coupure par la résolution

### 1.8.5 Indecidabilité

Bien que le calcul soit correct et complet, nos résultats n'amènent pas à la construction d'un algorithme—c'est à un quelque programme qui *s'arrête toujours* et qui donne la réponse souhaitée à la fin des calculs—pour décider si un ensemble de clauses universelles est satisfaisable ou non. Si nous essayons d'adapter l'algorithme de résolution propositionnelle, cf. **??**, on rencontre un problème majeur : cet algorithme pourrait ne jamais se terminer, en raison de la possibilité de produire une infinité de nouvelles de clauses. Pour s'en apercevoir, il suffit de considerer le langage $\mathcal{S}$ avec $\mathcal{S}_F = \{ (o, 0), (s, 1) \}$ et $\mathcal{S}_R = \{ (P, 1) \}$. Considérons l'ensemble $\mathcal{C}$ de clauses donné par

$$\mathcal{C} := \{ \neg P(x) \vee P(s(x)),\ P(o) \}.$$

L'algorithme engendrera, l'une après l'autres, toutes les clauses de la forme

$$P(\underbrace{s(\ldots s(}_{n \text{ fois}} o) \ldots) )$$

En fait, nous ne pouvons simplement pas trouver un algorithme ; les prochains théorèmes pourront être mieux compris dans le cadre du chapitre suivant, autour de la calculabilité, où nous formaliserons la notion d'algorithme.

**Théorème 1.99.** *Il n'existe aucun algorithme tel que, étant donné une formule du premier ordre close $\varphi$, il répond oui si $\varphi$ admet un modèle, et non si $\varphi$ est insatisfaisable.*

Car décider de la satisfaisabilité d'un formule du premier ordre se réduit (via la mise en forme prenexe, la Skolemisation, et la mise en forme clausale) à décider de la satisfaisabilité d'un ensemble de clauses, nous pouvons déduire cet autre théorème à partir du précédent :

**Théorème 1.100.** *Il n'existe aucun algorithme tel que, étant donné un ensemble fini de clauses $\mathcal{C}$, il répond oui si $\mathcal{C}$ admet un modèle, et non si $\mathcal{C}$ est insatisfaisable.*

### 1.8.6 Utilisation d'un démonstrateur automatique

**Exemple 1.101** (L'île misterieuse)**.** Écoutez cette histoire.

> *Chaque habitant de cette île est soit un cavaliers, soit un escroc. Il peut être un loup garou (il est donc dangereux, car il mange les hommes pendant les nuits de lune pleine). Un loup garou est lui aussi soit un cavalier soit un escroc. Les cavaliers disent toujours la vérité, les escrocs mentent toujours. Un explorateur debarque sur cette île et rencontre Albert, Bernard et Charles. Il est au courant qu'un des trois est un loup garou.*
>
> — *Albert prétend que Bernard est un loup ;*
> — *Bernard dit qu'il n'est pas un loup ;*
> — *Charles dit qu'au moins deux entre eux sont des escrocs.*
>
> *Qui doit choisir l'explorateur comme guide de son voyage ?*

Nous avons formalisé cette histoire en logique du premier ordre, dans un fichier prêt à être lu par le démonstrateur automatique `Prover9`. Ce fichier apparaît dans la Figure 1.4. Le prouveur automatique confirme que Albert est un loup garou, et donc l'explorateur ne choisira pas Albert comme

```
set(binary_resolution).

formulas(assumptions).
        Cavalier(x) | Escroc(x).
        LoupGarou(albert)  | LoupGarou(bernard) | LoupGarou(charles).
        Cavalier(albert) -> LoupGarou(bernard).
        Escroc(albert) -> -LoupGarou(bernard).
        Cavalier(bernard) -> -LoupGarou(bernard).
        Escroc(bernard) -> LoupGarou(bernard).
        Cavalier(charles) -> (
                (Escroc(albert) & Escroc(bernard))
                | (Escroc(albert) & Escroc(charles))
                | (Escroc(bernard) & Escroc(charles))
                 ).
        Escroc(charles) -> -(
                (Escroc(albert) & Escroc(bernard))
                | (Escroc(albert) & Escroc(charles))
                | (Escroc(bernard) & Escroc(charles))
                 ).
end_of_list.

formulas(goals).
        LoupGarou(albert).
end_of_list.
```

FIGURE 1.4 – Fichier d'entrée pour `Prover9`

guide. La preuve construite par le prouveur apparait dans la Figure 1.5. Le lecteur y reconnaîtra plusieurs instances de la règle de résolution. L'analyse de la preuve montre que l'hypothèse 'Albert n'est pas un loup garou' n'a pas été utilisée. Cela veut dire que la connaissances à disposition de l'explorateur, (qui est modélisée dans la liste des assomptions) est elle même incohérente.

