Logique du premier ordre
2ème leçon

Jean-Pierre Jouannaud
École Polytechnique
91400 Palaiseau, France

email: jouannaud@lix.polytechnique.fr
http: //w^3.lix.polytechnique.fr/Labo/Jean-Pierre.Jouannaud

Project LogiCal, Pôle Commun de Recherche en Informatique du Plateau de Saclay, CNRS, École Polytechnique, INRIA, Université Paris-Sud.

February 6, 2006

Outline

Feuille de route

Un théorème est constitué

- d'un ensemble d'hypothèses $H_1, \ldots, H_n$
- d'une conclusion C,
- d'une preuve de validité de la formule $H_1 \wedge \ldots \wedge H_n \implies C$

Première étape

$H_1 \wedge \ldots \wedge H_n \implies C$ est valide ssi
$H_1 \wedge \ldots \wedge H_n \wedge \neg C$ est insatisfiable.

Exemple : *tous les hommes sont grecs ; Socrate est un homme ; donc Socrate est grec.*

$$(\forall x H(x) \implies G(x)) \wedge H(S) \implies G(S)$$

et on est donc amené à vérifier que la formule

$$(\forall x H(x) \implies G(x)) \wedge H(S) \wedge (\neg G(S))$$

est insatisfiable.

Première étape

$H_1 \wedge \ldots \wedge H_n \implies C$ est valide ssi
$H_1 \wedge \ldots \wedge H_n \wedge \neg C$ est insatisfiable.

Exemple : *tous les hommes sont grecs ; Socrate est un homme ; donc Socrate est grec.*

$$(\forall x H(x) \implies G(x)) \wedge H(S) \implies G(S)$$

et on est donc amené à vérifier que la formule

$$(\forall x H(x) \implies G(x)) \wedge H(S) \wedge (\neg G(S))$$

est insatisfiable.

Première étape

$H_1 \wedge \ldots \wedge H_n \implies C$ est valide ssi
$H_1 \wedge \ldots \wedge H_n \wedge \neg C$ est insatisfiable.

Exemple : *tous les hommes sont grecs ; Socrate est un homme ; donc Socrate est grec.*

$$(\forall x H(x) \implies G(x)) \wedge H(S) \implies G(S)$$

et on est donc amené à vérifier que la formule

$$(\forall x H(x) \implies G(x)) \wedge H(S) \wedge (\neg G(S))$$

est insatisfiable.

Première étape

$H_1 \wedge \ldots \wedge H_n \implies C$ est valide ssi
$H_1 \wedge \ldots \wedge H_n \wedge \neg C$ est insatisfiable.

Exemple : *tous les hommes sont grecs ; Socrate est un homme ; donc Socrate est grec.*

$$(\forall x H(x) \implies G(x)) \wedge H(S) \implies G(S)$$

et on est donc amené à vérifier que la formule

$$(\forall x H(x) \implies G(x)) \wedge H(S) \wedge (\neg G(S))$$

est insatisfiable.

Seconde étape

On transforme la formule

$$H_1 \wedge \ldots \wedge H_n \wedge \neg C$$

en une conjonction de clauses $C_1 \wedge \ldots \wedge C_p$, c'est-à-dire de formules de la forme

$$\forall \overline{x} A_1 \vee \ldots \vee A_M \vee \neg B_1 \vee \ldots \vee \neg B_n$$

où les A_i et les B_i sont des formules atomiques, tout en préservant l'insatisfiabilité.

Cela est équivalent à transformer chaque élément de la conjonction en clauses. Dans l'exemple précédent, on obtient :

$$\{H(x) \implies G(x), H(S), \neg G(S)\}$$

Seconde étape

On transforme la formule

$$H_1 \wedge \ldots \wedge H_n \wedge \neg C$$

en une conjonction de clauses $C_1 \wedge \ldots \wedge C_p$, c'est-à-dire de formules de la forme

$$\forall \overline{x} A_1 \vee \ldots \vee A_M \vee \neg B_1 \vee \ldots \vee \neg B_n$$

où les A_i et les B_i sont des formules atomiques, tout en préservant l'insatisfiabilité.

Cela est équivalent à transformer chaque élément de la conjonction en clauses. Dans l'exemple précédent, on obtient :

$$\{H(x) \implies G(x), H(S), \neg G(S)\}$$

Troisième étape

On construit un ensemble fini insatisfiable d'instances closes des clauses obtenues grâce au théorème de compacité de Herbrand, à partir duquel on va chercher à déduire la clause vide qui est la plus petite clause insatisfiable.

