CS5311 Homework #1 Solution (Spring 2001)

Due: Thursday, January 31, 2002 **before class**

100 points

1. **[6 points]** Suppose we have an infinite number of sets S_0, S_1, S_2, S_3, ..., each of which satisfies a property $\mathcal{P}$. Moreover, we also know that the union of two sets that satisfy property $\mathcal{P}$ also satisfies property $\mathcal{P}$. The following induction proves that $A = \bigcup_{i \geq 0} S_i$ satisfies property $\mathcal{P}$.

   (a) **Base Step $n = 0$ and $n = 1$:** Since $A = S_0$ and S_0 satisfies property $\mathcal{P}$, so does A. For $n = 1$, we have $A = S_0 \cup S_1$ and A satisfies property $\mathcal{P}$.

   (b) **Induction Step:** Suppose the proposition holds for n. Then, we have $A = S_0 \cup S_1 \cup \cdots \cup S_n \cup S_{n+1} = (S_0 \cup S_1 \cup \cdots \cup S_n) \cup S_{n+1}$. By the induction hypothesis, $B = S_0 \cup S_1 \cup \cdots \cup S_n$ satisfies property $\mathcal{P}$. Because B and S_{n+1} satisfy property $\mathcal{P}$, their union $A = B \cup S_{n+1} = S_0 \cup S_1 \cup \cdots \cup S_n \cup S_{n+1}$ satisfies property $\mathcal{P}$

   (c) From the induction, since the proposition holds for every $n \geq 0$, it holds for $A = \bigcup_{i \geq 0} S_i$, and A satisfies property $\mathcal{P}$.

   Is this proof completely correct? If it is not completely correct, correct the mistakes to make this proof work. If it is completely incorrect, point out the problem and elaborate why it is a problem.

   Answer: This proof is completely incorrect. It only proves that the sets S_0, $S_0 \cup S_1$, $S_0 \cup S_1 \cup S_2$, ..., $S_0 \cup S_1 \cup S_2 \cup \cdots S_n$, $\cdots$ satisfy property $\mathcal{P}$. More precisely, it proves that finite unions satisfy property $\mathcal{P}$; however, it does not establish the fact that the infinite union A satisfies property $\mathcal{P}$. ∎

2. (a) **[4 points]** Given a constant $k \geq 0$, show that the constant function $f_k(x) = k$ is computable.

   Answer: Since k is given, using the $V \leftarrow V + 1$ instruction k times solves the problems:

$$\left.\begin{array}{lcl} Y & \leftarrow & Y+1 \\ Y & \leftarrow & Y+1 \\ & \vdots & \\ Y & \leftarrow & Y+1 \end{array}\right\} k \text{ times}$$

   Therefore, constant function $f_k(x) = k$ is computable. ∎

   (b) **[10 points]** A *straightline program* is a program that does not contain the IF-GOTO instruction. Show that if the length of a straightline program $\mathcal{P}$ is k, then $\psi_{\mathcal{P}}^{(1)}(x) \leq k$ for all k. This means the output of a straightline program of k instructions must be less than or equal to k. Note that we have only three types of instructions $V \leftarrow V + 1$, $V \leftarrow V - 1$ and IF $V \neq 0$ GOTO L. (**Hint:** use induction on program length k)

   Answer: The induction hypothesis is: "if the length of a program $\mathcal{P}$ is k, then $\psi_{\mathcal{P}}^{(1)}(x) \leq k$."

   Base Cases $k = 0$: If $\mathcal{P}$ has no instruction, the output in Y is 0. Therefore, $\psi_{\mathcal{P}}^{(1)}(x) \leq 0$.

   Induction Step: Suppose the induction hypothesis is true for all programs with length $\leq k$. Let $\mathcal{P}$ be a program of length $k+1$. Let the program formed by the first k instructions of $\mathcal{P}$ be $\mathcal{P}'$. Since $\mathcal{P}'$ has k instructions, its output is at most k. After program $\mathcal{P}'$ completes, instruction $k+1$ of program $\mathcal{P}$ may increase Y by 1 by instruction $Y \leftarrow Y + 1$. Otherwise, the output is the same as that of program $\mathcal{P}'$. As a result, the output of program $\mathcal{P}$ is at most $k+1$. Therefore, the proposition is proved. ∎

(c) **[6 points]** Show that if $\mathcal{P}$ is a straightline program that computes f_k, then the length of $\mathcal{P}$ is at least k.

