Harvard School of Engineering and Applied Sciences — CS 152: Programming Languages

Sketch of Notes on Dependent Types

Spring 2017

---

1 The core calculus

(Basically λP2 in the λ-cube)

Syntax:

$$\begin{array}{lr} \text{expressions, types} & e, \rho ::= x \mid \lambda x\!:\!\rho.\, e \mid e_1\ e_2 \mid * \mid \Pi x\!:\!\rho.\, \rho' \\ \text{values} & v, \tau ::= \lambda x\!:\!\tau.\, e \mid * \mid \Pi x\!:\!\tau.\, \tau' \mid n \\ & n ::= x \mid n\ v \end{array}$$

Abbreviations:

$$\begin{array}{rll} \rho_1 \to \rho_1 &= \Pi z\!:\!\rho_1.\, \rho_2 & \quad (z \notin FV(\rho_2)) \\ \forall \alpha.\rho &= \Pi \alpha\!:\!*.\, \rho & \end{array}$$

Large-step operational semantics:

$$\frac{}{x \Downarrow x} \qquad \frac{\rho \Downarrow \tau \quad e \Downarrow v}{\lambda x\!:\!\rho.\, e \Downarrow \lambda x\!:\!\tau.\, v} \qquad \frac{e_1 \Downarrow \lambda x\!:\!\tau.\, v \quad e_2 \Downarrow v}{e_1\ e_2 \Downarrow v}$$

$$\frac{e_1 \Downarrow n \quad e_2 \Downarrow v}{e_1\ e_2 \Downarrow nv} \qquad \frac{}{* \Downarrow *} \qquad \frac{\rho \Downarrow \tau \quad \rho' \Downarrow \tau'}{\Pi x\!:\!\rho.\, \rho' \Downarrow \Pi x\!:\!\tau.\, \tau'}$$

Typing contexts:

$$\Gamma ::= \cdot \mid \Gamma, x\!:\!\tau$$

Type judgment $\Gamma \vdash \mathbf{ok}$:

$$\frac{}{\cdot \vdash \mathbf{ok}} \qquad \frac{\Gamma \vdash \mathbf{ok} \quad \Gamma \vdash \tau : *}{\Gamma, x\!:\!\tau \vdash \mathbf{ok}}$$

Type judgment $\Gamma \vdash e : \tau$:

$$\frac{\Gamma \vdash \mathbf{ok}}{\Gamma \vdash x : \tau}\ x\!:\!\tau \in \Gamma \qquad \frac{\Gamma, x\!:\!\tau \vdash e : \tau' \quad \Gamma \vdash \tau : * \quad \rho \Downarrow \tau}{\Gamma \vdash \lambda x\!:\!\rho.\, e : \Pi x\!:\!\tau.\, \tau'}$$

$$\frac{\Gamma \vdash e_1 : \Pi x\!:\!\tau.\, \tau' \quad \Gamma \vdash e_2 : \tau \quad \tau'\{e'/x\} \Downarrow \tau''}{\Gamma \vdash e_1\ e_2 : \tau''} \qquad \frac{\Gamma \vdash \mathbf{ok}}{\Gamma \vdash * : *}$$

$$\frac{\Gamma \vdash \rho : * \quad \rho \Downarrow \tau \quad \Gamma, x\!:\!\tau \vdash \rho' : *}{\Gamma \vdash \Pi x\!:\!\rho.\, \rho' : *}$$

Adding natural numbers

Syntax:

$$\begin{aligned} e, \rho &::= \cdots \mid \textbf{nat} \mid 0 \mid \mathsf{succ}\ e \mid \mathsf{nelim}\ e_1\ e_2\ e_3\ e_4 \\ v, \tau &::= \cdots \mid \textbf{nat} \mid 0 \mid \mathsf{succ}\ v \end{aligned}$$

Large-step operational semantics:

$$\frac{}{\textbf{nat} \Downarrow \textbf{nat}} \qquad \frac{}{0 \Downarrow 0} \qquad \frac{e \Downarrow v}{\mathsf{succ}\ e \Downarrow \mathsf{succ}\ v}$$

$$\frac{k \Downarrow 0 \quad mz \Downarrow v}{\mathsf{nelim}\ m\ mz\ ms\ k \Downarrow v} \qquad \frac{k \Downarrow \mathsf{succ}\ l \quad ms\ l\ (\mathsf{nelim}\ m\ mz\ ms\ l) \Downarrow v}{\mathsf{nelim}\ m\ mz\ ms\ k \Downarrow v}$$

$$\frac{k \Downarrow n}{\mathsf{nelim}\ m\ mz\ ms\ k \Downarrow \mathsf{nelim}\ m\ mz\ ms\ n}$$

