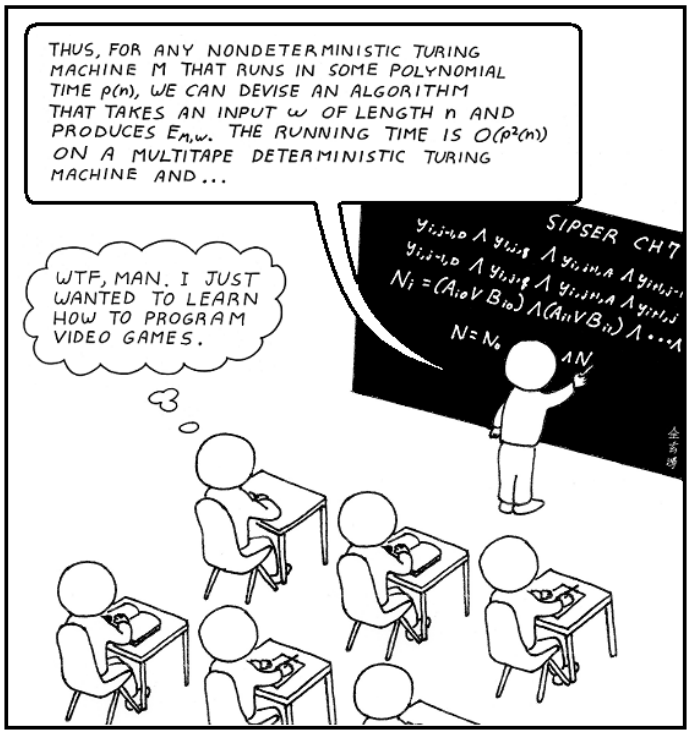
Logique des prédicats : aide mémoire



1 Syntaxe de la logique des prédicats

1.1 langage du premier ordre

C'est un ensemble $\mathcal{L} = \mathcal{C}_{st} \cup Prd$ de symboles constitué :
- d'un ensemble $\mathcal{C}_{st}$ de constantes (ou symboles de fonctions 0-aires) : $\{a, b, c \ldots\}$
  Dans le cadre de cette première partie nous ne considérons pas les symboles de fonctions d'arité ≥ 1 : $\{f,\ g,\ h \ldots\}$
- d'un ensemble $\mathcal{P}_{rd}$ de symboles de prédicats avec une arité associée : $\{P_1, Q_2, R_2 \ldots\}$
- $\mathcal{C}_{st} \cap \mathcal{P}_{rd} = \emptyset$

1.2 Termes

Un terme est une expression désignant un objet du domaine modélisé
En plus des constantes, on dispose d'un ensemble infini $\mathcal{V}_{rb}$ de variables disjoint de $\mathcal{L} : \{x, y, z \ldots\}$

L'ensemble des termes d'un langage $\mathcal{L} =_{def} \mathcal{C}_{st} \cup \mathcal{P}_{rd}$ est l'ensemble $T_m =_{def} V \ \cup \ \mathcal{C}_{st}$
Dans le cadre de cette introduction les termes sont élémentaires ; en logique des prédicats les termes peuvent être complexes : par exemple $f(x, g(a, y))$ où f et g sont des fonctions d'arité 2.

