CS5311 Quiz 1 Solutions (Spring 2002)

1. **[20 points]** Let f be defined as follows:

$$f(x) = \text{the number of perfect squares} \leq x$$

x is a perfect square if and only if there is a y such that $x = y^2$. Thus, 0, 1, 4, 9, 16, 25, ... are perfect squares. Show that f is primitive recursive.

Answer: A number p is a perfect square if there exists a t such that $t^2 = p$. Of course, $t \leq p$. To find the number of perfect squares $\leq x$, we can simply count the number of t^2's that satisfy $t^2 \leq x$. For example, if $x = 20$, then we see $0^2 \leq 20$, $1^2 \leq 20$, $2^2 \leq 20$, $3^2 \leq 20$, and $4^2 \leq 20$. Therefore, the perfect squares are 0, 1, 4, 9 and 16, and $f(20) = 5$. In other words, function f is simply

$$f(x) = \sum_{t=0}^{x} \left(t^2 \leq x\right)$$

Since $\sum$ and $\leq$ are primitive recursive, f is primitive recursive. ∎

2. **[20 points]** Define predicate CRASH as follows:

$$\text{CRASH}(p, x) \Leftrightarrow \text{program } p \text{ running on input } x \text{ crashes}$$

Show that CRASH(p, x) is not computable. Therefore, it is impossible to design a super program that can determine if a given program running on certain input will crash by scanning the given program.

Answer: This problem is exactly the duplication of the anti-virus problem. If CRASH is computable, we can write a program and use it in another program. Consider the following program:

IF CRASH(X,X) GOTO E
crash the program by whatever means

Let the program number of the above program be p. Then, we have the following:

$$\begin{aligned} \text{CRASH}(p, x) &\Leftrightarrow \text{program } p \text{ running on } x \text{ crashes} \\ &\Leftrightarrow \text{CRASH}(x,x) \text{ is false in } p \\ &\Leftrightarrow \neg\text{CRASH}(x, x) \end{aligned}$$

Since x is an arbitrary input, replacing x with p yields

$$\text{CRASH}(p, p) \Leftrightarrow \neg\text{CRASH}(p, p)$$

This is a contradiction and CRASH is not computable. ∎

3. **[20 points]** Define set $A + B$ as follows:

$$A + B = \{\ x + y \mid x \in A \text{ and } y \in B\ \}$$

Show that if A and B are r.e., then $A+B$ is also r.e. (**Hint**: Use the strong form of Theorem 4.8 proved in class. We call it the Projection Theorem. See the last page for the details.)

Answer: This is a simple application of the Projection Theorem. Since A and B are both r.e., there exist two primitive recursive predicates $R_A(x,t)$ and $R_B(x,t)$ such that

$$\begin{aligned} A &= \{\ x \mid (\exists t) R_A(x,t)\ \} \\ B &= \{\ x \mid (\exists t) R_B(x,t)\ \} \end{aligned}$$

Therefore, we have

$$\begin{aligned} z \in A + B &\Leftrightarrow \exists_{x,y}[z = x + y \ \wedge\ x \in A \ \wedge\ y \in B] \\ &\Leftrightarrow \exists_{x,y}[z = x + y \ \wedge\ (\exists u) R_A(x,u) \ \wedge\ (\exists v) R_B(y,v)] \end{aligned}$$

Now, there are four variables x, y, u and v. We can use Gödel number to encode them into one number as shown below:

$$z \in A + B \Leftrightarrow (\exists t)[\mathrm{Len}(t) = 4 \ \wedge\ z = (t)_1 + (t)_2 \ \wedge\ R_A((t)_1,(t)_3) \ \wedge\ R_B((t)_2,(t)_4)]$$

where $(t)_1$, $(t)_2$, $(t)_3$ and $(t)_4$ represent x, y, u and v, respectively. Since we have only four numbers, we require $\mathrm{Len}(t) = 4$. Is this requirement absolutely necessary? Let $R_{A+B}(z,t) = (\exists t)[\mathrm{Len}(t) = 4 \ \wedge\ z = (t)_1 + (t)_2 \ \wedge\ R_A((t)_1,(t)_3) \ \wedge\ R_B((t)_2,(t)_4)]$. Then, $A + B = \{\ z \mid (\exists t) R_{A+B}(z,t)\ \}$, Since $R_{A+B}(z,t)$ is primitive recursive, $A+B$ is r.e. ∎

