Week 5: It's Complicated

Collaboration: You should work on the problems yourself, before discussing with others, including your cohorts at your cohort meeting. By the Assessed Cohort Meeting, you and all of your cohortmates, should be prepared to present and discuss solutions to all of the assigned problems. In addition to discussing with your cohortmates, you may discuss the problems with any other current CS3102 students you want, and use any resources you want except for any materials from previous offerings of this course or complete solutions that might be available on the web, which are not permitted. **You may not collaborate on the assigned writeup with your cohort-mates or anyone else, but you may use notes taken before or during your assessed cohort meeting.**

You writeups for this week will work a little differently from previous weeks. At the end of your assessed cohort meeting, your Cohort Coach will assign a *new* problem for the writeup. Since we will have a final exam in the course (and a midterm-like activity during the week of fall break) we wanted to give you experience solving problems on your own so that you can check your progress. As such, this week's writeup will not be taken directly from this week's problem set, and instead will be a new problem in the same style as Problem 4.

Problem 1: Cohort Communication

Please discuss any questions or concerns that came up as a result of your peer evals. If you have something you would like to discuss with your cohort, but are not sure how to mention it, Professor Brunelle or Professor Evans would be happy to help you make a plan. Just show up to office hours, or send a DM to schedule a time to chat.

Problem 2: Complexity by Circuit Depth

In the *Complexity Classes: SIZE* video we defined the complexity $SIZE(s)$ to be the set of all functions which can be implemented as a NAND circuit containing s or fewer gates. We defined $SIZE^{AON}(s)$ to be the set of all functions which can be implemented as an AON circuit containing s or fewer gates.

Once a circuit has been implemented in hardware, all gates at the same level will evaluate in parallel with one another, but each gate must "wait" on gates at shallower levels before it can be evaluated. The *depth* of a circuit is defined as the maximum number of gates along a path from an input to output. For this reason, circuit depth is actually a better metric for estimating running time than circuit size would be.

For this problem we will measure complexity by circuit depth rather than by number of gates. These two definitions define sets of complexity classes based on circuit depth:

Definition 1 (*$DEPTH^{NAND}$*) *A function $f : \{0,1\}^n \to \{0,1\}^m$ belongs to class $DEPTH^{NAND}(d)$ if it can be implemented as a NAND circuit with depth d or less.*

Definition 2 (*$DEPTH^{AON}$*) *A function $f : \{0,1\}^n \to \{0,1\}^m$ belongs to class $DEPTH(d)$ if it can be implemented as an AON circuit with depth d or less.*

Answer the following using these complexity classes:

(a) What is the smallest natural number d for which `OR` $: \{0,1\}^2 \to \{0,1\}$ will be in class $DEPTH^{NAND}(d)$

(b) What is the smallest natural number d for which `AND` $: \{0,1\}^2 \to \{0,1\}$ will be in class $DEPTH^{NAND}(d)$

(c) What is the smallest natural number d for which `NOT` $: \{0,1\} \to \{0,1\}$ will be in class $DEPTH^{NAND}(d)$

(d) What is the smallest natural number d for which `NAND` $: \{0,1\}^2 \to \{0,1\}$ will be in class $DEPTH^{AON}(d)$

(e) In the *First Complexity Proof lecture* we showed $SIZE(\frac{s}{2}) \subseteq SIZE^{AON}(s) \subseteq SIZE(3s)$. Use your answers above to perform a similar argument for *DEPTH* by identifying functions f and g that will allow you to show that $DEPTH^{NAND}(f(d)) \subseteq DEPTH^{AON}(d) \subseteq DEPTH^{NAND}(g(d))$.

Problem 3: Implementations are not unique

Show that for any function $f : \{0,1\}^n \to \{0,1\}$ there are an infinite number of NAND circuits which implement that function.

With this proof in mind, explain why we define the complexity of a function in terms of its most efficient implementation.

Problem 4: Asymptotic Operators

For each sub-problem, indicate if the statement is *true* or *false* and support your answer with a convincing argument.

(a) $17n \in O(723n + \log n)$

(b) $\min(n^n, 3012) \in O(1)$

(c) $n^2 \in \Theta(n^3)$

(d) $2.0001^n \in O(2^n)$

(e) Half a bonus point: $\log_n 10 \in \Theta(\log_{2n} 17)$

Problem 5: Constant "time"

Show whether $O(1) = \Theta(1)$.

It is left up to you to determine if the two sets are equivalent (the proposition may be either true or false), and provide a convincing proof to support your answer.

Problem 6: little-o

Another useful notation is "little-o" which is designed to capture the notion that a function g grows much faster than f:

Definition 3 (o) *A function $f(n) : \mathbb{N} \to \mathbb{R}$ is in $o(g(n))$ for any function $g(n) : \mathbb{N} \to \mathbb{R}$ if and only if for every positive constant c, there exists an $n_0 \in \mathbb{N}$ such that:*

$$\forall n > n_0 . f(n) < cg(n).$$

In other words, $f(n) \in o(g(n))$ provided that no matter what positive constant is chosen for c, eventually there comes a point were $c \cdot g(n) > f(n)$ forever more.

Provide a proof for each of the following sub-problems.

(a) Prove that for any function f, $f \notin o(f)$.

(b) Prove that $n \in o(n \log n)$.