Programmation fonctionnelle – TD2

S5 – 2023/2024

1 Typage d'expressions [★]

Pour chacune de ces expressions, déterminez le type puis donnez la preuve de typage :

1. $3 + (5 + 1)$
2. *if* $(3 < 1)$ *then true else* $(2 < 3)$
3. $(\mathit{if}\ \mathit{false}\ \mathit{then}\ 1\ \mathit{else}\ (2 + 4)) < 2$
4. $5 + (\mathit{if}\ (1 < (2 + 1))\ \mathit{then}\ 1\ \mathit{else}\ 2)$

2 Lambda-termes typés [★]

Pour chacun de ces types, donnez un exemple de lambda-terme :

1. $\mathbb{N}$
2. $\mathbb{B}$
3. $\mathbb{N} \to \mathbb{B}$
4. $\mathbb{N} \to \mathbb{N} \to \mathbb{N}$
5. $(\mathbb{N} \to \mathbb{B}) \to \mathbb{N} \to \mathbb{B}$

3 Vérification de type [★]

Pour chacun de ces jugements de typage, donnez une preuve :

1. $\vdash \lambda(n : \mathbb{N}).\, n + 1 : \mathbb{N} \to \mathbb{N}$
2. $\vdash (\lambda(b : \mathbb{B}).\ \mathit{if}\ b\ \mathit{then}\ 0\ \mathit{else}\ 1)\ \mathit{false} : \mathbb{N}$
3. $\vdash \lambda(n_1 : \mathbb{N}).\lambda(n_2 : \mathbb{N}).\ \mathit{if}\ (n_1 < n_2)\ \mathit{then}\ n_1\ \mathit{else}\ n_2 : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$
4. $f : \mathbb{N} \to \mathbb{N} \to \mathbb{N} \vdash f\ (5 + 2) : \mathbb{N} \to \mathbb{N}$
5. $g : \mathbb{N} \to \mathbb{B} \to \mathbb{N} \vdash \lambda(b : \mathbb{B}).\, \lambda(n : \mathit{nat}).\, g\, n\, b : \mathbb{B} \to \mathbb{N} \to \mathbb{N}$

4 Inférence de type [★★]

Pour chacun de ces lambda-termes, déterminez le type puis donnez la preuve de typage :

1. $\lambda(x : \mathbb{N}).\, x + 1$
2. $(\lambda(x : \mathbb{N}).\, x < 0)\ 3$
3. $\lambda(p : \mathbb{N} \to \mathbb{B}).\lambda(n : \mathbb{N}).\ \mathit{if}\ (p\, n)\ \mathit{then}\ n\ \mathit{else}\ 0$
4. $\lambda(b : \mathbb{B}).\ \mathit{if}\ b\ \mathit{then}\ (\lambda(n : \mathbb{N}).\, n + 1)\ \mathit{else}\ (\lambda(n : \mathbb{N}).\, 2)$

5 Programmer avec des types abstraits [★★]

On étend le système de typage avec trois types α, β et γ, dont on ignore les valeurs. Pour chacun des types suivants, donnez un lambda-terme :

1. $\alpha \to \alpha$
2. $\alpha \to \mathbb{N}$
3. $\alpha \to \beta \to \alpha$
4. $(\alpha \to \beta) \to (\beta \to \gamma) \to \alpha \to \gamma$
5. $(\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma$

6 Les limites du lambda calcul typé [★★]

Dans le TD précédent, nous avons vu que le lambda-terme $(\lambda x.\, x\, x)$ permettait une forme basique de récursion. Est-il possible de trouver une type $T_?$ qui permette de typer le terme suivant ? Expliquez pourquoi.

$$\lambda(x : T_?).\, x\, x$$

7 Pairs typées [★★★]

Dans le TD précédent, nous avons utilisé les termes *pair*, *first*, *second*, *true* et *false* pour coder des paires.

1. Dans le lambda-calcul typé, on souhaite à présent coder des paires d'éléments de type $\mathbb{N}$. Déterminez le type des cinq lambda-termes précédents qui permette ce codage (il faudra ajouter des annotations de type pour les variables des lambda-abstractions).
2. Est-il possible de changer les types afin de coder des paires où le premier élément est de type $\mathbb{N}$ et le second de type $\mathbb{B}$? Expliquez pourquoi.