Un procédé analogue peut être utilisé pour montrer que une liste de spécifications d'un programme/logiciel est incohérente, et donc ne peut pas être assuré par n'importe quel programme. Réfléchir avant se mettre à programmer!!! □

**Exemple 1.102** (Le mystère du Château Letot)**.** Écoutez cette autre histoire.

— *Quelqu'un qui habite Château Letot a tué tante Agate.*
— *Agate, le majordome, et Charles habitent Château Letot, et ils sont les seuls qui l'habitent.*
— *Un tueur haït toujours sa victime, et il n'est jamais plus riche que sa victime.*
— *Charles haït personne que tante Agate haït.*
— *Agate haït tous sauf le majordome.*
— *Le majordome haït tous ceux qui ne sont pas plus riches de tante Agate.*
— *Le majordome haït tous ceux qui tante Agate haït.*
— *Personne haït tous le monde.*
— *Agate n'est pas le majordome.*

*Qui a tué tante Agate ?*

Nous avons utilisé le prouveur automatique `Prover9` pour montrer que Agate s'est suicidé ; en fait, 'Agate a tué Agate' est une conséquence logique des faits décrits concernant le Château Letot.

Le procédé est comme auparavant (voir l'île mystérieuse). Nous avons d'abord modélisé l'histoire (*base de connaissances*, ou *ontologie*) en logique du premier ordre, en construisant ainsi un ensemble d'*assomptions* ; la phrase 'Agate a tuée Agate' étant la formule *but* à démontrer[4]. Le code qui a été fourni en entrée à `Prover9` apparaît dans la figure 1.6. `Prover9` transforme d'abord cet ensemble de formules dans un ensemble de clauses universelles. Observez donc l'introduction de

4. Dans la notation que nous avons utilisé dans le cours, les assomptions correspondent à l'ensemble $\Gamma$ et le but à $\varphi$ quand on se pose la question si $\varphi$ est une conséquence logique de $\Gamma$ ($\Gamma \models \varphi$ ?)

