Exam 2 Practice Problems

These problems are provided to help you prepare for Exam 2.

Exam 2 will be **Wednesday, 6 November** at the normal class time in the normal classroom. We will aim to start the exam right at 3:30pm, so you will benefit from arriving early for class Wednesday to be settled and ready to start the exam.

Exam 2 will cover material from Classes 1-16 (through 28 October), Problem Sets 1-6, and the textbook chapters 0-6 and parts of 8 (8.1-8.4). Most of the questions on the exam will cover topics that you will have seen in at least two different places (e.g., both in class and on a problem set, or in multiple classes, or at least both in the textbook and in class or a problem set). The exam will emphasize material that was not covered by Exam 1 (so focusing on the material on problem sets 4-6), but will also build on material earlier in the class, and you should be surprised if it does not include induction proofs. These practice problems are not necessarily comprehensive, but should give you a good idea of the kinds of problems you will see on the exam.

As a reminder from the syllabus, you may construct a one-page (letter-size, two-sided) reference sheet for use during the exam, but all other resources are forbidden (no internet, textbook, other humans, magnification instruments, etc.). We expect that students will benefit from thinking about what to put on your reference sheet in preparing for the exam, and you may work with anyone you want (including other students in the class) to prepare a reference sheets together.

Asymptotic Operators

Problem 1 *Asymptotic Operator Definitions*

For each pair of functions, f and g, indicate which of these are true: $f \in O(g)$, $f \in \Omega(g)$, $f \in \Theta(g)$, $f \in \widetilde{O}(g)$, $f \in o(g)$ (see Problem Set 4 for the definitions of $\widetilde{O}$ and o). You should be able to support your answer with a clear argument, using the definition of the asymptotic operator.

(a) $f(n) := n; g(n) := 2n$.

(b) $f(n) := n^2; g(n) := n \log n$.

(c) $f(n) := \sin(n); g(n) := \cos(n)$.

(d) $f(n) := 2^n; g(n) := n^n$.

(e) $f(n) :=$ number of cs3102 students who obtain a score $\geq n$ on Exam 2; $g(n) := 77$.

Problem 2 *Properties of Asymptotic Operators*

(a) Prove that for any function f, $f \in O(f)$.

(b) Prove that for any functions f_1, f_2, and g, if $f_1 \in \Theta(g)$ and $f_2 \in \Theta(g)$, then $f_+ \in \Theta(g)$ where $f_+(n) := f_1(n) + f_2(n)$.

Counting Functions

Problem 3 *Hard Functions are "Randomish"*

Prove that if $F : \{0,1\}^n \longrightarrow \{0,1\}$ is a function that cannot be computed by a NAND-CIRC program with fewer than $2^n/1000n$ lines, the number of inputs for which $F(x) = 0$ must be between $2^n/100000n$ and $2^n \cdot 99999/100000n$. (Hint: how hard would it be to implement a function that outputs 0 for all inputs? For all but one inputs? etc.)

Problem 4 *Size hierarchy theorem for multibit functions (TCS Exercise 5.6)*

Use the ideas of Remark 5.4 to show that for every $\epsilon > 0$ and sufficiently large s, n, m,

$$|SIZE_{n,m}(s)| < 2^{(2+\epsilon)s\log s + n\log n + m\log s}.$$

Conclude that the implicit constant in Theorem 5.2 can be made arbitrarily close to 5. (Hint: Using the adjacency list representation, a graph with n in-degree zero vertices and s in-degree two vertices can be represented using roughly $2s\log(s+n) \leq 2s(\log s + O(1))$ bits. The labeling of the n input and m output vertices can be specified by a list of n labels in $[n]$ and m labels in $[m]$.)

Problem 5 *Tighter counting lower bound (TCS Exercise 5.7)*

Prove that for every $\delta < 1/2$, if n is sufficiently large then there exists a function $f : 0,1^n \to 0,1$ such that $f \notin SIZE_{n,1}\left(\frac{\delta 2^n}{n}\right)$. (Hint: Use the results of the previous problem and the fact that in this regime $m = 1$ and $n \ll s$.)