Quatrième étape

Complétude de la *résolution close* :

$$\frac{A \vee C \qquad \neg A \vee D}{C \vee D}$$

où A dénote un atome, C et D des clauses. Les clauses $A \vee C$, $\neg A \vee D$, $C \vee D$ sont supposées en forme normale vis à vis des règles de l'algèbre de Boole : $A, \neg A \notin C \vee D$.

La complétude de la résolution s'énonce comme l'appartenance de la clause vide à l'ensemble $Res^*(\{C'_1, \ldots, C'_{p'}\})$ des clauses inférables par résolution à partir des clauses $\{C'_1, \ldots, C'_{p'}\}$.

Complétude de la *résolution close* :

$$\frac{A \vee C \qquad \neg A \vee D}{C \vee D}$$

où A dénote un atome, C et D des clauses.
Les clauses $A \vee C$, $\neg A \vee D$, $C \vee D$ sont supposées en forme normale vis à vis des règles de l'algèbre de Boole : $A, \neg A \notin C \vee D$.

La complétude de la résolution s'énonce comme l'appartenance de la clause vide à l'ensemble $Res^*(\{C'_1, \ldots, C'_{p'}\})$ des clauses inférables par résolution à partir des clauses $\{C'_1, \ldots, C'_{p'}\}$.

Cinquième étape

Le lemme de relèvement permet de généraliser la résolution aux clauses avec variables:

$$\frac{P_1 \vee \ldots \vee P_p \vee C \qquad \neg N_1 \vee \ldots \vee \neg N_n \vee D}{C\sigma \vee D\sigma}$$

où σ désigne l'unificateur principal du problème d'unification

$$\begin{gathered} P_1 = P_2 \wedge \ldots \wedge P_1 = P_p \wedge \\ P_1 = N_1 \wedge N_1 = N_2 \wedge \ldots \wedge N_1 = N_n \end{gathered}$$

c-a-d la substitution σ la plus générale t.q.

$$P_1\sigma = P_2\sigma = \ldots = P_p\sigma = N_1\sigma = \ldots = N_n\sigma$$

Exemple

Deux preuves possibles :

$$\dfrac{\dfrac{G(x) \vee \neg H(x) \quad H(S)}{G(S)} \qquad \neg G(S)}{\square}$$

$$\dfrac{\dfrac{G(x) \vee \neg H(x) \quad \neg G(S)}{\neg H(S)} \qquad H(S)}{\square}$$

Dernière étape

L'opération d'unification permet en fait d'assurer que si deux clauses ont des instances closes qui peuvent se résoudre par résolution close, alors la résolvante (close) est une instance close de celle obtenue par résolution (générale) des deux closes de départ. Il ne sera donc nul besoin de *connaitre* les instances closes fournies par le théorème de Herbrand, seule leur existence est importante.

La dernière étape sera donc celle de l'unification, dont nous détaillerons les propriétés algorithmiques.

Dernière étape

L'opération d'unification permet en fait d'assurer que si deux clauses ont des instances closes qui peuvent se résoudre par résolution close, alors la résolvante (close) est une instance close de celle obtenue par résolution (générale) des deux closes de départ. Il ne sera donc nul besoin de *connaitre* les instances closes fournies par le théorème de Herbrand, seule leur existence est importante.

La dernière étape sera donc celle de l'unification, dont nous détaillerons les propriétés algorithmiques.

Mise sous forme clausale

Forme prénexe

Une formule est en forme *prénexe* si elle est non quantifiée, ou de l'une des deux formes $\forall x \phi$ ou $\exists x \phi$ où ϕ est une formule prénexe.

Lemma

Toute formule logique est équivalente à une formule prénexe.

Exercice

Écrire un programme PROLOG qui transforme une formule quelconque en une formule prénexe équivalente.

Une formule est en forme *prénexe* si elle est non quantifiée, ou de l'une des deux formes $\forall x \phi$ ou $\exists x \phi$ où ϕ est une formule prénexe.

Lemma

Toute formule logique est équivalente à une formule prénexe.

Exercice

Écrire un programme PROLOG qui transforme une formule quelconque en une formule prénexe équivalente.

Une formule est en forme *prénexe* si elle est non quantifiée, ou de l'une des deux formes $\forall x \phi$ ou $\exists x \phi$ où ϕ est une formule prénexe.

