Réécriture

La résolution

Il faut tenir à une résolution parce qu'elle est bonne,

et non parce qu'on l'a prise.

La Rochefoucauld

version du 13 décembre 2004 – 15 h 23

La résolution en calcul propositionnel

La règle cut

En calcul des séquents, la coupure (ou cut) est la règle

$$\frac{A, \Gamma \vdash \Delta \qquad \Xi \vdash A, \Theta}{\Gamma, \Xi \vdash \Delta, \Theta}$$

ce qui donne aussi

$$\frac{A, A_1, \ldots, A_m \vdash B_1, \ldots, B_n \qquad C_1, \ldots, C_p \vdash A, D_1, \ldots, D_q}{A_1, \ldots, A_m, C_1, \ldots, C_p \vdash B_1, \ldots, B_n, D_1, \ldots, D_q}$$

Interprétation des séquents

Le séquent

$$A_1, \ldots, A_m \vdash B_1, \ldots, B_n$$

s'interprète par

$$A_1 \wedge \ldots \wedge A_m \Rightarrow B_1 \vee \ldots \vee B_n$$

ou encore

$$\neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n$$

Introduction à la résolution en calcul propositionnel

Si on récrit cut dans cette interprétation

$$\frac{\neg A \vee \neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n \qquad A \vee \neg C_1 \vee \neg \ldots \vee \neg C_p \vee D_1 \ldots \vee D_q}{\neg A_1, \ldots \vee \neg A_m \vee C_1, \ldots \vee C_p \vee B_1 \vee \ldots \vee B_n \vee D_1 \vee \ldots \vee D_q}$$

ou encore si Φ et Ψ sont des disjonctions de propositions

$$\frac{\neg A \vee \Phi \qquad A \vee \Psi}{\Phi \vee \Psi}$$

Introduction à la résolution en calcul propositionnel

Si on récrit cut dans cette interprétation

$$\frac{\neg A \vee \neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n \qquad\qquad A \vee \neg C_1 \vee \neg \ldots \vee \neg C_p \vee D_1 \ldots \vee D_q}{\neg A_1, \ldots \vee \neg A_m \vee C_1, \ldots \vee C_p \vee B_1 \vee \ldots \vee B_n \vee D_1 \vee \ldots \vee D_q}$$

ou encore si Φ et Ψ sont des conjonctions de propositions

$$\frac{\neg A \vee \Phi \qquad\qquad A \vee \Psi}{\Phi \vee \Psi}$$

La spécificité de la résolution (en calcul propositionnel) est que tous les A_i sont des variables propositionnelles.

La résolution en calcul propositionnel

La résolution

$$\frac{\neg A \vee \Phi \qquad\qquad A \vee \Psi}{\Phi \vee \Psi}$$

est apparentée à notre bon vieux modus ponens

car on peut l'écrire

$$\frac{A \Rightarrow \Phi \qquad\qquad A \vee \Psi}{\Phi \vee \Psi}$$

Formes clausales

Pour la résolution on travaille sur les propositions mises

sous forme normale conjonctive,

c'est-à-dire sous la forme de conjonction de disjonctions de littéraux.

Un littéral est

– soit une variable propositionnelle,

– soit la négation d'une variable propositionnelle.

Formes clausales

Une forme clausale est donc une expression de la forme

$$\bigwedge_{i=1}^{n} (\bigvee_{j=1}^{p_i} A_{i\,j} \vee \bigvee_{j=1}^{q_i} \neg B_{i\,j})$$

où les $A_{i\,j}$ et les $B_{i\,j}$ sont des variables propositionnelles.

$A_{i\,j}$ est un littéral positif. $\neg B_{i\,j}$ est un littéral négatif.

Chaque $\bigvee_{j=1}^{p_i} A_{i\,j} \vee \bigvee_{j=1}^{q_i} \neg B_{i\,j}$ est appelé une clause.

Une forme clausale est donc vue comme un ensemble de clauses.

De même, une clause peut-être vue comme un ensemble de littéraux.

La résolution

Une étape de résolution consiste donc

– à identifier un littéral A dans une clause $A \vee \Phi$

– à identifier son opposé $\neg A$ dans une clause $\neg A \vee \Psi$

et à fabriquer un résolvant $\Phi \vee \Psi$.

