Unification dans des théories

5 avril 2002

1 Compléments sur l'unification

1.1 Ensemble complet d'unificateurs

Il existe des théories pour lesquels les problèmes n'ont pas un unificateur plus général, mais plusieurs (ou pas du tout !).

Un ensemble S de substitutions est un ensemble complet d'unificateurs d'un problème P si

1. $S \subseteq \mathcal{U}(P)$,
2. $(\forall \tau \in \mathcal{U}(P))\ (\exists \sigma \in S)\ (\exists \rho \in Sub(T(\Sigma, V))\ \tau = \rho \circ \sigma$.

2 Combinaison d'algorithmes d'unification

2.1 Hétérogénéité

Soit $F = F_0 \uplus F_1$. On suppose que l'on a deux théories

- E_0 sur $T(F_0, X)$
- et E_1 sur $T(F_1, X)$.

Un terme $t \in T(F, X)$ est pur si $t \in T(F_0, X) \cup T(F_1, X)$. Si t n'est pas pur, il est hétérogène.

Un sous-terme pur $t_{|p}$ est maximal
si $(\forall q \in Pos(t))\ q$ ***prefix*** $p \quad \Rightarrow \quad$ «$t_{|q}$ est hétérogène».

Une équation $s \stackrel{?}{=} t$ est pure

- si $\{s, t\} \subseteq T(F_0, X)$
- ou $\{s, t\} \subseteq T(F_1, X)$

Une équation $s \stackrel{?}{=} t$ est propre si ni s, ni t ne sont des variables.

2.2 Propriété des théories

Une théorie est sans collapsus si elle ne contient aucune identité de la forme $x = t$. Une théorie est régulière si dans chaque identité $s = t$ chaque variable qui apparaît dans s apparaît dans t et vice-versa.

2.3 Organisation d'un problème hétérogène

Un problème d'unification hétérogène s'écrit

$$P = P_0 \cup P_1 \cup P_V \cup P_H$$

où P_0 est formé des équations propres et pures dans $T(F_0, X)$,
P_1 est formé des équations propres et pures dans $T(F_1, X)$,
P_V est formé des équations de la forme $x \stackrel{?}{=} y$,
P_H est formé des équations hétérogènes.

Dans ce qui suit on travaille sur des problèmes en forme triangulaire. Donc on n'appelle pas Elimine.

2.4 L'unification dans des théories hétérogènes

Abstract	$P \cup \{s \stackrel{?}{=} t\}$	$\Rightarrow$	$\exists x\ P \cup \{s[x]_p \stackrel{?}{=} t, x \stackrel{?}{=} s_{\|p}\}$ $s_{\|p}$est un sous-terme maximal pur de s.
Solve$_0$	$P \cup P_0$	$\Rightarrow$	$P \cup P_0'$ P_0' est une forme résolue pure de P_0 dans E_0.
Solve$_1$	$P \cup P_1$	$\Rightarrow$	$P \cup P_1'$ P_1' est une forme résolue pure de P_1 dans E_1.
Coalesce	$P \cup \{x \stackrel{?}{=} y\}$	$\Rightarrow$	$P\{x \mapsto y\} \cup \{x \stackrel{?}{=} y\}$ $x \neq y$
Merge	$P \cup \{x \stackrel{?}{=} f(\vec{s}), x \stackrel{?}{=} g(\vec{t})\}$	$\Rightarrow$	$P\{x \stackrel{?}{=} f(\vec{s}), f(\vec{s}) \stackrel{?}{=} g(\vec{t})\}$

2.5 Présolutions et solutions

Une présolution est un problème dont toutes les équations sont de la forme $x \stackrel{?}{=} t$. Une présolution est une solution si elle n'admet pas de cycle de la forme

$$x_1 \stackrel{?}{=} s_1[x_2] \ldots x_i \stackrel{?}{=} s_i[x_{i+1}] \ldots x_{2n} \stackrel{?}{=} s_i[x_1]$$

où les symboles à la racine d'au moins deux s_i et s_j sont dans des théories différentes.