Typing rules:

$$\frac{\Gamma \vdash \textbf{ok}}{\Gamma \vdash \textbf{nat} : *} \qquad \frac{\Gamma \vdash \textbf{ok}}{\Gamma \vdash 0 : \textbf{nat}} \qquad \frac{\Gamma \vdash k : \textbf{nat}}{\Gamma \vdash \mathsf{succ}\ k : \textbf{nat}}$$

$$\frac{\Gamma \vdash m : \textbf{nat} \to * \quad m\ 0 \Downarrow \tau \quad \Gamma \vdash mz : \tau \quad \Pi l\!:\!\textbf{nat}.\, m\ k \to\ m\ (\mathsf{succ}\ l) \Downarrow \tau' \quad \Gamma \vdash ms : \tau' \quad m\ k \Downarrow \tau''}{\Gamma \vdash \mathsf{nelim}\ m\ mz\ ms\ k : \tau''}$$

Example:

$$\begin{aligned} \mathsf{plus} = \lambda x\!:\!\textbf{nat}.\, \lambda y\!:\!\textbf{nat}.\ \mathsf{nelim}\ &(\lambda z\!:\!\textbf{nat}.\, \textbf{nat} \to \textbf{nat}) \\ &(\lambda n\!:\!\textbf{nat}.\, n) \\ &(\lambda k\!:\!\textbf{nat}.\, \lambda rec\!:\!(\textbf{nat} \to \textbf{nat}).\, \lambda n\!:\!\textbf{nat}.\, \mathsf{succ}\ (rec\ n)) \end{aligned}$$

It is tedious but not difficult to check that:

- $\vdash \mathsf{plus} : \textbf{nat} \to (\textbf{nat} \to \textbf{nat})$
- $\mathsf{plus}\ (\mathsf{succ}\ (\mathsf{succ}\ 0))\ (\mathsf{succ}\ 0) \Downarrow \mathsf{succ}\ (\mathsf{succ}\ (\mathsf{succ}\ 0))$

Adding vectors

Syntax:

$$\begin{aligned} e, \rho &::= \cdots \mid \textbf{vec}\ \rho\ e \mid \mathsf{nil}\ \rho \mid \mathsf{cons}\ \rho\ e_1\ e_2\ e_3 \mid \mathsf{velim}\ e_1\ e_2\ e_3\ e_4\ e_5\ e_6 \\ v, \tau &::= \cdots \mid \textbf{vec}\ \tau\ v \mid \mathsf{nil}\ \tau \mid \mathsf{cons}\ \tau\ v_1\ v_2\ v_3 \end{aligned}$$

Large-step operational semantics:

$$\frac{\rho \Downarrow \tau \qquad e \Downarrow v}{\textbf{vec}\ \rho\ e \Downarrow \textbf{vec}\ \tau\ v} \qquad \frac{\rho \Downarrow \tau}{\mathsf{nil}\ \rho \Downarrow \mathsf{nil}\ \tau} \qquad \frac{\rho \Downarrow \tau \quad e_1 \Downarrow v_1 \quad e_2 \Downarrow v_2 \quad e_3 \Downarrow v_3}{\mathsf{cons}\ \rho\ e_1\ e_2\ e_3 \Downarrow \mathsf{cons}\ \tau\ v_1\ v_2\ v_3}$$

$$\frac{xs \Downarrow \mathsf{nil}\ \tau \quad mn \Downarrow v}{\mathsf{velim}\ \rho\ m\ mn\ mc\ k\ xs \Downarrow v} \qquad \frac{xs \Downarrow \mathsf{cons}\ \tau\ l\ v_1\ v_2 \quad mc\ l\ v_1\ v_2\ (\mathsf{velim}\ \rho\ m\ mn\ mc\ l\ v_2) \Downarrow v}{\mathsf{velim}\ \rho\ m\ mn\ mc\ k\ xs \Downarrow v}$$

$$\frac{xs \Downarrow n}{\mathsf{velim}\ \rho\ m\ mn\ mc\ k\ xs \Downarrow \mathsf{velim}\ \rho\ m\ mn\ mc\ k\ n}$$