Exemples de résolution

La clause $\{A, B\}$ et la clause $\{\neg A, \neg B\}$ ont deux résolvants

$\{A, \neg A\}$

et $\{B, \neg B\}$.

La clause $\{C\}$ et la clause $\{\neg C\}$ ont un seul résolvant

la clause vide que l'on note $\Box$.

Exemples de résolution

La clause $\{A, B\}$ et elle-même ont deux résolvants :

$$\{A, \neg A\}$$

et $$\{A, \neg A\}$$

Exemples de résolution

La clause $\{A, \neg A\}$ et elle-même ont deux résolvants :

$$\{A, \neg A\}$$

et $\{A, \neg A\}$

On n'a pas beaucoup progressé !

La procédure de résolution

La procédure de résolution consiste à enrichir un ensemble de clauses par leurs résolvants jusqu'à ce qu'on engendre la clause vide.

Quand on ajoute le résolvant de Φ et Ψ,
on ne supprime ni Φ ni Ψ.

En effet, après résolution, on doit obtenir un ensemble de clauses logiquement équivalent.

Preuve par réfutation

Quand on veut prouver une proposition,

- on prend sa négation
- on la met sous forme clausale,
- on prouve par résolution que cette forme clausale est contradictoire.

Preuve par réfutation et par résolution

donnée : S un ensemble de clauses

répéter

choisir Φ_1 et Φ_2 qui ont un résolvant Ψ.

$S := S \cup \{\Psi\}$

jusqu'à $\Psi = \Box$

Preuve par réfutation et par résolution

La mise en œuvre de la procédure de choix est une clé de l'implantation.

Cela correspond au parcours «astucieux» d'un graphe.

Le graphe de résolution

Le graphe de résolution d'un ensemble $\mathcal{E}$ de clauses est un graphe orienté sans cycle (DAG) étiqueté par des clauses

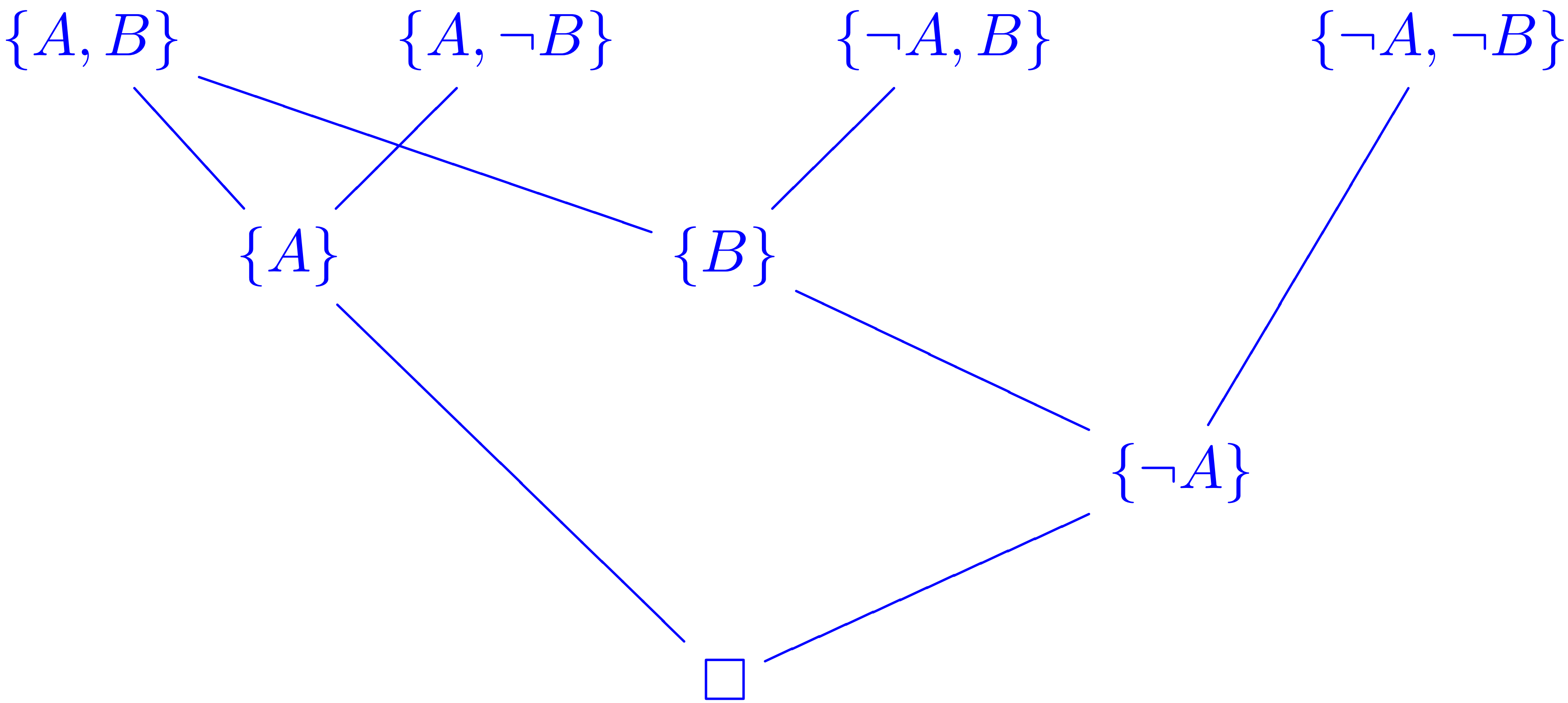
– dont les étiquettes des feuilles sont les clauses de $\mathcal{E}$,

