CS 332: *Elements of Theory of Computation* — Professor Sofya Raskhodnikova
Boston University — November 7, 2019

Homework 8 – Due Friday, November 15, 2019 before the noon

This homework contains 3 mandatory problems, worth 10 points each, and an announcement of an extra credit programming assignment, worth 30 (homework) points.

Reminder Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the instructor if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

Exercises Please practice on exercises 6.1-6.2, 7.1-7.11 and the following exercise.

Problems

1. (**Applications of recursion theorem**)
   (a) Let $\text{SMALL}_{\mathsf{TM}} = \{\langle M\rangle \mid M$ is a TM and there is no TM M' equivalent to M which has a much shorter description, that is, $|\langle M'\rangle| \leq \frac{1}{2}|\langle M\rangle|\}$. Show that every infinite subset of $\text{SMALL}_{\mathsf{TM}}$ is not Turing-recognizable.

   (b) Define the *prolificacy* of a Turing machine M as follows: If M does not halt on input ε, then the prolificacy of M is 0; otherwise, the prolificacy of M is the number of nonblank characters left on the tape of M after M halts on input ε. Let $p(n)$ denote the maximum prolificacy among all Turing machines with n states. Use recursion theorem to prove that $p(n)$ is not computable.

2. (**Review of asymptotic notation**) This problem will be graded automatically by Gradescope. Please enter your answers manually by completing the assignment Homework 8-Problem 2. For each of the following, select *true* or *false* using the radio buttons on Gradescope.

   (a) $2^{10} = O(n)$
   (b) $16n = O(n)$
   (c) $n^4 = O(n^2 \log n)$
   (d) $n \log n + 10n = O(n^2)$
   (e) $3^n = O(2^n)$
   (f) $3^n = 2^{O(n)}$
   (g) $2^{2^n} = O(2^{2n})$
   (h) $n^n = O(n!)$
   (i) $n = o(n)$
   (j) $2n = o(n^2)$
   (k) $2^n = o(3^n)$
   (l) $1 = o(n)$
   (m) $2 \log n = o(\log n)$
   (n) $\frac{1}{3} = o(1)$
   (o) $\log_2 n = \Theta(\log_3 n)$
   (p) $2^n = \Theta(4^n)$
   (q) $n^5 = \Theta(32^{\log_2 n})$
   (r) $n^3 = \Omega(n^3)$
   (s) $\log n = \Omega(\log(\log n))$
   (t) $2^{5^n} = \Omega(5^{2^n})$

3. (**Exponentiation cipher**) An exponentiation cipher encodes a message A using a ciphertext $C = A^e \pmod p$ where p is a prime number and e is an integer exponent. (Here A and C are also integers.) You are given integers A, C, e and p, and you would like to determine whether C is a valid ciphertext for message A.

   (a) Formulate this problem as a language EC.

   (b) Explain why the following algorithm for EC does not run in polynomial time: *Compute* A^e *using* $e - 1$ *multiplications. Take the result modulo* p *using one integer division, and compare the answer to* C.

   (c) Show that $EC \in \mathrm{P}$. Analyze the running time of your algorithm using O-notation.
   Hint: First, find an algorithm for the case when e is a power of 2.

4. (**Extra credit programming assignment**) On Tuesday, we will publish an extra credit programming assignment on the recursion theorem on the course webpage. You wil have about 2 weeks to complete it.