4. **[20 points]** Let A, B and C be three mutually exclusive sets (*i.e.*, $A \cap B = \emptyset$, $A \cap C = \emptyset$ and $B \cap C = \emptyset$). Assume there exists three partially computable functions f_1, f_2 and f_3 that satisfying the following properties:

$$f_1(x) = \begin{cases} 1 & \text{if } x \in A \cup B \\ 2 & \text{if } x \in C \\ \uparrow & \text{otherwise} \end{cases}$$

$$f_2(x) = \begin{cases} \uparrow & \text{if } x \in A \cup C \\ 0 & \text{otherwise} \end{cases}$$

$$f_3(x) = \begin{cases} \uparrow & \text{if } x \in B \cup C \\ 0 & \text{otherwise} \end{cases}$$

Show that A, B and C are all recursive. (**Hint**: Use Theorem 4.4.)

Answer: From f_1, we know that $A \cup B$, C and $A \cup B \cup C$ are all r.e., because they are the domains of restricted functions of f_1. From f_2 (*resp.*, f_3), we know that $\overline{A \cup C}$ (*resp.*, $\overline{B \cup C}$) is r.e.

Since $\overline{C} = \overline{A \cup C} \cup \overline{B \cup C}$, and since $\overline{A \cup C}$ and $\overline{B \cup C}$ are both r.e., $\overline{C}$ is r.e., and C is recursive.

Since $A = \overline{B \cup C} \cap (A \cup B)$, and since $\overline{B \cup C}$ and $A \cup B$ are both r.e., A is r.e. Since $\overline{A} = \overline{A \cup C} \cup C$, and since $\overline{A \cup C}$ and C are both r.e., $\overline{A}$ is r.e., and A is recursive.

Since $B = \overline{A \cup C} \cap (A \cup B)$, and since $\overline{A \cup C}$ and $A \cup B$ are both r.e., B is r.e. Since $\overline{B} = \overline{B \cup C} \cup C$, and since $\overline{B \cup C}$ and C are both r.e., $\overline{B}$ is r.e., and B is recursive. ∎

5. **[20 points]** Define a function f as follows:

$$f(x) = \begin{cases} \uparrow & \text{if } \Phi^{(1)}(x,x)\downarrow \\ 1 & \text{otherwise} \end{cases}$$

Show that f is not partially computable. (**<u>Hint</u>**: Use the enumeration property and diagonalization.)

<u>Answer</u>: If f is partially computable, there is program p that computes f. More precisely, there is a p such that $f(x) = \Phi^{(1)}(x, p)$. Therefore, from this program p, we have

$$f(x)\downarrow \Leftrightarrow \ \Phi^{(1)}(x,p)\downarrow \tag{1}$$

By the definition of f, we have

$$f(x)\downarrow \Leftrightarrow \ \Phi^{(1)}(x,x)\uparrow \tag{2}$$

Since x is an arbitrary input, replacing x with p yields

$$\begin{aligned} \Phi^{(1)}(p,p)\downarrow \ &\Leftrightarrow \ f(p)\downarrow && \text{Eqn (1)} \\ &\Leftrightarrow \ \Phi^{(1)}(p,p)\uparrow && \text{Eqn (2)} \end{aligned}$$

Therefore, since $\Phi^{(1)}(p,p)$ cannot be defined and undefined at the same time, we have a contradiction, and f is not partially computable. ∎

Projection Theorem

The following three propositions are equivalent:

- B is an r.e. set
- There exists a primitive recursive predicate $R(x, y)$ such that

$$B = \{ \ x \in \mathcal{N} \mid (\exists t) R(x,t) \ \}$$

- There exists a computable predicate $R(x, y)$ such that

$$B = \{ \ x \in \mathcal{N} \mid (\exists t) R(x,t) \ \}$$