# Interpreter of Primitive Recursive Functions

**1 Primitive recursive function symbols.** For every $n \geq 1$, the class $\mathrm{PR}^n$ of $n$-ary *primitive recursive function symbols* (*PR-function symbols* for short) is defined inductively as follows:

– $Z \in \mathrm{PR}^1$, $S \in \mathrm{PR}^1$ and $I^n_i \in \mathrm{PR}^n$ for $1 \leq i \leq n$,
– if $h \in \mathrm{PR}^m$ and $g_1, \ldots, g_m \in \mathrm{PR}^n$ then $Comp^n_m(h, g_1, \ldots, g_m) \in \mathrm{PR}^n$,
– if $g \in \mathrm{PR}^n$ and $h \in \mathrm{PR}^{n+2}$ then $Rec_{n+1}(g, h) \in \mathrm{PR}^{n+1}$.

We set $\mathrm{PR} = \bigcup_{n \geq 1} \mathrm{PR}^n$.

We interprete $n$-ary PR-function symbols by $n$-ary functions. The interpretation $f^{\mathcal{N}}$ of a PR-function symbol $f$ is defined by induction on the structure of PR-function symbols as follows:

– $Z^{\mathcal{N}}$ is the zero function $Z(x) = 0$,
– $S^{\mathcal{N}}$ is the successor function $S(x) = x + 1$,
– $(I^n_i)^{\mathcal{N}}$ is the identity function $I^n_i(\vec{x}) = x_i$,
– $\big(Comp^n_m(h, g_1, \ldots, g_m)\big)^{\mathcal{N}}$ is the $n$-ary function defined by composition:

$$\big(Comp^n_m(h, g_1, \ldots, g_m)\big)^{\mathcal{N}}(\vec{x}) = h^{\mathcal{N}}\big(g^{\mathcal{N}}_1(\vec{x}), \ldots, g^{\mathcal{N}}_m(\vec{x})\big),$$

– $\big(Rec_n(g, h)\big)^{\mathcal{N}}$ is the $n$-ary function defined by primitive recursion:

$$\begin{aligned} \big(Rec_n(g, h)\big)^{\mathcal{N}}(0, \vec{y}) &= g^{\mathcal{N}}(\vec{y}) \\ \big(Rec_n(g, h)\big)^{\mathcal{N}}(x + 1, \vec{y}) &= h^{\mathcal{N}}\big(x, \big(Rec_n(g, h)\big)^{\mathcal{N}}(x, \vec{y}), \vec{y}\big). \end{aligned}$$

In the sequel we will often drop the superscript in $f^{\mathcal{N}}$ and write shortly $f$ instead of $f^{\mathcal{N}}$.

It is easy to see that primitive recursive functions are exactly those functions which are denoted by PR-function symbols. In other words, the class of primitive recursive functions is just the set

$$\bigcup_{n \geq 1} \{ f^{\mathcal{N}} \mid f \in \mathrm{PR}^n \}.$$

**2 Arithmetization of primitive recursive function symbols.** Now we consider the problem of coding of PR-function symbols into $\mathbb{N}$. The symbols are arithmetized with the help of the following pair constructors:

| | |
|---|---|
| $\boldsymbol{Z} = \langle 0, 0 \rangle$ | *(zero)* |
| $\boldsymbol{S} = \langle 1, 0 \rangle$ | *(successor)* |
| $\boldsymbol{I}^n_i = \langle 2, n, i \rangle$ | *(identities)* |
| $\boldsymbol{\langle} g, gs \boldsymbol{\rangle} = \langle 3, g, gs \rangle$ | *(contraction)* |
| $\boldsymbol{Comp}^n_m(h, gs) = \langle 4, n, m, h, gs \rangle$ | *(composition)* |
| $\boldsymbol{Rec}_n(g, h) = \langle 5, n, g, h \rangle$. | *(primitive recursion)* |

The arities of the constructors are as shown in their definitions. We postulate that the binary constructor $\boldsymbol{\langle} g, gs \boldsymbol{\rangle}$ groups to the right and has the same precedence as the pairing function $\langle x, y \rangle$.