2.6 Du problème aux solutions

La relation $\Rightarrow$ est clairement indéterministe. En particulier, elle a un branchement multiple pour $Solve_i$ qui peut fournir plus qu'une solution ou pas du tout. On suppose que $Solve_i$ fournit un ensemble complet d'unificateurs.

2.7 $Solve_i$

$Solve_i$

- transforme P en Q (soit $P \Rightarrow Q$)
- et prend une équation dans P_i et retourne (de façon indéterministe mais complète)
  - des équations de la forme $x_j \stackrel{?}{=} s_j$ dans P_i
  - et des équations de la forme $y_k \stackrel{?}{=} y_l$
    que l'on va mettre dans P_V,
  - une transition existe pour chaque unificateur de l'ensemble complet,

2.8 Les échecs

L'algorithme échoue dans trois cas, c'est-à-dire qu'on est en présence d'un problème équationnel pour lequel aucune transition n'existe et pourtant le problème n'est pas lui-même une solution.

1. Seule une transition $Solve_i$ est applicable, mais elle ne retourne aucun unificateur,
2. Aucune transition n'est applicable, mais il reste une équation de la forme $s \stackrel{?}{=} t$ avec ni s, ni t qui ne soit une variable. Cela signifie que s et t ont des symboles racines dans deux théories différentes.
3. Le problème est une présolution qui n'est pas une solution (présence d'un cycle).

2.9 Du problème aux solutions

Si E_0 et E_1 sont des théories régulières et sans collapsus
alors l'ensemble $\{Q \text{ solution} \mid P \stackrel{*}{\Rightarrow} Q\}$
forme un ensemble complet d'unificateurs pour P.

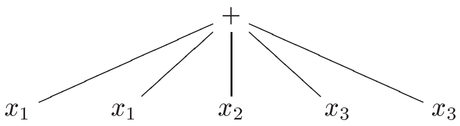
2.10 L'unification associative et commutative générale

Dans le cas de l'unification associative et commutative générale, seuls certains opérateurs sont AC (c'est-à-dire associatifs et commutatifs[1]). Il faudra donc combiner des algorithmes d'unification où seul un opérateur est AC. Si + est un opérateur associatif et commutatif, la résolution d'équations dans la théorie $T(\{+\}, X)$ s'appelle l'unification associative et commutative restreinte.

2.11 La représentation des termes

A priori un terme associatif et commutatif dans la théorie $T(\{+\}, X)$ est un arbre dont les noeuds internes sont des + et les feuilles des variables.

Par exemple $(x_1 + x_2) + (x_3 + (x_3 + x_1))$, se représente mieux par $x_1 + x_1 + x_2 + x_3 + x_3$ ou par l'arbre



ou par le terme $2x_1 + x_2 + 2x_3$.

2.12 La représentation des termes et des équations

Un terme s non réduit à une varibale est représenté par $s = \sum_{i=1}^{n} s_i x_i$ avec $\sum_{i=1}^{n} s_i \geq 2$

N. B. Un terme réduit à une variable correspond au cas où $\sum_{i=1}^{n} s_i = 1$

Le cas $\sum_{i=1}^{n} s_i = 0$ne correspond à aucun terme AC.

Une équation est représentée par $\sum_{i=1}^{n} s_i x_i = \sum_{i=1}^{n} t_i x_i$

2.13 La représentation des substitutions

Clairement un substitution σ idempotente applique à chaque variable x_i un terme AC a_i tel que $\{x_1, \ldots, x_n\} \cap Var(a_i) = \emptyset$ donc

$$\sigma(x_i) = \sum_{j=1}^{q} a_{ij} y_j$$

où $a_{ij} \in \mathbb{N}$, $x_i \notin \{y_1, \ldots, y_q\}$ et la condition (que l'on appelle $\mathcal{C}$) est satisfaite : $\sum_{j=1}^{q} a_{ij} \geq 1$.

