CSC 456/656 Fall 2021 Practice for the Third Examination November 17, 2021

The entire practice test is 590 points. The real test will be shorter. **Report any errors immediately.**

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time.

   (i) **F** Let L be the language over $\{a, b, c\}$ consisting of all strings which have more a's than b's and more b's than c's. There is some PDA that accepts L.

   (ii) **T** The language $\{a^n b^n \mid n \geq 0\}$ is context-free.

   (iii) **F** The language $\{a^n b^n c^n \mid n \geq 0\}$ is context-free.

   (iv) **T** The language $\left\{a^i b^j c^k \mid j = i + k\right\}$ is context-free.

   (v) **T** The intersection of any three regular languages is regular.

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

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

   (viii) **T** If L is a context-free language over an alphabet with just one symbol, then L is regular.

   (ix) **F** There is a deterministic parser for any context-free grammar.

   (x) **T** The set of strings that your high school algebra teacher would accept as legitimate expressions is a context-free language.

   (xi) **T** Every language accepted by a non-deterministic machine is accepted by some deterministic machine.

   (xii) **T** The problem of whether a given string is generated by a given context-free grammar is decidable.

   (xiii) **T** If G is a context-free grammar, the question of whether $L(G) = \emptyset$ is decidable.

   (xiv) **F** Every language generated by an unambiguous context-free grammar is accepted by some DPDA.

   (xv) **T** The language $\{a^n b^n c^n d^n \mid n \geq 0\}$ is recursive.

   (xvi) **T** The language $\{a^n b^n c^n \mid n \geq 0\}$ is in the class $\mathcal{P}$-TIME.

   (xvii) **O** There exists a polynomial time algorithm which finds the factors of any positive integer, where the input is given as a binary numeral.

   (xviii) **F** Every undecidable problem is $\mathcal{NP}$-complete.

   (xix) **F** Every problem that can be mathematically defined has an algorithmic solution.

   (xx) **F** The intersection of two undecidable languages is always undecidable.

(xxi) **T** Every $\mathcal{NP}$ language is decidable.

(xxii) **T** The clique problem is $\mathcal{NP}$-complete.

(xxiii) **T** The traveling salesman problem is $\mathcal{NP}$-hard.

(xxiv) **T** The intersection of two $\mathcal{NP}$ languages must be $\mathcal{NP}$.

(xxv) **F** If L_1 and L_2 are $\mathcal{NP}$-complete languages and $L_1 \cap L_2$ is not empty, then $L_1 \cap L_2$ must be $\mathcal{NP}$-complete.

(xxvi) **O** There exists a $\mathcal{P}$-TIME algorithm which finds a maximum independent set in any graph G.

(xxvii) **T** There exists a $\mathcal{P}$-TIME algorithm which finds a maximum independent set in any acyclic graph G.

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

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

(xxx) **O** $\mathcal{NP} = \mathcal{P}$-SPACE

(xxxi) **O** $\mathcal{P}$-SPACE = EXP-TIME

(xxxii) **O** EXP-TIME = EXP-SPACE

(xxxiii) **F** EXP-TIME = $\mathcal{P}$-TIME.

By the time hierachy theorem.

(xxxiv) **F** EXP-SPACE = $\mathcal{P}$-SPACE.

By the space hierachy theorem.

(xxxv) **T** The traveling salesman problem (TSP) is $\mathcal{NP}$-complete.

(xxxvi) **T** The knapsack problem is $\mathcal{NP}$-complete.

(xxxvii) **T** The language consisting of all satisfiable Boolean expressions is $\mathcal{NP}$-complete.

(xxxviii) **T** The Boolean Circuit Problem is in $\mathcal{P}$.

(xxxix) **O** The Boolean Circuit Problem is in $\mathcal{NC}$.

(xl) **F** If L_1 and L_2 are undecidable langugages, there must be a recursive reduction of L_1 to L_2.

(xli) **T** The language consisting of all strings over $\{a, b\}$ which have more a's than b's is LR(1).

(xlii) **T** 2-SAT is $\mathcal{P}$-TIME.

(xliii) **O** 3-SAT is $\mathcal{P}$-TIME.

(xliv) **T** Primality is $\mathcal{P}$-TIME.

(xlv) **T** There is a $\mathcal{P}$-TIME reduction of the halting problem to 3-SAT.

(xlvi) **T** Every context-free language is in $\mathcal{P}$.

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

(xlviii) **T** Addition of binary numerals is in $\mathcal{NC}$.

(xlix) **F** Every context-sensitive language is in $\mathcal{P}$.

(l) **F** Every language generated by a general grammar is recursive.

(li) **F** The problem of whether two given context-free grammars generate the same language is decidable.

