

UNIVERSITE DE LORRAINE

FACULTE DE SCIENCE ET TECHNOLOGIE

MASTER 1 INFORMATIQUE

Projet de LMC

ALGORITHME D'UNIFICATION

MARTELLI MONTANARI

Auteurs :

Baptiste LESQUOY

Nicolas WEISSENBACH

15 décembre 2015

Question 1

Les premiers pas vers l'unification

Dans un premier temps, nous avons décider d'écrire tous les prédicats de règle (qui permettent de savoir quelle règle utiliser) et les prédicats de réduction. Commencer par ces prédicats aura permis d'avoir une base de travail et d'emprunter un processus de développement itératif.

Exemple d'un prédicat *regle*

```
1 regle(E, simplify):-  
2   split(E, L, R),  
3   not(var(R)),  
4   var(L),  
5   not(compound(R))  
6   .
```

Listing 1 – Choix de la règle *simplify*

Ici nous avons le predicat *regle* pour l'unification *simplify*, le but de ce predicat est de renvoyer vrai si *simplify* est applicable a une equation *E*. Le predicat est faux si la règle n'est pas applicable. Toutes les règles ont été instanciées pour chacune des unifications possible.

Exemple d'un predicat *reduit*

```
1 reduit(expand, E, P, Q) :-  
2   split(E, X, T),  
3   X = T,  
4   delete_elem(E, P, Q)  
5   .
```

Listing 2 – Application de la reduction *expand*

Le prédicat *reduit* permet de d'appliquer la réduction approprié à l'équation *E*. Ici le prédicat *reduit* applique la transformation *expand* à l'équation *E*.

Unification

A la suite de l'implémentation de ces prédicats, nous avons mis en place le système de prédicat qui permet résoudre l'unificateur le plus général. Pour cela, nous avons implémenter le prédicat *unifie*. C'est ce prédicat qui fait le lien entre les prédicats de choix de transformation (*regle*) et ceux d'application de cette transformation.

```
1 unifie([E|P]) :-  
2   write("systeme: "),print([E|P]),nl,  
3   regle(E, R),  
4   write(R), write(": "), write(E), nl,  
5   reduit(R, E, P, Q),  
6   unifie(Q)  
7   .
```

Listing 3 – Prédicat d'unification d'un système d'équation

Question 2

La rapidité d'exécution de l'algorithme dépend du choix plus ou moins judicieux des règles à exécuter en premier sur le système d'équation. C'est pourquoi il faut mettre en place un système qui prenne en compte le l'ordre dans lequel les règles sont exécutés. Nous avons donc décidé d'implémenter quatre methode de choix :

- Choix du premier : la premiere règle possible est appliqué
- Choix pondéré : les règles sont évalué dans un ordre près établit qui est plus judicieux que l'ordre normal
- Choix aléatoire : les équations sont prise de façon aléatoire, on y applique la règle qui est possible
- Choix du dernier : la dernière équation possible est appliqué

Grâce à ces quatre possibilités, nous pouvons voir la difference d'exécution entre ces differents methodes de choix :

```

1  ?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], pondere).
2  systeme: [f(_G1769, _G1770, h(_G1766, _G1767)) ?= f(g(_G1773), h(a), h(e, r)), _G1773 ?= f(_G1770)]
3  decompose: f(_G1769, _G1770, h(_G1766, _G1767)) ?= f(g(_G1773), h(a), h(e, r))
4  systeme: [_G1769 ?= g(_G1773), _G1770 ?= h(a), h(_G1766, _G1767) ?= h(e, r), _G1773 ?= f(_G1770)]
5  decompose: h(_G1766, _G1767) ?= h(e, r)
6  systeme: [_G1766 ?= e, _G1767 ?= r, _G1769 ?= g(_G1773), _G1770 ?= h(a), _G1773 ?= f(_G1770)]
7  simplify: _G1766 ?= e
8  systeme: [_G1767 ?= r, _G1769 ?= g(_G1773), _G1770 ?= h(a), _G1773 ?= f(_G1770)]
9  simplify: _G1767 ?= r
10 systeme: [_G1769 ?= g(_G1773), _G1770 ?= h(a), _G1773 ?= f(_G1770)]
11 expand: _G1769 ?= g(_G1773)
12 systeme: [_G1770 ?= h(a), _G1773 ?= f(_G1770)]
13 expand: _G1770 ?= h(a)
14 systeme: [_G1773 ?= f(h(a))]
15 expand: _G1773 ?= f(h(a))
16 X = g(f(h(a))),
17 Y = h(a),
18 C = e,
19 V = r,
20 Z = f(h(a)).