Answer: Suppose there exists a program $\mathcal{Q}$ of length $n < k$ that computes f_k. By (b) above, we have $\psi^{(1)}_{\mathcal{Q}}(x) \leq n < k$. Therefore, we have a contradiction, and any straightline program that computes f_k must have a program length at least k. Consequently, the shortest straightline program that computes f_k has length k and the program that answers (a) is the shortest. ∎

A Note

The proof for the original proposition in which "straightline" is omitted is more difficult and is worth of more than 6 points. As a result, as long as you supply a correct proof with the straightline assumption, you get full credit. One of you tried not to use the straightline assumption; however, that proof is only partially correct.

(d) **[6 points]** Show that no straightline program can compute the function $g(x) = x + 1$. Therefore, the class of all straightline programs is a proper subset of the class of all computable functions.

Answer: Suppose there is a program $\mathcal{P}$ that computes $g(x) = x + 1$. Let the length of this program be m. Then, the output of this program is at most m. However, since for $x \geq m$, $g(x) = x + 1 > m$, we have a contradiction, and $g(x) = x + 1$ cannot be computed by any straightline program. ∎

3. Let COMP be the class of functions obtained from the initial functions by a finite sequence of compositions.

(a) **[15 points]** Show that every function $f(x_1, x_2, \ldots, x_n)$ in COMP, either it is a constant function $f(x_1, x_2, \ldots, x_n) = k$ for some constant k, or $f(x_1, x_2, \ldots, x_n) = x_i + k$ for some $1 \leq i \leq n$ and some constant k.

Answer: We can use induction on the number of compositions applied in the construction of function f. To this end, "the number of compositions" must be precisely defined. Let $f(x_1, x_2, \ldots, x_n) = g(h_1(x_1, x_2, \ldots, x_n), h_2(x_1, x_2, \ldots, x_n), \ldots, h_k(x_1, x_2, \ldots, x_n))$, where h_1, h_2, …, h_k and g functions are in COMP. Let function h_j, $1 \leq j \leq k$, be constructed using m_j function compositions, and g be constructed with m function compositions. Then, function f is constructed with $(m_1 + m_2 + \cdots m_k) + m + 1$ function compositions. After the "number of compositions" is precisely defined, we can use induction on this "number." Our induction hypothesis is: "if a function $f(x_1, x_2, \ldots, x_n)$ in COMP is constructed using t function compositions, then $f(x_1, x_2, \ldots, x_n) = k$ or $f(x_1, x_2, \ldots, x_n) = x_i + k$ for some $1 \leq i \leq n$ and constant k."

Base Case $t = 0$: Only the initial functions do not need function compositions. The successor function $s(x) = x + 1$ fits the second form (*i.e.*, $i = 1$ and $k = 1$), the zero function $n(x) = 0$ fits the first form with $k = 0$, and a projection function $\pi^h_i(x_1, x_2, \ldots, x_h) = x_i$ fits the second form with $k = 0$. Thus, the base case holds.

Induction Step: Suppose the induction hypothesis holds for all functions in COMP constructed with less than t function compositions. Let $f(x_1, x_2, \ldots, x_n)$ be a function in COMP constructed with t function compositions. Then, f is in a form of

$$f(x_1, x_2, \ldots, x_n) = g(h_1(x_1, x_2, \ldots, x_n), h_2(x_1, x_2, \ldots, x_n), \ldots, h_k(x_1, x_2, \ldots, x_n))$$

and functions h_j's and g are constructed with less than t function compositions. (Why?) Thus, h_j's and g must be one of the two given forms. If g is a constant function (*i.e.*, $g(x_1, x_2, \ldots, x_k) = s$, where s is a constant), then f is also a constant function (*i.e.*, $f(x_1, x_2, \ldots, x_n) = s$). If $g(x_1, x_2, \ldots, x_k) = x_j + s$ for some $1 \leq j \leq k$ and a constant s, then we have $f(x_1, x_2, \ldots, x_n) = h_j(x_1, x_2, \ldots, x_n) + s$. By the induction hypothesis, h_j must be in one of the two forms. If

$h_j(x_1, x_2, \ldots, x_n) = p$ for some constant p, then $f(x_1, x_2, \ldots, x_n) = p + s$ is a constant function. If $h_j(x_1, x_2, \ldots, x_n) = x_g + p$ for some $1 \leq g \leq n$ and a constant p, we have $f(x_1, x_2, \ldots, x_n) = x_g + (p+s)$, which is in the second form. Consequently, any function in COMP that is constructed with t function composition must be in one of the two two given forms. ■

