CPSC 513 Winter 2010 Midterm: Solutions

1. (a) Define what it means for a function to be computable.

   A function $f : \mathcal{N}^m \to N$ is computable if it is total, and there exists a program P such that $\psi_P(x_1 \ldots x_m) = f(x_1 \ldots x_m)$.

   (b) Show that the function

   $$f(x) = \text{the largest } n \text{ such that } n^3 \leq x$$

   is computable.

   We could write a program to show that this function is computable, but instead we will show it it is primitive recursive (and hence computable). The function f is given by

   $$(\min_{n \leq x}(n^3 > x)) - 1$$

   as this calculates the smallest n for which $n^3 > x$, then subtracts 1, giving the largest n such that $n^3 \leq x$. By results from class, the above expression is primitive recursive, so f is primitive recursive and hence computable.

2. (a) Define what it means for a partial function to be μ-recursive.

   A partial function is μ-recursive if it is given by a finite number of applications of composition, primitive recursion, and unbounded minimization to the initial functions.

   (b) Show that every partially computable function is μ-recursive.

   Let $f(x_1 \ldots x_n)$ be a partially computable function. By the Normal Form theorem, there exists a primitive recursive predictate R so that

   $$f(x_1 \ldots x_n) = l(\min_z R(z, x_1 \ldots x_n))$$

   Since R and l are primitive recursive, we are applying composition and unbounded minimization to primitive recursive functions; hence the result is μ-recursive. Thus, f is μ-recursive, as required.

3. Suppose a sequence $f(n)$ is given by $f(0) = 2, f(1) = 3, f(2) = 7$, and for $n > 2$,

   $$f(n) = f(n-1) + f(n-2) + f(n-3).$$

   Show that $f(n)$ is primitive recursive.

   We begin by defining a new function g given by

   $$g(n) = [f(n), f(n+1), f(n+2)]$$

We will first show that g is primitive recursive. Indeed,

$$g(0) = [2, 3, 7]$$

and

$$g(t+1) = [f(t+1), f(t+2), f(t+3)] = [g(t)_2, g(t)_3, g(t)_3 + g(t)_2 + g(t)_1]$$

Since the index functions $(-)_i$ are primitive recursive, we have shown that g is given by applying primitive recusion to primitive recursive functions - hence g itself is primitive recursive.

Finally, since $g(n) = [f(n), f(n+1), f(n+2)]$, we have $f(n) = g(n)_1$. So, since g is primitive recursive, f is as well.

4. Give the statements of:

   (a) the Universality Theorem,

   (b) the Parameter Theorem.

   The Universality Theorem states that the function

   $$\Phi(x_1 \ldots x_n, y) = \psi_P(x_1 \ldots x_n)$$

   (where the number of P is y) is partially computable.

   The Parameter Theorem states that for any $n, m > 0$, there exists a primitive recursive function S^n_m such that

   $$\Phi(x_1 \ldots x_n, u_1 \ldots u_m, y) = \Phi(x_1 \ldots x_n, S^n_m(u_1 \ldots u_m, y).$$

5. Let A and B be recursively enumerable sets. Show that the set

   $$A \cup B = \{x : x \in A \text{ or } x \in B\}$$

   is recursively enumerable.

   Since A and B are recursively enumerable, there exist partially computable functions f, g so that $x \in A \Leftrightarrow f(x)\downarrow$ and $x \in B \Leftrightarrow g(x)\downarrow$. Suppose f is partially computed by the program with number p, and g by the program with number q. Then consider the following program:

```
(A) IF STP(X,p,T) GOTO E
IF STP(X,q,T) GOTO E
T ← T + 1
GOTO A
```

```
(E)
```

If either program p or program q terminates on input X, then this program terminates; otherwise, it runs forever. Thus, this program halts exactly when $X \in A$ or $X \in B$, and so $A \cup B$ is recursively enumerable.

6. (a) State Rice's Theorem.

   Suppose Γ is a set of partially computable functions so that there exists some functions f, g with $f \in \Gamma$ and $g \notin \Gamma$. Then the set $R_\Gamma := \{n : \Phi_n \in \Gamma\}$ is not recursive.

   (b) Let $W_x = \{n : \Phi(n, x) \downarrow\}$. Show that the sets

   $$\{x : W_x \text{ is infinite}\} \text{ and } \{x : W_x \text{ is finite}\}$$

   are not recursive.

   By Rice's Theorem, it suffices to show that the set of infinitely-defined partially computable functions is non-trivial, and the set of finitely-defined partially computable functions is non-trivial. For the first, the function $f(x) = x$ is infinitely-defined, but the function $g(x)$ which is undefined everywhere is not. For the second, the function $f(x)$ which is undefined everywhere is finitely-defined, and the function $g(x) = x$ is not finitely-defined. Thus, in both cases, the sets are not recursive.