University of Nevada, Las Vegas Computer Science 456/656 Fall 2023

Practice Problems for the Examination on October 25, 2023

1. Review answers to homework3:
   `http://web.cs.unlv.edu/larmore/Courses/CSC456/S23/Assignments/hw3ans.pdf`

2. Review answers to homework4:
   `http://web.cs.unlv.edu/larmore/Courses/CSC456/S23/Assignments/hw4ans.pdf`

3. Review answers to homework5:
   `http://web.cs.unlv.edu/larmore/Courses/CSC456/S23/Assignments/hw5ans.pdf`

4. True or False. If the question is currently open, write "O" or "Open."

   (i) **O** $\mathcal{P} = \mathcal{NP}$.

   (ii) **O** $\mathcal{P} = \mathcal{NC}$.

   (iii) **T** Every regular language is $\mathcal{NC}$.

   (iv) **T** Every context-free language is $\mathcal{NC}$.

   (v) **O** The Boolean circuit problem is $\mathcal{NC}$

   (vi) **T** The complement of any $\mathcal{P}$-TIME language is $\mathcal{P}$-TIME.

   (vii) **O** The complement of any $\mathcal{NP}$ language is $\mathcal{NP}$.

   (viii) **T** The complement of any $\mathcal{P}$-SPACE language is $\mathcal{P}$-SPACE.

   (ix) **T** The complement of every recursive language is recursive.

   (x) **F** The complement of every recursively enumerable language is recursively enumerable.

   (xi) **T** If p is the pumping length of a regular language L, then $p + 1$ is also the pumping length of L.

   (xii) **T** If a language L is accepted by an NFA with p states, then p is the pumping length of L.

   (xiii) **T** Every language which is generated by a general grammar is recursively enumerable.

   (xiv) **F** The context-free membership problem is undecidable.

   (xv) **F** Given any context-free grammar G and any string $w \in L(G)$, there is always a unique leftmost derivation of w using G.

   (xvi) **T** For any non-deterministic finite automaton, there is always a unique minimal deterministic finite automaton equivalent to it.

   (xvii) **T** The union of any two context-free languages is context-free.

   (xviii) **F** The question of whether a given Turing Machine halts with empty input is decidable.

   (xix) **T** The class of languages accepted by non-deterministic finite automata is the same as the class of languages accepted by deterministic finite automata.

(xx) **F** The class of languages accepted by non-deterministic push-down automata is the same as the class of languages accepted by deterministic push-down automata.

(xxi) **T** Let π be the ratio of the circumference of a circle to its diameter. The problem of whether the n^{th} digit of the decimal expansion of π for a given n is equal to a given digit is decidable.

(xxii) **T** There cannot exist any computer program that can decide whether any two C++ programs are equivalent.

(xxiii) **F** An undecidable language is necessarily $\mathcal{NP}$-complete.

(xxiv) **T** Every context-free language is in the class $\mathcal{P}$-TIME.

(xxv) **T** Every regular language is in the class $\mathcal{NC}$

(xxvi) **F** Every Function that can be mathematically defined is recursive.

(xxvii) **F** Every bounded function from integers to integers is Turing-computable. (We say that f is bounded if there is some B such that $|f(n)| \leq B$ for all n.)

(xxviii) The language of all palindromes over $\{0, 1\}$ is inherently ambiguous.

(xxix) **F** The boolean satisfiability problem is undecidable.

(xxx) **T** If $\mathcal{P} = \mathcal{NP}$, then all one-way encoding systems are breakable in polynomial time.

(xxxi) **T** A language L is in $\mathcal{NP}$ if and only if there is a polynomial time reduction of L to SAT.

(xxxii) **F** Every subset of a regular language is regular.

(xxxiii) **T** The intersection of any context-free language with any regular language is context-free.

(xxxiv) **T** The question of whether two context-free grammars generate the same language is undecidable.

(xxxv) **T** There exists some proposition which is true but which has no proof.

(xxxvi) **T** If L_1 reduces to L_2 in polynomial time, and if L_2 is $\mathcal{NP}$, and if L_1 is $\mathcal{NP}$-complete, then L_2 must be $\mathcal{NP}$-complete.

(xxxvii) **F** Given any context-free grammar G and any string $w \in L(G)$, there is always a unique leftmost derivation of w using G.

(xxxviii) bf O The question of whether two regular expressions are equivalent is $\mathcal{NP}$-complete. (Do not guess. Look it up.)