Lemma

Toute formule logique est équivalente à une formule prénexe.

Exercice

Écrire un programme PROLOG qui transforme une formule quelconque en une formule prénexe équivalente.

Exemple

Mise en forme prénexe de la formule $(\forall x P(x)) \implies (\exists y Q(y))$:

$$\begin{aligned}
(\forall x P(x)) &\implies (\exists y Q(y)) \\
\forall x (P(x) &\implies (\exists y Q(y))) \\
\forall x \exists y (P(x) &\implies Q(y))
\end{aligned}$$

Mais la forme prénexe n'est pas unique:

$$\begin{aligned}
(\forall x P(x)) &\implies (\exists y Q(y)) \\
\exists y (\forall x P(x) &\implies Q(y)) \\
\exists y \forall x (P(x) &\implies Q(y))
\end{aligned}$$

Une *formule clausale* est une formule prénexe close sans quantificateur existentiel.

Theorem

Soit A une formule close. Il existe une formule clausale A' non unique insatisfiable ssi A l'est.

Soit $A = \forall\overline{x}\exists y\ B$ prénexe close, donc B est prénexe. Soit f un nouveau symbole de fonction (*de Skolem*) d'arité $|\overline{x}|$, et $C = \forall\overline{x}B\{y \mapsto f(\overline{x})\}$ prénexe close. $C \models A$. Par récurrence, il existe A' t.q. $A' \models C$. Par transitivité, $A' \models A$.

Exercice

Écrire un programme PROLOG qui transforme une formule prénexe en formule clausale.

Skolémisation

Une *formule clausale* est une formule prénexe close sans quantificateur existentiel.

Theorem

Soit A une formule close. Il existe une formule clausale A′ non unique insatisfiable ssi A l'est.

Soit $A = \forall \overline{x} \exists y\ B$ prénexe close, donc B est prénexe. Soit f un nouveau symbole de fonction (*de Skolem*) d'arité $|\overline{x}|$, et $C = \forall \overline{x} B\{y \mapsto f(\overline{x})\}$ prénexe close. $C \models A$. Par récurrence, il existe A' t.q. $A' \models C$. Par transitivité, $A' \models A$.

Exercice

Écrire un programme PROLOG qui transforme une formule prénexe en formule clausale.

Une *formule clausale* est une formule prénexe close sans quantificateur existentiel.

Theorem

Soit A une formule close. Il existe une formule clausale A' non unique insatisfiable ssi A l'est.

Soit $A = \forall\overline{x}\exists y\ B$ prénexe close, donc B est prénexe. Soit f un nouveau symbole de fonction (*de Skolem*) d'arité $|\overline{x}|$, et $C = \forall\overline{x}B\{y \mapsto f(\overline{x})\}$ prénexe close. $C \models A$. Par récurrence, il existe A' t.q. $A' \models C$. Par transitivité, $A' \models A$.

Exercice

Écrire un programme PROLOG qui transforme une formule prénexe en formule clausale.

Une *clause* est une formule clausale de la forme $\forall \overline{x}(A_1 \wedge \ldots \wedge A_n \implies B_1 \vee \ldots \vee B_p)$, ou $\forall \overline{x}(\neg A_1) \vee \ldots \vee (\neg A_n) \vee B_1 \vee \ldots \vee B_p$.
Les quantificateurs sont souvent omis. Une clause non quantifiée sans variable est *close*.
$C\gamma$ est une *instance* de la clause $\forall \overline{x} C$.

Lemma

Toute formule clausale est équivalente à une conjonction de clauses.

Exercice

Écrire un programme PROLOG qui transforme une formule clausale en conjonction de clauses.

Une *clause* est une formule clausale de la forme $\forall \vec{x}(A_1 \wedge \ldots \wedge A_n \implies B_1 \vee \ldots \vee B_p)$, ou $\forall \vec{x}(\neg A_1) \vee \ldots \vee (\neg A_n) \vee B_1 \vee \ldots \vee B_p$.
Les quantificateurs sont souvent omis. Une clause non quantifiée sans variable est *close*.
$C\gamma$ est une *instance* de la clause $\forall \vec{x} C$.

Lemma

Toute formule clausale est équivalente à une conjonction de clauses.

Exercice

Écrire un programme PROLOG qui transforme une formule clausale en conjonction de clauses.