(lii) **F** If G is a context-free grammar, the question of whether $L(G) = \Sigma^*$ is decidable, where Σ is the terminal alphabet of G.

(liii) **T** The language of all fractions (using base 10 numeration) whose values are less than π is decidable. (A *fraction* is a string. "314/100" is in the language, but "22/7" is not.)

(liv) **T** There exists a polynomial time algorithm which finds the factors of any positive integer, where the input is given as a unary ("caveman") numeral.

(lv) **T** For any two languages L_1 and L_2, if L_1 is undecidable and there is a recursive reduction of L_1 to L_2, then L_2 must be undecidable.

(lvi) **F** For any two languages L_1 and L_2, if L_2 is undecidable and there is a recursive reduction of L_1 to L_2, then L_1 must be undecidable.

(lvii) **F** If P is a mathematical proposition that can be written using a string of length n, and P has a proof, then P must have a proof whose length is $O(2^{2^n})$.

(lviii) **T** If L is any $\mathcal{NP}$ language, there must be a $\mathcal{P}$–TIME reduction of L to the partition problem.

(lix) **F** Every bounded function is recursive.

(lx) **O** If L is $\mathcal{NP}$ and also co-$\mathcal{NP}$, then L must be $\mathcal{P}$.

(lxi) **T** If L is in $\mathcal{RE}$ and also in co-$\mathcal{RE}$, then L must be decidable.

(lxii) **T** Every language is enumerable.

(lxiii) **F** If a language L is undecidable, then there can be no machine that enumerates L. **F**

(lxiv) **T** There exists a mathematical proposition that can be neither proved nor disproved.

(lxv) **T** There is a non-recursive function which grows faster than any recursive function.

(lxvi) **T** There exists a machine that runs forever and outputs the string of decimal digits of π (the well-known ratio of the circumference of a circle to its diameter).

(lxvii) **F** For every real number x, there exists a machine that runs forever and outputs the string of decimal digits of x.

(lxviii) **O** **Rush Hour**, the puzzle sold in game stores everywhere, generalized to a board of arbitrary size, is $\mathcal{NP}$–complete.

(lxix) **O** There is a polynomial time algorithm which determines whether any two regular expressions are equivalent.

(lxx) **O** If two regular expressions are equivalent, there must be a polynomial time proof that they are equivalent.

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

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

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

(lxxiv) **F** Let L be the language over $\{a, b, c\}$ consisting of all strings which have more a's than b's and more b's than c's. There is some PDA that accepts L.

(lxxv) **T** Every subset of any enumerable set is enumerable.

(lxxvi) **T** If L is a context-free language which contains the empty string, then $L\backslash\{\lambda\}$ must be context-free.

(lxxvii) ________ If L is any language, there is a reduction of L to the halting problem. (Warning: this is a trick question. Give it some serious thought.)

(lxxviii) **T** The computer language C++ has Turing power.

(lxxix) **T** Let Σ be the binary alphabet. Every $w \in \Sigma^*$ which starts with 1 is a binary numeral for a positive integer. Let $Sq : \Sigma^* \to \Sigma*$ be a function which maps the binary numeral for any integer n to the binary numeral for n^2. Then Sq is an $\mathcal{NC}$ function.

(lxxx) **T** If L is any $\mathcal{P}$-TIME language, there is an $\mathcal{NC}$ reduction of the Boolean circuit problem to L.

(lxxxi) **T** If an abstract Pascal machine can perform a computation in polynomial time, there must be some Turing machine that can perform the same computation in polynomial time.

(lxxxii) **T** The binary integer factorization problem is co-$\mathcal{NP}$.

(lxxxiii) **F** Let L be any $\mathcal{RE}$ language which is not decidable, and let M_L be a machine which accepts L.
(a) If there are no strings of L of length n, let $T(n) = 0$.
(b) Otherwise, let $T(n)$ be the largest number of steps it takes M_L to accept any string in L of length n.
Then T is a recursive function.

(lxxxiv) **O** There is a polynomial time reduction of the subset sum problem to the binary factorization problem.

(lxxxv) **F** The language of all palindromes over $\{a, b\}$ is an LR language.

(lxxxvi) **F** The Simplex algorithm for linear programming is polynomial time.

(lxxxvii) **F** Remember what a *fraction* is? It's a string consisting of a decimal numberal, followed by a slash, followed by another decimal numeral whose value is not zero. For example, the string "14/37" is a fraction. Each fraction has a value, which is a number. For example, "2/4" and "1/2" are different fractions, but has the same value. For any real number x, the set of fractions whose values are less than x is $\mathcal{RE}$.

