Qualifying Examination

Theoretical Computer Science

Tuesday, March 3, 2020

Part II: Automata and Complexity

Name	

Problem	Maximum Points	Points Earned	Grader
1	25		
2	25		
3	25		
4	25		
Total	100		

Instructions:

1. This is a closed book exam.
2. The exam is from 9am–4.30pm and has four problems of 25 points each. Read all the problems carefully to see the order in which you want to tackle them.
3. Write clearly and concisely. You may appeal to some standard algorithms/facts from text books unless the problem explicitly asks for a proof of that fact or the details of that algorithm.
4. If you cannot solve a problem, to get partial credit write down your main idea/approach in a clear and concise way. For example you can obtain a solution assuming a clearly stated lemma that you believe should be true but cannot prove during the exam. However, please do not write down a laundry list of half baked ideas just to get partial credit.

May the force be with you.

Problem 1:

1. (80 points) Given a language L over a finite alphabet Σ, define

$$\textsc{once}(L) = \Big\{a_1 \cdots a_n : \begin{array}{l} \text{there is exactly one index pair } (i,j) \text{ with } 1 \le i \le j \le n \\ \text{such that the substring } a_i \cdots a_j \text{ is in } L \quad (a_1, \ldots, a_n \in \Sigma)\end{array}\Big\}.$$

   (Informally, a string is in $\textsc{once}(L)$ iff it has exactly one substring that lies in L, however the precise definition is more subtle.)

   Prove that if L is regular, then $\textsc{once}(L)$ is regular.

2. (20 points) Consider the following alternative definitions:

$$\begin{aligned} \textsc{once}'(L) &= \Big\{x \in \Sigma^* : \begin{array}{l} \text{there is exactly one string } y \in L \\ \text{such that } y \text{ is a substring of } x\end{array}\Big\} \\ \textsc{once}''(L) &= \Big\{x \in \Sigma^* : \begin{array}{l} \text{there exists a string } y \in L \\ \text{such that } y \text{ occurs exactly once in } x\end{array}\Big\}. \end{aligned}$$

   Is it also true that if L is regular, then $\textsc{once}'(L)$ must be regular? And is it true that if L is regular, then $\textsc{once}''(L)$ must be regular? Explain.

Problem 2:

1. Prove that $\mathsf{coNEXP} \subseteq \mathsf{NEXP}/(n+1)$. *Hint:* Recall $\mathsf{NL} = \mathsf{coNL}$.

2. Given the above, prove that $\mathsf{NEXP}/\mathsf{poly} = \mathsf{coNEXP}/\mathsf{poly}$.

Problem 3: Define a ZPP-machine to be a probabilistic Turing machine that is permitted three types of output on each of its branches: *accept*, *reject*, and '*?*'. A ZPP-machine M decides a language A if M outputs the correct answer on every input string x (*accept* if $x \in A$ and *reject* if $x \notin A$) with probability at least $2/3$, and M never outputs the wrong answer. On every input, M may output '*?*' with probability at most $1/3$. Furthermore, the average running time over all branches of M on x must be bounded by a polynomial in the length of x. Show that $\mathsf{RP} \cap \mathsf{coRP} = \mathsf{ZPP}$, where ZPP is the collection of languages that are recognized by ZPP-machines.

Problem 4: Let A be a regular language over $\{0,1\}$. Show that A has circuits (over AND, OR, and NOT gates of fan-in 2) where the circuit C_n for A over $\{0,1\}^n$ is simultaneously $O(n)$-size and $O(\log n)$-depth.