1 Syntax

$$
\begin{array}{rrcl}
\text{expressions} & M, N, A, B & ::= & \lambda\, p\,.\, M \mid x \mid M\, N \mid \Pi\, p : A\,.\, B \mid \mathsf{U} \mid \\
 & & & M, N \mid M.1 \mid M.2 \mid \Sigma\, p : A\,.\, B \mid \\
 & & & 0 \mid \mathbf{1} \mid \\
 & & & c\, M \mid \mathsf{fun}\, S \mid \mathsf{Sum}\, S \mid \\
 & & & D; M \\
\text{patterns} & p & ::= & x \mid p, p \mid \_ \\
\text{choices} & S & ::= & () \mid (c\, M, S) \\
\text{declarations} & D & ::= & p : A = M \mid \mathsf{rec}\, p : A = M \\
\text{syntactic sugar} & A \to B & = & \Pi\, \_ : A\,.\, B \\
 & A \times B & = & \Sigma\, \_ : A\,.\, B \\
 & c\, A \mid S & = & (c, A), S \\
 & c \to M \mid S & = & (c, M), S
\end{array}
$$

Fig. 6.1. Syntax of Mini-TT

2 Values

$$
\begin{array}{rrcl}
\text{values} & u, v, t & ::= & [k] \mid \lambda f \mid \Pi\, t\, g \mid \mathsf{U} \mid \\
 & & & u, v \mid 0 \mid \Sigma\, t\, g \mid \mathbf{1} \mid \\
 & & & c\, v \mid \mathsf{fun}\, s \mid \mathsf{Sum}\, s
\end{array}
$$

$$
\begin{array}{rrcl}
\text{neutral values (accumulators)} & k & ::= & \mathsf{x}_n \mid k\, v \mid k.1 \mid k.2 \mid s\, k \\
\text{function closures} & f, g & ::= & \langle \lambda p.M, \rho \rangle \mid f \circ c \\
\text{choice closures} & s & ::= & \langle S, \rho \rangle \\
\text{environments} & \rho & ::= & () \mid \rho, p = v \mid \rho, D
\end{array}
$$

Fig. 6.2. Values

There is a function which instantiates a function closure to a value. It is defined by:

$$
\begin{array}{lcl}
\mathsf{inst}\langle \lambda p.M, \rho\rangle\, v & = & [\![M]\!](\rho, p = v) \\
\mathsf{inst}(f \circ c)\, v & = & \mathsf{inst}\, f (c\, v)
\end{array}
$$

Application $\mathsf{app}\, u\, v$ of values is defined using instantiation. Notice how a neutral value is built up in the case that the function is a neutral value:

$$
\begin{array}{lcl}
\mathsf{app}(\lambda f)\, v & = & \mathsf{inst}\, f\, v \\
\mathsf{app}(\mathsf{fun}\langle S, \rho\rangle)(c_i\, v) & = & \mathsf{app}([\![M_i]\!]\rho)\, v \\
 & & \text{where } S = (c_1 \rightarrow M_1 \mid \cdots \mid c_n \rightarrow M_n) \\
\mathsf{app}(\mathsf{fun}\, s)[k] & = & [s\, k] \\
\mathsf{app}[k]\, v & = & [k\, v]
\end{array}
$$

The projection function for pairs of values follows the same pattern:

$$
\begin{array}{lcl}
(u, v).1 & = & u \\
[k].1 & = & [k.1] \\
(u, v).2 & = & v \\
[k].2 & = & [k.2]
\end{array}
$$

3 Value Operations

4 Lookup Variable Function

5 Semantics

6 Readback Function

If x is in p,

$$\begin{array}{lcl}
(\rho, p = v)(x) & = & \mathsf{proj}^p_x(v) \\
(\rho, p : A = M)(x) & = & \mathsf{proj}^p_x([\![M]\!]\rho) \\
(\rho, \mathsf{rec}\, p : A = M)(x) & = & \mathsf{proj}^p_x([\![M]\!](\rho, \mathsf{rec}\, p : A = M))
\end{array}$$

If x is not in p,

$$\begin{array}{lcl}
(\rho, p = v)(x) & = & \rho(x) \\
(\rho, D)(x) & = & \rho(x)
\end{array}$$

The notation $\mathsf{proj}^p_x(v)$ is well-defined under the precondition that x is in p.