– chaque nœud est étiqueté par une clause Φ et est issu de deux autres nœuds étiquetés par deux clauses Φ_1 et Φ_2, et Φ est un résolvant de Φ_1 et Φ_2.

Si la racine du graphe sans cycle est étiquetée par la clause vide $\square$

on dit qu'il s'agit d'une réfutation par résolution de l'ensemble $\mathcal{E}$.

Un exemple de graphe de résolution



Insatisfiabilité

Une formule φ est insatisfiable s'il n'existe aucune interprétation $\mathfrak{I}$ telle que $\mathfrak{I} \models_K \varphi$

Correction

Si un ensemble $\mathcal{E}$ de clauses admet une réfutation par résolution alors il est insatisfiable

Complétude

Si une ensemble $\mathcal{E}$ de clauses est insatisfiable alors il admet une réfutation par résolution.

Complétude

Si une ensemble $\mathcal{E}$ de clauses est insatisfiable alors il admet une réfutation par résolution.

Idée de la démonstration :

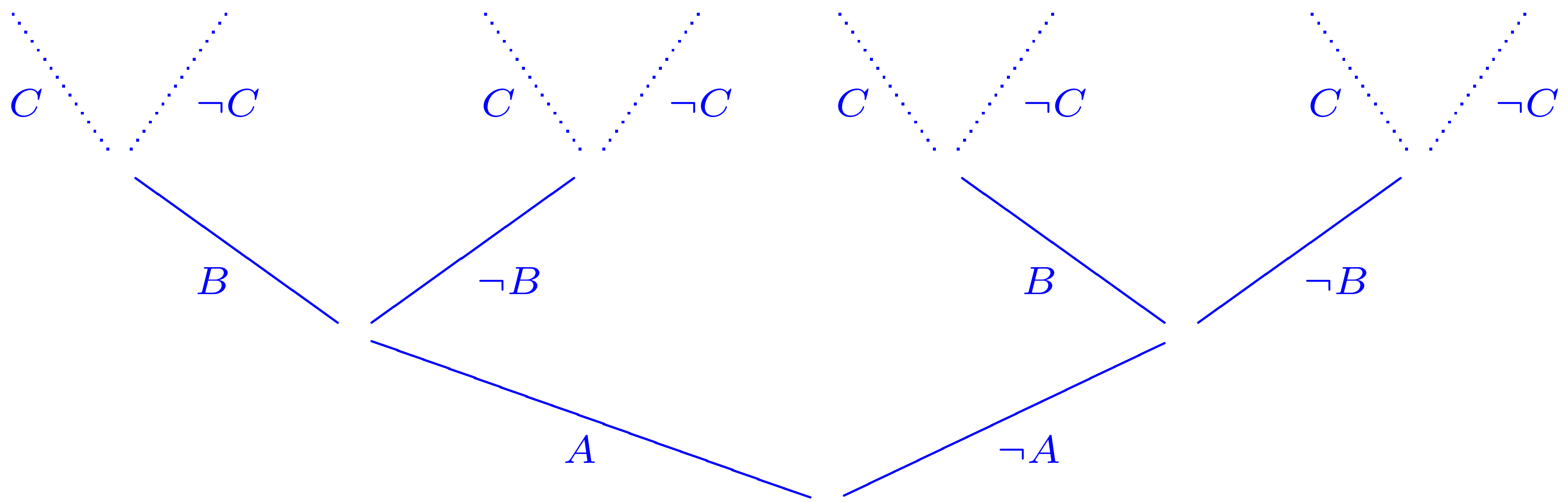
On note l'analogie entre un séquent

$$A_1, \ldots, A_m \vdash B_1, \ldots, B_n$$

où toutes les propositions qui interviennent sont des variables propositionnelles, et la clause

