Homework 9 – Due Tuesday, November 24th, 2020 before 2:00PM

Reminder Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the course staff 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 the following exercises and solved problems in Chapter 7: 7.1, 7.2, 7.4, 7.6, 7.8–7.11. The material they cover may appear on exams.

Problems There are 3 mandatory problems and one bonus problem.

1. (**Review of asymptotic notation**) This problem will be graded automatically by Gradescope. Please enter your answers manually by completing the assignment HW9P1. 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) $7^{\log n} = n^{O(1)}$

   (j) $2n = o(n^2)$

   (k) $2^n = o(3^n)$

   (l) $1 = o(n)$

   (m) $2 \log n = o(\log n)$

   (n) $3 = o(1)$

   (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) $2^{5^n} = \Omega(5^{2^n})$

2. (**Universal RAM TM**) Define a *RAM Turing machine* to be a Turing machine that has *random access memory*. We formalize this as follows: the machine has additional two symbols on its alphabet we denote by `R` and `W` and an additional state we denote by q_{access}. We also assume that the machine has an infinite array A that is initialized to all blanks. Whenever the machine enters q_{access}, if its address tape starts with $\langle i \rangle R$ then the value $A[i]$ is written in the cell next to the `R` symbol. If its tape contains $\langle i \rangle \texttt{W}\sigma$ where σ is some symbol in the machine's alphabet, then $A[i]$ is set to the value σ.

   Show that there exists some universal RAM TM $\mathcal{U}$ with constant overhead, that is, on input $\langle M, x \rangle$ where M is a RAM TM that takes T steps before either accepting or rejecting on input x, $\mathcal{U}$ accepts or rejects correspondingly in time $O(|M| + T \log |M|)$. (Justify its running time!)

3. (**Closure properties of P**) For both parts of this problem, analyze the running time of your algorithms using O-notation. Prove that P is closed under

   (a) concatenation;

   (b) star.

   Hint: Use dynamic programming. Let $A \in \mathrm{P}$. On input $y = y_1 \cdots y_n$ for $y_i \in \Sigma$, build a table indicating for each $i \leq j$ whether the substring $y_i \cdots y_j \in A^*$.

4* (**Optional, collaboration is *allowed***) Define a TM M to be *oblivious* if its head movement does not depend on the input but only on the input length. That is, M is oblivious if for every input $x \in \Sigma^*$ and $i \in \mathbb{N}$, the location of each of M's heads at the i^{th} step fo execution on input x is only a function of $|x|$ and i (in particular, it is oblivious to the content of x, or the head moving pattern "perfectly hides" the content of x). Show that if a language L is decided by a Turing machine whose running time is *exactly* $T(n) \geq n$ for every $|x| = n$, there is an oblivious TM that decides L in time $O(T(n)^2)$. Furthermore, show that this holds even if the language is decided by a *multi-tape* TM with running time $T(n) \geq n$.

   Think but not turn in: Could you prove the same conclusion if the running time is not fixed, but is known to be upper bounded by a time-constructible function $T(n) : \mathbb{N} \to \mathbb{N}$ instead? A function T is time-constructible if you can compute it in unary alphabet (i.e. mapping 1^n to $1^{T(n)}$) on a TM in time $O(T(n))$. This definition includes all "reasonable" functions like $n^2, 2^n, n^{n^n}$.