LRC - Annexe n°2

Autre système de preuve : la méthode de résolution

1 Logique propositionnelle

1.1 Composantes

La méthode de résolution est basée sur une unique règle d'inférence : C_1 et C_2 étant des formules et p une variable propositionnelle, on a

$$\frac{C_1 \vee p \qquad C_2 \vee \neg p}{C_1 \vee C_2}$$

1.2 Déduction

La déduction d'une formule A dans une théorie Δ se fait algorithmiquement de la façon suivante :

- Réfutation de la conclusion : $\neg A$
- Mise sous forme normale conjonctive des formules de Δ et de $\neg A$: obtention d'un ensemble de clauses $\mathcal{C} = \{C_1, ..., C_p\}$
- Itération de la règle de résolution jusqu'à obtention de la clause vide
  - sélection d'une paire de clauses (C_i, C_j) dans $\mathcal{C}$ contenant respectivement un littéral et sa négation : $C_i = C_1 \vee p$ et $C_j = C_2 \vee \neg p$
  - calcul de la résolvante $C' = C_1 \vee C_2$ par application de la règle de résolution
  - mise à jour de $\mathcal{C} \leftarrow \mathcal{C} \cup \{C'\}$

Il existe de nombreuses stratégies pour le choix des clauses et des littéraux considérés (par exemple stragtégies linéaire, ordonnée, input, SLD, ...).

Le principe de résolution est l'une des bases du langage Prolog.

2 Logique des prédicats du premier ordre

Etapes préliminaires La méthode de résolution en LPPO s'applique à des formules d'un type particulier, les formules closes dont toutes les variables sont universellement quantifiées.

On peut montrer que, pour toute formule F, il existe une formule F' qui est satisfiable si et seulement si F l'est : F et F' sont dites équisatisfiables (elles ne sont pas équivalentes, car elles ne sont pas définies par la même sémantique). La construction de F' se fait en 2 étapes, appelées mise sous forme prénexe et skolémisation.

On considère dans cette UE directement des formules closes dont toutes les variables sont universellement quantifiées et on omet d'écrire ces quantificateurs $\forall$.

2.1 Unification

On rappelle que deux termes t_1 et t_2 sont *unifiables* s'il existe une substitution des variables σ qui les rend identiques, c'est-à-dire telle que $t_1\sigma = t_2\sigma$.

Un problème d'unification peut avoir une infinité de solutions : par exemple $f(a, x) = f(a, g'y))$ est unifiable avec $\sigma_1 : x \mapsto g(f(a, a)), y \mapsto f(a, a)$, avec $\sigma_2 : x \mapsto g(f(a, w)), y \mapsto f(a, w)$, avec $\sigma_3 : x \mapsto g(z), y \mapsto z$. On s'intéresse à l'unificateur le plus général (*most general unifier*, *mgu*).

Il existe de nombreux algorithmes d'unification, de complexités variables, on présente ici l'algorithme de Martelli et Montanari (1982), défini par un ensemble de règles.

On note E un ensemble d'équations à unifier, on construit la substitution σ récursivement. Dans ce qui suit, on note x une variable, alors que u, u_i et v_i sont des termes.

L'algorithme n'est pas déterministe, cependant l'ordre des opérations n'a pas d'importance. Il termine toujours, soit pas un échec, soit avec $E_n = \emptyset$. S'il termine par $E_n = \emptyset$, alors σ_n est le *mgu* de E, s'il termine par échec, alors E n'a pas d'unificateur. La complexité est exponentielle.

Initialisation $E_0 = E$ et $\sigma_0 = id$

Itération Tant que $E_n \neq \emptyset$ appliquer l'une des règles suivantes

- Si $E_n = E' \cup \{f(u_1, .., u_q) \sim g(v_1, ..., v_p)\}$
  - si $f = g$ et $q = p$, poser $E_{n+1} = E' \cup \{u_1 \sim v_1, .., u_q \sim v_q\}$ et $\sigma_{n+1} = \sigma_n$ (règle : décomposer)
  - sinon, renvoyer "Echec" (règle : échec par conflit)
- Si $E_n = E' \cup \{x \sim x\}$, poser $E_{n+1} = E_n$ et $\sigma_{n+1} = \sigma_n$ (règle : supprimer)
- Si $E_n = E' \cup \{x \sim u\}$ avec $u \neq x$
  - si x n'apparaît pas dans u, poser $E_{n+1} = E'[x := u]$ et $\sigma_{n+1} = [x := u] \circ \sigma_n$ (règle : éliminer)
  - sinon, renvoyer "Echec" (règle : échec par *occurs-check*)

2.2 Règle d'inférence

$$\frac{C_1 \vee L_1 \qquad C_2 \vee L_2}{(C_1 \vee C_2)\sigma}\,\sigma$$

où σ est une substitution telle que $L_1\sigma = \neg(L_2\sigma)$, c'est à dire un unificateur de L_1 et $\neg L_2$.

2.3 Déduction

Elle est identique à celle établie dans le cas de la logique des propositions, à la différence des étapes intermédiaires et de l'unification :

La déduction d'une formule A dans une théorie Δ se fait algorithmiquement de la façon suivante :

- Réfutation de la conclusion : $\neg A$
- Mise sous forme prénexe, skolémisée et normale conjonctive des formules de Δ et de $\neg A$: obtention d'un ensemble de clauses $\mathcal{C} = \{C_1, ..., C_p\}$
- Itération de la règle de résolution jusqu'à obtention de la clause vide
  - sélection d'une paire de clauses (C_i, C_j) dans $\mathcal{C}$ contenant deux clauses $C_i = C_1 \vee L_1$ et $C_j = C_2 \vee L_2$ telles qu'il existe une substitution σ telle que $L_1\sigma = \neg(L_2\sigma)$
  - calcul de la résolvante $C' = (C_1 \vee C_2)\sigma$ par application de la règle de résolution
  - mise à jour de $\mathcal{C} \leftarrow \mathcal{C} \cup \{C'\}$