$$\neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots, \vee B_n$$

Considérons l'arbre d'affectation des variables propositionnelles qui définit toutes les interprétations.



On voit que chaque branche est recouverte par une affectation qui ne satisfait pas l'une des clauses de $\mathcal{E}$.

Si ce n'était pas le cas, $\mathcal{E}$ serait satisfiable.

Ainsi si l'on traduit cela en calcul des séquents, on voit que l'arbre d'affectation des variables propositionnelles devient un arbre de preuve en calcul des séquents,

– dont la racine est le séquent $\vdash$,
– dont le feuilles sont des clauses de $\mathcal{E}$.

Donc si $\mathcal{E}$ est insatisfiable, il y a un arbre de preuve de $\vdash$ dans le calcul des séquents où l'on utilise que la règle de coupure

$$\frac{A, \Gamma \vdash \Delta \qquad \Xi \vdash A, \Theta}{\Gamma, \Xi \vdash \Delta, \Theta}$$

et où l'on utilise comme axiome les différentes clauses de l'ensemble $\mathcal{E}$.

Étant donné une arbre de preuve, on lui associe le graphe sans cycle qui correspond à la fusion maximum des preuves des séquents identiques.

Par exemple

$$\dfrac{\dfrac{\vdash A, B \qquad B \vdash A}{\vdash A} \qquad \dfrac{\dfrac{A \vdash B \qquad A, B \vdash}{\vdash B} \qquad A, B \vdash}{A \vdash}}{\vdash}$$

Étant donné une arbre de preuve, on lui associe le graphe sans cycle qui correspond à la fusion maximum des preuves des séquents identiques.

Par exemple

$$\dfrac{\dfrac{\vdash A, B \qquad B \vdash A}{\vdash A} \qquad \dfrac{\dfrac{A \vdash B \qquad \boxed{A, B \vdash}}{\vdash B} \qquad \boxed{A, B \vdash}}{A \vdash}}{\vdash}$$

On «fusionne» dans l'arbre de preuve les branches correspondant à la même sous-preuve. On crée un graphe de preuves sans cycle.

Étant donné une arbre de preuve, on lui associe le graphe sans cycle qui correspond à la fusion maximum des preuves des séquents identiques.

Par exemple

$$\dfrac{\dfrac{B \vdash A \qquad \vdash A, B}{\vdash A} \qquad \dfrac{\dfrac{A \vdash B \qquad \boxed{A, B \vdash}}{\vdash B} \qquad \bullet}{A \vdash}}{\vdash}$$

On «fusionne» dans l’arbre de preuve les branches correspondant à la même sous-preuve. On crée un graphe de preuves sans cycle.

On montre qu'il y a une correspondance entre les arbres de preuves du calcul de séquents et les graphes de résolution.

Autrement dit, si un ensemble de clauses est insatisfiable
    il existe une réfutation par résolution.

La résolution en calcul des prédicats du premier ordre

Formes prénexes

Une formule est en forme prénexe,

- si elle ne contient aucun quantificateur,
- ou bien si elle est de forme $Q_1 x_1 \ldots Q_n x_n B$ où
  - B est une formule sans quantificateurs,
  - $x_1, \ldots, x_n$ sont des variables
  - et $Q_i \in \{\forall, \exists\}$ pour $i = 1, \ldots, n$.

Formes prénexes

Pour toute formule Φ, il existe une formule Ψ en forme prénexe telle que $\Phi \Leftrightarrow \Psi$.

Exercice

Mettre les formules suivantes en forme prénexe :

$$(\neg \forall x P(x)) \Rightarrow \forall y Q(y)$$

$$\forall x (\forall y Q(x, y) \Rightarrow P(x))$$

Skolémisation

La Skolémisation ou mise en forme de Skolem consiste à supprimer les quantificateurs de la façon suivante.

