

Sémantique et traduction des langages

Interprétation d'un sous-ensemble de Caml : mini-ML

1 Grammaire du langage complet

La syntaxe du langage mini-ML que nous utiliserons comme support est définie par la grammaire suivante :

$$\begin{aligned}
 Expr &\rightarrow Ident \\
 &\quad | \quad Const \\
 &\quad | \quad Expr \text{ Binaire } Expr \\
 &\quad | \quad Unaire Expr \\
 &\quad | \quad \mathbf{fun} \text{ Ident } \rightarrow Expr \\
 &\quad | \quad (Expr) Expr \\
 &\quad | \quad \mathbf{if} Expr \mathbf{then} Expr \mathbf{else} Expr \\
 &\quad | \quad \mathbf{let} \text{ Ident } = Expr \mathbf{in} Expr \\
 &\quad | \quad \mathbf{letrec} \text{ Ident } = Expr \mathbf{in} Expr \\
 &\quad | \quad (Expr) \\
 &\quad | \quad Expr ; Expr \\
 &\quad | \quad \mathbf{ref} Expr \\
 &\quad | \quad ! Expr \\
 &\quad | \quad Ident := Expr \\
 \\
 Const &\rightarrow entier \mid boolean \\
 \\
 Unaire &\rightarrow - \mid ! \\
 \\
 Binaire &\rightarrow + \mid - \mid * \mid / \mid \% \mid \& \mid | \\
 &\quad | \quad == \mid != \mid < \mid <= \mid > \mid >=
 \end{aligned}$$

2 Typage

Les types possibles pour une expression du langage mini-ML sont décrits par la syntaxe suivante :

$$\begin{aligned}
 \tau &\rightarrow \alpha \mid \mathbf{int} \mid \mathbf{bool} \\
 &\quad | \quad \tau \rightarrow \tau \\
 &\quad | \quad (\tau)
 \end{aligned}$$

Un jugement de typage s'écrit sous la forme $\sigma \vdash e : \tau$.

Rappelons l'ensemble des axiomes et des règles de déduction correspondant au typage de mini-ML.

Accès à l'environnement

$$\frac{x \in \sigma \quad \sigma(x) = \tau}{\sigma \vdash x : \tau}$$

Opérateur binaire

$$\frac{\sigma \vdash e_1 : \tau_1 \quad \sigma \vdash e_2 : \tau_2 \quad \tau_1 \times \tau_2 = \text{dom } op \quad \tau = \text{codom } op}{\sigma \vdash e_1 \text{ op } e_2 : \tau}$$

Opérateur unaire

$$\frac{\sigma \vdash e : \tau \quad \tau = \text{dom } op \quad \tau' = \text{codom } op}{\sigma \vdash op \ e : \tau'}$$

Conditionnelle

$$\frac{\sigma \vdash e_1 : \text{bool} \quad \sigma \vdash e_2 : \tau \quad \sigma \vdash e_3 : \tau}{\sigma \vdash \text{if } e_1 \text{ then } e_2 \text{ else } e_3 : \tau}$$

Définition locale

$$\frac{\sigma \vdash e_1 : \tau_1 \quad \sigma :: \{x : \tau_1\} \vdash e_2 : \tau_2}{\sigma \vdash \text{let } x = e_1 \text{ in } e_2 : \tau_2}$$

Définition de fonction

$$\frac{\sigma :: \{x : \tau_1\} \vdash e : \tau_2}{\sigma \vdash \text{fun } x \rightarrow e : \tau_1 \rightarrow \tau_2}$$

Appel de fonction

$$\frac{\sigma \vdash e_1 : \tau_1 \rightarrow \tau_2 \quad \sigma \vdash e_2 : \tau_1}{\sigma \vdash (e_1) e_2 : \tau_2}$$

Définition récursive

$$\frac{\sigma :: \{x : \tau_1\} \vdash e_1 : \tau_1 \quad \sigma :: \{x : \tau_1\} \vdash e_2 : \tau_2}{\sigma \vdash \text{letrec } x = e_1 \text{ in } e_2 : \tau_2}$$

Gestion des erreurs

Il faut ajouter à ces règles, celles d'apparition et propagation des erreurs.