Opérations élémentaires

Plan

- **Introduction**
- **Substitution**
  - Définitions
  - Exemples
  - Composition de substitutions
- **Unification**
  - Définitions
  - Exemples
  - Algorithmes
- **Règle de résolution**
  - Résolution dans L0
  - Formes clausales
  - Résolution dans L1

Introduction

Problématique

Soient les fbfs suivantes :

$\forall x \ (P(x) \Rightarrow Q(x))$ et $P(a)$

Question : comment déduire $Q(a)$?

1) Il faut d'abord appliquer la règle de spécialisation universelle : ceci suppose que l'on sache par quelle constante remplacer x, puis comment effectuer ce remplacement
2) Il faut ensuite inférer le résultat conjecturé, par application d'une règle d'inférence (ici le modus ponens)

Plus généralement, les "techniques" mises à contribution seront : *la substitution, l'unification et la règle de résolution*

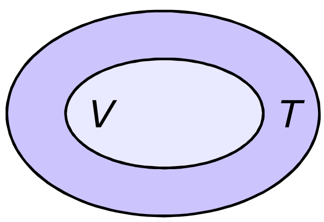
Substitution

Définitions

Notion de substitution

Une substitution de variables σ est une application de V dans T qui est l'application identique, sauf en un nombre fini de variables
$\sigma = \{ < v_1 . t_1 > ; < v_2 . t_2 > ; \ldots ; < v_n . t_n > \}$

- pour $i \neq j$, $v_i \neq v_j$
- $< v_i . t_i >$ est une liaison sur la variable v_i
- si aucun t_i ne comporte de variable, σ est une substitution *concrète*
- la substitution *identique* est un ensemble vide de liaisons. On la note ε
- une substitution est *pure* si les termes t_i sont libres des variables v_i
- *V* ensemble des variables,
  T ensemble des termes



Définitions

Substitution d'un terme

$\bar{\sigma}$ prolongement de σ à T est définie inductivement par :

– $\bar{\sigma}(c) = c$ où c constante

– $\bar{\sigma}(x) = \sigma(x)$ où x variable

– $\bar{\sigma}(f(t_1, t_2, \ldots, t_n)) = f(\bar{\sigma}(t_1), \bar{\sigma}(t_2), \ldots, \bar{\sigma}(t_n))$

Le résultat $\bar{\sigma}(t)$ de l'application de σ à un terme t est appelé *instance* de t

Substitution d'une matrice

Une matrice M est une formule bien formée ne comportant pas de quantificateur

L'application de la substitution σ à la matrice M est la matrice obtenue en remplaçant chaque terme rencontré dans M par son instance par σ

Définitions

Substitution d'une fbf

On peut étendre la notion d'instance à une fbf comportant des quantificateurs, à condition de ne substituer que les variables libres. De plus, les termes substitués ne doivent pas contenir de variables liées dans la fbf initiale

- *Exemple :*

  $A(x) = P(x) \vee \forall y\, \exists x\, Q(x,y)$

  $t = f(y,u)$

  $A(t) = P(f(y,u)) \vee \forall z\, \exists v\, Q(v,z)$

- *Lien avec la sémantique :*

  Une fbf est valide ssi toutes ses instances sont valides

  Une fbf est satisfiable ssi une au moins de ses instances est satisfiable

Définitions

Variantes alphabétiques

Deux matrices E et F sont des variantes alphabétiques s'il existe des substitutions σ et θ telles que $E = \sigma(F)$ et $F = \theta(E)$

- *Exemple :*

  $P(f(x,y),g(z),A)$ variante de $P(f(y,x),g(u),A)$

  $P(x,x)$ n'est pas une variante de $P(x,y)$

- *Renommage :*

  Une substitution $\sigma = \{< v_1 \,.\, t_1 >;\ldots < v_n \,.\, t_n >\}$ est un renommage si les t_i sont des variables

