Résolution et logique du premier ordre

Pour quoi faire ? Et comment ?

- Démontrer automatiquement des théorèmes
- Basé sur
  - Principe de résolution (Robinson, 1965)
- A conduit à
  - Langage de programmation logique Prolog (Colmerauer et équipe de l'université de Versailles)
- Principe
  - Montrer qu'une formule est un théorème en montrant qu'elle est valide
  - F est valide ssi **¬F** est inconsistante
- Pour montrer qu'une formule **F1** est inconsistante
  - La mettre sous forme d'une conjonction de clauses (I)
  - Par applications successives du principe de résolution (II), aboutir à la clause vide, indiquant l'inconsistance de F1
    - Cela passe par *l'unification* (III) de littéraux
  - C'est une procédure de preuve par *réfutation* pouvant être réalisée selon différentes stratégies (IV) plus ou moins efficaces

Clauses

- La forme clausale est souvent utilisée en démonstration automatique de théorèmes et en programmation logique
- Définition :
  - un **littéral** est un atome (= formule atomique), appelé littéral positif, ou la négation d'un atome, appelé littéral négatif
- Définition :
  - une **clause** est une formule close de la forme : $L_1 \vee L_2 \vee \ldots \vee L_m$
  - L_i est un littéral
- $x_1, \ldots, x_S$ sont des variables apparaissant dans $\forall x_1 \ldots \forall x_S \left(L_1 \vee L_2 \vee \ldots \vee L_m \right)$
- Si m=0, la clause est dite vide, notée $\lozenge$
- La clause vide est interprétée comme étant une formule toujours fausse, c'est-à-dire qu'elle est une formule insatisfiable

Notations

- Une clause $\forall x_1 \ldots \forall x_S \left(A_1 \vee \ldots \vee A_k \vee \neg B_1 \vee .. \vee \neg B_n \right)$ où $x_1, \ldots, x_S$ sont des variables , $A_1, \ldots, A_k, \neg B_1, .., \neg B_n$ sont des atomes, est équivalente à :

$$\boxed{\forall x_1 \ldots \forall x_S \left(B_1 \wedge .. \wedge B_n \rightarrow A_1 \vee \ldots \vee A_k \right)}$$

  - Et s'écrit : $\boxed{A_1, \ldots, A_k \leftarrow B_1, .., B_n}$
  - Les virgules dans $A_1, \ldots, A_k$ correspondent à des disjonctions
  - Les virgules dans $B_1, .., B_n$ correspondent à des conjonctions
  - Les A_i sont des conclusions de la clause
  - Les B_i sont des conditions

I. Mise sous forme normale

- Toute formule peut être mise sous forme normale

Deux formes normales sont requises par le processus de transformation :

  - la forme prénex
  - la forme normale conjonctive

- Exemple :

On considère la formule :

$$\forall x \left(\exists y P(x,y) \vee \neg \left(\exists y \left(\neg Q(x,y) \rightarrow R\left(f(x,y)\right)\right)\right)\right)$$

  - Cette formule est transformée en 8 étapes, tout d’abord sous forme prenex, puis sous forme normale conjonctive en utilisant les équivalences relatives aux quantificateurs et aux connecteurs

$$\forall x\,(\exists y P(x,y) \vee \neg(\exists y(\neg Q(x,y) \rightarrow R(f(x,y)))))$$

Etapes (1/3)

- Eliminer tous les symboles d'implication en utilisant l'équivalence :

$$\forall x(\exists y P(x,y) \vee \neg(\exists y(Q(x,y) \vee R(f(x,y)))))$$

- Réduire la portée des symboles de négation, de manière à ce qu'ils ne concernent qu'un symbole de prédicat/proposition. On utilise les équivalences

$$\neg(\exists x F(x)) \equiv \forall x \neg F(x), \neg(\neg F) \equiv F$$

et les lois de De Morgan (négation d'une conjonction ou d'une disjonction)

$$\forall x\left(\exists y P(x,y) \vee \forall y\left(\neg Q(x,y) \wedge \neg R(f(x,y))\right)\right)$$

- Renommer les variables dans la formule en utilisant les équivalences

$$\forall x F(x) \equiv \forall y F(y) \text{ et } \exists x F(x) \equiv \exists y F(y)$$

de manière à ce que chaque quantificateur ait ses propres variables

$$\forall x\left(\exists y P(x,y) \vee \forall z\left(\neg Q(x,z) \wedge \neg R(f(x,z))\right)\right)$$

