

Q7 3.1 Rewrite systems, rewriting of terms, definition.

But :

- automatiser les preuves
- évaluer les expressions (en connaissant les propriétés(termes) de cette expression)
- (- prototyping. On a la définition du programme voulu, et on vérifie que c'est bien ce qu'on veut faire)

-> **transformer les équations (axiomes) en rewrite rules** $a=b \rightarrow a$ est réécrit en b

Ex : $\text{not true} = \text{false} \rightarrow \text{not}(\text{true}) \sim_1 \text{false}$

Problèmes :

- Quelle direction? (réécrire le terme en un terme de plus faible complexité)
- Ensemble des rewriting rules est complet?
- terminaison (nombre fini d'étapes?)
- confluence (une solution finale (normale) unique pour chaque réécriture)

Abstract rewriting system : adaptation de la théorie équationnelle :

Avant : théorèmes déduits des axiomes

Ici : théorèmes obtenus en transformant un terme en un autre

Definition :

Soit $\Sigma = \langle S, F \rangle$ une signature, X un S -sorted set de variables

-> un ARS est $A = (T_{\Sigma, X}, \rightarrow)$, avec $\rightarrow \subseteq T_{\Sigma, X} \times T_{\Sigma, X}$

Décomposé en steps $l \sim r \in \rightarrow$

On évite la symétrie et la réflexivité sinon on risque d'avoir qqch qui ne se termine pas

Closure : système obtenu à partir des règles basiques en appliquant :

La substitution

La substitutivité

La transitivité (Si (t, t') et $(t', t'') \in \rightarrow^* \Rightarrow (t, t'') \in \rightarrow^*$)

$\text{Closure}(A) = (T_{\Sigma, X}, \rightarrow^*)$

$\text{Rew}_{\Sigma, X} \subseteq T_{\Sigma}(X) \times T_{\Sigma}(X)$ ensemble des rewrite rules

Déduction de théorèmes depuis le ARS :

2 termes sont égaux dans cette théorie si :

Si $t_1 = t_2 \in \text{Th}_{\rightarrow}(\text{Spec}) \Leftrightarrow \exists t \in T_{\Sigma}(X)$ tel que $t_1 \rightarrow^* t$ AND $t_2 \rightarrow^* t$

(Théorème valide dans ARS \Rightarrow valide dans la théorie équationnelle $\text{Th}_{\rightarrow}(\text{Spec}) \subseteq \text{Th}_{\text{eq}}(\text{Spec})$)

Processus pour le rewriting :

- **filtering** : choix de la règle : $\text{not}(\text{not}(\text{true}))$

- **substitution** : On met la partie droite de la règle correspondante : $\sim \text{not}(\text{false})$

Rewriting terms : steps

Soit $l \sim r$ une rewrite rule avec $l, r \in T_{\Sigma}(X)$

- $\text{filter}(t, l) = \langle \sigma, c \rangle \Leftrightarrow \exists \sigma \in X_{s \rightarrow} (T_{\Sigma, X})_s, \exists c$ tel que

$t = c[\sigma]$ (t = contexte appliqué à l) et $t' = c[\sigma r]$

- $\langle t, t' \rangle \in \text{Rew}_{l \sim r}$ un rewrite step

contexte c = terme avec un trou dedans où on peut remplacer le sous-terme. Ex: $\text{not}(\text{not}(\text{true}))$

$l = \text{not}(\text{true})$, $\sigma =$ une substitution, $t = \text{not}(\text{not}(\text{true}))$, $t' = \text{not}(\text{false})$, $r = \text{false}$

La procédure n'est pas unique (non déterministe) car le choix du contexte pas unique

Ex: $\text{and}(\text{not}(\text{true}), \text{not}(\text{false}))$

Exemple : $S = \{\text{Nat}, \text{Bool}\}$, $X_{\text{Nat}} = \{x, y, z\}$, $X_{\text{Bool}} = \{a, b\}$

$\text{OP} = \{+ : \text{nat}, \text{nat} \rightarrow \text{nat}, 0 : \rightarrow \text{nat}, 1 : \rightarrow \text{nat}, \text{true} : \rightarrow \text{Bool}, \text{false} : \rightarrow \text{Bool}, > : \text{nat}, \text{nat} \rightarrow \text{Bool}\}$

Axiomes : (0) $x + 0 = x$ (1) $x + \text{Succ}(y) = \text{Succ}(x + y)$ (2) $1 = \text{Succ}(0)$ (3) $(\text{Succ}(x) > 0) = \text{true}$

-> Rewrite rules : $x + 0 \sim_0 x$, $x + \text{Succ}(y) \sim_1 \text{Succ}(x + y)$, $1 \sim_2 \text{Succ}(0)$, $(\text{Succ}(x) > 0) \sim_3 \text{true}$

Rewriting terms :

- $1 > 0 \sim_2 \text{Succ}(0) > 0 \sim_3 \text{true}$

- $1 + 1 \sim_2 \text{Succ}(0) + 1 \sim_2 \text{Succ}(0) + \text{Succ}(0) \sim_1, x = \text{Succ}(0), y = 0 \text{Succ}(\text{Succ}(0) + 0) \sim_0, x = \text{Succ}(0) \text{Succ}(\text{Succ}(0))$