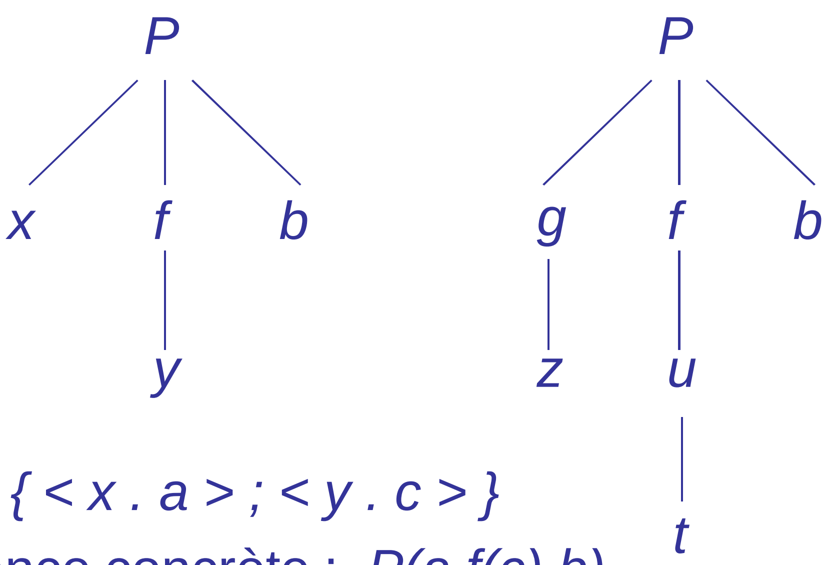
- *Propriété :*

  Soient E et F des variantes alphabétiques. Alors il existe deux renommages σ et θ tels que $E = \sigma(F)$ et $F = \theta(E)$

Exemples

Soit *P(x,f(y),b)*

- $\sigma = \{ < x \,.\, z > ; < y \,.\, w > \}$
  Instance et variante : *P(z,f(w),b)*
- $\sigma = \{ < y \,.\, a > \}$ Instance : *P (x,f(a),b)*
- $\sigma = \{ < x \,.\, g(z) > < y \,.\, u(t) > \}$
  Instance et variante : *P(g(z),f(u(t)),b)*



- $\sigma = \{ < x \,.\, a > ; < y \,.\, c > \}$
  Instance concrète : *P(a,f(c),b)*
- $\sigma = \{ < x \,.\, y > ; < y \,.\, x > \}$
  Instance et variante : *P(y, f(x),b)*

Composition de substitutions

Définition de la composée

Soit σ et θ deux substitutions. Alors la composée de σ par θ est définie par :

pour tout terme t, $(\theta \text{ o } \sigma)(t) = \theta[\sigma(t)]$

Définition technique :

Soient $\theta = \{< v_1 . s_1 > ; \{< v_2 . s_2 > ; \ldots ; < v_m . s_m > \}$ et $\sigma = \{< u_1 . t_1 >; \ldots ; < u_n . t_n > \}$ deux substitutions. La composée $\theta \text{ o } \sigma$ de σ par θ est la substitution obtenue à partir de l'ensemble $\{ < u_1 . \theta(t_1) >; \ldots ; < u_n . \theta(t_n) >; < v_1 . s_1 >; \ldots ; < v_m . s_m > \}$ en supprimant toute liaison $< v_j . s_j >$ où $v_j \in \{u_1, \ldots, u_n\}$

Composition de substitutions

Exemples

$\{ < x\ .\ a > ; < y\ .\ b >; < z\ .\ c >\}\ o\ \{ < w\ .\ g(x, y) >\} =$

$\{ < x\ .\ a > ; < y\ .\ b > ; < z\ .\ c > ; < w\ .\ g(a,b) > \}$

$\{ < x\ .\ b > ; < z\ .\ f(x) > ; < v\ .\ h(d) > ; < w\ .\ f(x)> \}$

$o\ \ \{ < x\ .\ a > ; < v\ .\ g(x,z,u) > \} =$

$\{ < x\ .\ a >; < z\ .\ f(x) >; < v\ .\ g(b,f(x),u) > ; < w\ .\ f(x) > \}$

Propriétés

– La composition est associative, n'est pas commutative, admet ε *pour* élément neutre. L'ensemble Σ des substitutions muni de *o* constitue donc un demi-groupe non commutatif

– La composée de deux substitutions impures peut être pure, et celle de deux substitutions pures peut être impure

Unification

Unification

- **Terminologie**

  On rencontre notamment les termes :

  – pattern-matching

  – appariement de formes

  – mise en concordance, …

  – en logique : unification

- **Exemples**

  – Application d'une connaissance générale à un problème particulier : par exemple dans les systèmes experts

  – Utilisation de règles de réécriture :

  $(a + b)^2 \rightarrow a^2 + 2ab + b^2$

  $(x + 3)^2 \rightarrow$ ………..