Typing rules:

$$\frac{\Gamma \vdash \rho : * \quad \Gamma \vdash e : \textbf{nat}}{\Gamma \vdash \textbf{vec}\ \rho\ e : *} \qquad\qquad \frac{\Gamma \vdash \rho : * \quad \rho \Downarrow \tau}{\Gamma \vdash \mathsf{nil}\ \rho : \textbf{vec}\ \tau\ 0}$$

$$\frac{\Gamma \vdash \rho : * \quad \rho \Downarrow \tau \quad \Gamma \vdash e_1 : \textbf{nat} \quad e_1 \Downarrow k \quad \Gamma \vdash e_2 : \tau \quad \Gamma \vdash e_3 : \textbf{vec}\ \tau\ k}{\Gamma \vdash \mathsf{cons}\ \rho\ e_1\ e_2\ e_3 : \textbf{vec}\ \tau\ (\mathsf{succ}\ k)}$$

$$\frac{\begin{array}{c}\Gamma \vdash \rho : * \quad \rho \Downarrow \tau \quad \Gamma \vdash m : \Pi k{:}\textbf{nat}.\,\textbf{vec}\ \tau\ k \to * \quad m\ 0\ (\mathsf{nil}\ \tau) \Downarrow \tau' \quad \Gamma \vdash mn : \tau \\ \Pi l{:}\textbf{nat}.\,\Pi x{:}\tau.\,\Pi xs{:}\textbf{vec}\ \tau\ l.\, m\ l\ xs \to m\ (\mathsf{succ}\ l)\ (\mathsf{cons}\ \tau\ l\ x\ xs) \Downarrow \tau'' \\ \Gamma \vdash ms : \tau'' \quad \Gamma \vdash k : \textbf{nat} \quad \Gamma \vdash xs : \textbf{vec}\ \tau\ k \quad m\ k\ xs \Downarrow \tau'''\end{array}}{\Gamma \vdash \mathsf{velim}\ \rho\ m\ mn\ mc\ k\ xs : \tau'''}$$

Example:

$$\begin{array}{l}
\mathsf{append} = \lambda\alpha{:}*.\,\lambda m{:}\textbf{nat}.\,\lambda v{:}\textbf{vec}\ \alpha\ m.\,\lambda n{:}\textbf{nat}.\,\lambda w{:}\textbf{vec}\ \alpha\ n.\\
\qquad\qquad (\mathsf{velim}\ \alpha\\
\qquad\qquad\quad (\lambda m{:}\textbf{nat}.\,\lambda_{:}\textbf{vec}\ \alpha\ m.\,\Pi n{:}\textbf{nat}.\,\textbf{vec}\ \alpha\ n \to \textbf{vec}\ \alpha\ (\mathsf{plus}\ m\ n))\\
\qquad\qquad\quad (\lambda m{:}\textbf{nat}.\,\lambda v{:}\textbf{vec}\ \alpha\ m.\,v)\\
\qquad\qquad\quad (\lambda m{:}\textbf{nat}.\,\lambda v{:}\alpha.\,\lambda vs{:}\textbf{vec}\ \alpha\ m.\,\lambda rec{:}(\Pi n{:}\textbf{nat}.\,\textbf{vec}\ \alpha\ n \to \textbf{vec}\ \alpha\ (\mathsf{plus}\ m\ n)).\\
\qquad\qquad\qquad \lambda n{:}\textbf{nat}.\,\lambda w{:}\textbf{vec}\ \alpha\ n.\,\mathsf{cons}\ \alpha\ (\mathsf{plus}\ m\ n)\ v\ (rec\ n\ w))\\
\qquad\qquad\quad m\ v)\ n\ w
\end{array}$$

Compare to OCaml's `append xs ys = fold_right (fun a r -> a::r) xs ys`.
It is mind-boggingly tedious but not difficult to check that:

- $\vdash \mathsf{append} : \Pi\alpha{:}*.\,\Pi m{:}\textbf{nat}.\,\Pi v{:}\textbf{vec}\ \alpha\ m.\,\Pi n{:}\textbf{nat}.\,\Pi w{:}\textbf{vec}\ \alpha\ n.\,\textbf{vec}\ \alpha\ (\mathsf{plus}\ m\ n)$
  (i.e., given vectors of length m and length n, append returns a vector of length $m + n$)
- $\mathsf{append}\ \textbf{nat}\ 2\ (\mathsf{cons}\ \textbf{nat}\ 1\ 33\ (\mathsf{cons}\ \textbf{nat}\ 0\ 66\ (\mathsf{nil}\ \textbf{nat})))\ 1\ (\mathsf{cons}\ \textbf{nat}\ 0\ 99\ (\mathsf{nil}\ \textbf{nat}))$
  $\Downarrow \mathsf{cons}\ \textbf{nat}\ 2\ 33\ (\mathsf{cons}\ \textbf{nat}\ 1\ 66\ (\mathsf{cons}\ \textbf{nat}\ 0\ 99\ (\mathsf{nil}\ \textbf{nat})))$