CS 5000: Theory of Computation

Assignment 10: Partially Computable & Computable Functions

Vladimir Kulyukin
Department of Computer Science
Utah State University

Learning Objectives

1. Programming Language L
2. Partially Computable & Computable Functions
3. Composition & Primitive Recursion

Problem 1 (1 point)

Let $f(x_1, \ldots, x_n)$ be a partially computable function. Show that f is computed by infinitely many L-programs whose length $l \geq k, k \geq 0, k \in N$. This problem offers a mathematical justification, after a fashion, of the TIMTOWTDI (there is more than one way to do it) principle to which some software engineers and architects adhere.

Problem 2 (2 point)

An *L*-program *P* is *straightline* if it contains no instructions, labeled or unlabeled, of the form IF V != 0 GOTO L, for some label L. Show that if P is a straightline L-program of length $k \in N$, then $\psi_P^{(1)}(x) \leq k$.

Problem 3 (1 point)

This problem will give you a flavor of the formal theory of compilation. Let *L++* be a programming language that extends the programming language *L* by adding one instruction type: V ← k, $k \geq 0, k \in N$. Show that a function is partially computable in *L++* if and only if it is partially computable.

Problem 4 (1 point)

Here is a fun problem to probe and, possibly, improve your understanding of primitive recursion. Let $k > 0$ be some natural number. Let C be a class of total functions of no more than k variables. Is C primitive recursively closed? Sketch a proof why it is or why it is not. When you think about this problem, keep in mind the difference b/w *primitive recursive* functions and *primitive recursively closed* classes of functions.