Les formules dans lesquelles seulement les variables liées diffèrent sont appelées variantes

$$\forall x\left(\exists y P(x,y) \vee \forall z\left(\neg Q(x,z) \wedge \neg R\left(f(x,z)\right)\right)\right)$$

Etapes (2/3)

- Eliminer tous les quantificateurs existentiels. Toutes les variables x quantifiées existentiellement, qui ne se trouvent pas dans le champ d'un quantificateur universel, peuvent être remplacées par une constante c. On supprime alors le quantificateur existentiel.

$$\exists x P(x) \text{ donne } P(c)$$

- Si la variable y est quantifiée existentiellement et figure dans le champ de une ou plusieurs variables quantifiées universellement x_i, alors y est fonctionnellement dépendante des x_i et peut être remplacée par une fonction $g(x_1, x_2..., x_n)$ appelée fonction de Skolem

$$\forall x\left(P\left(x, g(x)\right) \vee \forall z\left(\neg Q(x,z) \wedge \neg R\left(f(x,z)\right)\right)\right)$$

$$\forall x\left(P(x,g(x))\vee\forall z\left(\neg Q(x,z)\wedge\neg R(f(x,z))\right)\right)$$

Etapes (3/3)

- Transformer la formule en forme normale prenex en plaçant les quantificateurs universels devant la formule (à cette étape, la portée des quantificateurs est toute la formule)

$$\forall x\forall z\left(P(x,g(x))\vee\left(\neg Q(x,z)\wedge\neg R(f(x,z))\right)\right)$$

- mettre la formule sous forme normale conjonctive en utilisant les lois de distributivité

$$\forall x\forall z\left(\left(P(x,g(x))\vee\neg Q(x,z)\right)\wedge\left(P(x,g(x))\vee\neg R\left(f(x,z)\right)\right)\right)$$

- négliger le préfixe dans la formule

$$\left(\left(P(x,g(x))\vee\neg Q(x,z)\right)\wedge\left(P(x,g(x))\vee\neg R\left(f(x,z)\right)\right)\right)$$

- Traduire la formule en un ensemble de clauses en remplaçant les formules de la forme $\mathrm{F}\wedge\mathrm{G}$ par un ensemble de clauses $\{\mathrm{F}',\mathrm{G}'\}$, où F’ et G’ indiquent que F et G sont maintenant représentées dans les notations de la programmation logique

$$\{P(x,g(x))\leftarrow Q(x,z), P(x,g(x))\leftarrow R(f(x,z))\}$$

Clause de Horn

- Une clause de Horn est une clause ayant l'une des formes suivantes :

  (1) $A \leftarrow$

  (2) $\leftarrow B_1, .., B_n, \; n \geq 1$

  (3) $A \leftarrow \; B_1, .., B_n, \; n \geq 1$

- A est quelquefois appelé **tête de la clause**, et $B_1, B_2, \ldots.Bn$ est appelé **corps de la clause**. Une clause de la forme (1) est appelée **clause unité**, une clause de la forme (2) **clause but**

- Les clauses de Horn sont employées dans le langage Prolog

Résolution et logique propositionnelle

- On considère un ensemble de formules S sous forme clausale. On veut montrer qu'une formule G sous forme clausale peut être dérivée de S : $S \supset G$
- Cela consiste à montrer que l'ensemble de clauses W, constitué de S et $\neg G$ est insatisfiable
  - On vérifie que W contient la clause vide $\Diamond$. Si oui, W est insatisfiable et donc $S \supset G$
  - Sinon, la règle de résolution est appliquée à deux clauses de W. On obtient une nouvelle clause qui est ajoutée à W. On recommence jusqu'à ce que l'on trouve la clause $\Diamond$
- Exemple : On considère l'ensemble suivant de clauses :

$$\{C_1 = P \vee R, C_2 = \neg P \vee Q\}$$

  - Ces clauses contiennent des littéraux complémentaires, c'est à dire des littéraux ayant des valeurs de vérité opposées, P et $\neg P$. En appliquant la résolution, on obtient une nouvelle clause $C_3 = R \vee Q$ qui est ajoutée à l'ensemble initial des clauses

Table de vérité de $((P \vee R) \wedge (\neg P \vee Q)) \supset (R \vee Q)$

A B

P	Q	R	$P \vee R$	$\neg P \vee Q$	$A \wedge B$	$R \vee Q$	F
1	1	1	1	1	1	1	1
1	1	0	1	1	1	1	1
1	0	1	1	0	0	1	1
1	0	0	1	0	0	0	1
0	1	1	1	1	1	1	1
0	1	0	0	1	0	1	1
0	0	1	1	1	1	1	1
0	0	0	0	1	0	0	1

II. Principe de résolution

- On considère les 2 clauses C_1 et C_2 contenant les littéraux L_1 et L_2 respectivement, où L_1 et L_2 sont complémentaires.

