Math 29: Computability Theory **Spring 2024**

PSET 5 — 05/12/2024

Prof. Miller ***Student: Amittai Siavava***

Problem 1.

Show that if C is a computable set, then $C \leq_m X$ for any set X which is nonempty and has nonempty complement.

Let C be a computable set and X be a set which is nonempty and has nonempty complement. Since X is nonempty and has nonempty complement, there is some $x \in X$ and some $y \notin X$. Define the function f by

$$f(n) = \begin{cases} x & \text{if } n \in C, \\ y & \text{if } n \notin C. \end{cases}$$

Since C is computable, there is some Turing machine φ_e which computes χ_C, hence given any input n, φ_e can determine whether $n \in C$ or $n \notin C$. Therefore, f is computable. We claim that $C \leq_m X$ via f. To see this, note that $n \in C$ if and only if $f(n) = x \in X$, and $n \notin C$ if and only if $f(n) = y \notin X$. Thus, $n \in C$ if and only if $f(n) \in X$, which shows that $C \leq_m X$.

Problem 2.

Let B be an infinite c.e. set. Is there an immune set I such that $B \leq_1 I$? Justify your answer.

No, there is no immune set I such that $B \leq_1 I$.

Suppose for the sake of contradiction that there is an immune set I such that $B \leq_1 I$, with $f_I : \omega \to \omega$ being an computable, injective function that witnesses the reduction. Since B is c.e., there exists a total, computable, injective function $f_B : \omega \to \omega$ such that $B = \textbf{range}\,(()\, f_B)$ (*we proved this result in a previous assignment*).

Consider the composition $f \coloneqq f_B \circ f_I$.

1. Since both f_B and f_I are injective, so is f.
2. Since f_B is total and computable, and f_I is total and computable, so is f.
3. Since $\textbf{range}\,(f_B) = B$ and f_I is injective from B to I;

$$\textbf{range}\,(f) = f_I(\textbf{range}\,(f_B)) f_I(B) = I$$

Thus, I is the range of f, which implies that I is c.e., hence not immune. This contradicts the fact that I is immune. Therefore, the assumption that such an I exists must be wrong.

Notation wise, given a set S, we denote by $f(S)$ to be the image of S under f:

$$f(S) = \{\ f(n) \mid n \in S\ \}.$$

Problem 3.

Are there uncountably many Turing degrees? Justify your answer.

Yes, there are uncountably many Turing degrees.

For any set A, there are countably many sets B such that $B \equiv_T A$ since for each B, $\chi_A = \Phi_e^B$ for some $e \in \omega$, and there are countably many such e. Thus, each Turing degree contains countably many sets, yet there are $2^{|\omega|} = 2^{\aleph_0}$ subsets of ω. Since $2^{\aleph_0}$ is uncountable, and each Turing degree can only contain countably many sets, there must be uncountably many Turing degrees.

Problem 4.

$A \oplus B$, "A join B", is defined as

$$A \oplus B = \{\, 2x \mid x \in A \,\} \cup \{\, 2x+1 \mid x \in B \,\}\,.$$

Prove that the Turing degree of $A \oplus B$ is a least upper bound of the Turing degrees of A and B. In other words, show that it computes both A and B, and that any C which computes both A and B also computes $A \oplus B$.

Let $A, B \subseteq \omega$ be sets. We will show that $A \oplus B$ is the least upper bound of A and B.

1. *We claim that $A \oplus B$ computes A and B.*

Let $x \in A$. Then $2x \in A \oplus B$. Let $y \in B$. Then $2y+1 \in A \oplus B$. Thus, $A \leq_T A \oplus B$ and $B \leq_T A \oplus B$. Specifically, we can compute χ_A and χ_B as follows:

Let φ_e be the machine that computes $\chi_{A\oplus B}$.

(i) To compute $\chi_A(x)$, Define Φ_k^B to be the oracle machine that, given x, simulates $\varphi_e(2x)$ and returns the result. *If $2x \in A \oplus B$, then $x \in A$. Otherwise, $x \notin A$.*
In other words, we compute

$$\chi_A(x) = \Phi_k^B(x) = \varphi_e(2x).$$

(ii) To compute $\chi_B(x)$, Define Φ_ℓ^B to be the oracle machine that, given x, simulates $\varphi_e(2x+1)$ and returns the result. *If $2x+1 \in A \oplus B$, then $x \in B$. Otherwise, $x \notin B$.*
In other words, we compute

$$\chi_B(x) = \Phi_\ell^B(x) = \varphi_e(2x+1).$$

Thus, $A \leq_T A \oplus B$ and $B \leq_T A \oplus B$.

2. *We claim that any set C which computes both A and B also computes $A \oplus B$.*

Let C be a set such that $A \leq_T C$ and $B \leq_T C$. We will show that $A \oplus B \leq_T C$.
Let $x \in A \oplus B$. By definition of $A \oplus B$, for every $x \in A \oplus B$, either $x = 2y$ for some $y \in A$ or $x = 2z+1$ for some $z \in B$.

(i) Suppose $x = 2y$ for some $y \in A$. Since $A \leq_T C$, C can compute y.
Thus, C can compute $2y = x$.

(ii) Suppose $x = 2z+1$ for some $z \in B$. Since $B \leq_T C$, C can compute z.
Thus, C can compute $2z+1 = x$.

Therefore, C can compute all $x \in A \oplus B$, hence $A \oplus B \leq_T C$.

Problem 5.

Given a countable sequence of sets $\{ A_i \}_{i\in\omega}$, define the *infinite join* $A = \bigoplus_{i\in\omega} A_i$ by

$$A = \{ \langle i, n\rangle \mid n \in A_i \} .$$

Prove that there are sequences $\{ A_i \}_{i\in\omega}$ and $\{ B_i \}_{i\in\omega}$ such that $A_i \equiv_T B_i$ for all i but $A \not\equiv_T B$. In other words, this operation is defined on sets, but not on degrees (unlike the finite joins).

Hint: make A computable but B not computable.

For each $i \in \omega$, let

$$A_i = \{ i \}$$

and

$$B_i = \{ e \mid \varphi_{e,i}(e) \downarrow \} .$$

First, we show that each $A_i \equiv_T B_i$.

1. $A_i \leq_T B_i$. Note that each A_i is computable, since it is a singleton set containing i. Thus, for any A_i and B_i, we can define $\Phi_e^{B_i}$ to be an oracle machine that, given n, ignores B_i and computes $f : \omega \to \omega$ as follows:

$$f(n) = \begin{cases} 1 & \text{if } n = i, \\ 0 & \text{otherwise.} \end{cases}$$

2. $B_i \leq_T A_i$. Each B_i is also computable since we limit the number of execution steps to a maximum of i. We can define $\Phi_e^{A_i}$ to be an oracle machine that, given n, ignores A_i and simulates $\varphi_{n,i}(n)$ up to a maximum of i steps, computing the following function:

$$f(n) = \begin{cases} 1 & \text{if } \varphi_{n,i}(n) \downarrow, \\ 0 & \text{otherwise.} \end{cases}$$

Next, we show that $A \not\equiv_T B$.

Let

$$A = \{ \langle i, n\rangle \mid n \in A_i \} = \{ \langle i, i\rangle \mid i \in \omega \}$$

and

$$B = \{ \langle i, n\rangle \mid n \in B_i \} = \{ \langle i, n\rangle \mid \varphi_{n,i}(n) \downarrow \} .$$

Since the number of execution steps in B is not restricted to a specific i, B has the capability to solve the halting problem, which is known to be non-computable. Thus, B is not computable, while A is computable. Therefore, $A \not\equiv_T B$.