2.14 L'unification associative et commutative restreinte

Un unificateur idempotent de $s \stackrel{?}{=} t$ dans le problème P signifie que

$$\sum_{i=1}^{n} s_i \sum_{j=1}^{q} a_{ij} y_j = \sum_{i=1}^{n} t_i \sum_{j=1}^{q} a_{ij} y_j$$

où $\{y_1, \ldots, y_q\} \cap Var(P) = \emptyset$

Soit encore

$$\sum_{j=1}^{q} y_j \sum_{i=1}^{n} s_i a_{ij} = \sum_{j=1}^{q} y_j \sum_{i=1}^{n} t_i a_{ij}$$

.

Donc si $(a_{1j} \ldots, a_{ij}, \ldots, a_{nj})$ (pour un j fixé) est une solution génératrice de l'équation diophantienne

– linéaire (tous les monômes de degré 1),

– homogène (pas de termes constants),

[1] et pas automate cellulaire comme le croient certains.

$$\sum_{i=1}^{n} s_i x_i = \sum_{i=1}^{n} t_i x_i$$

.

Donc si $(a_{1j} \dots, a_{ij}, \dots, a_{nj})$ (pour un j fixé) est une solution génératrice de l'équation diophantienne
- linéaire (tous les monômes de degré 1),
- homogène (pas de termes constants),

$$\sum_{i=1}^{n} s_i x_i = \sum_{i=1}^{n} t_i x_i$$

.

La solution générale est une combinaison $\mathbb{N}$-linéaire des ces solutions.

Étant donné un problème d'unification P, l'ensemble des solutions du système d'équations diophantiennes associé qui satisfont la condition $\mathcal{C}$ est l'ensemble des unificateurs de P.

Éant donné un problème d'unification P, l'ensemble des solutions du système d'équations diophantiennes associé qui satisfont la condition $\mathcal{C}$ est l'ensemble des unificateurs de P.

Reste à trouver un ensemble fini de générateurs de l'ensemble des solutions qui satisfont $\mathcal{C}$pour avoir un ensemble complet et fini d'unificateurs de P.

Pour cela on cherche d'abord un ensemble de générateurs des solutions du système dans $\mathbb{N}$. Ensuite on s'en sert pour engendrer un générateur des solutions qui satisfont $\mathcal{C}$.

3 L'unification en lambda calcul ou unification d'ordre supérieur

3.1 Objectif

On peut vouloir résoudre des équations comme

$$\lambda x.sin(cos(x)) \stackrel{?}{=} \lambda x.sin(F(x))$$

qui a pour solution $\lambda y.cos(y)$.

Alors que

$$\lambda x.F \stackrel{?}{=} \lambda x.sin(cos(x))$$

n'a pas de solution.

3.2 Exemple

Plus généralement, on va résoudre des équations du type

$$\lambda x.F(\lambda y.xya) \stackrel{?}{=} \lambda z.z(zaa)$$

où $a : \tau$ et $\vdash \lambda z.z(zaa) : (\tau \to \tau \to \tau) \to \tau \to \tau$.

F est une variable à instancier (dite aussi méta-variable). Quel est son type ?

3.3 Forme $\beta\eta$ longue

Les termes que l'on cherche à unifier
- sont typés,
- contiennent des variables libres que nous appellerons méta-variables et que nous noterons F, G, H (l'ensemble de méta-variables de t est noté $FV(t)$)
- et l'on travaille modulo β et η.

On considère donc la forme β-normale de chaque terme.

D'autre part on veut que si $\vdash M : \sigma_1 \to \ldots \to \sigma_n \to \tau$ alors le terme étudié soit de la forme

$$\lambda x_1 \ldots x_n.N$$

Si toutes variables ne sont pas présente il faut faire une ou plusieurs η-extensions. En résumé pour obtenir la forme $\beta\eta$ longue d'un terme

$$\vdash M : \sigma_1 \to \ldots \to \sigma_n \to \tau$$

On calcule la forme β-normale $\lambda x_1 \ldots x_k.N$.