- La procédure de résolution est la suivante :
  - Effacer L_1 de C_1 et L_2 de C_2. On obtient les clauses C'_1 et C'_2
  - Former la disjonction C' de C'_1 et C'_2
  - Effacer les éventuels littéraux redondants de C'. On obtient ainsi la clause C.
  - La clause obtenue est appelée *résolvante* de C_1 et C_2. Les clauses C_1 et C_2 sont appelées clauses *parent* de la résolvante.

C'est une forme de résolution « basique » (concerne 2 clauses) appelée résolution binaire

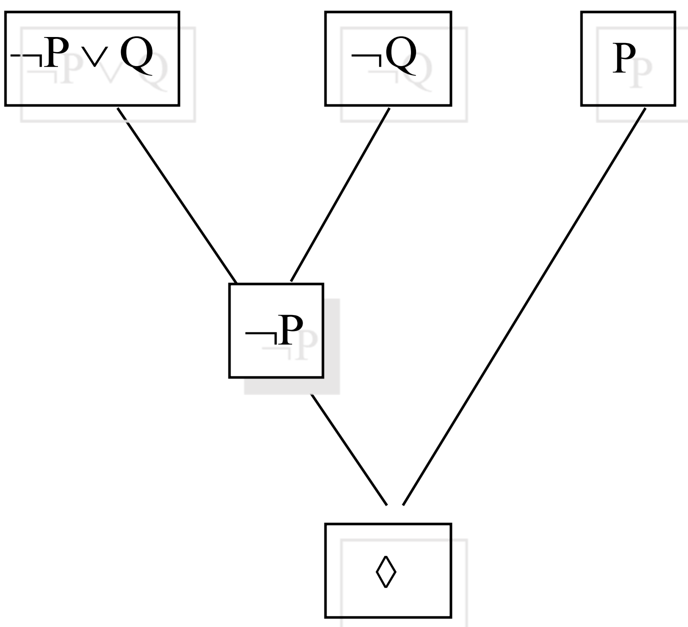
La résolution est une procédure d'inférence saine : deux clauses parent vraies donnent lieu à une résolvante vraie également

Réfutation

- Soit S un ensemble de clauses et C une clause unique.
- On veut montrer que $S \supset C$
- une dérivation de C à partir de E, constitué des clauses de S et $\neg C$,
  est une séquence finie de clauses C_1, C_2, …C_n, n ≥ 1 où chaque C_k est soit une clause de E, soit une résolvante de clauses parents C_i et C_j , i<k, j<k, i≠j, de la séquence et $C=C_n$
- Si $C_n = \lozenge$ alors la dérivation est appelée réfutation de E, indiquant que E est insatisfiable
- Exemple :
  - On considère l'ensemble suivant de clauses : $E = \{\neg P \vee Q, \neg Q, P\}$
  - De $C_1 = \neg P \vee Q$, de $C_2 = \neg Q$, on obtient la résolvante $C_3 = \neg P$
  - Des clauses C_3 et $C_4 = P$, on dérive $C_5 = \lozenge$
  - Donc E est insatisfiable
  - La séquence de clauses C_1, C_2, C_3, C_4, C_5 est une réfutation de E (parmi d'autres possibles)

Réfutation

- Une réfutation peut être décrite par un graphe décrivant les liens entre clauses parents et résolvantes. Lorsqu'un graphe est limité aux clauses ayant donné lieu à la clause vide, il est appelé graphe de réfutation.



III. Unification

- En logique des propositions, il est facile de calculer les clauses résolvantes :

  si on a P et ¬P, on fait la disjonction des littéraux restants.

- En logique des prédicats, si on veut comparer $\neg P(x)$ et $P(a)$, on utilise la substitution $\theta = \{x/a\}$ et on obtient $\neg P(a)$ et $P(a)$ qui deviennent complémentaires.