Clauses

Une *clause* est une formule clausale de la forme $\forall \vec{x}(A_1 \wedge \ldots \wedge A_n \implies B_1 \vee \ldots \vee B_p)$, ou $\forall \vec{x}(\neg A_1) \vee \ldots \vee (\neg A_n) \vee B_1 \vee \ldots \vee B_p$.
Les quantificateurs sont souvent omis. Une clause non quantifiée sans variable est *close*.
$C\gamma$ est une *instance* de la clause $\forall \vec{x} C$.

Lemma

Toute formule clausale est équivalente à une conjonction de clauses.

Exercice

Écrire un programme PROLOG qui transforme une formule clausale en conjonction de clauses.

Mise en forme clausale de la formule
$\neg(\forall x \exists y . P(x,y)) \vee \exists z . Q(z)$:

$$\begin{aligned} & \neg(\forall x \exists y . P(x,y)) \vee \exists z . Q(z) \\ \Longrightarrow \ & \exists x \forall y \exists z . \neg P(x,y) \vee Q(z) \\ \Longrightarrow \ & \forall y . \neg P(a,y) \vee Q(f(y)) \end{aligned}$$

Mais la forme clausale n'est pas unique:

$$\begin{aligned} & \neg(\forall x \exists y . P(x,y)) \vee \exists z . Q(z) \\ \Longrightarrow \ & \exists x . (\forall y . \neg P(x,y) \vee Q(x)) \\ \Longrightarrow \ & \forall y . \neg P(a,y) \vee Q(a) \end{aligned}$$

Interprétations de Herbrand (syntaxiques)

On suppose les vocabulaires $\mathcal{F}$, $\mathcal{X}$ et $\mathcal{A}$ dénombrables. Soit $\mathcal{T}(\mathcal{F})$ l'ensemble des termes clos, appellé univers de Herbrand et noté $\mathcal{G}$. Soit $\mathcal{A}(\mathcal{P},\mathcal{F})$ l'ensemble des atomes clos, appellé base de Herbrand et noté $\mathcal{B}$. Une interprétation H de Herbrand
- a pour domaine l'univers de Herbrand $\mathcal{G}$,
- interprète le symbole $f \in \mathcal{F}_n$ par l'opération f_H d'arité n telle que $f_H(\bar{t}) = f(\bar{t})$,
- interprète le symbole de prédicat $R \in \mathcal{P}_n$ par un sous-ensemble arbitraire R_H de $\mathcal{G}^n$.

Theorem (Premier Théorème de Herbrand)

Un ensemble de clauses est insatisfiable ssi il l'est dans toute interprétation de Herbrand.

On montre que si un ensemble de clauses est satisfiable dans une interprétation arbitraire I, alors il l'est dans une interprétation de Herbrand H définie comme suit :

$$P_H(t_1, \ldots, t_n) = T$$

ssi pour toute valuation v

$$P_I([t_1]_{I,v}, \ldots, [t_n]_{I,v}) = T$$

Arbre des interprétations

Lemma

Un ensemble S de clauses est satisfiable ssi il l'est dans une interprétation de Herbrand.

La base de Herbrand est dénombrable, soit $\mathcal{B} = \{A_i\}_{i \in N}$.

Étant donnée une énumération de $\mathcal{B}$, il existe un *arbre* binaire *des interprétations*, dont les branches sont en correspondance biunivoque avec les interprétations H : on étiquette les arcs issus d'un noeud à profondeur i par les valeurs de vérité de A_i.

Lemma

Un ensemble S de clauses est satisfiable ssi il l'est dans une interprétation de Herbrand.

La base de Herbrand est dénombrable, soit $\mathcal{B} = \{A_i\}_{i \in N}$.

Étant donnée une énumération de $\mathcal{B}$, il existe un *arbre* binaire *des interprétations*, dont les branches sont en correspondance biunivoque avec les interprétations H : on étiquette les arcs issus d'un noeud à profondeur i par les valeurs de vérité de A_i.

