919. Unification : algorithmes et applications.

1 Définitions

1.1 Termes du premier ordre et substitutions

Définition 1. Soit $\mathcal{V}$ un ensemble infini dénombrable de variables individuelles, $\mathcal{F}_n$ un ensemble au plus dénombrable de symboles de fonction d'arité $n \in \mathbb{N}$, $\mathcal{F} = \cup_{n \geqslant 0} \mathcal{F}_n$. On définit par induction l'ensemble des termes $T(\mathcal{V}, \mathcal{F})$:

1. $\mathcal{V} \subset T(\mathcal{V}, \mathcal{F})$;
2. $\forall n \in \mathbb{N}, \forall f \in \mathcal{F}_n, \forall t_1, \ldots, t_n \in T(\mathcal{V}, \mathcal{F})$, $f(t_1, \ldots, t_n) \in T(\mathcal{V}, \mathcal{F})$.

Définition 2. On appelle *substitution* une fonction de $\mathcal{V}$ dans $\mathcal{T}$ telle que le *domaine* $\operatorname{dom} \sigma = \{x \in V, x \neq \sigma(x)\}$ est fini. Si $\operatorname{dom} \sigma = \{x_1, \ldots, x_n\}$, on note aussi $\sigma = [x_1 \mapsto \sigma(x_1), \ldots, x_n \mapsto \sigma(x_n)]$. $\diamond$ désigne la substitution vide.

Remarque 3. On peut prolonger une substitution σ de manière unique sur $\mathcal{T}$ en posant pour tout $n \in \mathbb{N}$, pour tout $f \in \mathcal{F}_n$ et pour tous termes $t_1, \ldots, t_n \in \mathcal{T}$: $f(t_1, \ldots, t_n)\sigma = f(t_1\sigma, \ldots, t_n\sigma)$.

Définition 4. Soit t un terme. On appelle *instance* tout terme de la forme $t\sigma$, où σ est une substitution.

Définition 5. On définit par induction l'ensemble $\mathcal{P}os(t)$ des positions d'un terme t (c'est un ensemble de mots sur l'alphabet $\mathbb{N}^*$) :

- $\forall t \in T(\mathcal{V}, \mathcal{F}), \varepsilon \in \mathcal{P}os(t)$;
- $\forall n \in \mathbb{N}, \forall f \in \mathcal{F}_n, \forall t_1, \ldots, t_n \in T(\mathcal{V}, \mathcal{F}), \forall i \in [\![1, n]\!]$,

$$m \in \mathcal{P}os(t_i) \Rightarrow im \in \mathcal{P}os(f(t_1, \ldots, t_n)).$$

1.2 Unification

Définition 6. Deux termes s et t sont dits *unifiables* s'il existe une substitution σ telle que $s\sigma = t\sigma$. On dit que σ est un *unificateur* de s et t.

Définition 7. On dit qu'un terme s *filtre* un terme t lorsqu'il existe une substitution σ telle que $s\sigma = t$.

Exemple 8. $u = f(x, x, y)$ et $v = f(f(y, y, z), f(f(y, y, z), a)$ sont unifiables : la substitution $[x \mapsto f(a, a, z), y \mapsto a]$ les rend égaux.

Théorème 9. Soient u et v deux termes du premier ordre unifiables. Il existe un unificateur σ de u et v tel que pour tout unificateur σ' de u et v, il existe une substitution σ'' vérifiant $\sigma' = \sigma'' \circ \sigma$. On le note $mgu(u, v)$ (*most general unifier*).

2 Algorithmes d'unification

2.1 Une approche naïve

Algorithme naïf d'unification

```
procédure UnifierEnsemble(E, σ)        ▷ E est une pile d'équations ordonnées
    si E est vide alors
        renvoyer σ
    X, E' ← Dépiler(E)
    si X = f(u_1, ..., u_n) ~ g(v_1, ..., v_p) alors
        si f = g alors
            renvoyer UnifierEnsemble({u_1 ~ v_1} :: ... :: {u_n ~ v_n} :: E', σ)
        sinon
            renvoyer « s et t ne sont pas unifiables » (clash)
    sinon si X = x ~ x alors
        renvoyer UnifierEnsemble(E', σ)
    sinon si X = x ~ u ou X = u ~ x (où x ≠ u) alors
        si x n'apparaît pas dans u alors
            UnifierEnsemble(E'[u/x], [x ↦ u] ∘ σ)
        sinon
            renvoyer « s et t ne sont pas unifiables » (occur-check)
    renvoyer t

procédure Unifier(s, t)
    renvoyer UnifierEnsemble({s ~ t}, id)
```

Proposition 10. L'algorithme Unifier termine toujours.

