Bootstrap

Erik Helmers

February 16, 2023

Contents

1 NO Design et implémentation d'une syntaxe support

L'objectif du sujet est de développer un certain moteur de typage. Pour commencer, on va donc créer un petit language interpreté qui servira de base pour pratiquer nos expérimentations.

Pour rendre agréable le travail, on a ajouté de la syntaxe (`let id = ... in ...`, `let cons x y = x`, etc..) en veillant à ne pas ajouter de nouveau termes (par exemple les assignations sont juste des applications de fonctions).

$$\text{letter} ::= \text{A} \ldots \text{Z} \mid \text{a} \ldots \text{z}$$

$$\text{number} ::= 0 \ldots 9$$

$$\text{ident} ::= (\text{letter}|\text{number}|\_)\ \{\text{letter}|0\ldots 9|\_|'\}$$

2 Théorie

2.1 Syntaxe

$$
\begin{array}{rcll}
e, \rho, \kappa & ::= & e : \rho & \text{annotated term} \\
& | & x & \text{variable} \\
& | & \lambda x \to e & \text{lambda} \\
& | & e\ e' & \text{application} \\
& | & \Pi(x : \rho).\rho' & \text{pi type} \\
& | & (e, e') & \text{tuple} \\
& | & \text{fst } e & \text{fst} \\
& | & \text{snd } e & \text{snd} \\
& | & \Sigma(x : \rho).\rho' & \text{sigma type} \\
& | & * & \text{type of types}
\end{array}
$$

where e, ρ, κ represent general expressions, types and kinds respectively.

2.2 Contexte

$$
\begin{array}{rcll}
\Gamma & ::= & \epsilon & \text{empty context} \\
& | & \Gamma, x : \tau & \text{adding a variable}
\end{array}
$$

$$
\frac{}{\text{valid}(\epsilon)} \qquad \frac{\text{valid}(\Gamma) \qquad \Gamma \vdash \tau :_{\downarrow} \star}{\text{valid}(\Gamma, x : \tau)}
$$

2.3 Evaluation

$$
\begin{array}{rcll}
\nu, \tau & ::= & n & \text{neutral term} \\
& | & \lambda x \to \nu & \text{lambda} \\
& | & (\nu, \nu') & \text{tuple} \\
& | & \star & \text{type of types} \\
& | & \Pi(x : \tau).\tau' & \text{dependent function space} \\
& | & \Sigma(x : \tau).\tau' & \text{dependent pair space}
\end{array}
$$

$$\frac{}{\star \Downarrow *}\ (\textsc{Star}) \qquad \frac{}{x \Downarrow x}\ (\textsc{Var}) \qquad \frac{e \Downarrow \nu}{e : \rho \Downarrow \nu}\ (\textsc{Ann})$$

$$\frac{e \Downarrow \nu}{\lambda x \to e \Downarrow \lambda x \to \nu}\ (\textsc{Lam}) \qquad \frac{e \Downarrow \nu \qquad e' \Downarrow \nu'}{(e, e') \Downarrow (\nu, \nu')}\ (\textsc{Tuple})$$

$$\frac{e \Downarrow \lambda x \to \nu \qquad \nu[x \mapsto e'] \Downarrow \nu'}{e\ e' \Downarrow \nu'}\ (\textsc{App}) \qquad \frac{e \Downarrow n \qquad e' \Downarrow \nu'}{e\ e' \Downarrow n\ \nu'}\ (\textsc{NApp})$$

$$\frac{e \Downarrow (\nu, \nu')}{\text{fst } e \Downarrow \nu}\ (\textsc{Fst}) \qquad \frac{e \Downarrow (\nu, \nu')}{\text{snd } e \Downarrow \nu'}\ (\textsc{Snd})$$

$$\frac{\rho \Downarrow \tau \qquad \rho' \Downarrow \tau'}{\Pi(x : \rho).\rho' \Downarrow \Pi(x : \tau).\tau'}\ (\textsc{Pi}) \qquad \frac{\rho \Downarrow \tau \qquad \rho' \Downarrow \tau'}{\Sigma(x : \rho).\rho' \Downarrow \Sigma(x : \tau).\tau'}\ (\textsc{Sigma})$$