Size Hierarchy

Recall that $SIZE_n(s)$ is the set of all n-input Boolean functions that can be implemented with s or fewer $NAND$ gates.

Problem 6 *Prove that $SIZE_3(1) \subsetneq SIZE_3(2)$.*

Problem 7 *Prove that $SIZE_3(3102) = SIZE_3(4102)$.*

Problem 8 *($\star$) What is the largest k such that $SIZE_k(3102) = SIZE_k(4102)$?*

Turing Machines and Computability

Problem 9 *Naturals are Computable*

Prove that every natural number is computable, by showing how to define a Turing Machine, M_k that on a blank input tape outputs a representation of the natural number k for any $k \in \mathbb{N}$. (You can decide how a natural number should be represented, but should be able to precisely describe M_k.)

Problem 10 *Increment Machine*

Describe a Turing Machine that takes as input a tape containing the binary representation of a natural number, n, and outputs a tape containing the binary representation of $n + 1$. You machine should work for any input, $n \in \mathbb{N}$, and you should be able to explain how many steps your machine will take in the worst case for an input of length k bits.

Problem 11 *Fading Tape Machine*

Consider a variation of a Turing Machine where instead of writing symbols on a tape (where it is assumed the same symbol can always be read back), we have an infinite beach instead of an infinite tape. Each step the machine takes, symbols that were written on the tape fade away as the sand is blown by the wind. To keep things simple, assume the input symbols are written in a way that does not fade (so the machine can still process any length input), but all symbols written by the machine are perfectly readable for 100 steps, but are no longer readable after the machine has taken 100 steps since it was written.

How powerful is a fading tape machine?

Problem 12 *Input Reversing*

Prove that if $F : \{0,1\}^* \to 0,1$ is computable, then the function which takes the input in reverse direction, $F_{REV}\{0,1\}^* \to 0,1$ is computable. $F_{REV}(x) = F(reverse(x))$ where $reverse(x_1x_2 \ldots x_n) = x_nx_{n-1} \ldots x_2x_1$.

Problem 13 *Oblivious Turing Machines (challenging) (TCS Exercise 6.8)*

Define a Turing Machine M to be oblivious if its head movement are independent of its input. That is, we say that M is oblivious if there exists an infinite sequence $MOVE \in \{\mathsf{L}, \mathsf{R}, \mathsf{S}\}^\infty$ such that for every $x \in \{0,1\}^*$, the movements of M when given input x (up until the point it halts, if such point exists) are given by $MOVE_0, MOVE_1, MOVE_2, \ldots$.

Prove that for every function $F : 0,1^* \to 0,1^*$, if F is computable then it is computable by an oblivious Turing machine.

(Hint: R, L,R, R, L, L, R, R ,R, L, L, L, ...)

Problem 14 *Bounding Busy Beaver*

Recall our definition of BB from Problem Set 5:

Definition 1 (Busy Beaver Problem) *For any $n \in \mathbb{N}$, define $BB_2(n)$ as the maximum number of steps for which a Turing Machine with n states and 2 symbols can execute and halt, starting from a blank tape.*

Consider instead, this definition:

Definition 2 (Busy Beaver Bound) *For any $n \in \mathbb{N}$, define $BEYONDB_2(n)$ as the index of a tape cell that cannot be written to by any Turing Machine with n states and 2 symbols that eventually halts, starting from a blank tape.*

So, we can think of $BEYONDB_2(n)$ as an upper bound on the furthest tape cell that can be written by the machine. Since it is an upper bound, it would be correct to output any value larger than the furthest cell that can be written to by any halting Turing machine, thus there is an infinite set of functions that satisfy the $BEYONDB_2$ definition.

(a) Prove that there is some $z \in \mathbb{N}$ which is a correct value of $BEYONDB_2(n)$ for any $n \in \mathbb{N}$.

(b) Prove that computing any $BEYONDB_2$ function is uncomputable.