

Série d'exercices, ex-examen

IFT-2035

October 14, 2009

1 Syntaxe

Soient les expressions suivantes en notation préfixe:

1. $+ * a b c$
2. $* a + / b c d$
3. $\wedge a \wedge b - c 2$
4. $/ a / b + c d$
5. $= * a + b c \text{sqrt } d$

Après s'être souvenu que la mise à la puissance (représentée par le symbole \wedge) est associative à droite, écrire ces expressions en format postfixe, infix, et en arbre de syntaxe abstraite. Utilisez la notation a^b à la place de $a \wedge b$ dans la notation infix. Utilisez un strict minimum de parenthèses.

2 Types

Donner le type le plus générique des expressions Haskell suivantes:

1. $[]$
2. $\text{let } id \ x = x \ \text{in } id$
3. $(\lambda x \rightarrow [x]) \ 5$
4. $(:) \ 5$
5. $\lambda x \rightarrow \lambda y \rightarrow x \ y$
6. $\lambda x \rightarrow \lambda y \rightarrow x \ x$
7. $\text{map } (\lambda x \rightarrow 1)$
8. $\lambda f \rightarrow \lambda x \rightarrow f \ (x \ (\text{fst } f))$

9. $map\ fst$

10. $(\lambda x \rightarrow \lambda y \rightarrow y\ x)\ 5$

Précisions:

- $\lambda x \rightarrow e$ s'écrit en réalité $\backslash x \rightarrow e$
- La composition de fonction $e_1 \circ e_2$ s'écrit en réalité $e1 \ .\ e2$
- Les fonctions map et fst sont (pré)définies comme suit:

$$\begin{aligned}map\ f\ [] &= [] \\map\ f\ (x : xs) &= f\ x : map\ f\ xs \\fst\ (x, y) &= x\end{aligned}$$

3 Portée

Soit le code ci-dessous écrit dans un langage à la syntaxe identique à celle de Haskell:

```
let x = 5 in
let f1 x y = x + y in
let f2 x y = f1 y x in
f2 7 13
```

Montrer les étapes de l'évaluation d'abord dans le cas où le langage utilise la portée statique, puis dans le cas où il utilise la portée dynamique. Dans les deux cas, utiliser une notation basée sur les environnements et non sur les substitutions.

4 Fonctions et données

Définir les constructeurs booléens $myTrue$ et $myFalse$ et le destructeur correspondant $myIf$ en utilisant seulement des expressions λ . I.e. sans utiliser aucun type prédéfini (en particulier pas le type `Bool` et donc ni la forme `if` ni les constantes `True` ou `False`) ni déclarer de nouveau type.

$$\begin{aligned}myTrue &= \dots \\myFalse &= \dots \\myIf &= \dots\end{aligned}$$

de telle sorte que les équivalences ci-dessous soient vraies:

$$\begin{aligned}myIf\ myTrue\ e_1\ e_2 &\simeq e_1 \\myIf\ myFalse\ e_1\ e_2 &\simeq e_2\end{aligned}$$

5 Passages de paramètres

Pour chacune des 4 méthodes de passages de paramètres vues au cours:

- donner une description textuelle concise (au plus 3 lignes).
- donner un exemple de code (dans un langage hypothétique avec une syntaxe similaire à celle de C) pour lequel cette méthode de passage de paramètres donne un résultat différent de chacune des 3 autres méthodes.