Skolémisation

On considère le quantificateur le plus profond.

– Si ce quantificateur est un quantificateur universel, on le supprime et on transforme la variable quantifiée en une variable libre.

$$Q_1x_1 \dots Q_nx_n \forall x P(x_1, \dots, x_n, x)$$

donne

$$Q_1x_1 \dots Q_nx_nP(x_1, \dots, x_n, x).$$

Skolémisation

– Si ce quantificateur est un quantificateur existentiel, on remplace toutes les occurrences des variables associées à ce prédicat par un symbole de fonction qui dépend de toutes les variables quantifiées universellement qui précèdent.

$$Q_1\, x_1 \ldots Q_n x_n \exists x P(x_1, \ldots, x_n, x)$$

donne

$$Q_1 x_1 \ldots Q_n x_n P(x_1, \ldots, x_n, f(x_{j_1}, \ldots, x_{j_q}))$$

où les $x_{j_1}, \ldots, x_{j_q}$ sont les variables correspondant aux quantificateurs universels.

Skolémisation

$$\forall x_1 \exists x_2 \forall x_3 \exists x_4 R(x_1, x_2, x_3, x_4)$$

donne

$$R(x_1, g(x_1), x_3, f(x_1, x_3)).$$

Les fonctions f et g introduites s'appellent les fonctions de Skolem.

Petit truc

Lors de la mise en forme prénexe (qui précède a priori la skolémisation), on doit éviter de «sortir» les quantificateurs universels avant les quantificateurs existentiels, pour ne pas augmenter inutilement l'arité des fonctions de Skolem.

Skolémisation

Une proposition est satisfiable si et seulement si sa forme skolémisée est satisfiable.

Littéraux

Un littéral est une formule de la forme

– soit $P(t_1, ..., t_k)$,

– soit $\neg P(t_1, ..., t_k)$,

où $P(t_1, ..., t_k)$ est un atome, autrement dit P est un symbole de prédicats et t_1, ..., t_k, sont des termes.

On écrira $L(x_1, ..., x_m)$ et $\neg L(x_1, ..., x_m)$

où $FV(L(x_1, ..., x_m)) = \{x_1, ..., x_m\}$,

et où $L(x_1, ..., x_m)$ est un littéral positif ne commençant pas par le signe $\neg$.

Mise sous forme clausale

Étant donnée une formule du calcul des prédicats, la mise sous forme clausale consiste

1. à mettre la formule sous forme prénexe,
2. à skolémiser la formule prénexe,
3. à mettre la skolémisée sous forme clausale.

C'est la phase préparatoire à la résolution.

Mise sous forme clausale

La forme clausale de la formule

$$\forall x (\forall y Q(x, y) \Rightarrow P(x))$$

est la formule

$$P(x) \vee \neg Q(x, f(x))$$

Mise sous forme clausale

Soit la proposition

Toute relation irréflexive et transitive est antisymétrique

Mise sous forme clausale

Sans tenir compte de la caractérisation de l'identité, cela s'énonce par la formule

$$
\begin{aligned}
(\forall x\, \neg R(x,x)) \quad &\wedge \quad \forall y \forall z \forall u (R(y,z) \ \wedge \ R(z,u) \Rightarrow R(y,u)))) \\
&\Rightarrow \quad \forall v \forall w (R(v,w) \ \wedge \ R(w,v) \Rightarrow (v = w))
\end{aligned}
$$

dont la négation est

$$
\begin{aligned}
\neg[(\forall x\, \neg R(x,x)) \quad &\wedge \quad \forall y \forall z \forall u (R(y,z) \ \wedge \ R(z,u) \Rightarrow R(y,u)))) \\
&\Rightarrow \quad \forall v \forall w (R(v,w) \ \wedge \ R(w,v) \Rightarrow (v = w))]
\end{aligned}
$$

qui a pour forme prénexe

$$\exists v \exists w \forall x \forall y \forall z \forall u \, \neg[\neg R(x,x) \;\wedge\; (R(y,z) \;\wedge\; R(z,u) \Rightarrow R(y,u))]$$
$$\Rightarrow (R(v,w) \;\wedge\; R(w,v) \Rightarrow (v = w))]$$

Cette négation a pour formule skolémisée

$$\begin{aligned} \neg[(\neg R(x,x)) \quad &\wedge \quad (R(y,z) \;\wedge\; R(z,u) \Rightarrow R(y,u)))) \\ &\Rightarrow \quad (R(a,b) \;\wedge\; R(b,a) \Rightarrow (a = b))] \end{aligned}$$

où a et b sont de nouvelles constantes.