(lxxxviii) **F** The union of any two deterministic context-free languages must be a DCFL.

(lxxxix) **F** The intersection of any two deterministic context-free languages must be a DCFL.

(xc) **F** The complement of any DCFL must be a DCFL.

(xci) **F** Every DCFL is generated by an LR grammar.

(xcii) **T** The membership problem for a DCFL is in the class $\mathcal{P}$-TIME.

(xciii) **T** If $h : \Sigma_1 \to \Sigma_2^*$ is a function, $L_1 \in \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, $h(L_1) = L_2$ and L_1 is regular, then L_2 must be regular.

(xciv) **F** If $h : \Sigma_1 \to \Sigma_2^*$ is a function, $L_1 \in \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, $h(L_1) = L_2$ and L_2 is regular, then L_1 must be regular.

(xcv) **T** If $h : \Sigma_1 \to \Sigma_2^*$ is a function, $L_1 \in \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, $L_1 = h^{-1}(L_2)$ and L_2 is regular, then L_1 must be regular.

2. (10 points each) For each language or problem listed below, fill in the blank with a letter from A to G, where each letter has the following meaning:
A: Known to be $\mathcal{NC}$.
B: Known to be $\mathcal{P}$–TIME but not known to be $\mathcal{NC}$.
C: Known to be $\mathcal{NP}$, but not known to be $\mathcal{P}$–TIME, and not known to be $\mathcal{NP}$–complete.
D: Known to be $\mathcal{NP}$–complete.
E: Known to be $\mathcal{P}$–SPACE, but not known to be $\mathcal{NP}$.
F: Known to be decidable, but not known to be $\mathcal{P}$–SPACE
G: Undecidable.

(i) **A** The Dyck language.

(ii) **B** The Boolean Circuit problem.

(iii) **E** Equivalence of NFAs.

(iv) **D** Block Sorting.

(v) **A** The language $\{a^n b^n c^n \,:\, n > 0\}$.

(vi) **C** Factoring integers expressed as decimal numerals.

(vii) **A** Multiplication of binary numerals.

(viii) **F** All checkers positions where Black can force a win.

3.)10 points each) For each language or problem listed below, fill in the blank with a letter from H to L, where each letter has the following meaning:

H: Decidable.
I: $\mathcal{RE}$ but not decidable.
K: co-$\mathcal{RE}$ but not decidable.
L: Neither $\mathcal{RE}$ nor co-$\mathcal{RE}$.

(i) **I** The halting problem.

(ii) **K** The diagonal language.

(iii) **I** The complement of the diagonal language.

(iv) **H** The subset sum problem.

(v) **K** Equivalence of context-free grammars.

(vi) **H** Rush Hour (the sliding block puzzle).

4. [20 points] Design an LALR praser for the following context-free grammar where S is the start symbol and the alphabet of terminals is $\{a, b\}$. I have filled in a few entries.

1. $S \to a_2\, S_3\, b_4\, S_5$
2. $S \to \lambda$

	a	b	\$	S
0	$s2$			1
1			HALT	
2	$s2$	$r2$		3
3		$s4$		
4	$s2$	$r2$	$r2$	5
5		$r1$	$r1$	

5. [20 points] Give a $\mathcal{P}$-TIME reduction of the subset sum problem to the partition problem.

Let $(K, x_1, \ldots x_n)$ be an instance of the subset sum problem. Let $S = x_1 + \cdots x_n$. The reduction of that instance is $(K + 1, S - K + 1, x_1, \ldots x_n)$, and instance of the partition problem.

6. [20 points] State the pumping lemma for context-free languages.

For any context-free language L there exists a positive integer p such that for any $w \in L$ where $|w| \in L$ there exist strings u, v, x, y, z such that the following four conditions hold:
1. $w = uvxyz$,
2. $|vxy| \leq p$,
3. v and y are not both the empty string,
4. For any integer $i \geq 0$, $uv^ixy^iz \in L$.

7. [20 points] Suppose there is a machine that enumerates a language in canonical order. Prove that L is decidable.

If L is finite, it is clearly decidable. If L is infinite, let $w_1, w_2, \ldots$ be a canonical order enumeration of L. The following program accepts L.

read w
$n = |w|$
for i from 1 to 2^{n+1}
  if$(w == w_i)$
    Accept w and Halt
Reject w

8. [20 points]

Let L be the language generated by the Chomsky Normal Form (CNF) grammar given below.

$S \to IS$
$S \to XY$
$S \to WS$
$S \to a$
$X \to IS$
$Y \to ES$
$W \to w$
$I \to i$
$E \to e$

Use the CYK algorithm to prove that the string *iwiaea* is a member of L. Use the figure below for your work.