Théorème 11. Si l'appel $\text{Unifier}(s, t)$ termine sans échec, alors la substitution renvoyée est un unificateur le plus général de s et t. S'il échoue, alors s et t ne sont pas unifiables.

Exemple 12. On considère une suite (x_n) de variables et on définit par induction les suites (u_n) et (v_n) suivantes :

$$\begin{cases} u_0 & = & v_0 = x_0 \\ u_{n+1} & = & f(u_n, x_n) \\ v_{n+1} & = & f(x_n, v_n) \end{cases}$$

L'algorithme Unifier a un nombre d'appels exponentiel sur le terme $f(u_n, u_n)$.

2.2 L'algorithme pseudo-linéaire

Définition 13. Un ensemble E d'équations est dit *dirigé* s'il vérifie les conditions suivantes :
- E est fini ;
- les équations de E sont de la forme : $x \sim y$ ou $x \sim f(x_1, \dots, x_n)$, où $x, y, x_1, \dots, x_n$ sont des variables ;
- chaque variable apparaît au plus une fois en position gauche ;
- il n'y a pas, dans E, de cycle de la forme $x_0 \sim x_1 \sim x_2 \sim \dots \sim x_n \sim x_0$.

Définition 14. Un ensemble d'équations dirigé E est dit *acyclique* si la plus petite relation transitive $<_E$ vérifiant :
- si $x \sim y \in E$ alors $y <_E x$;
- si $x \sim f(x_1, \dots, x_n) \in E$ alors pour tout $i \in [\![1, n]\!]$, on a $x_i <_E x$;

ne contient pas de cycle.

Définition 15. Soit E un ensemble dirigé d'équations et x une variable. On note $\nu(E, x)$ le couple défini comme suit :
- si x n'apparaît pas comme membre gauche de E, alors $\nu(E, x) = (x, x)$;
- si $x \sim f(x_1, \dots, x_n) \in E$ alors $\nu(E, x) = (x, f(x_1, \dots, x_n))$;
- si $x \sim y \in E$ alors $\nu(E, x) = \nu(E, y)$.

Remarque 16. La dernière condition sur les ensembles dirigés assure que $\nu(E, x)$ est toujours défini.

Définition 17. $\tau(E, x)$ est le terme défini par :
- si $\nu(E, x) = (z, z)$, $\tau(E, x) = z$;
- si $\nu(E, x) = (z, f(x_1, \dots, x_n))$ et $t_i = \tau(E, x_i)$ pour $i \in [\![1, n]\!]$, alors $\tau(E, x) = f(t_1, \dots, t_n)$.

On dit que (E, x) *représente* le terme $\tau(E, x)$.

Lemme 18. Si E est un ensemble dirigé acyclique, alors $\tau(E, x)$ est toujours défini.

Théorème 19. Pour tout ensemble de termes $\{t_1, \dots, t_n\}$, on peut construire un ensemble dirigé acyclique E et des variables $x_1, \dots, x_n$ tel que pour tout $i \in [\![1, n]\!]$, $t_i = \tau(E, x_i)$.

Développement 1. L'algorithme UnifierPseudoLinéaire termine toujours et si E est dirigé, pour toutes variables x et y, $E' = \textsc{UnifierVariables}(E, x, y)$ est dirigé. Si de plus E' est acyclique, alors $\tau(E', x) = \tau(E', y)$.

Théorème 20. Si l'appel $\textsc{UnifierPseudoLinéaire}(s, t)$ termine sans échec, alors la substitution renvoyée est un unificateur le plus général de s et t en $O(n \log n)$.

Algorithme pseudo-linéaire d'unification

procédure $\textsc{UnifierVariables}(E, x, y)$
  $x', v \leftarrow \nu(E, x)$
  $y', w \leftarrow \nu(E, y)$
  si $x' = y'$ **alors renvoyer** E
  sinon si $x' = v$ **alors renvoyer** $E \sqcup \{x' \sim y'\}$
  sinon si $y' = w$ **alors renvoyer** $E \sqcup \{y' \sim x'\}$
  sinon
    $f(x_1, \dots, x_n) \leftarrow v$
    $g(x_1, \dots, x_p) \leftarrow w$
    si $f \neq g$ **alors renvoyer** « s et t ne sont pas unifiables » (*clash*)
    sinon
      $E' \leftarrow E \setminus \{x' \sim v\} \sqcup \{x' \sim y'\}$
      pour $i = 1, \dots, n$ **faire**
        $E' \leftarrow \textsc{UnifierVariables}(E', x_i, y_i)$
      renvoyer E'