  – Mise en concordance de deux formes : en imagerie par exemple

Unification

Définitions

Soit $E = \{ A_1 ; A_2 ; \ldots ; A_n \}$ un ensemble fini de termes (ou d'atomes) de L1. Une substitution σ est un unificateur de E si : $\sigma(A_1) = \sigma(A_2) = \ldots = \sigma(A_n)$

Si E admet au moins un tel unificateur, on dit que E est unifiable, par σ

Cas de l'unification unidirectionnelle :

L'unification la plus simple est celle de deux atomes (ou termes) où il suffit d'appliquer la substitution à un seul d'entre eux. C'est le cas avec les règles de réécriture

Exemple : $P(u,v)$ et $P(a,b)$

Unification

Exemples plus complexes

- Unification de deux termes :

  f(x,a) et *f(g(b),y)*

  $\sigma = \{ < x \,.\, g(b) > , < y \,.\, a > \}$

  Instance commune : *f(g(b),a)*

- Unification avec partage de variable :

  f(x,a) et *f(a,x)*

  $\sigma = \{ < x \,.\, a > \}$

  Instance commune : *f(a,a)*

- Unification de deux termes :

  f(g(x),h(x)) et *f(y,h(a))*

  $\sigma = \{ < x \,.\, a > , < y \,.\, g(a) > \}$

  Instance commune : *f(g(a),h(a))*

- Pas d'unificateur :

  P(x,a) et *P(b,x)*

  P(u,f(u)) et *P(a,f(b))*

  x et *f(x,a)*

Unification

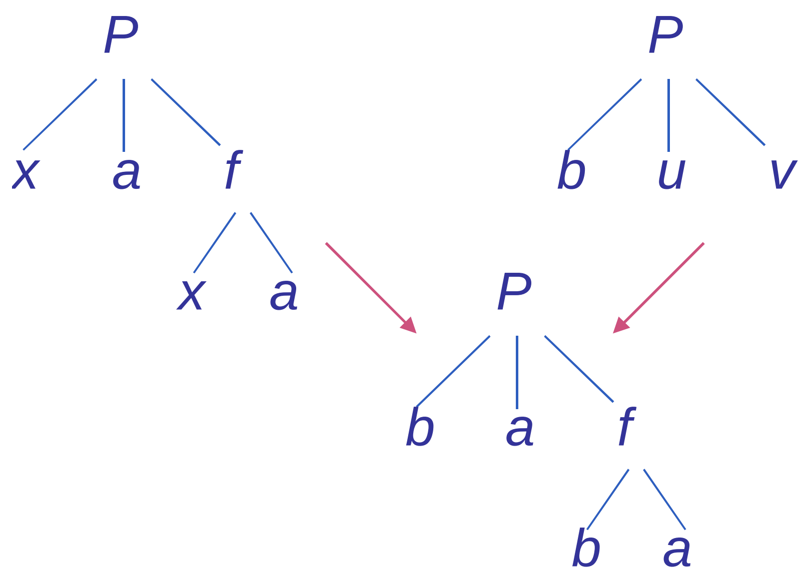
Exemples divers

- Unification de deux termes :

  P(x,a,f(x,a)) et *P(b,u,v)*

  $\sigma = \{ < x . b > , < u . a > , < v . f(b, a) > \}$

  Instance commune : *P(b,a,f(b,a))*



- Unification de plus de deux formules :

