Theory of Computation

November 16, 2010

This assignment gives you experience programming with primitive recursive functions. In the past I have assigned this as a pencil and paper exercise, however, some students have found it informative to approach this as a programming exercise.

In Chapter 1, Machtey and Young presents a version of the primitive recursive functions to define functions on strings over a k-symbol alphabet. I have written a Haskell interpreter for this formalization. It is in the file MY.hs.

In lecture and in the notes below, primitive recursive functions over natural numbers are defined. I have also written a Haskell interpreter for that formalization. It is in the file NaturalPR.hs.

For this assignment you have three options, any one of which is acceptable. You may do one of the following:

1. do the original assignment below with pencil and paper,
2. do the original assignment below as a programming exercise (possibly starting from NaturalPR.hs), or
3. work through the examples and exercises in Machtey and Young as a programming exercise including at least those problems that correspond to the problems listed below. You may use MY.hs if you wish.

1 Original Assignment

1. In lecture I presented five schemas for defining primitive recursive functions. They are as follows:

   (a) [Zero] There is a constant function zero of every arity.

   $$Z^k(x_1, \ldots, x_k) = 0$$

   (b) [Successor] There is a successor function of arity 1.

   $$S(x) = x + 1$$

(c) [Projection] There are projection functions for every argument position of every arity.

$$P_i^k(x_1, \ldots, x_k) = x_i \qquad \text{where } k > 0,\ i \le k$$

(d) [Composition (also called substitution)] The composition of the function f of arity k with functions $g_1, \ldots g_k$, each of arity l, defines a $f \circ_k^l [g_1 \ldots g_k]$ of arity l satisfying:

$$f \circ_k^l [g_1 \ldots g_k](x_1, \ldots, x_l) = f(g_1(x_1, \ldots, x_l), \ldots, g_k(x_1, \ldots, x_l))$$

(e) [Primitive Recursion] The arity k function defined by primitive recursion from a function g of arity $k-1$ and a function h of arity $k+1$ is indicated $\mathrm{PR}^k[g, h]$. It satisfies:

$$\begin{array}{lcl}\mathrm{PR}^k[g,h](0, x_2, \ldots, x_k) & = & g(x_2, \ldots, x_k) \\ \mathrm{PR}^k[g,h](x+1, x_2, \ldots, x_k) & = & h(x, \mathrm{PR}^k[g,h](x, x_2, \ldots, x_k), x_2, \ldots, x_k)\end{array}$$

In lecture we showed how to define addition by primitive recursion:

$$\mathrm{PR}^2[P_1^1, S \circ_1^3 [P_2^3]]$$

Using primitive recursion define:

(a) Multiplication

(b) If-then-else (e.g. $\mathrm{ITE}(1, x, y) = x, \mathrm{ITE}(0, x, y) = y$)

(c) Or

(d) And

(e) Define the bounded existential and universal quantifiers

  i. $\mathrm{BEQ}[\mathcal{P}] = \exists x < y.\mathcal{P}(x)$

  ii. $\mathrm{BUQ}[\mathcal{P}] = \forall x < y.\mathcal{P}(x)$

  Define these as functions of y. My solution uses primitive recursion on y, so the order of arguments is y first, x second. It helps to define if-then-else and "boolean" functions (I use 0 for false and 1 for true) first.

(f) Divides (use Bounded quantification to search for a divisor)

(g) Prime