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 decider 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 travaille et d'emprunter un processus de developpement iteratif.

Exemple d'un prédicat *regle*

```
regle(E, simplify):-
  split(E, L, R),
  not(var(R)),
  var(L),
  not(compound(R))
.
```

Listing 1 – Choix de la regle $simplify$

Ici nous avons le predicat regle pour l'unification $simplify$, le but de ce predicat est de revoyer vrai si simplify est applicable a une equation E. Le predicat est faux si la regle n'est pas applicable.
Toutes les regles on été instanciés pour chacunes des unifications possible.

Exemple d'un predicat *reduit*

```
reduit(expand, E, P, Q) :-
  split(E, X, T),
  X = T,
  delete_elem(E, P, Q)
.
```

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'equation 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.

```
unifie([E|P]) :-
write("systeme: "),print([E|P]),nl,
regle(E, R),
write(R), write(": "), write(E), nl,
reduit(R, E, P, Q),
unifie(Q)
.
```

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'execution entre ces differents methodes de choix :

```
?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], pondere).
systeme: [f(_G1769,_G1770,h(_G1766,_G1767))?=f(g(_G1773),h(a),h(e,r)),_G1773?=f(_G1770)]
decompose: f(_G1769,_G1770,h(_G1766,_G1767))?=f(g(_G1773),h(a),h(e,r))
systeme: [_G1769?=g(_G1773),_G1770?=h(a),h(_G1766,_G1767)?=h(e,r),_G1773?=f(_G1770)]
decompose: h(_G1766,_G1767)?=h(e,r)
systeme: [_G1766?=e,_G1767?=r,_G1769?=g(_G1773),_G1770?=h(a),_G1773?=f(_G1770)]
simplify: _G1766?=e
systeme: [_G1767?=r,_G1769?=g(_G1773),_G1770?=h(a),_G1773?=f(_G1770)]
simplify: _G1767?=r
systeme: [_G1769?=g(_G1773),_G1770?=h(a),_G1773?=f(_G1770)]
expand: _G1769?=g(_G1773)
systeme: [_G1770?=h(a),_G1773?=f(_G1770)]
expand: _G1770?=h(a)
systeme: [_G1773?=f(h(a))]
expand: _G1773?=f(h(a))
X = g(f(h(a))),
Y = h(a),
C = e,
V = r,
Z = f(h(a)).
```

Listing 4 – Exemple d'execution avec le choix pondéré

```
?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], dernier).
systeme: [f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(_G1530),h(a),h(e,r)),_G1530?=f(_G1527)]
expand: _G1530?=f(_G1527)
systeme: [f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(f(_G1527)),h(a),h(e,r))]
decompose: f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(f(_G1527)),h(a),h(e,r))
systeme: [_G1526?=g(f(_G1527)),_G1527?=h(a),h(_G1523,_G1524)?=h(e,r)]
decompose: h(_G1523,_G1524)?=h(e,r)
systeme: [_G1523?=e,_G1524?=r,_G1526?=g(f(_G1527)),_G1527?=h(a)]
expand: _G1527?=h(a)
systeme: [_G1523?=e,_G1524?=r,_G1526?=g(f(h(a)))]
expand: _G1526?=g(f(h(a)))
systeme: [_G1523?=e,_G1524?=r]
simplify: _G1524?=r
systeme: [_G1523?=e]
simplify: _G1523?=e
X = g(f(h(a))),
Y = h(a),
C = e,
V = r,
Z = f(h(a)).
```

Listing 5 – Exemple d'execution avec le choix du dernier d'habord

```
?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], aleatoire).
systeme: [f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(_G1530),h(a),h(e,r)),_G1530?=f(_G1527)]
decompose: f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(_G1530),h(a),h(e,r))
systeme: [_G1526?=g(_G1530),_G1527?=h(a),h(_G1523,_G1524)?=h(e,r),_G1530?=f(_G1527)]
decompose: h(_G1523,_G1524)?=h(e,r)
systeme: [_G1523?=e,_G1524?=r,_G1526?=g(_G1530),_G1527?=h(a),_G1530?=f(_G1527)]
expand: _G1526?=g(_G1530)
systeme: [_G1523?=e,_G1524?=r,_G1527?=h(a),_G1530?=f(_G1527)]
simplify: _G1523?=e
systeme: [_G1524?=r,_G1527?=h(a),_G1530?=f(_G1527)]
expand: _G1530?=f(_G1527)
systeme: [_G1524?=r,_G1527?=h(a)]
expand: _G1527?=h(a)
systeme: [_G1524?=r]
simplify: _G1524?=r
X = g(f(h(a))),
Y = h(a),
C = e,
V = r,
Z = f(h(a)).
```

Listing 6 – Exemple d'execution avec choix aléatoire

```
?- unifie([f(X, Y, h(C, V)) ?= f(g(Z), h(a), h(e, r)), Z ?= f(Y)], premier).
systeme: [f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(_G1530),h(a),h(e,r)),_G1530?=f(_G1527)]
decompose: f(_G1526,_G1527,h(_G1523,_G1524))?=f(g(_G1530),h(a),h(e,r))
systeme: [_G1526?=g(_G1530),_G1527?=h(a),h(_G1523,_G1524)?=h(e,r),_G1530?=f(_G1527)]
expand: _G1526?=g(_G1530)
systeme: [_G1527?=h(a),h(_G1523,_G1524)?=h(e,r),_G1530?=f(_G1527)]
expand: _G1527?=h(a)
systeme: [h(_G1523,_G1524)?=h(e,r),_G1530?=f(h(a))]
decompose: h(_G1523,_G1524)?=h(e,r)
systeme: [_G1523?=e,_G1524?=r,_G1530?=f(h(a))]
simplify: _G1523?=e
systeme: [_G1524?=r,_G1530?=f(h(a))]
simplify: _G1524?=r
systeme: [_G1530?=f(h(a))]
expand: _G1530?=f(h(a))
X = g(f(h(a))),
Y = h(a),
C = e,
V = r,
Z = f(h(a)).
```

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

Question 3

```
?- trace_unif([a ?= U, f(X, Y) ?= f(g(Z ),h(a )), Z ?= f(Y )], premier).
system: [a?=_G192,f(_G197,_G198)?=f(g(_G200),h(a )),_G200?=f(_G198)]
orient: a?=_G192
system: [_G192?=a,f(_G197,_G198)?=f(g(_G200),h(a )),_G200?=f(_G198)]
simplify: _G192?=a
system: [f(_G197,_G198)?=f(g(_G200),h(a )),_G200?=f(_G198)]
decompose: f(_G197,_G198)?=f(g(_G200),h(a ))
system: [_G197?=g(_G200),_G198?=h(a),_G200?=f(_G198)]
expand: _G197?=g(_G200)
system: [_G198?=h(a ),_G200?=f(_G198)]
expand: _G198?=h(a )
system: [_G200?=f(h(a ))]
expand: _G200?=f(h(a ))
Yes
U = a,
X = g(f(h(a ))),
Y = h(a ),
Z = f(h(a )).
```

Listing 8 – Exemple d'execution avec un niveau de debug

```
unif([a ?= U, f(X, Y) ?= f(g(Z),h(a)), Z ?= f(Y)], premier).
U = a,
X = g(f(h(a))),
Y = h(a),
Z = f(h(a)).
```

Listing 9 – Exemple d'execution