1.3 Formules Bien Formées

Soit $\mathcal{L} =_{def} \mathcal{C}_{st} \ \cup \ \mathcal{P}_{rd}$ un langage, $\mathcal{V}_{rb}$ un ensemble infini de variables, $\mathcal{C}_{log} =_{def} \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ les connecteurs, $Q =_{def} \{\forall, \exists\}$ les quantificateurs, et $\mathcal{P} =_{def} \{(,)\}$ un jeu de parenthèses.
On définit par induction $FBF(\mathcal{L})$, l'ensemble des formules bien formées, construites sur $\mathcal{L}$:
- base
  1. $FBF(\mathcal{L})$ contient l'ensemble des atomes
     $\{\ p(t_1, \ldots, t_n)\ \mid p \in \mathcal{P}_{rd}$ est un prédicat n-aire et $t_1, \ldots, t_n \in T_m$ sont des termes $\}$
  2. $FBF(\mathcal{L})$ contient $\{\top, \bot\}$ (optionnel).
  3. **le cas spécial du symbole d'égalité** $=$
     $FBF(\mathcal{L})$ contient l'ensemble des formules $\{t_1 = t_2$ avec t_1 et $t_2 \in T_m\}$
- induction : soit A et $B \in FBF(\mathcal{L})$ et soit $x \in \mathcal{V}_{rb}$:
  - $\neg A \in FBF(\mathcal{L})$
  - $(A \wedge B) \in FBF(\mathcal{L})$ (idem avec $(A \vee B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$)
  - $\forall x A \in FBF(\mathcal{L})$ (idem avec $\exists x A$)

1.4 Variables libres et liées

Portée d'un quantificateur
La portée d'un quantificateur est la sous-arborescence dont le couple
<ce quantificateur, sa variable associée> étiquette la racine.

définitions des variables libres et des variables liées
- Une occurrence d'une variable x est liée dans F si dans la branche qui va de la racine à la feuille (l'atome) où se trouve cette occurrence on trouve $\forall x$ ou $\exists x$ (autrement dit quand elle est dans la portée d'un $\forall x$ ou d'un $\exists x$). Elle est libre sinon.
- Une variable x est libre dans F si elle a au moins une occurrence libre dans F

fbf ouverte et fermée Une fbf est ouverte lorsqu'elle a au moins une variable libre et elle est fermée sinon (i.e si elle n'a aucune variable libre)

2 Sémantique de la logique des prédicats

Une *interprétation* I d'un langage $\mathcal{L}$ est un couple $I =_{def} (D, i)$ tel que :
- D (le domaine d'interprétation) est un ensemble non vide
- pour toute constante a de $\mathcal{L}$, $i(a)$ est un élément de D
- pour tout prédicat P de $\mathcal{L}$, $i(P)$ est une application de $D^{arit(p)}$ dans $\{faux, vrai\}$

assignation

Une *assignation* est une application de l'ensemble des variables dans le domaine d'interprétation D.

Définition par induction de la valeur d'une fbf

Soit $I =_{def} (D, i)$ une interprétation d'un langage $\mathcal{L}$ et σ une assignation de $\mathcal{V}_{rb}$ dans D, $v(F, I, \sigma)$ est telle que :

- base
  1. si $F =_{def} P(t_1, t_2, \ldots, t_n)$ est un atome, $v(F, I, \sigma) =_{def} (\phi(t_1, I, \sigma), \ldots, \phi(t_n, I, \sigma)) \in i(P)$, où :
     - si t est une constante $\phi(t, I, \sigma) =_{def} i(t)$
     - si t est une variable $\phi(t, I, \sigma) =_{def} \sigma(t)$
  2. $v(\bot, I, \sigma) =_{def} faux$
  3. $v(\top, I, \sigma) =_{def} vrai$
  4. **le cas spécial du symbole d'égalité** $=$
     $Val(t_1 = t_2, I, \sigma)$ vaut vrai si et seulement si $\phi(t_1) = \phi(t_2)$, (où ϕ est définie plus haut).
- induction
  - si $F =_{def} \neg A$, alors $v(F, I, \sigma) =_{def} NON(v(A, I, \sigma))$
  - si $F =_{def} (A \wedge B)$, $v(F, I, \sigma) =_{def} ET(v(A, I, \sigma), v(B, I, \sigma))$ (id. avec $\vee \rightarrow \leftrightarrow$)
  - si $F =_{def} \forall x\ A$, $v(F, I, \sigma) =_{def} vrai$ si et seulement si pour tout élément d de D, $v(A, I, \sigma_{x \triangleright d}) = vrai$ où $\sigma_{x \triangleright d}$ est l'assignation obtenue à partir de σ en donnant en plus à la variable x la valeur d
  - si $F =_{def} \exists x\ A$, $v(F, I, \sigma) =_{def} vrai$ si et seulement si il existe un élément d de D t.q.$v(A, I, \sigma_{x \triangleright d}) = vrai$

$Val(F, I)$

Si F est fermée, la valeur de $Val(F, I, \sigma)$ ne dépend pas de σ et on la note $Val(F, I)$.

