Homework #4: Computability
Due: see Canvas

Everyone should read Chapter 5, Section 6.3 (oracles), and Section 6.4 (Kolmogorv complexity). Students in CS524 should also read Section 6.1 (the Recursion Theorem). As usual, you may not need to solve all the problems below; see Canvas fo details.

Written Problems

Problem 1 Show a language L is recognizable if-and-only-if $L \leq_m A_{TM}$.

Problem 2 In Theorem 5.3, the book shows that REGULAR_{TM} is not decidable. (In that proof, they compute $\langle M_2 \rangle$ from $\langle M, w \rangle$. You can reuse that in one of the subproblems below. You might also look at the argument of Theorem 5.30.)

Problem 2(a). Show a reduction $\text{A}_{TM} \leq_m \text{REGULAR}_{TM}$.

Problem 2(b). Show a reduction $\overline{\text{A}_{TM}} \leq_m \text{REGULAR}_{TM}$ (equivalently, $\text{A}_{TM} \leq_m \overline{\text{REGULAR}_{TM}}$).

Problem 2(c). Review: which part ((a) or (b)?) implies REGULAR_{TM} is not recognizable, and which part implies it is not co-recognizable?

Problem 3 Given two languages $A, B \subseteq \Sigma^*$, recall that their "quotient" is $A/B = \{x \in \Sigma^* : \exists y \in B, xy \in A\}$. Find an example where A and B are decidable, but A/B is not. (Hint: use history strings.)

Problem 4 In Chapter 4, we saw that A_{TM} is recognizable but not decidable. We may assume it is encoded in binary, so $A_{TM} \subseteq \{0,1\}^*$. Using that, argue that some unary language $L \subseteq \{1\}^*$ is recognizable but not decidable. (Hint: first find a computable "one-to-one correspondence" from $\{1\}^*$ to $\{0,1\}^*$, sending each unary string 1^n to some binary string s_n, and then use that to define your language L.)

Problem 5 See Problem 5.21, and argue that AMBIG_{CFG} is not decidable. (The book gives an enormous hint, use it.) Furthermore, try to determine whether the language is recognizable or co-recognizable.

Problem 6 Exercise 6.4, showing A'_{TM} is undecidable relative to A_{TM}. (Hint: redo the argument in Chapter 4, with oracles in appropriate places.)

Remark: In the next two problems, $K(x)$ denotes the Kolmogorov complexity of binary string x, as defined in Section 6.4.

Problem 7 Suppose we have an oracle for A_{TM}. On input $x \in \{0,1\}^*$, I'll explain how to compute $k = K(x)$ using $O(2^k)$ oracle calls. Show how to do it with only $O(k)$ oracle calls. (Or even better, $O(\log k)$ oracle calls!)

Problem 8 Suppose $f(x)$ is a computable function, where its inputs x is a binary string, and its output is a non-negative integer. Suppose that $f(x) \leq K(x)$, for all $x \in \{0,1\}^*$. Argue there is a constant c such that $f(x) \leq c$, for all binary strings x. (Hint: otherwise, use f to find strings x with arbitrarily large $K(x)$.)

MPCP Problems

For the next two problems, see the program share/hw4/MPCP.py (or its online "notebook" version at `https://github.com/mgrigni/cs424s19`). It lets us define a language $L \subseteq \{a,b\}^*$, by setting `L_dominos`, a list of dominos (pairs of strings). For each of the following problems, find a value for `L_dominos` defining the requested L. You should test that your answer works, at least for short strings. For these problems you should submit a short python file defining `L_dominos`.

Problem 9 (ww.py) $L = \{ww : w \in \{a,b\}^*\} \subseteq \{a,b\}^*$.

Problem 10 (square.py) $L = \{a^n : n \text{ is a square}\} \subseteq \{a,b\}^*$.