Your Problems

Two obvious problems are **(1)** the hypothesis is not clearly stated, and **(2)** the hypothesis is "the proposition being true for the number of compositions." I will take points off for **(1)** should it happen in future homework. Item **(2)** is more serious. In your proofs, you said: "let $f(x_1, x_2, \ldots, x_n) = g(h_1(x_1, x_2, \ldots, x_n), h_2(x_1, x_2, \ldots, x_n), \ldots, h_k(x_1, x_2, \ldots, x_n))$ be a function that uses $i + 1$ compositions," and continue with "because all h_j's and g use less than i compositions, they must be one of the two forms." This is a big mistake, because h_j's and g may use more than $i+1$ compositions and are not covered by the induction hypothesis.

(b) **[6 points]** An n-ary function f is *monotonic* if for all n-tuples $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ such that $x_i \leq y_i$ for all $1 \leq i \leq n$, then $f(x_1, x_2, \ldots, x_n) \leq f(y_1, y_2, \ldots, y_n)$. Show that every function in COMP is monotonic.

<u>Answer:</u> By the above proposition, $f(x_1, x_2, \cdots, x_n) = k$ of $f(x_1, x_2, \ldots, x_n) = x_i + k$ for some $1 \leq i \leq n$ and constant k. If f is a constant function, it is already monotonic. If f is in the latter form, then we have $f(x_1, x_2, \ldots, x_n) = x_i + k \leq y_i + k = f(y_1, y_2, \ldots, y_n)$ if $x_j \leq y_j$ for all $1 \leq j \leq k$. Therefore, f is monotonic. ■

(c) **[6 points]** A predicate is *trivial* if and only if it is always TRUE or always FALSE. Show that no nontrivial predicate can be a member of COMP. Therefore, COMP is proper subset of the class of primitive recursive functions.

<u>Answer:</u> Suppose there is a non-trivial predicate $P(x_1, x_2, \ldots, x_n)$ in COMP. By the first proposition, $P(x_1, x_2, \ldots, x_n) = k$ or $P(x_1, x_2, \ldots, x_n) = x_i + k$ for some $1 \leq i \leq n$ and constant k. Because P is non-trivial by assumption, $P(x_1, x_2, \ldots, x_n) = x_i + k$. However, since P is total and x_i can be an arbitrary value, $x_i + k$ may be greater than 1. Consequently, P is not a predicate and all predicates in COMP must be trivial. ■

4. Prove or disprove the following propositions:

(a) **[8 points]** The class of all total functions is a PRC class.

<u>Answer:</u> This statement is true. Note that in our context, "functions" are always total; otherwise, a simple partial function would serve as a perfect counterexample.

Let the set of all "total" functions be $\mathcal{T}$. The initial functions are total functions and hence are members of $\mathcal{T}$. Next, we shall show that $\mathcal{T}$ is closed under composition and recursion.

Composition: Let $h(x_1, x_2, \ldots, x_n) = f(g_1(x_1, x_2, \ldots, x_n), g_2(x_1, x_2, \ldots, x_n), \ldots, g_k(x_1, x_2, \ldots, x_n))$, where f and g_i's are members of $\mathcal{T}$. Recall that the domain of h is the domain of all g_i's. Since $g_i \in \mathcal{T}$ for $1 \leq i \leq k$, f receives k values. Since f is in $\mathcal{T}$, f is defined on $g_1(x_1, x_2, \ldots, x_n)$, $g_2(x_1, x_2, \ldots, x_n)$, ..., $g_k(x_1, x_2, \ldots, x_n)$. Hence, h is total.

Recursion: Suppose h is obtained by an application of recursion:

$$\begin{aligned} h(x_1, \ldots, x_n, 0) &= f(x_1, \ldots, x_n) \\ h(x_1, \ldots, x_n, t+1) &= g(t, h(x_1, \ldots, x_n, t), x_1, \ldots, x_n) \end{aligned}$$

where f and g are in $\mathcal{T}$. We shall show h is total by induction on t.

Base case ($t = 0$): Since f is total, $h(x_1, \ldots, x_n, 0)$ is defined.

Induction step: Suppose $h(x_1, \ldots, x_n, t)$ is defined. Then, since g is total, $g(t, h(x_1, \ldots, x_n, t), x_1, \ldots, x_n)$ is defined, and, as a result, $h(x_1, \ldots, x_n, t+1)$ is defined. Therefore, h is total. ■

(b) **[8 points]** The intersection of two PRC classes is a PRC class.

