CSC4010: Introduction to Theoretical Computer Science
Problem Set 6

Recap

- A language H is *NP-hard* if $L \leq_{\mathrm{P}} H$ for every $L \in \mathsf{NP}$. It is *NP-complete* if it is both NP-hard and in NP.
- Boolean circuits use AND and OR gates of unrestricted fan-in and NOT gates. The *size* counts only AND and OR gates; the *depth* is the maximum number of AND and OR gates on an input-to-output path.
- A *Boolean formula* is a Boolean circuit in which each gate output is used as an input to at most one gate. The *size* counts the leaves in its tree representation.
- A language L is in PSIZE if it can be computed by a family of polynomial-size Boolean circuits.

1. Recall that
$$A_{\mathrm{TM}} = \big\{\langle M, w\rangle : M \text{ is a Turing machine that accepts } w\big\}.$$
Determine whether A_{TM} is NP-complete and whether it is NP-hard. Justify your answers.

2. For each integer $n \geq 1$, define $\mathrm{EXACTONE}_n : \{0,1\}^n \to \{0,1\}$ by
$$\mathrm{EXACTONE}_n(x_1, \ldots, x_n) = 1 \quad \Longleftrightarrow \quad \sum_{i=1}^{n} x_i = 1.$$
   (a) Construct a Boolean circuit computing $\mathrm{EXACTONE}_n$ and briefly explain why it is correct.
   (b) Give asymptotic bounds, as functions of n, on the size and depth of your circuit and on the size of the Boolean formula obtained by unfolding your circuit.

3. (Shannon's Theorem) Show that for every $n \geq 4$, there exists a boolean function $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by any Boolean circuit of size at most $\frac{2^{n/2}}{4}$.
   Hint: Use the counting argument alluded to in the lecture. That is, compare the number of Boolean circuits of size at most $2^{n/2}/4$ with the number of boolean functions on n variables.

4. (a) Describe an algorithm that, given an integer $n \geq 12$, outputs the truth table of an n-input Boolean function f_n that cannot be computed by a Boolean circuit of size at most $2^{\lfloor n/3 \rfloor}$.
   Hint: Consider a search over truth tables and Boolean circuits. You may use the counting theorem from Question 3.
   (b) Using the functions f_n produced by your algorithm in part (a), define
$$L = \big\{x \in \{0,1\}^* : |x| \geq 12 \text{ and } f_{|x|}(x) = 1\big\}.$$
   Show that L is decidable and that $L \notin \mathsf{PSIZE}$.
   (c) Use part (b) to prove that $A_{\mathrm{TM}} \notin \mathsf{PSIZE}$.
   Hint: Fix a decider D for L and consider inputs of the form $\langle D, x\rangle$.

5. (Formula Bounds for Parity) Consider the parity function on n bits: $\oplus_n : \{0,1\}^n \to \{0,1\}$ defined as $\oplus(x_1, \cdots, x_n) = \sum_{i=1}^{n} x_i \mod 2$.
   (a) Give an upper bound on the complexity of the boolean formula that computes $\oplus_n$? (Recall that the complexity measure for a boolean formula is the number of leaves it has. Give your bound in asymptotic notation, as function of n.)

(b) In this part, we will see a toy version of Subbotovskaya's proof of a superlinear lower bound on the formula complexity of the $\oplus_n$. For the purpose of this question, it will be useful to note the following identities. When an input variable to a gate is fixed, we can simplify a formula using identities: $0 \wedge G = 0; 1 \wedge G = G; 0 \vee G = G; 1 \vee G = 1$. **For parts (iii) onwards, let $n \geq 2$, and let F be a minimum-leaf-size formula computing $\oplus_n$.**

(i) Show that $L(\neg f) = L(f)$ holds for every Boolean function f.

(ii) Show that if any one input to $\oplus_n$ is fixed (to either 0 or 1), then the resulting function is $\oplus_{n-1}$ or $\neg\oplus_{n-1}$.

(iii) Consider F and let G be any gate which has as input a literal x_j or $\neg x_j$. Show that the subtree rooted at G contains a literal other than x_j or $\neg x_j$ as a leaf.

(iv) Show that there exists an index i, so that at least $L(F)/n$ leaves in F are labeled by x_i or $\neg x_i$.

(v) Using the above parts, show that $L(\oplus_{n-1}) \leq L(\oplus_n)(1 - \frac{1}{n}) - 1$.

(vi) Show that $L(\oplus_n) \geq n(1 + \sum_{j=1}^{n-1} \frac{1}{j}) = \Omega(n \log n)$.