Exercice

1. *Donner une enumération de la base de Herbrand pour le vocabulaire :*

0	*symbole de constante*
s	*symbole de fonction unaire*
P	*symbole de prédicat binaire*

2. *En déduire l'arbre des interprétations associé.*

Exercice

1. *Donner une enumération de la base de Herbrand pour le vocabulaire :*

0	*symbole de constante*
s	*symbole de fonction unaire*
P	*symbole de prédicat binaire*

2. *En déduire l'arbre des interprétations associé.*

Un chemin de la racine à un noeud $p \in (N^+)^*$ de l'arbre sera noté H_p et appellé une *interprétation partielle* des atomes A_0 à $A_{|p|-1}$. On dira que l'interprétation I_q *prolonge* l'interprétation I_p si le chemin p est un préfixe du chemin q. On définit un calcul des interprétations partielles dans une logique trivaluée (avec une nouvelle valeur $\perp$ qui se veut représenter la valeur indéfinie) de la manière suivante :

- $H_p(C) = T$ si $H(C) = T$ dans toute interprétation totale qui prolonge H_p.
- $H_p(C) = F$ si $H(C) = F$ dans toute interprétation totale qui prolonge H_p.
- $H_p(C) = \perp$ dans le cas contraire.

Interprétations partielles

Un chemin de la racine à un noeud $p \in (N^+)^*$ de l'arbre sera noté H_p et appellé une *interprétation partielle* des atomes A_0 à $A_{|p|-1}$. On dira que l'interprétation I_q *prolonge* l'interprétation I_p si le chemin p est un préfixe du chemin q. On définit un calcul des interprétations partielles dans une logique trivaluée (avec une nouvelle valeur $\perp$ qui se veut représenter la valeur indéfinie) de la manière suivante :

- $H_p(C) = T$ si $H(C) = T$ dans toute interprétation totale qui prolonge H_p.
- $H_p(C) = F$ si $H(C) = F$ dans toute interprétation totale qui prolonge H_p.
- $H_p(C) = \perp$ dans le cas contraire.

Exercice

1. *Soient C une clause clause close et H_p une interprétation partielle. Montrer que $[C]_{H_p} \neq \perp$ ssi H_p énumère tous les atomes de C.*
2. *Soit C une clause telle que $[\forall x C]_H = F$. Montrer qu'il existe une interprétation partielle H_p telle que $[\forall x C]_{H_p} = F$.*

Exercice

1. *Soient C une clause clause close et H_p une interprétation partielle. Montrer que $[C]_{H_p} \neq \perp$ ssi H_p énumère tous les atomes de C.*
2. *Soit C une clause telle que $[\forall x C]_H = F$. Montrer qu'il existe une interprétation partielle H_p telle que $[\forall x C]_{H_p} = F$.*

Arbre sémantique

p est un *noeud d'échec* pour l'ensemble $S = \{C_1, \ldots, C_n\}$ de clauses s'il existe une clause C_i et une substitution close γ telle que $[C_i\gamma]_{H_p} = F$ et la propriété n'est vraie pour aucun prédécesseur de p dans l'arbre des interprétations.

On appelle *arbre sémantique* d'un ensemble S de clauses l'arbre obtenu à partir de l'arbre des interprétations en élaguant les sous-arbres issus d'un noeud d'échec et en étiquettant ce dernier par les instances des clauses réfutées.

Un arbre sémantique sera dit *clos* si toute branche se termine en un noeud d'échec.

Exemples d'arbres clos

1. Supposons que $\mathcal{F}$ est réduit à un symbole de constante 0 et à un symbole unaire s, $\mathcal{P}$ étant réduit au symbole de prédicat unaire A. Soit S l'ensemble de clauses $\{\neg A(x) \vee A(s^2(x)), A(0), A(s(0)), \neg A(s^3(0)) \vee \neg A(s(0))\}$. Dessiner l'arbre sémantique associé pour une énumération naturelle des atomes clos de la base de Herbrand.
2. Cet ensemble de clauses est-il satisfiable ?
3. Quel est l'arbre sémantique clos obtenu pour l'ordre qui énumère tous les atomes de la forme $A(s^{2n+1}(0))$ avant d'énumérer les atomes de la forme $A(s^{2n}(0))$?

Theorem (Second Théorème de Herbrand)

un ensemble S de clauses est insatisfiable ssi il possède un arbre sémantique clos.

Si l'arbre est clos, toute interprétation de Herbrand est invalidée. D'après le premier théorème de Herbrand, S est insatisfiable.
Si S est insatisfiable, toute interprétation de Herbrand est invalidée par une instance close $C\gamma$ d'une clause C, et dès que les atomes de $C\gamma$ sont tous énumérés, la branche est stopppée.

Compacité

Theorem

Un ensemble S de clauses est insatisfiable si et seulement si cela est vrai d'un ensemble fini d'instances closes de clauses de S.

