Math 29: Computability Theory **Spring 2024**

PSET 7 — 05/24/2024

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

Problem 1.

Are there sets A and B such that $A' \leq_T B'$ but $A \not\leq_T B$? Justify your answer.

Yes.

Pick $B = \varnothing$, which is computable, and let A be any non-computable set that is **low**, meaning $\varnothing' \equiv_T A'$. *As we showed in class using the* ***Low Basis Theorem*** *and* ***Sacks' Splitting Theorem****, such sets exist (and in fact, there are many of them).* Then:

1. Since B is computable but A is non-computable,
   A has a higher Turing degree than B, so $A \not\leq_T B$.
2. Since, $B' = \varnothing'$ and $A' \equiv_T \varnothing'$, meaning $A' \equiv_T B'$. Therefore, $A' \leq_T B'$ and $B' \leq_T A'$.

Problem 2.

(a) Prove that if there is $g \leq_T X$ such that $\varphi_{g(x)} \neq \varphi_x$, then there is $h \leq_T X$ such that $h(e) \neq \varphi_e(e)$ for all e.

This means that g is fixed point-free. Define h as follows:

$$h(e) = g(\varphi_e(e)).$$

1. First, note that $\varphi_{g(x)} \neq \varphi_x$ means that $g(x) \neq x$ for all x, as that would trivially imply $\varphi_{g(x)} = \varphi_x$ for some x. Therefore, $h(e) = g(\varphi_e(e)) \neq \varphi_e(e)$ for all e.
2. Next, we show that $h \leq_T X$. Since $g \leq_T X$, we can use X as an oracle to compute $g(e)$ for any e. Specifically, there exists an oracle machine k that uses X as an oracle to compute g, such that $\Phi_k^X(e) = g(e)$ for all e. We can compute $h(e)$ by simulating $\mathit{\Phi}_k^X(\varphi_e(e))$ and returning the result, thus h can also be computed by an oracle machine that takes X as an oracle and, on input e, simulates $\mathit{\Phi}_k^X$ on $\varphi_e(e)$.

(b) Given $h \leq_T X$ such that $h(e) \neq \varphi_e(e)$ for all e, show that there is $f \leq_T X$ such that $W_{f(e)} \neq W_e$ for all e.

Hint: make $\left|W_{f(e)}\right| = 1$.

For each e, define e' such that:

$$\varphi_{e'}(x) = \begin{cases} h(e) & \text{if } x = e \\ \uparrow & \text{otherwise.} \end{cases}$$

This ensures that for any e, $\varphi_{e'}(e) = h(e) \neq \varphi_e(e)$, and $W_{e'} = \{e\}$ for all e and corresponding e'.

Finally, define f as follows:

$$f(x) = (x+1)'.$$

1. First, since $f(e) = (e+1)'$, $W_{f(e)} = \{e+1\} \neq \{e\} = W_e$.
2. Next, we show that $f \leq_T X$. Since X is an oracle for h, we can use X to compute $h(e)$ for any e. Specifically, there exists an oracle machine k that uses X as an oracle to compute h, such that $\Phi_k^X(e) = h(e)$ for all e. We can compute f as follows:
   On input e, construct an oracle machine that, on input x, simulates $\Phi_k^X(x)$ if $x = e+1$, and otherwise diverges.
   Return the code of this machine as $f(e)$.

Problem 3.

Verify that the f constructed in the High-Low lecture notes dominates every total computable function, but does not compute K.

We define $f = \bigcup_s \sigma_s$, where:

1. Define $\sigma_0 = \varnothing$.
2. Given σ_s, define σ_{s+1} as follows:
   (a) Say that e "looks total up to n" if there exists some t such that $\varphi_{e,t}(x) \downarrow$ for all $x \leq n$. *Note that $\varnothing'$ can determine if e is total up to n because this is a Σ^0_1 question.*
   (b) Look for a τ properly extending σ such that $\varphi_e(x) \leq \tau(x)$ for all $|\sigma| < x \leq |\tau|$ and all $e \leq s$ which look total up to $|\tau|$, and an x such that $\Phi^\tau_s(x) \downarrow \neq K(x)$. If there is such a τ and x, let $\sigma_{2s+1} = \tau$. If not, let $\sigma_{2s+1} = \sigma_{2s}$. *Similarly, this is a Σ^0_1 question, so $\varnothing'$ can determine if such a τ exists.*

We now show that (1) f dominates every total computable function, and (2) f does not compute K.

1. Let g be a total computable function. We show that there exists some x' such that $g(x) \leq f(x)$ for all $x > x'$.
   (a) Since g is total computable, there exists some e such that $\varphi_e = g$.
   (b) Accordingly, for every x, there exists some τ such that $|\sigma| < x \leq |\tau|$, so $\varphi_e(x) \leq \tau(x)$. By the definition that $\sigma_{2s+1} = \tau$, we have that $\varphi_e(x) \leq \sigma_{2s+1}(x) = f(x)$.
2. We show that f does not compute K.

   In particular, we know by Rice's theorem that K is non-computable, and f is computable (given any fixed input x, we can walk through the constructions of σ_s until we find the appropriate one that gives the value for $f(x)$). Therefore, f has a smaller Turing degree than K and cannot compute K.

Problem 4.

Prove that no ML-random set has an infinite c.e. subset (i.e., every ML-random set is immune).

Hint: use a lemma from the class notes.

In the class notes, we proved that no ML-random set has an infinite computable subset. We can use this result to prove that no ML-random set has an infinite c.e. subset by way of contradiction.

Suppose R is an ML-random set with an infinite c.e. subset $A \subseteq B$.

First, note that every c.e. set has a computable subset. Specifically, given A is c.e., then we can computably enumerate members of A. Let a_i be the ith element enumerated in A. Define

$$A' = \{a_i \mid a_i > a_j \text{ for all } j < i\}.$$

That is, A' is set of elements in a *unique* and *strictly increasing* sequence of members of A.

We claim that A' is computable. Specifically, given any x, we can enumerate members of A until we either list x, then we know $x \in A'$, or we list a number greater than x, then we know $x \notin A'$.

Thus: if an ML-random set R has an infinite c.e. subset A, then it necessarily has an infinite computable subset $A' \subseteq A \subseteq R$. However, the lemma tells us that no ML-random set can have an infinite computable subset, so it must be the case that R cannot have an infinite c.e. subset either.