Clairement, $k \leq n$ et la forme $\beta\eta$ longue est

$$\lambda x_1 \ldots x_k x_{k+1} \ldots x_n.N x_{k+1} \ldots x_n$$

La forme $\beta\eta$-longue du terme

$$\lambda x.F(\lambda y.xya) \stackrel{?}{=} \lambda z.z(zaa)$$

est

$$\lambda x_1 x_2.F(\lambda y.x_1 ya)x_2 \stackrel{?}{=} \lambda x_1 x_2.x_1(x_1 aa)x_2$$

3.4 Forme *flexible-rigide*

Si l'équation est de la forme $\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})$
où v n'est pas une méta-variable, on appelle cela un cas flexible-rigide

3.5 Forme *flexible-rigide* : projection

Si l'équation est de la forme $\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})$ où v n'est pas une méta-variable.

On peut faire l'hypothèse que v est le symbole de tête de l'un des t_i et que l'affectation que l'on fait à F vise à «atteindre» ce terme. On appelle cette règle projection.

Alors posons $\theta := F \mapsto \lambda\overrightarrow{y_n}.y_i(\overrightarrow{H_q(\overrightarrow{y_n})})$.

$$\{\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P$$

$$\Longmapsto$$

$$\{\lambda\overrightarrow{x_k}.\theta(F(\overrightarrow{t_n})) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\theta(v(\overrightarrow{t'_m}))\} \cup \theta P$$

Puisque $\theta(F(\overrightarrow{t_n})) \underset{\beta}{\longrightarrow} \lambda\overrightarrow{x_k}.\theta t_i(\overrightarrow{H_q(\overrightarrow{\theta t_n})})$

$$\{\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P$$

$$\Longmapsto$$

$$\{\lambda\overrightarrow{x_k}.\theta t_i(\overrightarrow{H_q(\overrightarrow{\theta t_n})}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{\theta t'_m})\} \cup \theta P$$

3.6 Forme *flexible-rigide* : imitation

Si l'équation est de la forme $\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})$ où v n'est pas une méta-variable.

On peut faire l'hypothèse que v est précisément le symbole par lequel on souhaite que F commence. Dans ce cas, v n'est pas une variable liée.

Alors posons $\theta := F \mapsto \lambda\overrightarrow{y_n}.v(\overline{H_i(\overrightarrow{y_n})})$.

$$\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}v(\overrightarrow{t'_m}) \cup P$$

$$\Longmapsto$$

$$\{\lambda\overrightarrow{x_k}.\theta(F(\overrightarrow{t_n})) \stackrel{?}{=} \lambda\overrightarrow{x_k}\theta(v(\overrightarrow{t'_m}))\} \cup \theta P$$

Soit $\theta := F \mapsto \lambda\overrightarrow{y_n}.v(\overline{H_i(\overrightarrow{y_n})})$.

$$\lambda\overrightarrow{x_k}.\theta(F(\overrightarrow{t_n})) \stackrel{?}{=} \lambda\overrightarrow{x_k}\theta(v(\overrightarrow{t'_m}))\} \cup \theta P$$

$$\overset{Decompose}{\Longmapsto}$$

$$\{\lambda\overrightarrow{x_k}.H_1(\overrightarrow{\theta t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{\theta t'_1}, \dots, \lambda\overrightarrow{x_k}.H_m(\overrightarrow{\theta t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{\theta t'_m}\} \cup \theta P$$

donc

$$\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}v(\overrightarrow{t'_m}) \cup P$$

$$\Longmapsto$$

$$\{\lambda\overrightarrow{x_k}.H_1(\overrightarrow{\theta t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{\theta t'_1}, \dots, \lambda\overrightarrow{x_k}.H_m(\overrightarrow{\theta t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{\theta t'_m}\} \cup \theta P$$

On appelle cette règle imitation.