  P(x,f(y),g(b)), *P(x,f(c),g(z))* et *P(x,f(v),w))*

  $\sigma = \{<x . a><y . c><z . b><v . c><w . g(b) >\}$

  Instance commune : *P(a,f(c),g(b))*

Unification

Unificateur le plus général

Soit $E = \{ A_1 ; A_2 ; \ldots ; A_n \}$ un ensemble fini de termes (ou d'atomes) de L1. Une substitution σ est un unificateur le plus général (ou principal) de E si, pour tout autre unificateur σ' de E, il existe une substitution τ telle que $\sigma' = \tau \circ \sigma$

Exemple :

Soit $P(a,y,z)$ et $P(x,b,z)$

$\sigma_1 = \{< x . a > ; < y . b > ; < z . c >\}$

$\sigma_2 = \{< x . a > ; < y . b > ; < z . d >\}$

$\sigma_3 = \{< x . a > ; < y . b > ; < z . h(d) >\}$ ......

$\rightarrow \quad \sigma = \{< x . a > ; < y . b >\}$

$\sigma_1 = \{< z . c >\} \circ \sigma$

$\sigma_2 = \{< z . d >\} \circ \sigma$

$\sigma_3 = \{< z . h(d) >\} \circ \sigma$, ...

Instance commune la plus générale : $P(a,b,z)$

Unification

Théorème

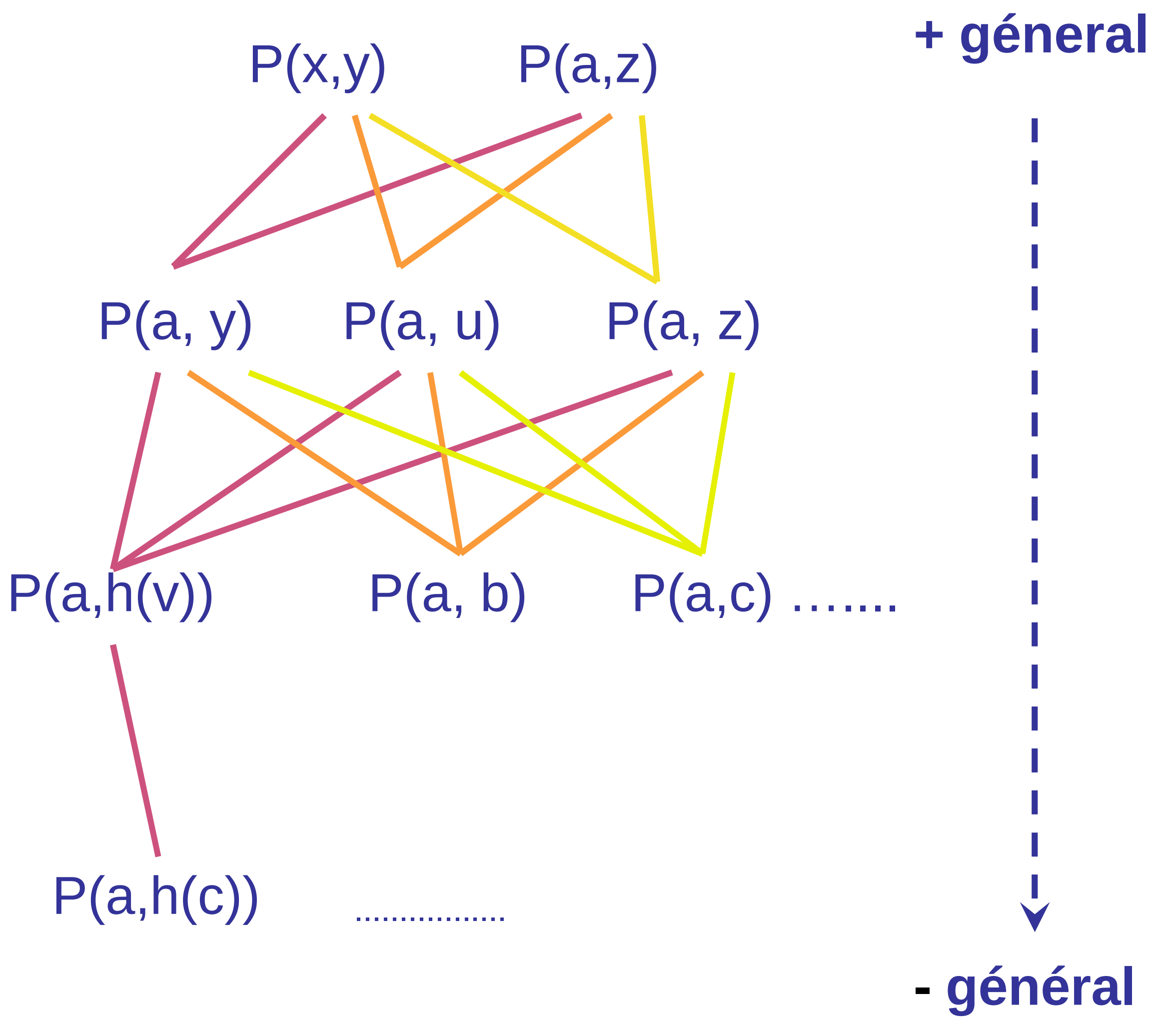
Si un ensemble *E* d'atomes est unifiable, alors il existe pour *E* un unificateur σ le plus général, noté *upg* ou *mgu*