- L'algorithme d'unification est une méthode générale pour comparer les expressions. L'algorithme calcule les substitutions requises pour rendre les expressions syntaxiquement égales

Définitions

- Substitution
  - Une substitution σ est une liste de couples $\{x_i/t_i\}$
  - Si F est une formule bien formée, $F\sigma$ est obtenue en remplaçant dans F toutes les occurences libres de x_i par le terme t_i
- Produit de substitutions
  - Le produit de 2 substitutions θ et λ,est noté $\theta\ \lambda$
  - il consiste à appliquer θ puis λ à F
- Unificateur
  - Une substitution σ est appelée un **unificateur** si pour un ensemble donné d'expressions $\{E_1,\ldots,E_m\}$, $E_1\sigma = \ldots = E_m\sigma$, $m \geq 2$
  - Un ensemble d'expressions est **unifiable** s'il possède un unificateur
- Unificateur le plus général
  - Un unificateur θ d'un ensemble d'expressions unifiables $E= \{E_1,\ldots,E_m\}$, est dit **unificateur le plus général** (pgu) si pour chaque unificateur σ de E, il existe une substitution λ telle que $\sigma= \theta\ \lambda$
  - Un pgu est unique

Produit de substitutions

- Considérons $\theta=\{x_1/t_1, x_2/t_2, \ldots\}$ et $\lambda=\{y_1/v_1, y_2/v_2, \ldots\}$

- La substitution $\theta\,\lambda$ consiste à réaliser les substitutions $x_i/t_i\lambda$
  1. appliquer λ aux termes t dans θ
     - Les y_i dans chaque t sont remplacés par les v_i
  2. Ajouter à la substitution $\theta\,\lambda$ les y_i/v_i chaque fois que y_i ne sera pas égal à un x_i de θ

- Le produit de substitutions n'est pas commutatif, mais il est associatif : $F\theta\lambda=(F\theta)\lambda$

Algorithme d’unification de littéraux

- En entrée : les littéraux à unifier, un unificateur initialement vide
- Calcul en plusieurs étapes.

Procédure unifie(littéraux, unificateur vide)

- Si tous les littéraux sont égaux alors renvoyer l’unificateur. *stop*.
- Sinon calculer l’ensemble de discordance D

  (C’est l’ensemble des premières sous-formules qui diffèrent)

  - Si dans D, il y a une variable x et un terme t ne contenant pas x, remplacer dans les littéraux les occurrences de x par t
    - Ajouter x/t à l’unificateur
    - Unifier à nouveau les littéraux, sur la base de l’unificateur mis à jour (appel à unifie(littéraux à jour, unificateur à jour))
  - Sinon les littéraux ne sont pas unifiables
- Fin

Unificateur le plus général : exemple

- On considère l'expression {R(x,f(a,g(y)), R(b,f(z,w))}
- Des unificateurs possibles de cet ensemble sont :

  - $\sigma 1 = \{x/b$,z/a, w/g(c), y/c}
  - $\sigma 2$ ={x/b, z/a, y/f(d), w/g(f(d))}
  - $\sigma 3 = \{x/b\,, z/a\,, w/g(y)\}$

  $\sigma 3$ est le pgu :

  - $\sigma 3$ composé avec {y/c} donne $\sigma 1$.
  - $\sigma 3$ composé avec {y/f(d)} donne $\sigma 2$

Résolvante de clauses avec variables

- Considérons l'ensemble suivant de clauses :
$$C1 = \{P(x) \vee Q(x)\} \quad C2 = \{\neg P(f(y)) \vee R(y)\}$$
- Ces clauses ne contiennent pas de littéraux complémentaires. Mais les atomes $P(x)$ dans $C1$ et $\neg P(f(y))$ dans $C2$ sont unifiables
- Si on applique $\sigma = \{x/f(a), y/a\}$, on obtient :
$$C1\sigma = \{P(f(a)) \vee Q(f(a)\} \quad C2\sigma = \{\neg P(f(a)) \vee R(a)\}$$
- On en déduit la résolvante : $C3 = \{Q(f(a)) \vee R(a)\}$
- Il est important de renommer des variables de même nom dans des clauses différentes :
- <u>Exemple</u> : soit $Q(x, y)$ et $Q(x, f(y))$ apparaissant dans deux clauses différentes. L'unification n'est ici pas possible. Elle le devient en renommant dans $Q(x, f(y))$: x en « u » et $f(y)$ en « v ».

