COMP-170: Homework #6

Ben Tanen - March 12, 2017

Problem 5

Prove that the function $C(x) = \min |\langle M, d \rangle|$, where M is a Turing machine which outputs x when started on input d is computable by machine that has oracle access to $\mathrm{A_{TM}}$

* * *

To show that the function $C(x) = \min |\langle M, d \rangle|$ is computable by a machine that oracle access to A_{TM}, we will first construct a recognizer for the language $L = \{\langle M, d, x \rangle \mid M \text{ outputs } x \text{ when started on input } d\}$. Using this recognizer and a properly phrased question for our A_{TM} oracle, we will then be able to construct a machine that can compute $C(x)$, proving that $C(x)$ is computable.

Consider the language $L = \{\langle M, d, x \rangle \mid M \text{ outputs } x \text{ when started on input } d\}$. We claim L is recognizable. In order to prove this, we will build a machine R_L that can recognize L. Let R_L be defined as follows:

R_L **on input** $\langle M, d, x \rangle$**:**

Run M on d

If M accepts or rejects d, check if output of M is equal to x

If output is equal to x, *ACCEPT*

If output is not equal to x, *REJECT*

We will now claim that R_L is a recognizer of L. To show this, consider the following cases:

1. Let $\langle M, d, x \rangle \in L$ such that M outputs x when started on d. Because we know that M outputs x from input d, we know that M either accepts or rejects d. Thus, we can see, once this happens, we can check the output and determine that the output is equal to x (which is given from our definition of M, d, and x). Thus, in this case, we will see that R_L will correctly accept $\langle M, d, x \rangle$.
2. Let $\langle M, d, x \rangle \notin L$ such that M accepts or rejects d but does not output x. Similar to #1, we can see that we will not loop when we run M on d. However, because of our definition of M, d, and x, when we check if the output from M is equal to x, we will see that the output is not equal to x so R_L will correctly not accept (reject) $\langle M, d, x \rangle$.
3. Let $\langle M, d, x \rangle \notin L$ such that M loops on d. Because M loops on d, we know R_L will also loop. However, this is okay because $\langle M, d, x \rangle \notin L$ so R_L is correctly not accepting (by looping) on $\langle M, d, x \rangle$.

Thus, given our construction and cases, we can see R_L is a recognizer of the language L.

Now, given our recognizer R_L for the language L, we will construct a new machine C^{ATM} (with oracle access) that can compute $C(x)$. Let C^{ATM} be defined as follows:

C^{ATM} **on input** $\langle x \rangle$**:**

Define M' as follows

M' **on input** $\langle x' \rangle$**:**

Output x and $ACCEPT$

Set $k = |\langle M', \lambda \rangle|$

For $\langle M_i, d_i \rangle$ in the sorted list (based on size of $|\langle M_i, d_i \rangle|$) of machine and input pairs

If $|\langle M_i, d_i \rangle| < k$, ask A_{TM} oracle $\langle R_L, \langle M_i, d_i, x \rangle \rangle$

If YES, $OUTPUT\ |\langle M_i, d_i \rangle|$

If NO, continue

If $|\langle M_i, d_i \rangle| \geq k$, $OUTPUT\ k$

Given our definition C^{ATM}, we claim C^{ATM} is computable because M' is a valid TM and the steps of C^{ATM} are all finite. First, we can see defining M' and setting k are finite. Next, because we were able to find a finite value for k, we can see that we will stop iterating over our sorted list of machine / input pairs after a finite number of times. Since each of our iterations only includes query our oracle (which is finite), comparing values, and outputting, we can see an iteration contains finite steps. Thus, we can see that C^{ATM} is made up of finite steps so we can see C^{ATM} is a computable function.

Given that we are able to define a computable C^{ATM} (which has oracle access) that computes $C(x) = \min |\langle M, d \rangle|$, we can see that $C(x)$ is computable by a machine that has oracle access to A_{TM}.