- *Unicité :*
  L'unificateur le plus général est unique au renommage des variables près

- *Hiérarchie des unificateurs :*
  Soient deux atomes. Les unificateurs de niveau 1 s'expriment en fonction des *upgs* (niveau 0) ; ceux de niveau 2 s'expriment en fonction d'unificateurs de niveau 1 : et ainsi de suite…..

Unification

Exemple de hiérarchie



Unification

Définition alternative d'un upg

$$\mathrm{upg}(c_i,c_j) = \perp \quad ; \quad \mathrm{upg}(c_i,c_i) = T$$

$$\mathrm{upg}(x_i,x_i) = T \quad ; \quad \mathrm{upg}(T,T) = T$$

$$\mathrm{upg}(x,t) = \{ < x\ .\ t > \}$$

$$\mathrm{upg}(t,x) = \{ < x\ .\ t > \}$$

$$\mathrm{upg}(p(\ldots),q(\ldots)) = \perp$$

$$\mathrm{upg}(p(u_1,u_2,\ldots),q(v_1,v_2,\ldots)) =$$

$$\quad \mathrm{upg}(u_1,v_1) \oplus$$

$$\quad \mathrm{upg}(\mathrm{upg}(u_1,v_1)(p(u_2,\ldots),\mathrm{upg}(u_1,v_1)(q(v_2,\ldots)))$$

avec $\oplus$ défini par :

$$(a_1,a_2,\ldots) \oplus (b_1,b_2,\ldots) = (a_1,a_2,\ldots,b_1,b_2,\ldots)$$

$$T \oplus (a_1,a_2,\ldots) = (a_1,a_2,\ldots)$$

$$\perp \oplus (a_1,a_2,\ldots) = \perp$$

Unification

Algorithme de Robinson (1965)

σ substitution recherchée, A_1 et A_2 deux atomes à unifier

$\sigma \leftarrow \varnothing$ (substitution identique)

Tant que $\sigma(A_1) \neq \sigma(A_2)$ faire

- trouver le premier symbole de $\sigma(A_1)$ qui soit différent du symbole correspondant de $\sigma(A_2)$
- déterminer alors les termes respectifs t_1 et t_2 de $\sigma(A_1)$ et $\sigma(A_2)$ débutant à ce rang
- si t_1 et t_2 ne sont pas des variables : ECHEC
- si l'un des deux termes est une variable x contenue dans l'autre terme t : ECHEC #
- sinon, l'un des termes étant forcément une variable, soit x, et l'autre un terme t, composer σ avec $< x . t >$

Renvoi de σ

Unification

Exemple

Soient les littéraux $L_1 = P(x,a)$ et $L_2 = P(f(u),u)$, auxquels on applique l'algorithme de Robinson

Itération n°1 :