IV. Stratégies de résolution

- Elles déterminent l'efficacité de la résolution. L'algorithme le plus simple génère des clauses pouvant être redondantes

- Exemple :
  - On considère l'ensemble de clauses $S = \{P, \neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\}$ numérotées de 1 à 4
  - Si on utilise le principe de résolution en générant systématiquement tous les résolvants, les résolvants suivants sont successivement générés :

$$\boxed{\begin{array}{l}
(5)\, Q\ (1+2) \\
(6)\, \neg Q \vee R\ (1+3) \\
(7)\, \neg P \vee R\ (2+3) \\
(8)\, \neg P \vee \neg Q\ (3+4) \\
(9)\, R\ (1+7) \\
(10)\, \neg Q\ (1+8) \\
(11)\, \neg P \vee R\ (2+6) \\
(12)\, \neg P\ (2+8)
\end{array}}$$

Stratégies de résolution

$(13)\ \neg P \vee R\ (3+5)$

$(14)\ \neg Q\ (4+6)$

$(15)\ \neg P\ (4+7)$

$(16)\ R\ (5+6)$

$(17)\ \neg P\ (5+8)$

$(18)\ R\ (1+11)$

$(19)\ \Diamond\ (5+14)$ $\Rightarrow$ 15 résolvants réductibles à $\{2+3, 4+5, 1+6\}$

Résolution sémantique

- Désigne une classe de résolution dans lesquelles le processus de résolution est commandé par la sémantique des clauses à traiter, liée à une interprétation
- Dans une interprétation I, certaines clauses d'un ensemble E seront vraies, d'autres fausses. Comme $E = S \wedge \neg G$ est supposé insatisfiable, S et $\neg G$ ne seront pas toutes vraies. E est divisé en deux ensembles $E1$ et $E2$: $E1$ pour les clauses fausses, $E2$ pour les clauses vraies. Les clauses parents sont choisies dans $E1$ et $E2$.
- Exemple :
  - On considère l'ensemble insatisfiable de formules suivant :

  $E = \{P, \neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\}$
  - On considère l'interprétation I, définie par
    - I(P)=faux
    - I(Q)=faux
    - I(R)=faux
  - On divise l'ensemble E en deux sous-ensembles $E1$ et $E2$:

  $E1 = \{P\}\ E2 = \{\neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\}$
  - Seules les résolvantes $\{Q, \neg Q \vee R, R\}$ et $\lozenge$ seront générées

Améliorations

- Imposer un ordre entre les symboles des propositions

- Stratégie de l'ensemble de soutien :
  - Lorsque l'on veut prouver $S \supset G$, on utilise une réfutation à partir de
    $$W = S \sqcup \{\neg G\}$$
  - Dans cette stratégie, on utilise le fait que S est satisfiable pour diminuer le nombre de résolvants générés
  - W est divisé en :
    - S ensemble d'origine
    - T ensemble qui contient les clauses à prouver (ensemble de soutien)
  - A chaque étape de la résolution, au moins une clause doit faire partie de T
  - Chaque clause résolvante est ajoutée à T
  - C'est une stratégie saine et complète

On considère l' ensemble de clauses :

$W = \{P, \neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\}$

Le sous - ensemble suivant S de W est satisfiable :

$S = \{P, \neg P \vee Q, \neg P \vee \neg Q \vee R\}$

(ex : choisir une interprétation I dans laquelle I(P) = I(Q) = I(R)

= vrai). La clause restante de W constitue l' ensemble de soutien

$T = \{\neg R\}$; ainsi $S \cup T = W$

On numérote les expressions

(1) P

(2) $\neg P \vee \neg Q \vee R$

(3) $\neg P \vee Q$

(4) $\neg R$

La résolution utilisant cette stratégie génère successivement les résolvants :

(5) $\neg P \vee \neg Q$ $(2 + 4)$

(6) $\neg Q$ $(1 + 5)$

(7) $\neg P$ $(3 + 5)$

(8) $\Diamond$

Stratégies de résolution linéaire

- On parle de stratégie de résolution SLD (Sélection Linéaire Définie) : à chaque étape de la résolution, le dernier résolvant généré est considéré comme une clause parent. L'autre clause parent est une clause de S ou une clause résolvante

