cs579: Computational Complexity	Assigned: *Thu., Feb. 14, 2019*

Problem Set #3

Prof. Michael A. Forbes	Due: *Thu., Feb. 28, 2019 (3:30pm)*

1. (Sipser #8.17) Let A be the language of properly nested parentheses. For example, (()) and (()(()))() in A, but)(is not. Show that A is in L.

2. (Sipser #9.19) Define the unique-sat problem to be $\mathsf{USAT} = \{\langle\varphi\rangle : \varphi$ is a boolean formula that has a unique satisfying assignment$\}$. Show that $\mathsf{USAT} \in \mathsf{P}^{\mathsf{SAT}}$.

3. (Arora-Barak Problem 6.3) Describe a decidable language in $\mathsf{P/poly}$ that is not in P.

4. (Sipser #9.13, #9.14): Consider the function $\text{pad} : \Sigma^* \times \mathbb{N} \to \Sigma^*\#^*$ that is defined as follows. Let $\text{pad}(s, \ell) = s\#^j$, where $j = \max(0, \ell - m)$ and m is the length of s. Thus $\text{pad}(s, \ell)$ simply adds enough copies of the new symbol # to the end of s so that the length of the result is at least ℓ. For any language A and function $f : \mathbb{N} \to \mathbb{N}$ define the language $\text{pad}(A, f(m))$ as

$$\text{pad}(A, f(m)) = \{\text{pad}(s, f(m)) : s \in A, m = |s|\}$$

   (a) Prove that, if $A \in \mathsf{TIME}(n^6)$, then $\text{pad}(A, n^2) \in \mathsf{TIME}(n^3)$.

   (b) Recall that $\mathsf{EXP} = \mathsf{TIME}(2^{\mathsf{poly}(n)})$, and define $\mathsf{NEXP} = \mathsf{NTIME}(2^{\mathsf{poly}(n)})$. Prove that, if $\mathsf{EXP} \neq \mathsf{NEXP}$, then $\mathsf{P} \neq \mathsf{NP}$.

5. (Sipser #9.16) Prove that $\mathsf{TQBF} \notin \mathsf{SPACE}(n^{1/3})$.

6. (Sipser #9.24, #9.25)

   (a) Define the function $\text{majority}_n : \{0,1\}^n \to \{0,1\}$ as $\text{majority}_n(x_1, \ldots, x_n) = 1$ iff $\sum_i x_i \geq n/2$. Thus the majority_n function returns the majority vote of the inputs. Directly show that the majority_n function can be computed by size $O(n^2)$ size circuits.

   (b) Recall that you may consider circuits that output strings over $\{0,1\}$ by designating several output gates. Let $\text{add}_n : \{0,1\}^{2n} \to \{0,1\}^{n+1}$ take the sum of two n-bit binary integers and produce the $n+1$ bit result. Show that you can compute the add_n function with $O(n)$ size circuits.

   (c) By recursively dividing the number of inputs in half, and using part (b), show that the majority_n function can be computed by size $O(n \log n)$ size circuits.