1 Pattern Matching

A *new* syntacic category: *patterns* $\mathscr{P}$:

$$\begin{array}{rcll}\mathscr{P} & = & \mathbb{N} & \text{— number}\\ & & \_ & \text{— wildcard}\\ & & [\,\mathscr{P}^*\,] & \text{— array}\\ & & \mathscr{C}\,\mathscr{P}^* & \text{— S-expression}\\ & & \mathscr{X}\,\texttt{@}\,\mathscr{P} & \text{— named pattern}\end{array}$$

Concrete syntax mainly repeats the abstract; in S-expression patterns extra round brackets are used to delimit the constructor's name from arguments (if any); arguments of array/S-expression patterns are delimited by commas, and extra round brackets can be used to group subpatterns. Additionally, one derived form is used: an identifier x is treated as a pattern $x@\_$.

Pattern matching expression:

$$\mathscr{E} += \texttt{case}\ \mathscr{E}\ \texttt{of}\ (\mathscr{P} \times \mathscr{E})^+\ \texttt{esac}$$

In a concrete syntax branches of case expression are delimited by "—", and in each branch " $\rightarrow$ " is used to delimit pattern from expression.

Well-formedness of case expressions is established in an obvious manner:

$$\frac{e : \textbf{Val} \quad e_i : a}{\texttt{case}\ e\ \texttt{of}\ p_1 \rightarrow e_1\ ...\ p_k \rightarrow e_k\ \texttt{esac} : a}$$

2 Operational Semantics

There are two aspects that have to be covered in semantic description of pattern matching:

- the criterion for a *scrutinee* to be matched by a pattern;
- the descipline of binding support.

The latter aspect is covered by a desugaring. First, we define a mapping

$$\beta : \mathscr{P} \rightarrow \mathscr{E} \rightarrow \mathscr{X} \rightarrow \mathscr{E}$$

in a following manner:

$$\begin{array}{rcl}\beta\, n\, e & = & \lambda\_.\bot\\ \beta\, \_\, e & = & \lambda\_.\bot\\ \beta\, ([p_0 \dots p_k])\, e & = & \\ \beta\, (C\, p_0 \dots p_k)\, e & = & (\beta\, p_0\, e[0])[\beta\, p_1\, e[1]] \dots [\beta\, p_k\, e[k]]\\ \beta\, (x@p)\, e & = & (\beta\, p\, e)[x \leftarrow e]\end{array}$$

This function determines a proper *subvalue* of an expression bound by an identifier in a pattern. For example, for a pattern $[\_, x, C\,(\_, y)]$ and scrutenee s the value of $\beta\,[\_, x, C\,(\_, y)]\,s$ can be described by the following table:

$$\begin{array}{lcl} x & \rightarrow & s[0] \\ y & \rightarrow & s[1][1] \end{array}$$

Then, given a pattern-matching expression **`case`** e **`of`** ... we, first, bind the expression e to a *fresh* variable, say, s:

$$\begin{array}{l} \texttt{var}\ s = e; \\ \texttt{case}\ s\ \texttt{of}\ \ldots \end{array}$$

Then, we transform each branch $p \rightarrow e$ into the following:

$$\begin{array}{lll} p & \rightarrow \texttt{var}\ b_1 & = \beta\, p\, s\, b_1\,, \\ & & \ldots \\ & \phantom{\rightarrow \texttt{var}\ } b_k & = \beta\, p\, s\, b_k\,; \\ & \phantom{\rightarrow \texttt{var}\ } e & \end{array}$$

where $b_1, \ldots, b_k$ are all bindings in p.

Now, for determining the descipline of matching we need an extra relation

$$match \subseteq \mathscr{P} \times \mathscr{V}$$

between patterns and values. We define it in a following way:

$$match\,(\_, v)$$

$$match\,(n, n)$$

$$\frac{match\,(p_i, v_i)}{match\,([p_1, \ldots, p_k], [v_1, \ldots, v_k])}$$

$$\frac{match\,(p_i, v_i)}{match\,(C\, p_1, \ldots, p_k, C\, v_1, \ldots, v_k)}$$

$$\frac{match\,(p, v)}{match\,(x@p, v)}$$

Finally, the operational semantics of pattern-matching expression can be given by the following rules:

$$\frac{c \xRightarrow{e} \langle c', v\rangle \qquad v \vdash c' \xrightarrow{(p_1, e_1)\ldots(p_k, e_k)}_{\mathscr{P}} c''}{\langle c, l\rangle \xRightarrow{\texttt{case}\ e\ \texttt{of}\ p_1 \rightarrow e_1\ \ldots\ p_k \rightarrow e_k\ \texttt{esac}} c''}$$

where an additional transition " $\longrightarrow_{\mathscr{P}}$ " is defined as follows:

$$\frac{\begin{array}{c} match(p, v) \\ c \overset{e}{\Longrightarrow} c' \end{array}}{v \vdash c \xrightarrow{(p, e)ps}_{\mathscr{P}} c'}$$

$$\frac{\begin{array}{c} \neg match(p, v) \\ v \vdash c \xrightarrow{ps}_{\mathscr{P}} c' \end{array}}{v \vdash c \xrightarrow{(p, e)ps}_{\mathscr{P}} c'}$$