```

Listing 4 – Exemple d'exécution avec le choix pondéré

```

1  ?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], dernier).
2  systeme: [f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(_G1530), h(a), h(e, r)), _G1530 ?= f(_G1527)]
3  expand: _G1530 ?= f(_G1527)
4  systeme: [f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(f(_G1527)), h(a), h(e, r))]
5  decompose: f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(f(_G1527)), h(a), h(e, r))
6  systeme: [_G1526 ?= g(f(_G1527)), _G1527 ?= h(a), h(_G1523, _G1524) ?= h(e, r)]
7  decompose: h(_G1523, _G1524) ?= h(e, r)
8  systeme: [_G1523 ?= e, _G1524 ?= r, _G1526 ?= g(f(_G1527)), _G1527 ?= h(a)]
9  expand: _G1527 ?= h(a)
10 systeme: [_G1523 ?= e, _G1524 ?= r, _G1526 ?= g(f(h(a)))]
11 expand: _G1526 ?= g(f(h(a)))
12 systeme: [_G1523 ?= e, _G1524 ?= r]
13 simplify: _G1524 ?= r
14 systeme: [_G1523 ?= e]
15 simplify: _G1523 ?= e
16 X = g(f(h(a))),
17 Y = h(a),
18 C = e,
19 V = r,
20 Z = f(h(a)).

```

Listing 5 – Exemple d'exécution avec le choix du dernier d'abord

```

1  ?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], aleatoire).
2  systeme: [f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(_G1530), h(a), h(e, r)), _G1530 ?= f(_G1527)]
3  decompose: f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(_G1530), h(a), h(e, r))
4  systeme: [_G1526 ?= g(_G1530), _G1527 ?= h(a), h(_G1523, _G1524) ?= h(e, r), _G1530 ?= f(_G1527)]
5  decompose: h(_G1523, _G1524) ?= h(e, r)
6  systeme: [_G1523 ?= e, _G1524 ?= r, _G1526 ?= g(_G1530), _G1527 ?= h(a), _G1530 ?= f(_G1527)]
7  expand: _G1526 ?= g(_G1530)
8  systeme: [_G1523 ?= e, _G1524 ?= r, _G1527 ?= h(a), _G1530 ?= f(_G1527)]
9  simplify: _G1523 ?= e
10 systeme: [_G1524 ?= r, _G1527 ?= h(a), _G1530 ?= f(_G1527)]
11 expand: _G1530 ?= f(_G1527)
12 systeme: [_G1524 ?= r, _G1527 ?= h(a)]
13 expand: _G1527 ?= h(a)
14 systeme: [_G1524 ?= r]
15 simplify: _G1524 ?= r
16 X = g(f(h(a))),
17 Y = h(a),
18 C = e,
19 V = r,
20 Z = f(h(a)).

```

Listing 6 – Exemple d'exécution avec choix aléatoire

```

1  ?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], premier).
2  systeme: [f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(_G1530), h(a), h(e, r)), _G1530 ?= f(_G1527)]
3  decompose: f(_G1526, _G1527, h(_G1523, _G1524)) ?= f(g(_G1530), h(a), h(e, r))
4  systeme: [_G1526 ?= g(_G1530), _G1527 ?= h(a), h(_G1523, _G1524) ?= h(e, r), _G1530 ?= f(_G1527)]
5  expand: _G1526 ?= g(_G1530)
6  systeme: [_G1527 ?= h(a), h(_G1523, _G1524) ?= h(e, r), _G1530 ?= f(_G1527)]
7  expand: _G1527 ?= h(a)
8  systeme: [h(_G1523, _G1524) ?= h(e, r), _G1530 ?= f(h(a))]
9  decompose: h(_G1523, _G1524) ?= h(e, r)
10 systeme: [_G1523 ?= e, _G1524 ?= r, _G1530 ?= f(h(a))]
11 simplify: _G1523 ?= e
12 systeme: [_G1524 ?= r, _G1530 ?= f(h(a))]
13 simplify: _G1524 ?= r
14 systeme: [_G1530 ?= f(h(a))]
15 expand: _G1530 ?= f(h(a))
16 X = g(f(h(a))),
17 Y = h(a),
18 C = e,
19 V = r,
20 Z = f(h(a)).

```

Listing 7 – Exemple d'exécution avec comme choix le premier

Question 3

```
1  ?- trace_unif([X ?= e, u ?= I, f(a) ?= g(Y)], aleatoire).
2  system: [_G191?=e,u?=_G198,f(a)?=g(_G205)]
3  simplify: _G191?=e
4  system: [u?=_G198,f(a)?=g(_G205)]
5  orient: u?=_G198
6  system: [_G198?=u,f(a)?=g(_G205)]
7  simplify: _G198?=u
8  system: [f(a)?=g(_G205)]
9  clash: f(a)?=g(_G205)
10 No
11 true.
```

Listing 8 – Exemple d'exécution avec choix aléatoire

```
1  ?- trace_unif([X ?= e, u ?= I, f(a) ?= g(Y)], premier).
2  system: [_G191?=e,u?=_G198,f(a)?=g(_G205)]
3  simplify: _G191?=e
4  system: [u?=_G198,f(a)?=g(_G205)]
5  orient: u?=_G198
6  system: [_G198?=u,f(a)?=g(_G205)]
7  simplify: _G198?=u
8  system: [f(a)?=g(_G205)]
9  clash: f(a)?=g(_G205)
10 No
11 true.
```

Listing 9 – Exemple d'exécution avec choix aléatoire

```
1  ?- trace_unif([X ?= e, u ?= I, f(a) ?= g(Y)], pondere).
2  system: [_G191?=e,u?=_G198,f(a)?=g(_G205)]
3  clash: f(a)?=g(_G205)
4  No
5  true.
```

Listing 10 – Exemple d'exécution avec choix aléatoire