$\sigma(L_1) = P(x,a)$ et $\sigma(L_2) = P(f(u),u)$

Donc : $\sigma(L_1) \neq \sigma(L_2)$

Premiers symboles non concordants : *x* et *f*

Termes : *x et f(u)* Liaison : $< x \,.\, f(u) >$

$\sigma = \{ < x \,.\, f(u) > \}$

Itération n°2 :

$\sigma(L_1) = P(f(u),a)$ et $\sigma(L_2) = P(f(u),u)$

Donc : $\sigma(L_1) \neq \sigma(L_2)$

Premiers symboles non concordants : *a* et *u*

Termes : *a* et *u* Liaison : $< u \,.\, a >$

$\sigma = \{< u \,.\, a >\} \circ \{< x \,.\, f(u) >\} = \{< x \,.\, f(a) > ; < u \,.\, a >\}$

Fin, avec renvoi de :

$\sigma = \{< x \,.\, f(a) > ; < u \,.\, a >\}$

Unification

Propriété

Appliqué à deux atomes A_1 *et* A_2 unifiables, l'algorithme d'unification de Robinson termine et renvoie un unificateur le plus général

Commentaires

- Cet algorithme est le plus simple qui soit, fondé sur la construction itérative de l'upg
- Il est de complexité exponentielle (en taille des termes à unifier), en particulier à cause de l'étape de vérification d'occurrence (#)
- Il existe d'autres algorithmes d'unification aux performances meilleures, dont un algorithme récursif quasi linéaire (à l'étape # près)

Unification

Unification généralisée

On peut étendre, de diverses manières, l'algorithme précédent de façon à ce qu'il s'applique à n atomes, $n \geq 2$. Cette extension repose notamment sur le lemme suivant :

Soit $t_1, t_2, \ldots, t_n$ n termes, avec $n \geq 2$. Il existe une substitution σ telle que $\sigma(t_1) = \sigma(t_2) = \ldots = \sigma(t_n)$ ssi il existe σ_1 tel que $\sigma_1(t_1) = \sigma_1(t_2)$, et σ_2 tel que $\sigma_2(\sigma_1(t_1)) = \sigma_2(\sigma_1(t_2))$, et σ_3 … etc ……

Propriété

Soit *E* un ensemble fini de formules atomiques. Si *E* est unifiable, alors l'algorithme d'unification généralisée termine et donne un *upg* pour *E*. Si *E* n'est pas unifiable, alors l'algorithme termine en le déclarant

Règle de résolution dans L0 et dans L1

Résolution dans L0

Règle de résolution

Problématique :

Soit les formules propositionnelles :

$p \Rightarrow q$ et $\neg p \Rightarrow r$

Quelle inférence est-il légitime de faire ?

Enoncé de la règle :

Soit 2 clauses $\neg P \vee Q$ et $P \vee R$. On infère alors la clause $Q \vee R$, appelée *résolvant*

$$\frac{\neg P \vee Q \qquad P \vee R}{Q \vee R}$$

Cette règle, dont il existe d'autres présentations, remplace plusieurs règles d'inférence

Résolution dans L0

Forme clausale

Un ensemble de formules est une forme clausale si chaque formule est une clause. Une clause est une disjonction de littéraux

Forme clausale : { clause-1 ; …… ; clause-n }

Clause : { littéral-1 $\vee$ .... $\vee$ littéral-p }

Par convention : *{ }* est la clause vide, qui est toujours fausse

Toute formule de la Logique des Propositions peut être mise sous forme clausale

Exemple :

Formule de départ : *(P => Q) => R*

Forme clausale : *{ P $\vee$ R, ¬Q $\vee$ R }*

Résolution dans L1

Forme clausale dans L1

Toute formule du Calcul des Prédicats peut être mise sous forme clausale

Clausification :

A partir d'une fbf close, en 7 étapes :

1. Elimination des symboles $\Rightarrow$ et $\Leftrightarrow$
2. Réduction de la portée des négations
   → *par utilisation des lois de Morgan*
3. Standardisation des variables
4. Elimination des quantificateurs existentiels
   → *étape de Skolémisation*
5. Elimination des quantificateurs universels
6. Conversion en forme conjonctive
   → *utilisation des lois de distributivité*
7. Elimination des $\wedge$ et renommage

Résolution dans L1

A propos de la skolémisation

Il faut supprimer les quantificateurs existentiels.