D'après le second théorème de Herbrand, un ensemble S de clauses est insatisfiable ssi il possède un arbre sémantique clos dont les feuilles, en nombre fini, sont étiquettées par des instances closes de clauses de S qui invalident toute interprétation de Herbrand, donc toute interprétation par le premier théorème de Herbrand.

Theorem

Un ensemble S de clauses est insatisfiable si et seulement si cela est vrai d'un ensemble fini d'instances closes de clauses de S.

D'après le second théorème de Herbrand, un ensemble S de clauses est insatisfiable ssi il possède un arbre sémantique clos dont les feuilles, en nombre fini, sont étiquettées par des instances closes de clauses de S qui invalident toute interprétation de Herbrand, donc toute interprétation par le premier théorème de Herbrand.

Complétude

Theorem

La résolution close est complète.

Soit S un ensemble insatisfiable de clauses closes. Par correction de la résolution, $Res^*(S)$ est insatisfiable, donc possède un arbre sémantique clos d'après le théorème de Herbrand.

1. Si la racine de l'arbre est noeud d'échec, alors $\square \in Res^*(S)$.
2. Sinon, il existe un noeud H de l'arbre dont les deux successeurs L et R sont des noeuds d'échec étiquettées par les clauses $\neg A \vee C$ et $A \vee D$. On vérifie alors que $C \vee D$ réfute le noeud H, ce qui est impossible.

Exercice

1. *Montrer que tout ensemble insatisfiable de clauses contient au moins une clause dont tous les littéraux sont négatifs.*
2. *En déduire la complétude réfutationnelle de la résolution close négative, où l'une des clauses a tous ses littéraux négatifs.*

Exercice : Résolution négative

Exercice

1. *Montrer que tout ensemble insatisfiable de clauses contient au moins une clause dont tous les littéraux sont négatifs.*
2. *En déduire la complétude réfutationnelle de la résolution close négative, où l'une des clauses a tous ses littéraux négatifs.*

Relèvement et Résolution

Lemma

Soient $A \vee C\gamma$ *et* $\neg A \vee D\eta$ *deux instances closes (en forme normale disjonctive) des clauses* $P \vee C$ *et* $N \vee D$*, telles que (i)* $\mathcal{V}ar(P \vee C) \cap \mathcal{V}ar(N \vee D) = \emptyset$*, et (ii) les atomes de* $P\gamma$ *soient tous égaux à* A *et ceux de* $N\eta$ *à* $\neg A$*. Alors* $P = N$ *possède un plus grand unificateur* σ *tel que* $C\sigma \vee D\sigma \ \in \ Res(P \vee C, N \vee D)$*, et* $C\gamma \vee D\eta = (C\sigma \vee D\sigma)\tau$ *pour une certaine substitution* τ*.*

Preuve du relèvement

Comme on peut toujours supposer que les clauses $P \vee C$ et $N \vee D$ ont des variables toutes distinctes, on en déduit que la substitution close $\gamma \cup \eta$ égale à γ pour les variables libres de $P \vee C$ et à η pour celles de $N \vee D$ unifie le problème $P = N$. Soit σ l'unificateur principal du problème, et τ la substitution telle que $\gamma \cup \eta = \sigma\tau$. On vérifie sans peine la propriété annoncée.

Theorem

Un ensemble S de clauses est insatisfiable ssi la clause vide appartient à $Res^*(S)$.

Par le théorème de Herbrand, S est insatisfiable ssi cela est le cas de l'ensemble fini E des instances closes de S. Par le lemme de relèvement, $Res^*(E)$ est inclus dans les instances closes de $Res^*(S)$. La clause vide ne pouvant être l'instance que d'elle même, on en déduit le résultat.

L'utilisation du lemme de relèvement impose l'élimination par résolution de tous les atomes qui deviennent égaux dans l'unification.

Théorème de Robinson

Theorem

Un ensemble S de clauses est insatisfiable ssi la clause vide appartient à $Res^*(S)$.

Par le théorème de Herbrand, S est insatisfiable ssi cela est le cas de l'ensemble fini E des instances closes de S. Par le lemme de relèvement, $Res^*(E)$ est inclus dans les instances closes de $Res^*(S)$. La clause vide ne pouvant être l'instance que d'elle même, on en déduit le résultat.

L'utilisation du lemme de relèvement impose l'élimination par résolution de tous les atomes qui deviennent égaux dans l'unification.

