CS 535: Complexity Theory, Fall 2020

Homework 6

Due: 8:00PM, Friday, October 23, 2020.

Reminder. Homework must be typeset with LaTeX preferred. Make sure you understand the course collaboration and honesty policy before beginning this assignment. Collaboration is permitted, but you must write the solutions *by yourself without assistance*. You must also identify your collaborators. Assignments missing a collaboration statement will not be accepted. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

Problem 0 (Term Paper)**.** Now would be a good time to start thinking about the term paper assignment: what topic/paper you might be interested in writing about, and who you might want to work with. Instructions for the term paper are here: `https://cs-people.bu.edu/mbun/courses/535_F20/handouts/term_paper.pdf` and a list of suggested topics is here: `https://piazza.com/class/keda2wyieyz10e?cid=277`.

Problem 1 (Exponential-Size Circuits for Every Function)**.** In this problem, you will prove that every Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be computed by a circuit of size $O(2^n/n)$. As we'll see, this is essentially tight: in fact, "most" functions require circuits of size $\Omega(2^n/n)$.

(a) First show that every Boolean function $f(x_1,\ldots,x_n)$ can be written in the form $(\overline{x_n} \wedge f_0(x_1,\ldots,x_{n-1})) \vee (x_n \wedge f_1(x_1,\ldots,x_{n-1}))$ for some functions $f_0, f_1 : \{0,1\}^{n-1} \to \{0,1\}$. (2 points)

(b) Use part (a) recursively to show that every function $f : \{0,1\}^k \to \{0,1\}$ is computed by a circuit of size $O(2^k)$. (2 points)

(c) There are exactly 2^{2^k} different functions $f : \{0,1\}^k \to \{0,1\}$. Combine this fact with part (b) and another recursive application of part (a) to show that every function $f : \{0,1\}^n \to \{0,1\}$ for $n \geq k$ can be computed a circuit of size $O(2^{n-k}) + O(2^k \cdot 2^{2^k})$. Hint: You can assume each gate has unbounded fan-out, so you can "reuse" the output of a subcircuit as many times as you want. (4 points)

(d) Set k appropriately in part (c) to conclude that every Boolean function $f : \{0,1\}^n \to \{0,1\}$ is computed by a circuit of size $O(2^n/n)$. (2 points)

Problem 2 (More Time-Space Tradeoffs)**.** In class (and in Arora-Barak) we saw that $\mathbf{NTIME}(n) \not\subseteq \mathbf{TISP}(n^{1.2}, n^{0.2})$, and hence SAT cannot be solved by a deterministic TM running in, say, time $O(n^{1.1})$ and space $O(n^{0.1})$ simultaneously. In this problem, you'll see how far you can push the technique to get different tradeoffs. Assume every function you encounter is time- and space-constructible.

(a) Generalize Claim 5.11.1 in Arora-Barak to prove that for $T(n) \geq n^2$ and $S(n) \geq \log n$, we have $\mathbf{TISP}(T, S) \subseteq \mathbf{\Sigma_2 TIME}(\sqrt{TS})$. (3 points)

(b) Generalize Claim 5.11.2 in Arora-Barak to prove that if $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^c)$ for some $c > 1$, then $\mathbf{\Sigma_2 TIME}(f(n)) \subseteq \mathbf{NTIME}((f(n))^c)$. (3 points)

(c) First we'll see how large we can make the time requirement. Use parts (a) and (b) to prove that for every $c < \sqrt{2}$, there exists a $\delta > 0$ such that $\mathbf{NTIME}(n) \not\subseteq \mathbf{TISP}(n^c, n^\delta)$. You don't have to show it, but this implies that SAT cannot be solved by an algorithm using $O(n^{1.41\dots})$ time and $n^{o(1)}$ space. Hint: Note that δ is allowed to depend on c. You'll want to choose δ small enough so that $c(c + \delta) < 2$. (2 points)

(d) Now we'll see how far we can push the space requirement. Prove that for every $c < 1$, there exists a $\delta > 0$ such that $\mathbf{NTIME}(n) \not\subseteq \mathbf{TISP}(n^{1+\delta}, n^c)$. This result implies that SAT cannot be solved by an algorithm using $n^{1+o(1)}$ time and $O(n^{0.999})$ space. Hint: This time, choose δ small enough so that $(c + 1 + \delta)(1 + \delta) < 2$. (2 points)

Problem 3 (*Bonus* Improved Time-Space Tradeoffs)**.** Now let's see how we can get even better tradeoffs by repeatedly trading alternations for time. Note that by combining Problems 2(a) and (b), we get the statement: If $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^c)$ for some $c > 1$, then $\mathbf{TISP}(T, S) \subseteq \mathbf{NTIME}((TS)^{c/2})$.

(a) Suppose $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^c)$ for some $c > 1$. Use the above statement to show that $\mathbf{TISP}(T, S) \subseteq \mathbf{coNTIME}((TS^2)^{c^2/(2+c)})$. Hint: Let C_0, C_f be the start and accept configurations of a deterministic TM running in time T. Then C_f is reachable from C_0 in T time steps iff *for all* $C' \neq C_f$, we have that C' is *not* reachable from C_0 in T time steps.

(b) Conclude that $\mathbf{NTIME}(n) \not\subseteq \mathbf{TISP}(n^c, n^{o(1)})$ whenever $c^3 < 2 + c$, i.e., $c < 1.521\dots$.

(c) Generalize the above argument inductively to show that $\mathbf{NTIME}(n) \not\subseteq \mathbf{TISP}(n^c, n^{o(1)})$ whenever $c(c - 1) < 1$, i.e., $c < \phi = 1.618\dots$.