2.1 Un peu de vocabulaire

pour des formules fermées
- F vaut vrai (resp. faux) pour I ssi $Val(F, I)$=vrai (resp. faux)
- I est un *modèle* de F ssi $Val(F, I)$=vrai
- I est un *contre-modèle* de F ssi $Val(F, I)$=faux
- F est *satisfiable* ssi elle a au moins un modèle
- F est *contingente* ssi elle a au moins un modèle et au moins un contre modèle
- F est *valide* ssi elle a n'a aucun contre modèle
- F est *insatisfiable* ssi elle a n'a aucun modèle

2.2 Equivalence et conséquence logique

Définition Deux fbf $\mathcal{F}_1$ et $\mathcal{F}_2$ (construites sur un même langage) **fermées** sont logiquement équivalentes ssi pour toute interprétation I, $Val(\mathcal{F}_1, I) = Val(\mathcal{F}_2, I)$
On note $\mathcal{F}_1 \equiv \mathcal{F}_2$

Propriété $A \equiv B$ si et seulement si $A \leftrightarrow B$ est valide.

2.3 Conséquence logique

Si $H_1, H_2, \ldots, H_n$ et C sont des fbf fermées d'un langage $\mathcal{L}$ du premier ordre on dit que C est conséquence logique de $H_1, H_2, \ldots, H_n$, lorsque toute interprétation I de $\mathcal{L}$ qui est un modèle de tous les H_i est un modèle de C : $\{H_1, H_2, \ldots, H_n\} \models C$.

2.3.1 Equivalence et conséquence logique

si F_1 et F_2 sont des fbf fermées d'un langage $\mathcal{L}$ du premier ordre, on a immédiatement
$F_1 \equiv F_2$ si et seulement si $F_1 \models F_2$ et $F_2 \models F_1$

2.4 Théorème : conséquence logique et implication syntaxique

Si $H_1,\ H_2, \ldots,\ H_n$ et C sont des fbf fermées d'un langage du premier ordre les propriétés ci-dessous sont équivalentes :

1. $\{H_1,\ H_2, \ldots, H_n\} \models C$
2. $H_1 \wedge H_2 \wedge \ldots \wedge H_n \rightarrow C$ est valide
3. $H_1 \wedge H_2 \wedge \ldots \wedge H_n \wedge \neg C$ est insatisfiable

Un raisonnement consiste à obtenir une conclusion à partir d'un ensemble d'hypothèses : on formule les hypothèses $H_1,\ H_2 \ldots H_N$ (chacune étant une fbf) et qu'on en conclut C (une fbf). Ce raisonnement est correct (ou valide) si C est une conséquence logique de $H_1,\ H_2 \ldots H_N$.

3 Démonstrations d'équivalence

3.1 Renommage des variables liées

Soit une formule $Q_{tf}x\ G$ où Q_{tf} est un quantificateur et x une variable. Soit z une variable **qui n'est pas une variable libre de** G. Alors en appelant $G_{x\leftarrow z}$ la formule obtenue à partir de G en y substituant toutes les occurrences de x par un z, on a $Q_{tf}x\ G \equiv Q_{tf}z\ G_{x\leftarrow z}$