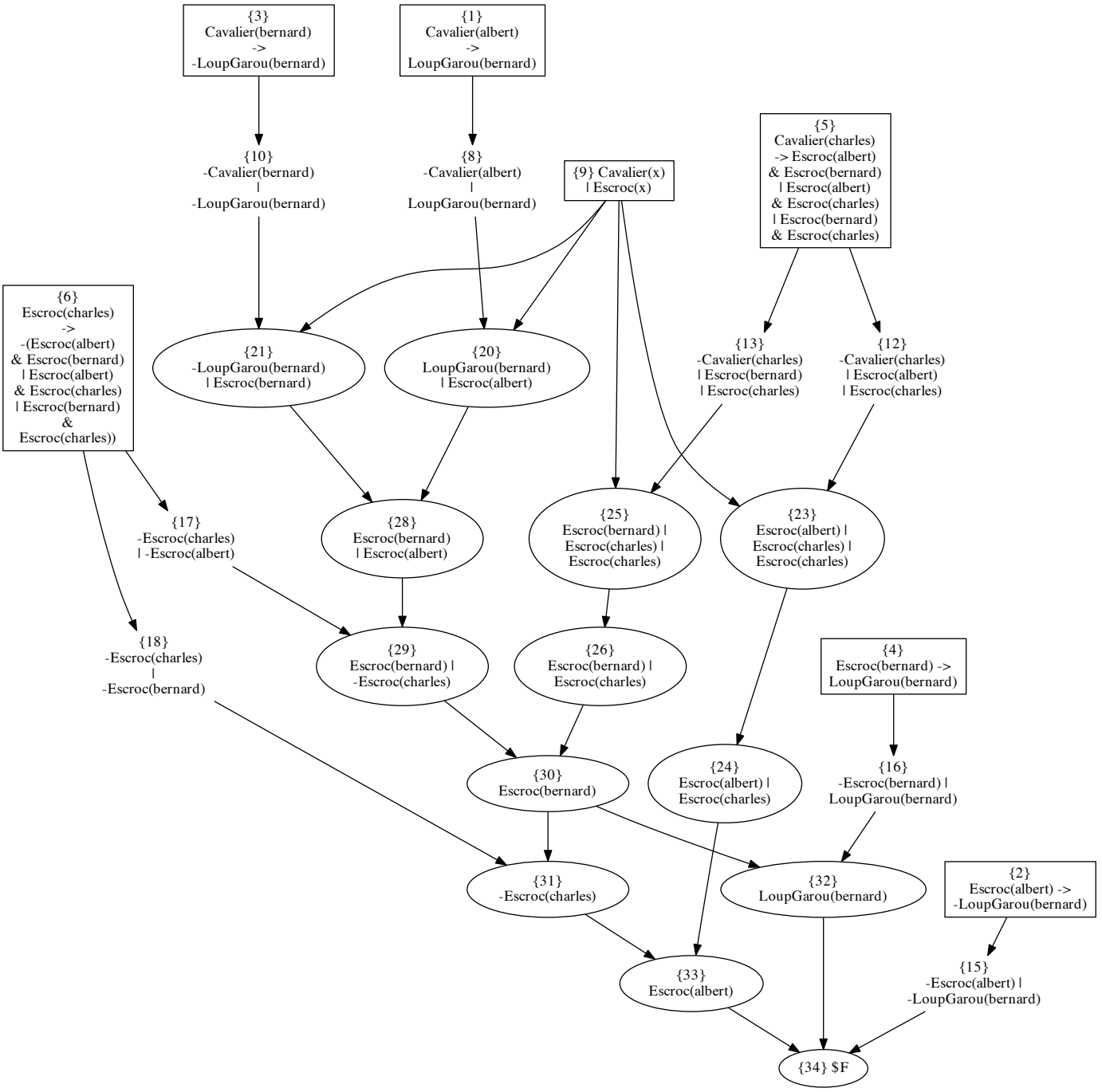



FIGURE 1.5 – Visualisation de la preuve construite par `Prover9`

```
set(binary_resolution).

formulas(assumptions).
        exists x (HabiteCL(x) & Tue(x,agate)).
        HabiteCL(agate) & HabiteCL(maj) & HabiteCL(charles)  &
                    (HabiteCL(x) -> (x = agate | x = maj | x = charles)).
        Tue(x,y) -> (Hait(x,y) & -PlusRiche(x,y)).
        Hait(agate,z) -> -Hait(charles,z).
        (-Hait(agate,x)) -> x=maj.
        x != maj -> Hait(agate,x).
        -PlusRiche(x,agate) -> Hait(maj,x).
        Hait(agate,x) -> Hait(maj,x).
        - (exists x all y Hait(x,y)).
        agate != maj.
end_of_list.

formulas(goals).
        Tue(agate,agate).
end_of_list.
```

FIGURE 1.6 – Fichier entrée pour le château Letot

```
HabiteCL(c1).  [clausify(1)].
Tue(c1,agate).  [clausify(1)].
HabiteCL(agate).  [clausify(2)].
HabiteCL(maj).  [clausify(2)].
HabiteCL(charles).  [clausify(2)].
-HabiteCL(x) | agate = x | maj = x | charles = x.  [clausify(2)].
-Tue(x,y) | Hait(x,y).  [clausify(3)].
-Tue(x,y) | -PlusRiche(x,y).  [clausify(3)].
-Hait(agate,x) | -Hait(charles,x).  [clausify(4)].
Hait(agate,x) | maj = x.  [clausify(5)].
maj = x | Hait(agate,x).  [clausify(6)].
PlusRiche(x,agate) | Hait(maj,x).  [clausify(7)].
-Hait(agate,x) | Hait(maj,x).  [clausify(8)].
-Hait(x,f1(x)).  [clausify(9)].
agate != maj.  [assumption].
-Tue(agate,agate).  [deny(10)].
```

FIGURE 1.7 – Château Letot : forme clausale

nouvelles constantes et de symboles de fonction par élimination de quantificateurs existentiels (Skolemization), la mise sous forme clausale, et l'inclusion du but parmi les assomptions, via sa négation (voir figure 1.7). La preuve que le but est une conséquence logique des assomptions apparaît dans la figure 1.8 Nous avons utilisé les outils GVIZIFY (pour transformer la sortie du `Prover9`—très souvent assez difficile à décrypter—vers un graphe décrit dans le langage `dot`) est GRAPHVIZ (pour dessiner des graphes à partir de leur description en langage `dot`) afin de présenter cette preuve sous forme de diagramme.

Un dernier remarque s'impose. Parmi les symboles de relation de notre langage nous avons utilisé le symbole $=$ sans avoir ajouté, parmi les assomptions, aucune hypothèse sur ce symbole. Notamment, nous aurions du ajouter des clauses explicitant le fait que l'égalité est réflexive, transitive, symétrique, et congruentielle. Par exemple, nous aurions du expliciter que si $x = y$ et $Hait(x, z)$ alors $Hait(y, z)$, et tous les inférences de ce type. `Prover9` reconnaît qu'il s'agit du symbole d'égalité et ajoute ces assomptions automatiquement. Car le traitement de l'égalité n'est pas optimal en utilisant la résolution seulement, on se sert aussi de la réglé de paramodulation,