The assignment of the code $\ulcorner f \urcorner$ to the PR-function symbol $f$ is defined inductively on the structure of PR-function symbols:

$$\begin{aligned}
\ulcorner Z \urcorner &= \boldsymbol{Z} \\
\ulcorner S \urcorner &= \boldsymbol{S} \\
\ulcorner I_i^n \urcorner &= \boldsymbol{I}_i^n \\
\ulcorner \mathit{Comp}_m^n(h, g_1, \dots, g_m) \urcorner &= \boldsymbol{Comp}_m^n\big(\ulcorner h \urcorner, \boldsymbol{\langle} \ulcorner g_1 \urcorner, \dots, \ulcorner g_m \urcorner \boldsymbol{\rangle}\big) \\
\ulcorner \mathit{Rec}_n(g, h) \urcorner &= \boldsymbol{Rec}_n(\ulcorner g \urcorner, \ulcorner h \urcorner).
\end{aligned}$$

Note that the binary operator $\boldsymbol{\langle} g, gs \boldsymbol{\rangle}$ plays a similar role as the pairing function $\langle x, y \rangle$ does for $n$-tuples of natural numbers. Its sole purpose is to represent the $m$-tuple $\ulcorner g_1 \urcorner, \dots, \ulcorner g_m \urcorner$ of the codes of PR-function symbols by its *contraction* which is the number of the form $\boldsymbol{\langle} \ulcorner g_1 \urcorner, \dots, \ulcorner g_m \urcorner \boldsymbol{\rangle}$.

**3 Interpreter of p.r. function symbols.** In this paragraph we give a definition of a binary function $e \bullet x$ which effectively realizes the interpretation of PR-function symbols. The application $\ulcorner f \urcorner \bullet \langle x_1, \dots, x_n \rangle$ takes the code of an $n$-ary PR-function symbol $f$ and the contraction of an $n$-tuple $x_1, \dots, x_n$ of numbers, and yields the number $f(x_1, \dots, x_n)$ as the result, ie

$$\ulcorner f \urcorner \bullet \langle x_1, \dots, x_n \rangle = f(x_1, \dots, x_n).$$

To improve readibility we will write $e_1 \bullet e_2 \bullet x$ instead of $e_1 \bullet (e_2 \bullet x)$, that is we let the operator associates right.

The interpreter $e \bullet x$ of primitive recursive functions is defined by

$$\begin{aligned}
&\boldsymbol{Z} \bullet x = 0 \\
&\boldsymbol{S} \bullet x = x + 1 \\
&\boldsymbol{I}_i^n \bullet x = [x]_i^n \\
&\boldsymbol{\langle} g, gs \boldsymbol{\rangle} \bullet x = \langle g \bullet x, gs \bullet x \rangle \\
&\boldsymbol{Comp}_m^n(h, gs) \bullet x = h \bullet gs \bullet x \\
&\boldsymbol{Rec}_n(g, h) \bullet \langle 0, y \rangle = g \bullet y \\
&\boldsymbol{Rec}_n(g, h) \bullet \langle x + 1, y \rangle = h \bullet \langle x, \boldsymbol{Rec}_n(g, h) \bullet \langle x, y \rangle, y \rangle.
\end{aligned}$$

This is an example of regular recursive definition which is into the lexicographical order $(x_1, y_1) <_{\text{lex}} (x_2, y_2)$ of natural numbers. This is because the first argument of each recursive application except the one in the last racursive clause goes down. In the recursive application of the last recursive clause the first argument $\boldsymbol{Rec}_n(g, h)$ stays the same and the second argument goes down since $\langle x, y \rangle < \langle x + 1, y \rangle$. We have therefore

$$\big(\boldsymbol{Rec}_n(g, h), \langle x, y \rangle\big) <_{\text{lex}} \big(\boldsymbol{Rec}_n(g, h), \langle x + 1, y \rangle\big).$$