3.2 Formulaire de la logique des propositions

- Idempotence de $\wedge$ et $\vee$: $A \wedge A \equiv A$ et $A \vee A \equiv A$
- Associativité de $\wedge$ et $\vee$:
  $((A \wedge B) \wedge C) \equiv (A \wedge (B \wedge C))$ et $((A \vee B) \vee C) \equiv (A \vee (B \vee C))$
- Commutativité de $\wedge$ et $\vee$: $A \wedge B \equiv B \wedge A$ et $A \vee B \equiv B \vee A$
- Double négation : $\neg\neg A \equiv A$
- Négation, $\top$, $\bot$ et $\rightarrow$:
  - $(A \vee \neg A) \equiv \top$
  - $(A \wedge \neg A \equiv \bot$
  - $\top \wedge A \equiv A$
  - $\top \vee A \equiv \top$
  - $\bot \wedge A \equiv \bot$
  - $\bot \vee A \equiv A$
  - $\neg A \equiv (A \rightarrow \bot$
- Distributivité du $\vee$ par rapport à $\wedge$ (et vice versa) :
  $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$ et $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
- Lois de De Morgan : $\neg(A \vee B) \equiv (\neg A \wedge \neg B)$ et $\neg(A \wedge B) \equiv (\neg A \vee \neg B)$
- Implication et équivalent :
  $(A \rightarrow B) \equiv (\neg A \vee B)$ et $(A \leftrightarrow B) \equiv ((A \rightarrow B) \wedge (B \rightarrow A))$

3.3 Théorème des tautologies

- Soit F une fbf valide de la logique des **propositions** ayant $p_1,\ p_2, \ldots, p_n$ pour symboles propositionnels.
- Soient $F_1,\ F_2, \ldots, F_n$ des fbf (pas nécessairement fermées) d'un langage du premier ordre $\mathcal{L}$
- Alors F' la fbf obtenue à partir de F en substituant, pour tout i, F_i à p_i, est valide.

3.4 Substitution de variables

Définition On appelle substitution de variables l'opération qui consiste à remplacer dans une formule F toutes les occurrences libres d'une variable x par un terme t
On note ${}_{\{x\leftarrow t\}}F$ le résultat de la substitution de x par t dans F.

3.5 Théorème de substitution

Le théorème de substitution permet d'obtenir, par substitution d'une fbf à une fbf équivalente, des équivalences entre fbf du premier ordre à partir d'équivalences entre fbf du premier ordre : Soient F une fbf, S une sous-fbf de F, et S' une fbf équivalente à S. Toute fbf F' obtenue à partir de F en remplaçant une occurrence de S par une occurrence de S' est équivalente à F.

3.6 Formulaire sur les prédicats

- $\forall x\ P(x) \equiv \neg\exists x\ \neg P(x)$ et $\exists x\ P(x) \equiv \neg\forall x\ \neg P(x)$
- $\forall x\ (P(x) \wedge Q(x)) \equiv (\forall x\ P(x)) \wedge (\forall x\ Q(x))$
- $\exists x\ (P(x) \vee Q(x)) \equiv (\exists x\ P(x)) \vee (\exists x\ Q(x))$
- $(\forall x\ P(x)) \vee (\forall x\ Q(x)) \models \forall x\ (P(x) \vee Q(x))$
- $\exists x\ (P(x) \wedge Q(x)) \models (\exists x\ P(x)) \wedge (\exists x\ Q(x))$
- $\exists x\ (P(x) \rightarrow Q(x)) \ \equiv\ (\forall x\ P(x)) \rightarrow (\exists x\ Q(x))$
- si x n'est pas une variable de Q
  - $Q_{tf} x\ (F \wedge Q) \equiv (Q_{tf} x\ F) \wedge Q$ et $Q_{tf} x\ (F \vee Q) \equiv (Q_{tf} x\ F) \vee Q$
  - $\exists x\ (F \rightarrow Q(x)) \ \equiv\ F \rightarrow (\exists x\ Q(x))$
  - $\forall x\ (F \rightarrow Q(x)) \ \equiv\ F \rightarrow (\forall x\ Q(x))$