Théorème de Robinson

Theorem

Un ensemble S de clauses est insatisfiable ssi la clause vide appartient à $Res^*(S)$.

Par le théorème de Herbrand, S est insatisfiable ssi cela est le cas de l'ensemble fini E des instances closes de S. Par le lemme de relèvement, $Res^*(E)$ est inclus dans les instances closes de $Res^*(S)$. La clause vide ne pouvant être l'instance que d'elle même, on en déduit le résultat.

L'utilisation du lemme de relèvement impose l'élimination par résolution de tous les atomes qui deviennent égaux dans l'unification.

Exercices

1. Décrire un algorithme qui engendre comme précédemment un ensemble de clauses insatisfiables pour lequel le nombre de symboles de Skolem ajoutés soit minimal.
2. Montrer que tout ensemble insatisfiable de clauses contient au moins une clause dont tous les littéraux sont positifs.
3. Montrer la correction du raisonnement suivant : Les Crétois sont tous des menteurs. Je suis Crétois. Donc je suis menteur.

4. Soit $Freres(x, y)$ une relation symétrique, transitive et antiréflexive. Montrer la correction du raisonnement suivant qui se passe en Crète, royaume des menteurs :
Hyp 1 : tout Crétois a un frère menteur ;
Hyp 2 : tout Crétois a les cheveux blonds ou bruns ;
Hyp 3 : deux frères ont la même couleur de cheveux ssi ils sont menteurs tous deux ou pas menteurs tous deux ;
Conclusion : Donc, en Crète, si deux frères n'ont pas la même couleur de cheveux, alors ils ont un troisième frère.

Corrigé de l'exercice 5

$$\forall xy\ F(x,y) \Rightarrow F(y,x) \qquad \forall x \neg F(x,x)$$

$$\forall xyz\ F(x,y) \wedge F(y,z) \Rightarrow F(x,z)$$

$$\forall x\ C(x) \Rightarrow \exists y\ F(x,y) \wedge M(y)$$

$$\forall x\ C(x) \Rightarrow \exists Bl(x) \vee Br(x)$$

$$\forall xy\ F(x,y) \Rightarrow \left\{ \begin{array}{l} ((Bl(x) \wedge Bl(y)) \vee (Br(x) \vee Br(y))) \\ \Leftrightarrow ((M(x) \wedge M(y)) \end{array} \right.$$

$$\forall xy\ F(x,y) \wedge Bl(x) \wedge Br(y) \Rightarrow \exists z\ F(x,z) \wedge F(y,z)$$

Devoirs

1. Devoir du cours (groupe de deux élèves) : Écrire en PROLOG un démonstrateur basé sur la résolution qui utilise une skolémisation optimale.
2. Travail d'option (individuel) : Justifier et écrire en PROLOG un démonstrateur basé sur la résolution ordonnée avec sélection.

Devoirs

1. Devoir du cours (groupe de deux élèves) : Écrire en PROLOG un démonstrateur basé sur la résolution qui utilise une skolémisation optimale.
2. Travail d'option (individuel) : Justifier et écrire en PROLOG un démonstrateur basé sur la résolution ordonnée avec sélection.

Pour en savoir plus

Prouveurs en logique du premier ordre disponibles sur le Web :
ACL2, SATURATE, SPASS, CiME.

Sur les systèmes de preuve existants, leur mécanismes, les grands problèmes que l'on se pose, leur capacité à prouver de vrais théorèmes :
Sur ma page Web, articles “formal mathematics” et “Twenty years later”.
Page Web de Roberto Nieuwenhuis.

Pour en savoir plus

Prouveurs en logique du premier ordre disponibles sur le Web :
ACL2, SATURATE, SPASS, CiME.

Sur les systèmes de preuve existants, leur mécanismes, les grands problèmes que l'on se pose, leur capacité à prouver de vrais théorèmes :
Sur ma page Web, articles “formal mathematics” et “Twenty years later”.
Page Web de Roberto Nieuwenhuis.

Bibliographie essentielle

Robert Kowalski.
Semantic trees in automatic theorem-proving.
Machine Intelligence, 4:86–101, 1969.

J. A. Robinson.
A machine-oriented logic based on the resolution principle.
Journal of the ACM, 12(1):23–41, 1965.

J.A. Robinson.
The generalized resolution principle. *Machine Intelligence*, 3, 1968.