procédure $\textsc{UnifierPseudoLinéaire}(s, t)$
  $E_0, x_0, y_0 \leftarrow \textsc{ConstruireDAG}(t, u)$ ▷ Tel que $s = \tau(E_0, x_0)$ et $t = \tau(E_0, y_0)$
  si $\textsc{UnifierVariables}(E_0, x_0, y_0)$ est acyclique **alors**
    renvoyer σ telle que $x\sigma = \tau(E, x)$ ▷ $s\sigma = \tau(E, x_0) = \tau(E, y_0) = t\sigma$
  sinon
    renvoyer « s et t ne sont pas unifiables » (*occur-check*)

3 Application à la réécriture

On considère un système de réécriture de termes R.

Définition 21. On dit que deux termes u et v sont joignables (noté $u \downarrow v$) s'il existe w tel que $u \rightarrow^* w \leftarrow^* v$.

Définition 22. R est dit *localement confluent* si pour tous termes t, u, v :

$$u \leftarrow t \rightarrow v \quad \Rightarrow \quad u \downarrow v.$$

Définition 23. R est dit *confluent* si pour tous termes t, u, v :

$$u \leftarrow^* t \rightarrow^* v \quad \Rightarrow \quad u \downarrow v.$$

Définition 24. On définit les sous-termes d'un terme t à partir des positions $\mathcal{P}os(t)$ comme suit :
- $t|_\varepsilon = t$;
- si $t = f(t_1, \dots t_n)$, $t|_{im} = t_i|_m$.

Définition 25. Pour tous termes t, u et toute position $p \in \mathcal{P}os(t)$, $t[u]_p$ désigne le terme obtenu en remplaçant le sous-terme de t à la position p par u :

— $t[u]_\varepsilon = u$;
— si $t = f(t_1, \dots t_n)$, $t[u]_{im} = f(t_1, \dots, t_i[u]_m, \dots, t_n)$.

Définition 26 (Paire critique)**.** On considère deux règles $\ell_1 \to r_1$ et $\ell_2 \to r_2$ dont les variables ont été renommées pour que $\mathcal{V}ar(\ell_1, r_1) \cap \mathcal{V}ar(\ell_2, r_2) = \varnothing$. Soit $p \in \mathcal{P}os(\ell_1)$ tel que $\ell_1|_p$ ne soit pas une variable et $\sigma = mgu(\ell_1|_p, \ell_2)$. Alors on appelle *paire critique* le couple $\langle r_1\sigma, (\ell_1\sigma)[r_2\sigma]_p\rangle$.

Exemple 27. Si l'on considère le système de règles suivant :

— $f(f(x, y), z) \to f(x, f(y, z))$
— $f(i(x_1), x_1) \to e$

alors $mgu(\ell_1|_1, \ell_2) = mgu(f(x, y), f(i(x_1), x_1)) = [x \mapsto i(x_1), y \mapsto x_1]$ et les deux dérivations possibles de $f(f(i(x_1), x_1), z)$ donnent lieu à la paire critique :

$$\langle f(i(x_1, f(x_1, z)), f(e, z)\rangle.$$

Lemme 28 (Lemme des paires critiques)**.** Si $v \leftarrow t \to u$ alors soit $u \downarrow v$, soit u, v s'écrivent respectivement $s[c_1]_p, s[c_2]_p$ pour une certaine position $p \in \mathcal{P}os(s)$ où $\langle c_1, c_2\rangle$ ou $\langle c_2, c_1\rangle$ est une instance d'une paire critique de R.

Théorème 29 (Théorème des paires critiques)**.** Un système de réécriture de termes est localement confluent si et seulement si toutes ses paires critiques sont joignables.

Lemme 30 (Newman)**.** Un système de réduction terminant est confluent si et seulement s'il est localement confluent.

Corollaire 31. Un système de réécriture de termes terminant est confluent si et seulement si toutes ses paires critiques sont joignables.

Développement 2. La confluence d'un système de réécriture de termes fini et terminant est décidable.

Développements	**Références**
Algorithme pseudo-linéaire d'unification	
Décidabilité de la confluence d'un SRT fini terminant	David Baader