4 Les fonctions

4.1 Langage logique : syntaxe

4.1.1 Définition

Un langage logique est maintenant un triplet $\mathcal{L} = (\mathcal{C}_{ste},\ \mathcal{P}_{red},\ \mathcal{F}_{ctn})$ où
- C_{ste} est un ensemble de constantes
- P_{red} est un ensemble de symboles de prédicats avec une arité associée : $\{P_1, Q_2, R_2 \ldots\}$
- $\mathcal{F}_{ctn}$ est un ensemble des symboles de fonction chacun avec une arité associée ≥ 1
- ces trois ensembles sont disjoints deux à deux

4.1.2 Définition par induction de $\mathcal{T}$ermes($\mathcal{L}$)

- base : $\mathcal{C}_{ste} \cup \mathcal{V}_{rb} \subset \mathcal{T}\text{ermes}(\mathcal{L})$
- règle de construction : si f est un symbole de fonction $\in \mathcal{F}_{ctn}$ d'arité k et si $e_1,\ e_2 \ldots e_k$ appartiennent à $\mathcal{T}\text{ermes}(\mathcal{L})$, alors $f(e_1,\ e_2 \ldots e_k) \in \mathcal{T}\text{ermes}(\mathcal{L})$

4.1.3 Formules Bien Formées

Soit $\mathcal{L} =_{def} \mathcal{C}_{ste} \cup P_{red} \cup \mathcal{F}_{ctn}$ un langage, $\mathcal{V}_{rb}$ un ensemble infini de variables, $C =_{def} \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ les connecteurs, $\mathcal{Q}_{tf} =_{def} \{\forall, \exists\}$ les quantificateurs, et $\mathcal{P} =_{def} \{(,)\}$ un jeu de parenthèses.
On définit par induction $FBF(\mathcal{L})$, l'ensemble des formules bien formées, construites sur $\mathcal{L}$:

- base
  1. $FBF(\mathcal{L})$ contient l'ensemble des atomes
     $\{\ p(t_1, \ldots, t_n) \mid p \in \mathcal{P}_{rd}$ est un prédicat n-aire et $t_1, \ldots, t_n \in \mathcal{T}$ermes$(\mathcal{L})$ sont n termes $\}$
  2. $FBF(\mathcal{L})$ contient $\{\top, \bot\}$ dans la mesure où ces symboles sont admis.
  3. **le cas spécial du symbole d'égalité** $=$
     $FBF(\mathcal{L})$ contient l'ensemble des formules $\{t_1 = t_2$ avec t_1 et $t_2 \in \mathcal{C}_{ste} \cup \mathcal{V}_{rb}\}$
     Attention, pour des raisons qu'on verra lors de l'étude de la sémantique, on ne peut comparer deux termes quelconques, seulement des constantes et des variables.
- induction : soit A et $B \in FBF(\mathcal{L})$ et soit $x \in \mathcal{V}_{rb}$:
  - $\neg A \in FBF(\mathcal{L})$
  - $(A \wedge B) \in FBF(\mathcal{L})$ (idem avec $(A \vee B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$)
  - $\forall x A \in FBF(\mathcal{L})$ (idem avec $\exists x A$)

4.2 Langage logique : sémantique

4.2.1 Définition d'une interprétation

pour tout f de $\mathcal{F}$, $\mathcal{I}(f)$ est une application de $\mathcal{D}^{arit(f)}$ dans $\mathcal{D}$.

4.2.2 Définition d'une valeur de vérité d'une fbf pour une interprétation $\mathcal{I}$ et une assignation σ

rien ne change formellement, simplement les termes sont maintenant des éléments dont l'interprétation peut nécessiter un calcul.