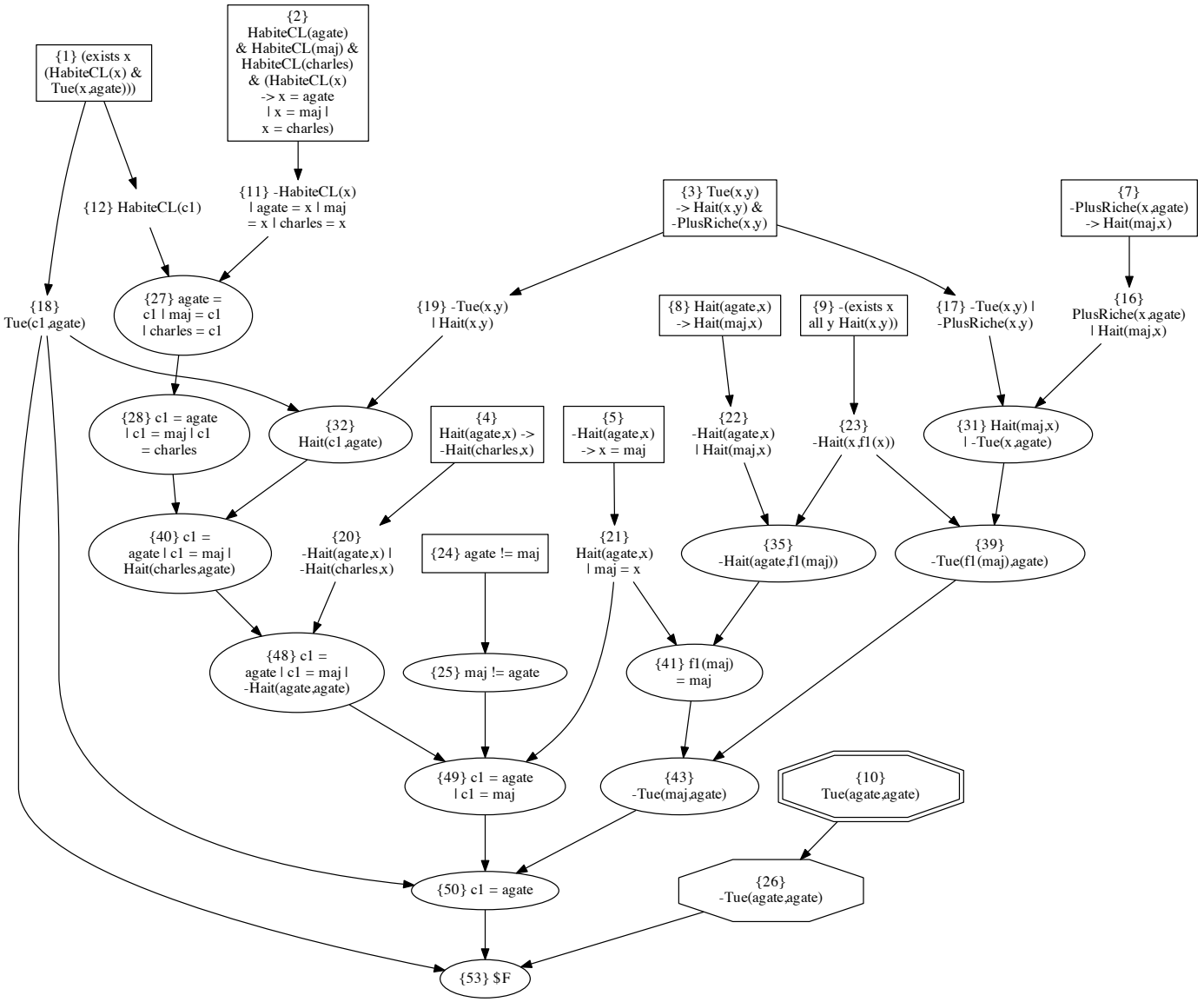



FIGURE 1.8 – La preuve trouvée par `Prover9`

que nous présentons ci-dessous.

$$\frac{C \vee t_1 = t_2 \qquad D(t_3)}{(C \vee D(t_2))\sigma}\,\texttt{Paramodulation}$$

où $\sigma$ est un unificateur de $t_1$ et $t_3$.

Dans notre exemple, nous avons que l'inférence de la clause 40,

$$\frac{c1 = agate \vee c1 = maj \vee c1 = charles \qquad Hait(c1, agate)}{c1 = agate \vee c1 = maj \vee Hait(charles, agate)}$$

est une instance de la règle de paramodulation. □

# Bibliographie

[Miq05] Alexandre Miquel. L'intuitionnisme : où l'on construit une preuve. *Pour la Science*, (49), 2005. http://www.dossierpourlascience.fr/ewb_pages/f/fiche-article-l-intuitionnisme-ou-l-on-construit-une-preuve-21944.php.