$$\begin{array}{lcll}
\mathsf{proj}^x_x(v) & = & v & \\
\mathsf{proj}^{(p_1,p_2)}_x(v) & = & \mathsf{proj}^{p_1}_x(v.1) & \text{if } x \text{ is in } p_1, \\
\mathsf{proj}^{(p_1,p_2)}_x(v) & = & \mathsf{proj}^{p_2}_x(v.2) & \text{if } x \text{ is in } p_2
\end{array}$$

$$\begin{array}{lcl}
[\![\lambda\, p\,.\, M]\!]\rho & = & \langle \lambda p.M, \rho\rangle \\
[\![x]\!]\rho & = & \rho(x) \\
[\![M\, N]\!]\rho & = & \mathsf{app}([\![M]\!]\rho)([\![N]\!]\rho) \\
[\![\Pi\, p : A\,.\, B]\!]\rho & = & \Pi\,([\![A]\!]\rho)\,\langle \lambda p.B, \rho\rangle \\
[\![\mathsf{U}]\!]\rho & = & \mathsf{U} \\
[\![D; M]\!]\rho & = & [\![M]\!](\rho, D)
\end{array}$$

$$\begin{array}{lcl}
[\![M, N]\!]\rho & = & ([\![M]\!]\rho, [\![N]\!]\rho) \\
[\![0]\!]\rho & = & 0 \\
[\![M.1]\!]\rho & = & ([\![M]\!]\rho).1 \\
[\![M.2]\!]\rho & = & ([\![M]\!]\rho).2 \\
[\![\Sigma\, p : A\,.\, B]\!]\rho & = & \Sigma\,([\![A]\!]\rho)\,\langle \lambda p.B, \rho\rangle \\
[\![\mathbf{1}]\!]\rho & = & \mathbf{1}
\end{array}$$

$$\begin{array}{lcl}
[\![c\, M]\!]\rho & = & c\,([\![M]\!]\rho) \\
[\![\mathsf{fun}\, S]\!]\rho & = & \mathsf{fun}\langle S, \rho\rangle \\
[\![\mathsf{Sum}\, S]\!]\rho & = & \mathsf{Sum}\langle S, \rho\rangle
\end{array}$$

Fig. 6.3. Semantics of Mini-TT

$$
\begin{array}{lll}
E & ::= & \lambda\, \mathsf{x}_i \,.\, E \mid \Pi\, \mathsf{x}_i : E_1 \,.\, E_2 \mid \mathsf{U} \mid [K] \\
 & & E_1, E_2 \mid 0 \mid \Sigma\, \mathsf{x}_i : E_1 \,.\, E_2 \mid \mathbf{1} \\
 & & c\, E \mid \mathsf{fun}\langle S, \alpha\rangle \mid \mathsf{Sum}\langle S, \alpha\rangle \\
K & ::= & \mathsf{x}_i \mid K E \mid K.1 \mid K.2 \mid \langle S, \alpha\rangle\, K \\
\alpha & ::= & () \mid (\alpha, p = E) \mid (\alpha, D)
\end{array}
$$

Fig. 6.4. Normal expressions

$$
\begin{array}{lll}
\mathsf{R}_i(\lambda f) & = & \lambda\, \mathsf{x}_i \,.\, \mathsf{R}_{i+1}(\mathsf{inst}\, f[\mathsf{x}_i]) \\
\mathsf{R}_i(u, v) & = & (\mathsf{R}_i\, u, \mathsf{R}_i\, v) \\
\mathsf{R}_i\, 0 & = & 0 \\
\mathsf{R}_i(c\, v) & = & c\, (\mathsf{R}_i\, v) \\
\mathsf{R}_i(\mathsf{fun}\langle S, \rho\rangle) & = & \mathsf{fun}\langle S, \mathsf{R}_i\, \rho\rangle \\
\mathsf{R}_i(\mathsf{Sum}\langle S, \rho\rangle) & = & \mathsf{Sum}\langle S, \mathsf{R}_i\, \rho\rangle \\
\mathsf{R}_i\, \mathsf{U} & = & \mathsf{U} \\
\mathsf{R}_i\, \mathbf{1} & = & \mathbf{1} \\
\mathsf{R}_i(\Pi\, t\, g) & = & \Pi\, \mathsf{x}_i : \mathsf{R}_i\, t \,.\, \mathsf{R}_{i+1}(\mathsf{inst}\, g[\mathsf{x}_i]) \\
\mathsf{R}_i(\Sigma\, t\, g) & = & \Sigma\, \mathsf{x}_i : \mathsf{R}_i\, t \,.\, \mathsf{R}_{i+1}(\mathsf{inst}\, g[\mathsf{x}_i]) \\
\mathsf{R}_i[k] & = & [\mathsf{R}_i\, k] \\
 & & \\
\mathsf{R}_i\, \mathsf{x}_j & = & \mathsf{x}_j \\
\mathsf{R}_i(k\, v) & = & (\mathsf{R}_i\, k)(\mathsf{R}_i\, v) \\
\mathsf{R}_i(k.1) & = & (\mathsf{R}_i\, k).1 \\
\mathsf{R}_i(k.2) & = & (\mathsf{R}_i\, k).2 \\
\mathsf{R}_i(\langle S, \rho\rangle\, k) & = & \langle S, \mathsf{R}_i\, \rho\rangle\, (\mathsf{R}_i\, k) \\
 & & \\
\mathsf{R}_i(\rho, p = v) & = & \mathsf{R}_i\, \rho,\ p = \mathsf{R}_i\, v \\
\mathsf{R}_i(\rho, D) & = & \mathsf{R}_i\, \rho, D \\
\mathsf{R}_i() & = & ()
\end{array}
$$

Fig. 6.5. The readback notation