Principles of Programming, Spring 2006
Practice 6. Solution

Daejun Park, Heejong Lee
Programming Research Lab.@SNU

April 13, 2006

Consider the following procedures:

$$\begin{array}{rcll} d & \in & \textit{digit} & \rightarrow & 0 \mid 1 \mid \cdots \mid 9 \\ s & \in & \textit{string} & \rightarrow & \epsilon \mid d \mid d \cdot s \\ c & \in & \textit{code} & \rightarrow & \epsilon \mid d \mid c \cdot c \mid c \,|\, c \mid c\texttt{+} \end{array}$$

$$\begin{array}{rcl} \texttt{match} & : & \textit{string} \times \textit{code} \rightarrow \texttt{bool} \\ \texttt{matchs} & : & \textit{string} \times \textit{code set} \rightarrow \texttt{bool} \\ \texttt{first} & : & \textit{code} \rightarrow \textit{digit set} \\ \texttt{rest} & : & \textit{digit} \times \textit{code} \rightarrow \textit{code set} \end{array}$$

Informally, the `matchs` extends the `match` to *code set*, the `first(`c`)` means a set of first digit of a string that matches with c, and the `rest(`d,c`)` means a set of code representing the string s, such that $d \cdot s$ matches with c. The following is the formal definitions of these procedures.

$$\begin{array}{rcl} \| \epsilon \| & \doteq & \{\epsilon\} \\ \| d \| & \doteq & \{d\} \\ \| c_1 \cdot c_2 \| & \doteq & \{s_1 \cdot s_2 \mid s_1 \in \| c_1 \| \ \wedge \ s_2 \in \| c_2 \|\} \\ \| c_1 | c_2 \| & \doteq & \| c_1 \| \ \cup \ \| c_2 \| \\ \| c\texttt{+} \| & \doteq & \| c \| \ \cup \ \{s \cdot s \mid s \in \| c \|\} \ \cup \ \{s \cdot s \cdot s \mid s \in \| c \|\} \ \cup \ \cdots \end{array}$$

$$\begin{array}{rcl} \texttt{match}(s,c) & \doteq & \begin{cases} true, & s \in \| c \| \\ false, & \text{otherwise} \end{cases} \\ \texttt{matchs}(s,C) & \doteq & \bigvee_{c \in C} \texttt{match}(s,c) \\ \texttt{first}(c) & \doteq & \{d \mid d \cdot s \in \| c \|\} \\ \texttt{rest}(d,c) & \doteq & \{c' \mid d \cdot s \in \| c \| \ \wedge \ s \in \| c' \|\} \end{array}$$

By the above definitions, we can conclude the following equality.

$$\| c \| \quad = \quad \{d \cdot s \mid d \in \texttt{first}(c) \ \wedge \ c' \in \texttt{rest}(d,c) \ \wedge \ s \in \| c' \|\}$$

However we cannot use the above definitions to implement the procedures, since the size of $\| c \|$ is infinite. Therefore we have to modify the definition into a computable one. The following is the modified definitions.

$$
\begin{array}{rcl}
\mathtt{match}(s, \epsilon) & \doteq & s = \epsilon \\
\mathtt{match}(s, d) & \doteq & s = d \\
\mathtt{match}(s, c_1 \cdot c_2) & \doteq & s = d \cdot s' \ \wedge \ d \in \mathtt{first}(c_1 \cdot c_2) \ \wedge \ \mathtt{matchs}(s', \mathtt{rest}(d, c_1 \cdot c_2)) \\
\mathtt{match}(s, c_1 | c_2) & \doteq & \mathtt{match}(s, c_1) \ \vee \ \mathtt{match}(s, c_2) \\
\mathtt{match}(s, c\mathtt{+}) & \doteq & s = d \cdot s' \ \wedge \ d \in \mathtt{first}(c\mathtt{+}) \ \wedge \ \mathtt{matchs}(s', \mathtt{rest}(d, c\mathtt{+}))
\end{array}
$$

$$
\begin{array}{rcl}
\mathtt{matchs}(s, \phi) & \doteq & false \\
\mathtt{matchs}(s, C) & \doteq & \bigvee_{c \in C} \mathtt{match}(s, c)
\end{array}
$$

$$
\begin{array}{rcl}
\mathtt{first}(\epsilon) & \doteq & \{\epsilon\} \\
\mathtt{first}(d) & \doteq & \{d\} \\
\mathtt{first}(c_1 \cdot c_2) & \doteq & \mathtt{first}(c_1) \\
\mathtt{first}(c_1 | c_2) & \doteq & \mathtt{first}(c_1) \ \cup \ \mathtt{first}(c_2) \\
\mathtt{first}(c\mathtt{+}) & \doteq & \mathtt{first}(c)
\end{array}
$$

$$
\begin{array}{rcl}
\mathtt{rest}(d, \epsilon) & \doteq & \phi \\
\mathtt{rest}(d, d') & \doteq & \begin{cases} \{\epsilon\}, & d = d' \\ \phi, & d \neq d' \end{cases} \\
\mathtt{rest}(d, c_1 \cdot c_2) & \doteq & \{c \cdot c_2 \mid c \in \mathtt{rest}(d, c_1)\}^{\dagger} \\
\mathtt{rest}(d, c_1 | c_2) & \doteq & \mathtt{rest}(d, c_1) \ \cup \ \mathtt{rest}(d, c_2) \\
\mathtt{rest}(d, c\mathtt{+}) & \doteq & \mathtt{rest}(d, c) \ \cup \ \{c' \cdot c\mathtt{+} \mid c' \in \mathtt{rest}(d, c)\}^{\dagger}
\end{array}
$$

† Note that if $\mathtt{rest}(d, c) = \phi$, then $\{c_1 \cdot c_2 \mid c_1 \in \mathtt{rest}(d, c)\} = \phi$.