C'est pourquoi il faut rajouter au calcul de la valeur de vérité d'une formule :

si A est un atome $p(t_1, t_2, ..., t_n)$, $Val(p(t_1, t_2, ..., t_n), \mathcal{I}, \sigma) =$ vrai ssi :
il existe $(d_1 \ldots d_n) \in \mathcal{I}(p)$ avec, pour tout i, $d_i = \mathcal{I}_s(t_i)$ où $\mathcal{I}_s$ associe un élément de $\mathcal{D}$ à tout terme :

- si t_i est une constante, $\mathcal{I}_s(t_i) = \mathcal{I}(t_i)$ (on prend l'interprétation de la constante)
- si t_i est une variable, $\mathcal{I}_s(t_i) = \sigma(t_i)$ (on prend l'assignation de la variable)
- si $t_i = f(t_1, ..., t_k)$ alors $\mathcal{I}_s(t_i) = \mathcal{I}(f)\ (\mathcal{I}_s(t_1), \ldots, \mathcal{I}_s(t_k))$

5 Une méthode de preuve : la méthode de résolution

5.1 Forme prénexe

5.1.1 Définition d'une forme prénexe

Une fbf est sous forme prénexe lorsqu'elle s'écrit sous forme d'une accumulation de quantificateurs suivie d'une fbf sans quantificateur : $Q_{tf_1} x_1 \; Q_{tf_2} x_2 \ldots Q_{tf_n} x_n \; G$
où $Q_{tf_i} \in \{\forall, \exists\}$ et G ne contient pas de quantificateurs.

5.1.2 Schéma d'algorithme de mise sous forme prénexe

Données: une fbf **fermée** F
Résultat: une fbf **fermée** F' sous forme prénexe telle que $F \equiv F'$

1. Renommer les variables de façon à ce qu'aucune variable ne soit liée par deux quantificateurs
2. éliminer les connecteurs logiques binaires autre que $\vee$ et $\wedge$ (c'est à dire éliminer $\rightarrow$ et $\leftrightarrow$)
3. faire "rentrer" les $\neg$ de façon à ce qu'ils ne portent que sur des atomes et en profiter pour éliminer les doubles négations
4. faire "remonter" les quantificateurs en tête de formule

Formulaire utilisé pour le pas 3
- $\neg[\forall x \; P] \equiv \exists x \; \neg P$
- $\neg[\exists x \; P] \equiv \forall x \; \neg P$
- $\neg[P \wedge Q] \equiv \neg P \vee \neg Q$
- $\neg[P \vee Q] \equiv \neg P \wedge \neg Q$

5.2 Forme clausale

Rappel
Une **clause** est une disjonction de littéraux

Définition d'une forme clausale
C'est la fermeture universelle d'une conjonction de clauses

le problème
Éliminer les $\exists$

Propriété
On ne peut pas toujours mettre une fbf sous une forme clausale équivalente.

5.2.1 Schéma d'algorithme de skolémisation

Données: une fbf **fermée** F
Résultat: une fbf **fermée** F' sous forme clausale, appelée forme de Skolem de F

1. mettre F sous forme prénexe
2. distribuer les $\wedge$ et les $\vee$ de façon à obtenir une conjonction de clauses
3. (c'est ici qu'on perd l'équivalence avec F) :
   Remplacer les $\exists x$ par des **fonctions de Skolem**
   - qui sont de nouvelles fonctions (n'appartenant pas au langage initial)
   - en procédant de la gauche vers la droite :
     - soient $\forall y_1\ \forall y_2 \ldots\ \forall y_n$ les quantificateurs universels qui précèdent $\exists x$
     - supprimer $\exists x$ et remplacer chaque occurrence de x par le même terme $f(y_1, y_2 \ldots y_n)$ dans lequel f est un nouveau symbole de fonction
     - si $n = 0$ (aucun quantificateur universel ne précède $\exists x$) alors f est une nouvelle constante (une fonction d'arité 0)
   - la fbf obtenue est appelée (une) forme de Skolem de F