  - Stratégie linéaire par entrée : chaque étape est réalisée entre le dernier résolvant généré (clause but) et une clause de l'ensemble original des clauses (clauses d'entrée)

    C'est une résolution complète pour les clauses de Horn

Stratégie SLD pour les clauses de Horn (1/3)

Définition :

- Soit $\{Ci\}$ un ensemble de clauses de la forme $\{Ci \leftarrow B_1, \dots Bp, p \geq 0\}$ et soit G_0 une clause but de la forme $\{G_0 = \leftarrow A_1, \dots A_q, q \geq 0\}$.
- Une dérivation SLD est une séquence finie ou infinie de clauses but G_0 , G_1…. ; une séquence C_1 , C_2…. de variantes de clauses d'entrée, et une séquence Θ_1 , Θ_2…. de pgu tels que G_{i+1} est dérivée de G_i =$\leftarrow A_1, \dots Ak$ et C_{i+1} en utilisant Θ_{i+1} si les conditions suivantes sont remplies :

1. A_j est l'atome de la clause but G_i choisi par la règle de sélection comme étant l'atome à résoudre
2. C_{i+1} est une clause d'entrée de la forme : $C_{i+1} = \mathrm{B} \leftarrow B_1, \dots Bp$ (dans laquelle éventuellement des variables ont été renommées), telle que $A_j\ \Theta_{i+1}$ = $B\ \Theta_{i+1}$, où Θ_{i+1} est le pgu de A_j et B

Stratégie SLD pour les clauses de Horn (2/3)

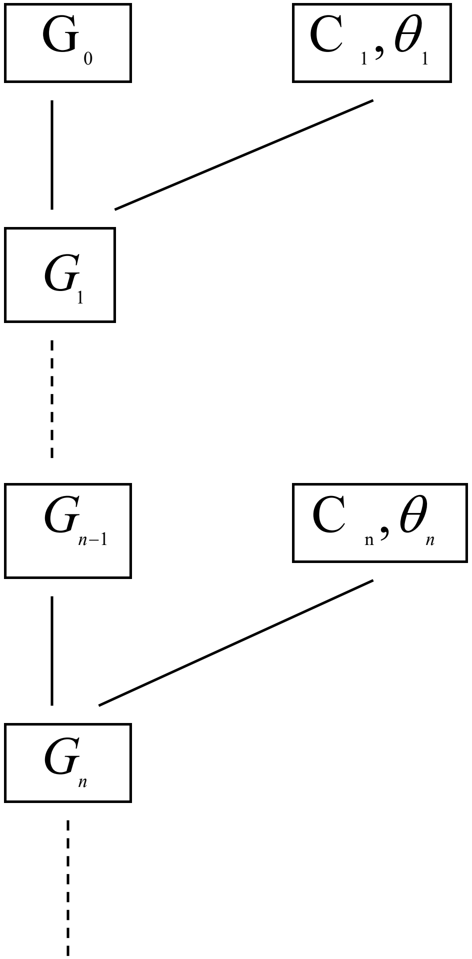
3. G_{i+1} est la clause

$$G_{i+1} = (A_1, \ldots Aj_{-1}, B_1, \ldots, B_p, A_{j+1, \ldots} A_k)\Theta_{i+1}$$

S'il existe $n, n \geq 0, Gn = \Diamond$ alors la dérivation est appelée réfutation SLD et le nombre n est appelé longueur de la réfutation.

Stratégie SLD pour les clauses de Horn (3/3)

Forme générale



Exemple (1/2)

- On considère l'ensemble de clauses de Horn suivant :

$$\{R(g(x)) \longleftarrow T(x, y, f(x)), T(a, b, f(a)), P(v, w) \longleftarrow R(v)\}$$

- On considère la clause but suivante :

$\leftarrow P(u, b)$

L'ensemble de clauses obtenu par ajout de la clause but à l'ensemble original de clauses est insatisfiable. Peut être prouvé par la stratégie de résolution SLD.

Exemple (2/2)

$\neg P(u, b)$ $\qquad$ $P(v, w) \longleftarrow R(v)$ {u/v, w/b}

$\neg R(u)$ $\qquad$ $R(g(x)) \longleftarrow T(x, y, f(x))${u/g(x)}

$\neg T(x, y, f(x))$ $\qquad$ $T(a, b, f(a))${x/a, y/b}

$\Diamond$