Logique 2023-2024

Département Informatique, ENS Paris Saclay

TD 07 : Calcul des séquents, Unification

Nicolas Margulies nicolas.margulies@lmf.cnrs.fr
Théo Vignon theo.vignon@lmf.cnrs.fr

Nous considérons le calcul des séquents tel que vu dans le cours, c'est-à-dire le calul des séquents sans coupures.

1 Preuves en calcul des séquents

Donnez des preuves dans le calcul des séquents des formules suivantes :

1. $A \vee (A \Rightarrow B)$
2. $A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$
3. $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$
4. $\neg\neg A \Rightarrow A$
5. $\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$
6. $\neg\forall x.\, R(x) \Rightarrow \exists x.\, \neg R(x)$
7. $\forall x.\, (Q \vee R(x)) \Rightarrow (Q \vee \forall x.\, R(x))$
8. $\exists x.\, [(R(a) \vee R(b)) \Rightarrow R(x)]$

9. Montrez que si $a \neq b$, alors il n'existe pas de preuve de 8. qui n'utilise pas de contractions.

2 Quelques exemples d'unification

Un *problème d'unification* est un ensemble E d'équations de la forme $t \stackrel{?}{=} u$.
Un *unificateur* (i.e. une solution au problème d'unification) d'un ensemble E est une substitution σ telle que pour toute équation $t \stackrel{?}{=} u$ dans E, $t\sigma = u\sigma$.

La procédure d'unification vue en classe est rappelée à la fin de la feuille de TD (Algorithme 1).

Appliquez la procédure aux problèmes d'unification suivants (votre réponse devrait être soit `fail` soit la substitution retournée par la procédure) :

— $E_1 = f(x, g(a, y)) \stackrel{?}{=} f(h(y), g(y, a))\,;\; g(x, h(y)) \stackrel{?}{=} g(z, z)$

— $E_2 = f(x, x) \stackrel{?}{=} f(g(y), z)\,;\; h(z) \stackrel{?}{=} h(y)$

— $E_3 = f(x, a) \stackrel{?}{=} f(b, y)\,;\; f(x) \stackrel{?}{=} f(y)$

3 Théorème d'interpolation

Si φ est une formule, on appelle $L(\varphi)$ l'ensemble des variables libres, symboles de fonction et symboles de prédicat apparaissant dans φ. Par extension, si Γ est un multi-ensemble de formules, on définit $L(\Gamma) = \bigcup_{\varphi \in \Gamma} L(\varphi)$.

On souhaite montrer que si $L(\Gamma_1 \cup \Gamma_2 \cup \Delta_1 \cup \Delta_2)$ ne contient pas de symboles de fonction, et si $\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2$ est prouvable, alors il existe une formule ξ telle que :

— $\Gamma_1 \vdash \xi, \Delta_1$ and $\Gamma_2, \xi \vdash \Delta_2$ sont prouvables

— $L(\xi) \subseteq L(\Gamma_1 \cup \Delta_1) \cap L(\Gamma_2 \cup \Delta_2)$

1. Prouvez ce résultat. On considèrera en particulier les cas suivants : ax ; $\Rightarrow_{right}$; $\Rightarrow_{left}$; $\forall_{left}$; $\forall_{right}$; $\exists_{left}$; $\exists_{right}$.
2. Prouvez le *théorème d'interpolation* : si φ et ψ sont des formules sans symboles de fonctions, et si $\vdash \varphi \Rightarrow \psi$ est prouvable, alors il existe une formule ξ telle que :

   — $\vdash \varphi \Rightarrow \xi$ et $\vdash \xi \Rightarrow \psi$ sont prouvables

   — $L(\xi) \subseteq L(\varphi) \cap L(\psi)$.
3. ($\star$) Nous appliquons ceci à la preuve du théorème de BETH.

   Soient P et P' soit $\Gamma(P)$ un ensemble de formules closes ne contenant pas le symbole P' ou de symboles de fonctions. On note $\Gamma(P')$ l'ensemble de symboles obtenu en remplaçant toutes les occurences de P par P' dans $\Gamma(P)$.

   On dit que $\Gamma(P)$ *définit implicitement* P si $\Gamma(P), \Gamma(P') \vdash \forall x.\ (P(x) \Leftrightarrow P'(x))$ est prouvable. On dit que $\Gamma(P)$ *définit explicitement* P s'il existe une formule $\varphi(x)$ n'utilisant ni P ni P' telle que $\Gamma(P) \vdash \forall x.\ (\varphi(x) \Leftrightarrow P(x))$.