3.7 Le système PT0 de transformation de problèmes

Supprime

$\{u \stackrel{?}{=} u\} \cup P \quad \Rightarrow \quad P$

Decompose

$\{\lambda\overrightarrow{x_k}.v(\overrightarrow{t_m}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P \quad \Rightarrow \quad \bigcup_{i=1}^{m}\{\lambda\overrightarrow{x_k}.\overrightarrow{t_i}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{t'_i}\} \cup P$

Elimine

$\{F \stackrel{?}{=} t\} \cup P \quad \Rightarrow \quad \theta P \quad$ si $\theta = \{F \mapsto t\}$ et $F \notin FV(t)$

Projette

$\{\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P \quad \Rightarrow \quad \{\lambda\overrightarrow{x_k}.\theta(F(\overrightarrow{t_n})) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\theta(v(\overrightarrow{t'_m}))\} \cup \theta P$
où $\theta = \{F \mapsto \lambda\overrightarrow{y_n}.y_i(\overline{H_q(\overrightarrow{y_n})})\}$

Imite

$\{\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P \quad \Rightarrow \quad \{\lambda\overrightarrow{x_k}.\theta(F(\overrightarrow{t_n})) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\theta(v(\overrightarrow{t'_m}))\} \cup \theta P$
où $\theta := \{F \mapsto \lambda\overrightarrow{y_n}.v(\overline{H_m(\overrightarrow{y_n})})\}$

3.8 Le système PT de transformation de problèmes

Supprime

$\{u \stackrel{?}{=} u\} \cup P \quad \Rightarrow \quad P$

Decompose

$\{\lambda\overrightarrow{x_k}.v(\overrightarrow{t_m}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P \quad \Rightarrow \quad \bigcup_{i=1}^{m}\{\lambda\overrightarrow{x_k}.\overrightarrow{t_i}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{t'_i}\} \cup P$

Elimine

$\{F \stackrel{?}{=} t\} \cup P \quad \Rightarrow \quad \theta P \quad$ si $\theta = \{F \mapsto t\}$ et $F \notin FV(t)$

Projette

$\{\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P \quad \Rightarrow \quad \{\lambda\overrightarrow{x_k}.\theta t_i(\overline{H_q(\overrightarrow{\theta t_n})}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{\theta t'_m})\} \cup \theta P$
où $\theta = \{F \mapsto \lambda\overrightarrow{y_n}.y_i(\overline{H_q(\overrightarrow{y_n})})\}$

Imite

$\{\lambda\overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.v(\overrightarrow{t'_m})\} \cup P \quad \Rightarrow \quad \bigcup_{i=1}^{m}\{\lambda\overrightarrow{x_k}.H_i(\overrightarrow{\theta t_n}) \stackrel{?}{=} \lambda\overrightarrow{x_k}.\overrightarrow{\theta t'_i}\} \cup \theta P$
où $\theta := \{F \mapsto \lambda\overrightarrow{y_n}.v(\overline{H_m(\overrightarrow{y_n})})\}$

3.9 Forme ***flexible-flexible***

Les équations de la forme $\lambda \overrightarrow{x_k}.F(\overrightarrow{t_n}) \stackrel{?}{=} \lambda \overrightarrow{x_k}.G(\overrightarrow{t'_m})$ sont dites flexibles-flexibles.
On montre

1. qu'elles ont toujours au moins une solution,
2. qu'elles ont une infinité de solutions sans pourtant posséder un ensemble complet fini d'unificateurs.

PT une procédure de préunification. σ est un préunificateur de P si σP produit par Supprime, Decompose et Elimine un ensemble d'équations flexibles-flexibles.

3.10 Cohérence

Si $P \Rrightarrow P'$ et si θ est un préunificateur de P' alors θ est un préunificateur de P.

3.11 Complétude

Dans le cas où la transformation est Elimine, Projette, Imite
on écrit $\Rrightarrow^{\theta}$.
Dans les autres cas on écrit $\Rrightarrow^{id}$.

Si θ est un préunificateur de P alors il existe P' composé uniquement d'équations flexibles-flexibles tel que $P \Rrightarrow_{\theta} P'$.