5.3 Unification de termes

5.3.1 Définition d'une substitution

Une substitution est un ensemble de couples $s = (x_1, t_1), (x_2, t_2), \ldots, (x_k, t_k)$ où les x_i sont des variables et les t_i des termes (variables, constantes, fonctions d'arité non nulle avec leurs arguments), avec : pour tout i et tout $j \neq i$, on a $x_i \neq x_j$ («on ne remplace pas 2 fois une même variable»)

Appliquer $\sigma = (x_1, t_1), (x_2, t_2), \ldots, (x_k, t_k)$ à une formule A consiste à remplacer toute variable x_i apparaissant dans A par t_i.Le résultat est noté $\sigma(A)$. On peut de la même façon appliquer σ à un terme, ou à une liste de termes. Attention, lors de l'application d'une substitution, on remplace simultanément (autrement dit "en parallèle") chaque variable.

5.3.2 Définition d'un unificateur

Des listes de termes $l_1, \ldots, l_n$ sont unifiables s'il existe une substitution σ qui les rend identiques : $\sigma(l_1) = \ldots = \sigma(l_n)$. σ est appelé un unificateur de ces listes de termes.
On étend cette définition à des atomes : des atomes sont unifiables s'ils ont même prédicat et si leurs listes de termes sont unifiables.

Définition d'un upg
Un unificateur u d'un ensemble $\mathcal{E}$ est un unificateur le plus général (upg) si tout autre unificateur u' de $\mathcal{E}$ s'obtient par une substitution supplémentaire σ : $u'\mathcal{E}) = \sigma(u(\mathcal{E}))$, ou encore $u'(\mathcal{E}) = \sigma \circ u(\mathcal{E})$.

$\mathcal{E}$ peut avoir plusieurs upg : dans ce cas, ils s'obtiennent les uns des autres avec un renommage de variables (c'est-à-dire : dans la définition ci-dessus, σ ne comporte que des couples de variables, et deux variables différentes sont remplacées par des variables différentes).

5.4 Algorithme de Robinson

Comment obtenir l'upg de deux termes ? (s'ils sont unifiables)

Ensemble de désaccord

On voit les deux termes comme des mots. Soit L_P leur plus grand préfixe commun ; l'ensemble de désaccord est formé des deux **sous-termes** commençant après ce préfixe

Données: Deux atomes avec le même prédicat (ou deux listes de termes) L_1 et L_2
Résultat: un upg de L_1 et L_2 s'ils sont unifiables ; sinon le signal "échec"

$U = \emptyset$; `// l'upg éventuel`
tant que *l'ensemble de désaccord de* L_1 *et* L_2 *n'est pas vide* **faire**
  si *l'un des termes de l'ensemble de désaccord est une variable* x *et l'autre un terme* t ***qui ne contient pas*** x **alors**
    $\sigma \leftarrow \{(x,t)\}$; $L_1 = \sigma(L_1)$; $L_2 = \sigma(L_2)$; $U = \sigma \circ U$;
  sinon
    retourner *"Échec"*
  fin
fin
retourner U;

5.5 Théorème de Robinson

Une forme clausale S est insatisfiable si et seulement si on peut obtenir la clause vide en utilisant un nombre fini de fois la règle de résolution à partir des clauses de S.

Schéma d'algorithme de la méthode de résolution

Données: $\mathcal{S}$ un ensemble de clauses skolémisées
Résultat:
– "vrai" si $\mathcal{S}$ est insatisfiable
– "faux" si $\mathcal{S}$ est satisfiable
– peut ne jamais s'arrêter dans ce deuxième cas

