University of Nevada, Las Vegas Computer Science 456/656 Spring 2021

Practice Problems for the Examination on March 8, 2023

Part I

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. State the pumping lemma for context-free languages. If the logic is wrong, you might get no partial credit, even if all the correct words are there.

4. Prove that the halting problem is undecidable.

5. Prove that any decidable language is enumerable in canonical order by some machine.

   Suppose $L \subseteq \Sigma^*$ is decidable. Let $w_1, w_2, \ldots$ be the strings over Σ in canonical order. The following program enumerates L in canonical order.

   For integers i = 1, . . .
   If $(w_i \in L)$ write w_i.

   The program will not get stuck because the loop condition can be decided.

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

   (i) **O** The Boolean circuit problem is in Nick's class.

   (ii) **T** Let $L = \{\langle G_1 \rangle G_2 : G_1 \text{ isnotequivalentto } G_2\}$ Then L is recursively enumerable.

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

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

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

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

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

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

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

   (x) **T** The factoring problem, where inputs are written in binary notation, is co-$\mathcal{NP}$.

   (xi) **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.

   (xii) **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.

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

(xiv) **F** The question of whether two regular expressions are equivalent is known to be $\mathcal{NP}$-complete.

(xv) **T** The halting problem is recursively enumerable.

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

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

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

(xix) **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.

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

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

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

(xxiii) **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.

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

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

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

(xxvii) **F** The language of all binary strings which are the binary numerals for prime numbers is context-free.

(xxviii) **F** The language of all binary strings which are the binary numerals for prime numbers is regular.

(xxix) **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.)~~

(xxx) **F** ~~The language of all palindromes over $\{0, 1\}$ is inherently ambiguous.~~

(xxxi) **F** ~~Every context-free grammar can be parsed by some deterministic top-down parser.~~

(xxxii) **T** ~~Every context-free grammar can be parsed by some non-deterministic top-down parser.~~

(xxxiii) **F** Commercially available parsers cannot use the LALR technique, since most modern programming languages are not context-free.

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

(xxxv) **T** If a string w is generated by a context-free grammer G, then w has a unique leftmost derivation if and only if it has a unique rightmost derivation.

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

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

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

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

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

(xli) **T** The set of all binary numerals for prime numbers is in the class $\mathcal{P}$.

(xlii) **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.

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

(xliv) **O** The question of whether two regular expressions are equivalent is $\mathcal{NP}$-complete.

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

(xlvi) **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.

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

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

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

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

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

(lii) **T** Let π be the ratio of the circumference of a circle to its diameter. (That's the usual meaning of π you learned in school.) The problem of whether the n^{th} digit of π, for a given n, is equal to a given digit is decidable.

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

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

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

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

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

(lviii) **T** If anyone ever proves that $\mathcal{P} = \mathcal{NP}$, then all public key/private key encryption systems will be known to be insecure.

Read This.

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.