## 4.7 Unification

### 4.7.1 Substitions et MGUs

Dans la suite, on fixe $\mathcal{S}_\mathtt{f}$ 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 4.65.** 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 4.66.** Supposons $\mathcal{S}_\mathtt{f} = \{\, (f,2), (g,1) \,\}$ 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.

Pour simplifier, on ne décrira une substitution que pour les variables $x$ pour lesquelles $\sigma(x) \neq x$. Dans notre exemple, on décrira donc $\sigma$ en spécifiant simplement que $\sigma(x_0) = f(g(x_0), x_1)$ et $\sigma(x_1) = x_2$.

On représente souvent les substitutions par des listes de couples notées d'habitude par :

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

Ainsi, la substitution de l'exemple 4.66, sera notée par

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

**Définition 4.67.** 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 4.68.** On a

$$f(g(x), y)_{[z/x, g(y)/y]} = f(g(z), g(y)), \qquad g(f(x, f(y, x)))_{[g(w)/x]} = g(f(g(w), f(y, g(w)))).$$

**Définition 4.69.** 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 :

$$(\tau \circ \sigma)(x) = \tau(\sigma(x)) = (x_\sigma)_\tau. \tag{4.2}$$

**Exemple 4.70.** Soient

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

Calculons $\tau \circ \sigma$ :

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

On a donc

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

**Exercice 4.71.** Prouvez les relations suivantes :

$$t_{[]} = t\,,$$
$$t_{(\tau\circ\sigma)} = (t_\sigma)_\tau\,. \tag{4.3}$$

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

**Remarque 4.72.** 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 (4.2) et (4.3) qui, en renversant gauche et droite, pourraient apparaître à première vue un peu bizarres.

**Définition 4.73.** 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 4.74.** 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 4.75.** Un **problème d'unification** est une liste $p = (s_1, t_1), \ldots, (s_n, t_n)$ avec $s_i, t_i \in \mathcal{T}_{\mathcal{S}_t}(X)$. Une solution de ce problème – appelé **unificateur** de $p$ – est une substitution $\sigma$ telle que $s_i\sigma = t_i\sigma$, pour $i = 1, \ldots, n$. On notera $\mathsf{Unif}[\,p\,]$ l'ensemble des unificateurs de $p$.

**Exemple 4.76.**

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 4.77.** *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 4.78.** $\tau = [g(f(w))/x, g(f(w))/y] \in \mathsf{Unif}[\,(f(x,g(z)),f(g(z),y))\,]$, 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 $\rho = [f(w)/z]$.

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) :
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 4.2 – Algorithme d'unification

### 4.7.2 Algorithme d'unification

L'algorithme d'unification est illustré en figure 4.7.2. Nous donnons dans la suite des exemples de calcul de cet algorithme sur des problèmes d'unification.

**Exemple 4.79.** 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 4.80.** 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. $(f(y, k(y), g(x)), f(k(x), k(y), y))$.

**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.

### 4.7.3 Correction et completude

Nous souhaitons prouver les propositions suivantes.

**Proposition 4.81** (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 4.82** (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, ainsi que la Proposition 4.83, qui est le résultat nécessaire le moins évident à démontrer.

— $\Delta = \{\,(t, t) \mid t \in \mathcal{T}_{\mathcal{S}_t}(X)\,\}$ 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 4.83.** *Soient $\pi$ et $\psi$ deux problemes d'unification, soit $\sigma$ un* MGU *de $\pi$. Alors*

$$\mathsf{Unif}[\,\pi,\psi\,] = \{\,\rho\circ\sigma \mid \rho \in \mathsf{Unif}[\,\psi_\sigma\,]\,\}\,. \tag{4.4}$$

*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 $\pi$, $\sigma \le \tau$, 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 (4.4) est inclus dans celui de droite. Montrons donc l'autre inclusion. Si $\rho \in \mathsf{Unif}[\,\psi_\sigma\,]$, alors

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

donc $\rho\circ\sigma \in \mathsf{Unif}[\,\pi,\psi\,]$. Ceci complète la preuve de l'égalité (4.4).

Soient maintenant $\rho$ un MGU de $\psi_\sigma$ et soit $\tau \in \mathsf{Unif}[\,\pi,\psi\,]$. Grâce à l'égalité (4.4), nous pouvons écrire $\tau = \rho'\circ\sigma$ avec $\rho' \in \mathsf{Unif}[\,\psi_\sigma\,]$; on a alors $\rho \le \rho'$, donc $\rho' = \theta\circ\rho$ pour une substitution $\theta$ et, par conséquent, $\tau = \theta\circ\rho\circ\sigma$, ce qui montre que $\rho\circ\sigma \le \tau$. 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 4.84.** *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 4.85.** *On a*

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

**Lemme 4.86.** *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 solution, alors il en est de même pour $(x,x),(s_2,t_2),\ldots,(s_n,t_n)$.*

**Lemme 4.87.** *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 solution, alors il en est de même pour $(x,t),(s_2,t_2),\ldots,(s_n,t_n)$.*

Les Lemmes 4.86 et 4.87 sont des conséquences de la Proposition 4.83, 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 4.88.** A l'aide des Lemmes 4.84-4.87 complétez une preuve formelle de correction et complétude de l'algorithme d'unification.