   Prouvez que $\Gamma(P)$ définit implicitement P si et seulement si $\Gamma(P)$ définit explicitement P.
4. ($\star\star$) En réalitén le théorème d'interpolation reste vrai même quand les formules impliquées contiennent des symboles de fonctions. Voyez-vous comment adapter la preuve à ce cas ?

4 Étude de l'algorithme d'unification

Nous étudions les propriétés de l'algorithme d'unification `Unif`.

1. L'algorithme présenté est-il déterministe ?
2. Montrez que l'algorithme termine (en échouant ou en renvoyant une substitution).
3. On cherche maintenant à prouver que l'algorithme calcule effectivement un unificateur de l'ensemble d'équations donné en entrée (quand ceci existe). Nous prouvons même un résultat plus général.

   (a) Montrez que si E contient $f(u_1, ..., u_n) \stackrel{?}{=} g(t_1, ..., t_m)$, avec $f \neq g$, alors E n'est pas unifiable.

   (b) Montrez que si E contient $x \stackrel{?}{=} t$, où $x \in \mathcal{X}$, $t \notin \mathcal{X}$ et $x \in Var(t)$, alors E n'est pas unifiable.

   > Un *unificateur le plus général* (mgu) σ d'un problème d'unification E est un unificateur de E tel que pour tout unificateur τ de E, il existe η telle que $\tau = \eta \circ \sigma$.

   (c) Montrez que le problème d'unification $x \stackrel{?}{=} f(y)$ possède une infinité d'unificateurs. En existe-t-il un le plus général ? Lorsqu'il existe, y a-t-il unicité du mgu ?

(d) Montrez que si $E = E' \cup \{ x \stackrel{?}{=} x \}$, alors σ unifie E ssi σ unifie E'.
Observez de plus que ceci implique que σ est un mgu de E ssi σ est un mgu de E'.

(e) Montrez que si $E = E' \cup \{ f(u_1, \ldots, u_n) \stackrel{?}{=} f(t_1, \ldots, t_n) \}$, alors σ unifie E ssi σ unifie $E' \cup \{u_1 \stackrel{?}{=} t_1, \ldots, u_n \stackrel{?}{=} t_n\}$.
Observez de plus que ceci implique que σ est un mgu de E ssi σ est un mgu de E'.

(f) Montrez que pour toute substitution σ, variable x et terme t, si $x\sigma = t\sigma$ alors $\sigma = \sigma \circ \{x \mapsto t\}$.

(g) Montrez que si $E = E' \cup \{ x \stackrel{?}{=} t \}$ où $x \notin var(t)$, alors σ unifie $E'\{x \to t\}$ ssi $\sigma \circ \{x \mapsto t\}$ unifie E.
Montrez grâce à la question précédente que : σ est un mgu de $E'\{x \to t\}$ implique que $\sigma \circ \{x \mapsto t\}$ est un mgu de E.

(h) Montrez, étant donné une entrée E, que
— si l'algorithme retourne une substitution σ, alors σ est un mgu de E ;
— si l'algorithme échoue, alors le problème d'unification E n'a pas de solution.

L'algorithme d'unification

La procédure d'unification est la suivante :

Algorithm 1: `Unif`

Input : a unification problem E
if $E = E' \cup \{f(u_1, ..., u_n) \stackrel{?}{=} f(t_1, ..., t_n)\}$ **then**
  `Unif`($E' \cup \{u_1 \stackrel{?}{=} t_1, \ldots, u_n \stackrel{?}{=} t_n\}$)
else if $E = E' \cup \{f(u_1, ..., u_n) \stackrel{?}{=} g(t_1, ..., t_m)\}$ *where* $f \neq g$ **then**
  `fail`
else if $E = E' \cup \{x \stackrel{?}{=} x\}$ **then**
  `Unif`(E')
else if $E = E' \cup \{x \stackrel{?}{=} t\}$ *where* $x \in \mathcal{X}$ *and* $x \notin Var(t)$ **then**
  `Unif`($E'\{x \to t\}$) $\circ \{x \mapsto t\}$
else if $E = E' \cup \{x \stackrel{?}{=} t\}$ *where* $x \in \mathcal{X}$, $t \notin \mathcal{X}$ *and* $x \in Var(t)$ **then**
  `fail`
else if $E = \emptyset$ **then**
  id