2.4 Typing

In the following, $e :_\uparrow \tau$ is an expression with inferrable type τ while $e :_\downarrow \tau$ is checkable.

$$\frac{\Gamma \vdash x :_\uparrow \tau}{\Gamma \vdash x :_\downarrow \tau}\ (\textsc{Chk}) \qquad \frac{\Gamma \vdash \rho :_\downarrow * \qquad \rho \Downarrow \tau \qquad \Gamma \vdash e :_\downarrow \tau}{\Gamma \vdash (e : \rho) :_\uparrow \tau}\ (\textsc{Ann})$$

$$\frac{}{\Gamma \vdash * :_\uparrow *}\ (\textsc{Star}) \qquad \frac{\Gamma(x) = \tau}{\Gamma \vdash e :_\uparrow \tau}\ (\textsc{Var})$$

$$\frac{\Gamma, x : \tau \vdash e :_\downarrow \tau'}{\Gamma \vdash \lambda x \to e :_\downarrow \Pi(x : \tau)).\tau'}\ (\textsc{Lam})$$

$$\frac{\Gamma \vdash e :_\downarrow \tau \qquad \Gamma \vdash e' :_\downarrow \tau'}{\Gamma \vdash (e, e') :_\downarrow \Sigma(x : \tau).\tau'}\ (\textsc{Tuple})$$

$$\frac{\Gamma \vdash e :_\uparrow \Pi(x : \tau).\tau' \qquad \Gamma \vdash e' :_\downarrow \tau \qquad \tau'[x \mapsto e'] \Downarrow \tau''}{\Gamma \vdash e\ e' :_\uparrow \tau''}\ (\textsc{App})$$

$$\frac{\Gamma \vdash e :_\uparrow \Sigma(x : \tau).\tau'}{\Gamma \vdash \text{fst } e :_\uparrow \tau}\ (\textsc{Fst})$$

$$\frac{\Gamma \vdash e :_\uparrow \Sigma(x : \tau).\tau' \qquad \tau'[x \mapsto \text{fst } e] \Downarrow \tau''}{\Gamma \vdash \text{snd } e :_\uparrow \tau''}\ (\textsc{Snd})$$

$$\frac{\Gamma \vdash \rho :_\downarrow * \qquad \rho \Downarrow \tau \qquad \Gamma, x : \tau \vdash \rho' :_\downarrow *}{\Gamma \vdash \Pi(x : \rho).\rho' :_\uparrow *}\ (\textsc{Pi})$$

$$\frac{\Gamma \vdash \rho :_\downarrow * \qquad \rho \Downarrow \tau \qquad \Gamma, x : \tau \vdash \rho' :_\downarrow *}{\Gamma \vdash \Sigma(x : \rho).\rho' :_\uparrow *}\ (\textsc{Sigma})$$

Une reformulation équivalentes des règles (Pi) et (Sigma), plus adaptée pour l'implémentation :

$$\frac{\Gamma \vdash \rho :_\downarrow * \qquad \rho \Downarrow \tau \qquad \Gamma \vdash \rho' :_\downarrow \Pi(x : \tau).*}{\Gamma \vdash \Pi(x : \rho).\rho' :_\uparrow *}\ (\textsc{Pi})$$

$$\frac{\Gamma \vdash \rho :_\downarrow * \qquad \rho \Downarrow \tau \qquad \Gamma \vdash \rho' :_\downarrow \Pi(x : \tau).*}{\Gamma \vdash \Sigma(x : \rho).\rho' :_\uparrow *}\ (\textsc{Sigma})$$