<u>**Answer:**</u> Since $\mathcal{C}_1$ and $\mathcal{C}_2$ are PRC classes, both contains the initial functions. Therefore, initial functions are members of $\mathcal{C}_1 \cap \mathcal{C}_2$.

Suppose f is obtained by a finite number of applications of compositions and recursions of functions in $\mathcal{C}_1 \cap \mathcal{C}_2$. Since $\mathcal{C}_1 \cap \mathcal{C}_2$ is a subset of both $\mathcal{C}_1$ and $\mathcal{C}_2$, after applying finite number of compositions and recursions, the resulting function f is a member of PRC classes $\mathcal{C}_1$. The same holds for PRC class $\mathcal{C}_2$. Therefore, f belongs to both $\mathcal{C}_1$ and $\mathcal{C}_2$ (*i.e.*, $f \in \mathcal{C}_1 \cap \mathcal{C}_2$). Therefore, $\mathcal{C}_1 \cap \mathcal{C}_2$ is a PRC class. ■

5. **[10 points]** Prove or disprove: if function $f(x_1, x_2, \ldots, x_n)$ is a primitive recursive function, then function $g(x) = f(\underbrace{x, x, \ldots, x}_{n \text{ terms}})$ is also primitive recursive.

<u>**Answer:**</u> This is a very simple problem. Since $f(x_1, x_2, \ldots, x_n)$ and the projection function $\pi_1^1(x) = x$ are both primitive recursive, there composition $g(x) = f(\pi_1^1(x), \pi_2^1(x), \ldots, \pi_n^1(x))$ must also be primitive recursive. ■

6. **[15 points]** Suppose function f is defined as follows:

$$f(x, 0) = g(x) \qquad \text{and} \qquad f(x, y+1) = f(f(x, y), y)$$

where g is a primitive recursive function. Show that f is primitive recursive.

<u>**Answer:**</u> We need first to convert $f(x, y)$ to a manageable form. Let us expand $f(x, y)$:

$$\begin{aligned}
f(x,0) &= g(x) \\
f(x,1) &= f(f(x,0),0) = f(g(x),0) = g(g(x)) = g^2(x) \\
f(x,2) &= f(f(x,1),1) = f(g^2(x),1) = g^2(g^2(x)) = g^4(x) \\
f(x,3) &= f(f(x,2),2) = f(g^4(x),2) = g^4(g^4(x)) = g^8(x)
\end{aligned}$$

It looks like that the exponent (*i.e.*, number of composition) of g is 2^y. Thus, we expect that $f(x, y) = g^{(2^y)}(x)$. Next, we should prove this is correct (by induction).

Base Case: $y = 0$
Since $g^{(2^0)}(x) = g(x) = f(x, y)$, the base case holds.

Induction Step:
Assume $f(x, y) = g^{(2^y)}(x)$. We shall prove $f(x, y+1) = g^{(2^{y+1})}(x)$ as follows:

$$\begin{aligned}
f(x, y+1) &= f(f(x,y), y) && \text{by definition of } f \\
&= f(g^{(2^y)}(x), y) && \text{induction hypothesis} \\
&= g^{(2^y)}(g^{(2^y)}(x)) && \text{by definition of } f \\
&= g^{(2^y + 2^y)}(x) \\
&= g^{(2^{(y+1)})}(x)
\end{aligned}$$

Therefore, $f(x, y) = g^{(2^y)}(x)$ holds.

Now, we shall prove that f is primitive recursive:

$$\begin{aligned} f(x,0) &= g(x) \\ f(x,y+1) &= g^{2^{y+1}}(x) \\ &= g^2(g^{2^y}(x)) \\ &= g^2(f(x,y)) \\ &= h(y, f(x,y), x) \end{aligned}$$

where $h(u,v,w) = g^2(v) = g^2(\pi_2^3(u,v,w))$. Since $\pi_2^3()$ and g are primitive recursive, $h()$ is primitive recursive. As a result, f is constructed with primitive recursion and hence is primitive recursive. ∎

Your Problem

I saw a completely wrong proof. This proof shows that, for every i, $f(x,i)$ is primitive recursive. Then, it claims $f(x,y)$ is primitive recursive. This is terribly wrong. What this proof does is showing that functions $g_0(x) = f(x,0)$, $g_1(x) = f(x,1)$, $g_2(x) = f(x,2)$. …, $g_i(i) = f(x,i)$, … are primitive recursive. It does not prove the two-variable function $f(x,y)$ is primitive recursive.