CSE596, Fall 2022 Problem Set 5 (short) Due Thu. Nov. 10

Lectures: *For two more weeks*, lectures will be:

- **Tuesdays 8–9pm online** via Zoom.
- **Thursdays 4–5pm in 201A Capen**. (This is in-person.)
- **Fridays 4–5pm** as normal in O'Brien 112.

The Monday before Thanksgiving (Nov. 21) will also be moved online, not sure if in the same time period. The **second prelim exam** will be after Thanksgiving, on **Wed. Nov. 30**.

Reading: After covering NP-completeness, we do not yet move on to section 4.5 about completeness for space-bounded classes. Instead, we go back to the ALR Chapter 27 "Complexity Classes" chapter, to cover the complexity class relationships in sections 2.4–2.6 of chapter 27. The DSPACE and DTIME Hierarchy Theorems are especially difficult. Ignore the nondeterministic version in ALR. **Do** read the Week 11 lecture notes on the course webpage ahead of time, especially the Monday and Wednesday ones. Debray's notes are now secondary—note that sections 10–12 were skipped over. Now section 13 parallels next week's reading, while sections 14–16 may be useful for reviewing NP-completeness with some different examples. Note that the very last example in Debray's section 16 is Factoring being in NP∩co-NP, which I put early-on. That different people do things different ways is just a fact of life at this level.

————————*Assgt. 5, due Thu. 11/10 "midnight stretchy" on CSE Autograder*————————-

(1) **This problem is carried over from Assignment 4**. Given a language L, define $L^R = \{x^R : x \in L\}$, where x^R means reversing the string x. Note that if every string in L is a palindrome, then $L = L^R$ automatically, but it is possible to have $L = L^R$ without every string in it being a palindrome, e.g., $L = \{01, 10\}$. Prove that the language of the following decision problem is neither c.e. nor co-c.e.

INSTANCE: A deterministic Turing machine M.

QUESTION: Is $L(M) = L(M)^R$?

Hint: This problem has a high "re-usability" quotient—it is OK to almost violate the University policy against re-submitting previously-used materials, provided you explain how and why the reduction(s) apply to this case. (**New Rule**: Your answer **must** include at least one reduction diagram of the kind exhibited in lectures for reductions from languages like A_{TM} and/or D_{TM}. But as already said, you may find ones from notes "re-usable." 24 pts.)

(2) (12 pts.)

Define a function f such that, for any Boolean formula $\phi(x_1, \ldots, x_n)$ in the "loose" definition of 3CNF that allows clauses of only 1 or 2 literals—as was used in the Cook-Levin proof—$f(\phi)$ is a formula ϕ' in strict 3CNF so that ϕ' is satisfiable if and only if ϕ is satisfiable. Your

formula ϕ' may—indeed must—have more than n variables. (*Hint*: Make a formula ϕ_0 in strict 3CNF that can only be satisfied by making a particular variable z in ϕ_0 **false**. Then use z as "filler" in ϕ to make your whole ϕ'.)

(3) (36 pts. total, for 72 on the set)

Prove that the following decision problem is NP-complete.

Two-Coloring With Faults

Instance: An undirected graph G, an integer $k \geq 1$.

Question: Can G be colored with 2 colors so that at most k edges have both nodes of the same color?

Note: You must use 3SAT for your reduction but you may use any variant of 3SAT that was covered in the course, including the "1-2-3" hybrid used in the Cook-Levin proof given in lecture, or forms such as "not-all-equal 3SAT." But you are not allowed to use "any" problem from an Internet or otherwise unofficial source—and it is strongly recommended that you use the "ladder/clause-gadget" architecture covered in the course for reductions from (3)SAT.