Mise sous forme clausale

Puis elle a pour forme clausale

$$\begin{aligned} \neg R(x,x) \quad & \wedge \quad (\neg R(y,z) \vee \neg R(z,u) \vee R(y,u)) \\ & \wedge \quad R(a,b) \ \wedge \ R(b,a) \ \wedge \ \neg(a=b). \end{aligned}$$

Elle correspond à l'ensemble de clauses

$$S = \{\neg R(x,x), \quad \neg R(y,z) \vee \neg R(z,u) \vee R(y,u), \quad R(a,b), \quad R(b,a), \quad \neg(a=b)\}.$$

Résolution

La règle de résolution s'applique à des clauses.

Elle s'énonce ainsi :

$$\frac{\neg L(x_1, ..., x_m) \vee \Phi \qquad\qquad L'(y_1, ..., y_n) \vee \Psi}{\sigma(\Phi) \vee \sigma(\Psi)}$$

où $\sigma = mgu(L(x_1, ..., x_m), L'(y_1, ..., y_n))$.

Autrement dit, σ est l'unificateur le plus général

de $L(x_1, ..., x_m)$ et $L'(y_1, ..., y_n)$.

Un exemple

$$S_0 = \{\neg R(x,x),\quad \neg R(y,z) \vee \neg R(z,u) \vee R(y,u),\quad R(a,b),\quad R(b,a),\quad \neg(a=b)\}$$

avec $\sigma_1 = \{y \mapsto x,\quad u \mapsto x\}$,

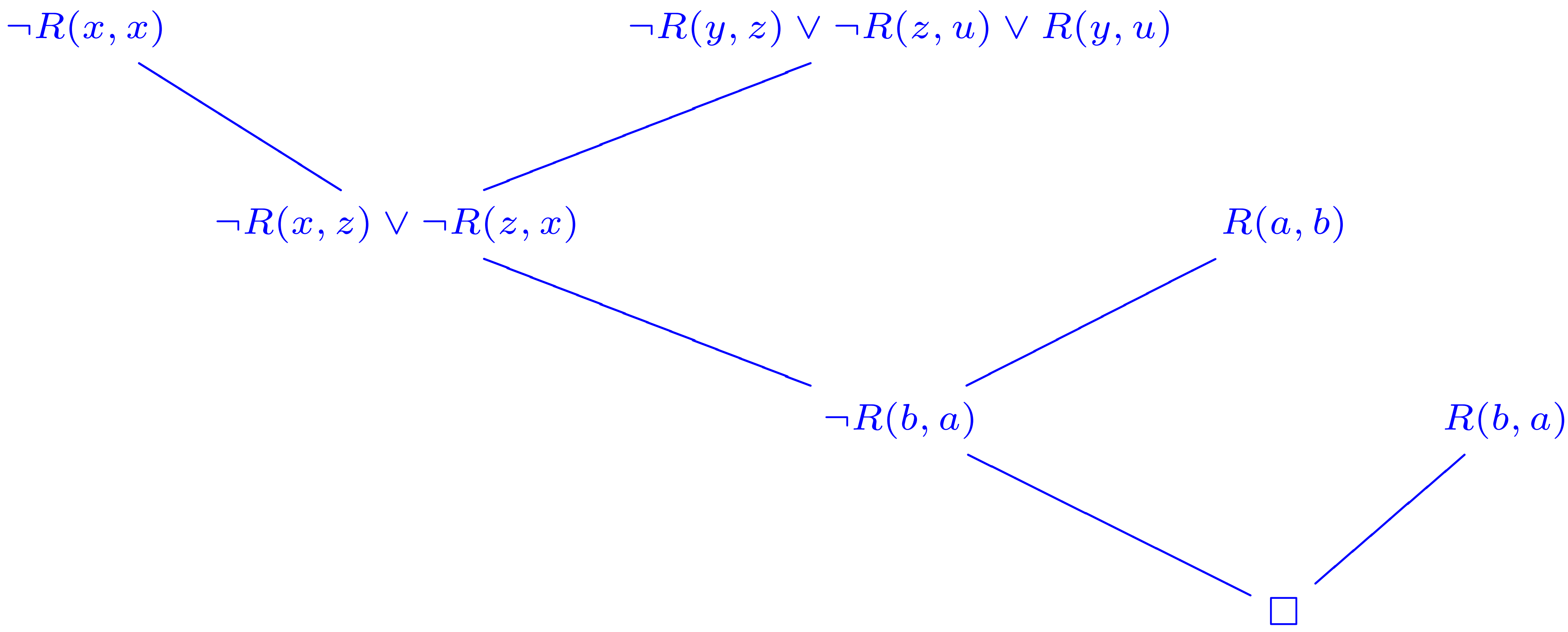
$$S_1 = \{\neg R(x,x),\quad \neg R(y,z) \vee \neg R(z,u) \vee R(y,u),\quad R(a,b),\quad R(b,a),\quad \neg(a=b),\quad \neg R(x,z) \vee \neg R(z,x)\}$$