*Exemple : démonstration de la Proposition 4.81, 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 4.84. Pour la ligne 13, on utilise le Lemme 4.87.

Les appels récursifs se trouvent aux lignes 6 et 9. On justifie la ligne 6 par le Lemme 4.85, et la ligne 9 par le e Lemme 4.86.

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. □

## 4.8 Résolution

### 4.8.1 Substitution, sur les formules propositionnelles

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 4.55). 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, donc d'une formule

$$C := \forall x_1, \ldots, \forall x_n (l_1 \vee \ldots \vee l_k) \text{ où les } l_i \text{ sont des littéraux.}$$

Par commodité, on travaillera ici sur la matrice $C_{\mathsf{mat}} = l_1 \vee \ldots \vee l_k$ de $C$ plutôt que sur $C$.

Par exemple,

$$\text{si } C := \forall x \forall y \forall z (R(x, y) \vee \neg Q(f(x), z)) \text{ alors } C_{\mathsf{mat}} := R(x, y) \vee \neg Q(f(x), z)$$

**Notation 4.89.** 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.

Reprenons l'exemple ci-dessus, si $\sigma = [f(y)/x, g(t)/z]$ alors

$$C_{\mathsf{mat}\sigma} := R(f(y), y) \vee \neg Q(f(f(y)), g(t)) \text{ et } C_\sigma := \forall y \forall z (R(f(y), y) \vee \neg Q(f(f(y)), g(t)))$$

**Définition 4.90.** Un **unificateur** de deux litteraux $l_0$ et $l_1$ est une substitution $\sigma$ telle que $l_{0\sigma} = l_{1\sigma}$.

**Lemme 4.91.** *$\sigma$ 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 $\sigma \in \mathsf{Unif}[(s_1, t_1), \ldots, (s_n, t_n)]$.*

Il découle directement de ce lemme que l'ensemble des unificateurs de deux littéraux – noté $\mathsf{Unif}[l_0, l_1]$ – ou bien est vide, ou bien possède une **substitution la plus générale**, qui sera appelé un $\mathsf{MGU}$ de $l_0$ et $l_1$.

### 4.8.2 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 $\perp$ depuis les clauses dans $\Gamma$. Les deux règles pour dériver des nouvelles clauses à partir des clauses déjà construites sont les suivantes :

$$\frac{C \vee A_0 \qquad C' \vee \neg A_1}{(C \vee C')_\sigma} \text{ (Résolution) où } \sigma \text{ est un } \mathsf{MGU} \text{ de } A_0 \text{ et } A_1 \,;$$

et

$$\frac{C \vee l_0 \vee l_1}{(C \vee l_0)_\sigma} \text{ (Factorisation) où } \sigma \text{ est un } \mathsf{MGU} \text{ des littéraux } l_0 \text{ et } l_1.$$

**Exemple 4.92** (Règle de Résolution)**.** La suivante est une instance de la règle de résolution :

$$\frac{\neg H(x) \vee T(x, a) \qquad \neg T(c, y) \vee L(x, y)}{\neg H(c) \vee L(c, a)}$$

Ici $A_0 = T(x, a)$ et $A_1 = T(c, y)$, $C = \neg H(x)$, $C' = L(x, y)$. La substitution appliquée est $\sigma = [c/x, a/y]$ qui est un $\mathsf{MGU}$ de $T(x, a)$ et $T(c, y)$.

**Exemple 4.93** (Règle de Factorisation)**.** La suivante est une instance de la règle de factorisation :

$$\frac{\neg H(x) \vee L(x,y) \vee T(x,a) \vee T(c,y)}{\neg H(c) \vee L(c,a) \vee T(c,a)}$$

Le MGU appliqué est encore une fois $[c/x, a/y]$.

**Théorème 4.94.** *La résolution est correcte est complète : pour toute formule clause $\varphi$, si $\mathcal{C}$ est un ensemble de clause équivalent à $\neg\varphi$ :*

$$\models \varphi \text{ ssi } \bot \text{ est dérivable par résolution à partir de } \mathcal{C}.$$

### 4.8.3 Indecidabilité

Bien que le calcul soit correct et complet, nos résultats n'amènent pas à la construction d'un algorithme – c'est à d'un 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, 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 4.95.** *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.*

Puisque décider de la satisfaisabilité d'une formule du premier ordre se réduit (via 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 4.96.** *Il n'existe aucun algorithme qui, étant donné un ensemble fini de clauses $\mathcal{C}$, répond oui si $\mathcal{C}$ admet un modèle, et non si $\mathcal{C}$ est insatisfaisable.*