$EnsClauses \leftarrow \mathcal{S}$;
tant que *Vrai* **faire**
  si *EnsClauses contient deux clauses* C_1 *et* C_2 *telles que, après un éventuel renommage des variables de* C_2 *de façon à ce que* C_1 *et* C_2 *n'aient aucunes variables en commun, on puisse produire par la règle de résolution une nouvelle clause* C *à partir de* C_1 *et* C_2*, c'et à dire que on peut trouver* $L_1 \subseteq C_1$ *et* $L_2 \subseteq C_2$ *tels que* L_1 *et* L_2 *sont unifiables selon un upg* U *et en appelant* C *la résolvante de* C_1 *et* C_2 *selon* L_1, L_2 *et* U *(*$C = Res(C_1, C_2, L_1, L_2)_{selon\ U}$*), au cas où* $C \notin EnsClauses$ *(à un renommage des variables près)* **alors**
    Ajouter C à $EnsClauses$;
    si $C = \emptyset$ *(clause vide)* **alors retourner** *vrai* `//` $\mathcal{S}$ `est insatisfiable`
  sinon
    retourner *faux* `//` $\mathcal{S}$ `est satisfiable`
  fin
fin

6 Les stratégies de résolution

Théorème de Church (1936)
Le problème de la validité d'une formule n'est pas décidable (il n'existe pas d'algorithme qui puisse décide pour **toute** formule et en un temps **fini** si la formule est **valide**.

Définition d'une stratégie de résolution
C'est une façon d'appliquer plusieurs fois de suite des résolutions.
En effet, le schéma d'algorithme donné est indéterministe, quand il y a plusieurs choix possibles pour les clauses C_1 et C_2 ou plusieurs façons d'unifier les deux clauses choisies.
En outre ce choix est crucial : si on fait les choix judicieux on peut produire la clause vide mais on peut aussi –à priori- procéder à des choix moins judicieux produisant indéfiniment des résolvantes sans ne jamais aboutir à la clause vide de sorte que l'utilisateur soit indéfiniment indécis sur une réponse à apporter. A ce stade, cette procédure n'est pas un algorithme !
Une stratégie est **complète** si elle permet de trouver la clause vide quand la forme clausale de départ est insatisfiable.
Elle sera dite **effectivement complète** lorsque l'utilisateur appliquant cette stratégie à une forme clausale invalide est assuré d'aboutir à la clause vide au bout d'un temps fini. Une stratégie complète non effectivement complète offre un choix à l'utilisateur qui peut le faire aboutir à une impasse (un bouclage par exemple).

Stratégie en largeur On construit une suite **d'ensembles** de clauses :
- $\mathcal{C}_0$ l'ensemble de toutes les clauses **initiales**
- $\mathcal{C}_1$ l'ensemble de toutes les clauses obtenues par résolution de deux clauses de $\mathcal{C}_0$
- $\mathcal{C}_2$ l'ensemble de toutes les clauses obtenues par résolution de deux clauses dont l'une dans $\mathcal{C}_1$ et l'autre dans $\mathcal{C}_0 \cup \mathcal{C}_1$
- ...
- $\mathcal{C}_i$ l'ensemble de toutes les clauses obtenues par résolution de deux clauses dont l'une dans $\mathcal{C}_{i-1}$ et l'autre dans $\bigcup_{k\in[0\ldots i-1]} \mathcal{C}_k$
- ...

Cette stratégie est
- complète : si la plus petite suite de résolutions permettant de construire la clause vide est de longueur l, cette stratégie construira la clause vide en construisant $\mathcal{C}_l$
- mais terriblement inefficace : $|\mathcal{C}_i| \geq 2^{|\mathcal{C}_0|}$

Autres stratégies Il existe des stratégies plus efficaces, mais alors elles ne seront pas complètes. Par exemple

Choisir toujours deux clauses, parmi celles qui ont le moins de littéraux : incomplète

Stratégie linéaire par entrée
A chaque étape, on utilise au moins une clause de l'ensemble initial.

Définition d'une clause de Horn
- Une clause de Horn est une clause ne comportant au plus qu'un littéral positif
- Une clause (de Horn) contenant exactement un littéral positif est appelée une clause **définie**.
- Une clause (de Horn) ne contenant aucun littéral positif est appelée clause négative ou **but**.

Table des matières