avec $\sigma_2 = \{x \mapsto a,\quad z \mapsto b\}$,

$$S_2 = \{\neg R(x,x),\quad \neg R(y,z) \vee \neg R(z,u) \vee R(y,u),\quad R(a,b),\quad R(b,a),\quad \neg(a=b),\quad \neg R(x,z) \vee \neg R(z,x),\quad \neg R(b,a)\}$$

avec σ_3 la substitution identité,

$$S_3 = \{\neg R(x,x),\quad \neg R(y,z) \vee \neg R(z,u) \vee R(y,u),\quad R(a,b),\quad R(b,a),\quad \neg(a=b),\quad \neg R(x,z) \vee \neg R(z,x),\quad \neg R(b,a),\quad \Box\}$$


$\neg R(x,x)$
$\neg R(y,z) \vee \neg R(z,u) \vee R(y,u)$
$\neg R(x,z) \vee \neg R(z,x)$
$R(a,b)$
$\neg R(b,a)$
$R(b,a)$
$\square$

On remarque que l'on n'a pas utilisé les propriétés de l'égalité.

A la place de $a = b$ on aurait pu mettre n'importe quelle propriété.

Stratégies

La stratégie qui consiste à toujours choisir la dernière clause produite s'appelle la résolution linéaire.

La stratégie linéaire est complète.

Stratégie SLD

et

programmation logique

Clause de Horn

Une clause qui contient au plus un littéral positif est appelée une clause de Horn.

Une clause qui contient un et un seul littéral positif est appelée une clause définie.

Au lieu de $\neg A_1 \vee \ldots \vee \neg A_n \vee B$ on écrit $B \leftarrow A_1, \ldots, A_n$.

Une clause qui contient seulement des littéraux négatifs est appelée un but.

Au lieu de $\neg A_1 \vee \ldots \vee \neg A_n$ on écrit $\leftarrow A_1, \ldots, A_n$.

La stratégie SLD

Soit S un ensemble de clauses de Horn composé d'un ensemble D de clauses définies, et d'un ensemble $\{G_1, \ldots, G_q\}$ de buts.

Une SLD-dérivation est une suite de buts $\langle N_0, \ldots, N_p \rangle$ tels que

– $N_0 = G_j$ est l'un des buts,

– si $N_i = \ \leftarrow A_1, \ldots, A_{k-1}, A_k, A_{k+1}, \ldots, A_n$

et s'il y a une clause définie $A \leftarrow B_1, \ldots, B_m$, telle que σ est l'unificateur le plus général de A_k et A,

alors (cas $m > 0$)

$$N_{i+1} = \ \leftarrow \sigma(A_1, \ldots, A_{k-1}, B_1, \ldots, B_m, A_{k+1}, \ldots, A_n)$$

ou (cas $m = 0$)

$$N_{i+1} = \ \leftarrow \sigma(A_1, \ldots, A_{k-1}, A_{k+1}, \ldots, A_n).$$

La stratégie SLD

Une SLD-réfutation est une SLD-dérivation dans laquelle $N_p = \Box$.

La SLD-résolution est la méthode qui consiste à montrer qu'un ensemble de clauses de Horn est insatisfiable en trouvant une SLD-réfutation.