(xxxix) **F** No language which has an ambiguous context-free grammar can be accepted by a DPDA.

(xl) **T** The intersection of any two regular languages is regular.

(xli) **F** The intersection of any two context-free languages is context-free.

(xlii) **T** If L_1 reduces to L_2 in polynomial time, and if L_2 is $\mathcal{NP}$, then L_1 must be $\mathcal{NP}$.

(xliii) **T** Let $F(0) = 1$, and let $F(n) = 2^{F(n-1)}$ for $n > 0$. Then F is recursive.

(xliv) **T** Every language which is accepted by some non-deterministic machine is accepted by some deterministic machine.

(xlv) **F** The language of all regular expressions over the binary alphabet is a regular language.

(xlvi) **T** There cannot exist any computer program that decides whether any two given C++ programs are equivalent.

(xlvii) **F** An undecidable language is necessarily $\mathcal{NP}$-complete.

(xlviii) **T** Every context-free language is in the class $\mathcal{P}$-TIME.

(xlix) **F** Every function that can be mathematically defined is recursive.

(l) **F** Every bounded function from integers to integers is recursive. (We say that f is bounded if there is some B such that $|f(n)| \leq B$ for all n.)

(li) **F** Every function that can be mathematically defined is recursive.

(lii) **T** The language of all binary strings which are the binary numerals for multiples of 23 is regular.

(liii) **F** Let β be the busy beaver function. You know that β is not recursive, but there is some recursive function F such that $\beta(n) = O(F(n))$.

5. Which of the following languages or problems are **known** to be $\mathcal{NP}$-complete? Write "T" if it is known to be $\mathcal{NP}$-complete, "F" otherwise. ("O" is not an option for this problem.) You may have to seach the internet.

(i) **T** SAT

(ii) **F** 2-SAT

(iii) **T** 3-SAT

(iv) **T** 4-SAT

(v) **T** 5-SAT

(vi) **F** Boolean Circuit.

(vii) **F** Context-free membership.

(viii) **F** The language of all strings generated by a given unrestricted grammar.

(ix) **F** The set of all solvable configurations of RUSH HOUR.

(x) **T** Given a big rectangle and a set of smaller rectangles, is it possible to place all the small rectangles into the big rectangle with no overlap?

(xi) **T** The block sorting problem. Given a list of n items and a number K, a "block move" moves a contiguous subset of items into another location in the list. Can the list be sorted with no more than K block moves? For example, ABCLMNODEFGHIJK can be sorted with 1 block move.

(xii) **F** Given a configuration in a game of generalized checkers (that means, any size board) can the black player force a win?

(xiii) **T** The firehouse problem. Given a graph $G = (V, E)$ and numbers K and d, is there a set $F \subseteq V$ of size K such that every vertex is within at most d steps of some member of F?

(xiv) **T** The traveling salesman problem.

(xv) **F** Given a finite sequence σ of distinct integers, does σ have an increasing subsequence?

6. State the pumping lemma for regular languages.

   See **hw5ans.pdf**.

7. Give a polynomial time reduction of the subset sum problem to the partition problem.

   See **hw5ans.pdf**.

8. Give a polynomial time reduction of 3-SAT to the independent set problem.

   See **hw5ans.pdf**.

9. This is not a question, but you must understand it!

   A deteriministic machine has at most one computation for a given input, but a non-deterministic machine could have many possible computations. We say that a non-deterministic machine M accepts a string w if, given w as input, M has at least one computation that ends in an accepting state. If L is a language, we say M accepts L if M accepts every $w \in L$ and accepts no other strings.

   If L is a language, we say that a non-deterministic machine M accepts L in polynomial time if M accepts L, and there is some constant k such that, for each $w \in L$, there is an accepting computation of M with input w consisting of $O(n^k)$ steps, where $n = |w|$.

   $\mathcal{NP}$–TIME (or simply $\mathcal{NP}$) is defined to be the class of all languages which are accepted by some machine in polynomial time.

10. I believe you will find this problem hard. But don't worry, it won't be on the test, although I really hope someone solves it.

    Let L be the language consisting of all binary numerals for multiples of 5. Then L is regular and has pumping length 5. Let $w = 1001011 \in L$. (Note that w is the binary numeral for 75.) Find the pumpable substring of w. (In the statement of the pumping lemma, the pumpable substring is usually denoted y, that is, $xy^iz \in L$ for all $i \geq 0$.)