PS5: Eval

Collaboration: You should work on the problems yourself, before discussing with others, including your cohorts at your cohort meeting. 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. Sharing and subsequently submitting any text, code, images, figures, etc. constitutes plagiarism, so make sure all submitted materials are created exclusively by members of your cohort.

Problem 1: EVAL in Python

Download the *eval.py* program. Follow the instructions in the comments, being sure to implement every function marked with "TODO". This program walks you though the implementation of EVAL in Python.

By the time you're finished with this, you'll have built a Python interpreter for the NAND-straightline programming language and have an interpreter powerful enough to compute any finite function!

Problem 2: 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 3: 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 4: Asymptotic Operators

For each sub-problem, indicate if the statement is *true* or *false* and support your answer with a convincing argument that uses the formal definitions of the various asymptotic operators from class.

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

(b) $17n \in \Omega(723n + \log n)$

(c) $\min(n^n, 3102) \in O(1)$

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

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