$\boxed{\text{La SLD-résolution est complète pour les clauses de Horn.}}$

La programmation logique

La programmation logique est une méthode qui consiste

– à présenter des prédicats récursivement par un ensemble des clauses définies avec le même symbole relationnel en position de littéral positif,

– à poser des questions par des buts.

Les clauses définies sans antécédents sont souvent vues comme l'énoncé de faits (comme dans une base de données).

La programmation logique

PROLOG est donc un langage de programmation

où les procédures sont des clauses définies,

où l'appel se fait par unification, c'est-à-dire que l'on effectue une unification (et pas seulement un filtrage comme en CAML) entre le paramètre de la procédure appelée et la valeur fournie par l'appelant.

La question est plutôt une contrainte à résoudre, que l'appelé identifie avec les contraintes qu'il sait résoudre.

PROLOG fait donc partie des langages de résolution de contraintes.

PROLOG est lié aux langages d'interrogation de bases de données.

Un exemple : la théogonie d’Hésiode

mère(Mnémosyné,Gaia) $\longleftarrow$

mère(Rhéa,Gaia) $\longleftarrow$

mère(Cronos,Gaia) $\longleftarrow$

mère(Histié,Rhéa) $\longleftarrow$

mère(Déméter,Rhéa) $\longleftarrow$

mère(Héra,Rhéa) $\longleftarrow$

mère(Zeus,Rhéa) $\longleftarrow$

mère(Poséïdon,Rhéa) $\longleftarrow$

mère(Hadès,Rhéa) $\longleftarrow$

mère(Perséphone,Déméter) $\longleftarrow$

mère(Hébé,Héra) $\longleftarrow$

mère(Arès,Héra) $\longleftarrow$

mère(Ilithye,Héra) $\longleftarrow$

mère(Clio,Mnémosyné) $\longleftarrow$

mère(Calliope,Mnémosyné) $\longleftarrow$

père(Mnémosyné,Ouranos) $\longleftarrow$

père(Rhéa,Ouranos) $\longleftarrow$

père(Cronos,Ouranos) $\longleftarrow$

père(Histié,Cronos) $\longleftarrow$

père(Déméter,Cronos) $\longleftarrow$

père(Héra,Cronos) $\longleftarrow$

père(Zeus,Cronos) $\longleftarrow$

père(Poséïdon,Cronos) $\longleftarrow$

père(Hadès,Cronos) $\longleftarrow$

père(Perséphone,Zeus) $\longleftarrow$

père(Hébé,Zeus) $\longleftarrow$

père(Arès,Zeus) $\longleftarrow$

père(Ilithye,Zeus) $\longleftarrow$

père(Clio,Zeus) $\longleftarrow$

père(Calliope,Zeus) $\longleftarrow$

grand-père(x, y) $\longleftarrow$ père(x,z), père(z,y)

grand-père(x, y) $\longleftarrow$ mère(x,z), père(z,y)

grand-mère(x, y) $\longleftarrow$ père(x,z), mère(z,y)

grand-mère(x, y) $\longleftarrow$ mère(x,z), mère(z,y)

cousin(x,y) $\longleftarrow$ grand-père(x,z), grand-père(y,z)

cousin(x,y) $\longleftarrow$ grand-mère(x,z), grand-mère(y,z)

ancêtre(x,y) $\longleftarrow$ père(x,y)

ancêtre(x,y) $\longleftarrow$ mère(x,y)

ancêtre(x,y) $\longleftarrow$ père(x,z), ancêtre(z,y)

ancêtre(x,y) $\longleftarrow$ mère(x,z), ancêtre(z,y)

Quelques questions

Simple à solution unique

$\longleftarrow$ mère(Clio,x)

$\longleftarrow$ mère(Persphone,x), père(x,y), reponse(x)

Simple à solutions multiples

$\longleftarrow$ père(x,Zeus)

$\longleftarrow$ grand-père(Perséphone, x)

$\longleftarrow$ grand-père(x, Zeus)

$\longleftarrow$ grand-père(x, y)

Quelques questions

Solutions d'une relation (question non linéaire)

$\leftarrow$ cousin(x,y), père(x,y)

$\leftarrow$ oncle(x,x)   si on a déjà défini oncle.

oncle(x,y) $\leftarrow$ grand-père(x,z), père(y,z)

oncle(x,y) $\leftarrow$ grand-mère(x,z), mère(y,z)