- *Deux cas de figure :*

  $\exists x\, P(x) \rightarrow P(a)$ (a constante)

  $\forall x\, \exists y\, P(x,y) \rightarrow P(x,f(x))$ (f fonction)

- *Forme de Skolem :*

  La forme de Skolem A^S d'une fbf close A est obtenue en remplaçant chaque variable x quantifiée en $\exists$ par $f_{\mathbf{x}}(x_{i1}, x_{i2}, \ldots, x_{ip})$ où les variables x_{ij} sont quantifiées par des $\forall$ dont la portée inclut $\exists x$. Attention : elle n'est pas équivalente à la formule originelle !

- *Théorème de Skolem :*

  A est satisfiable ssi A^S est satisfiable

Résolution dans L1

Exemple de mise sous forme clausale

$\forall x\ \exists y\ [(P(x) \lor Q(y)) \Rightarrow R(x,y)]$

$\forall x\ \exists y\ [\neg(P(x) \lor Q(y)) \lor R(x,y)]$

$\forall x\ \exists y\ [(\neg P(x) \land \neg Q(y)) \lor R(x,y)]$

$\forall x\ (\neg P(x) \land \neg Q(f(x)) \lor R(x,f(x))$

$(\neg P(x) \land \neg Q(f(x)) \lor R(x,f(x))$

$(\neg P(x) \lor R(x,f(x))) \land (\neg Q(f(x)) \lor R(x,f(x)))$

D'où, après renommage, les clauses suivantes :

$\{ \neg P(x) \lor R(x,f(x)) \}$

$\{ (\neg Q(f(u)) \lor R(u,f(u)) \}$

Résolution dans L1

Résolution pleine et entière

Soient deux clauses parentes C_1 et C_2 :

$C_1 = P'_1 \vee \ldots \vee P'_m \vee Q'_1 \vee \ldots . \vee Q'_n$

$C_2 = \neg P''_1 \vee \ldots \vee \neg P''_q \vee R'' \vee \ldots . \vee R''_s$

S'il existe un upg σ tel que :

$\sigma(P'_1) = \ldots = \sigma(P'_m) = \sigma(P''_1) = \ldots = \sigma(P''_q)$

alors on infère le résolvant :

$\sigma(Q'_1) \vee \ldots . \vee \sigma(Q'_n) \vee \sigma(R''_1) \vee \ldots \vee \sigma(R''_s)$

avec standardisation à part

- *Standardisation à part* : renommage des variables du résolvant pour que ses variables soient différentes de celles des parents
- *Résolution binaire* : un seul littéral P'_i dans C_1 et un seul littéral $\neg P''_j$ dans C_2
- *Equivalence* : résolution binaire + règle de factorisation = "full" résolution

Résolution dans L1

Exemples

Où la résolution binaire suffit :

– $C_1 = \{p(x,f(x)) \vee q(x)\}$ $C_2 = \{\neg p(g(u),f(g(a)) \vee r(u)\}$
$\sigma = \{< x . g(a) >, < u . a >\}$ Rés : $\{q(a) \vee r(a)\}$

– $C_1 = \{p(x,a)\}$ et $C_2 = \{\neg p(y,y)\}$
$\sigma = \{< x . a) >, < y . a >\}$ Rés : $\{\ \}$

Avec la résolution pleine et entière :

$C_1 = \{p(x,a) \vee p(f(a),y) \vee q(x,y)\}$
et $C_2 = \{r(z) \vee \neg p(f(z),z)\}$

– binaire : $\sigma_1 = \{< x . f(a) >, < z . a >\}$
Rés : $\{p(f(a),y_1) \vee q(f(a),y_1) \vee r(a)\}$

– binaire : $\sigma_2 = \{< y . a >, < z . a >\}$
Rés : $\{p(x_1,a) \vee q(x_1,a) \vee r(a)\}$

– pleine et entière :
$\sigma_3 = \{< x . f(a) >, < y . a >, < z . a >\}$
Rés : $\{q(f(a),a) \vee r(a)\}$