University of Nevada, Las Vegas
Computer Science 456/656 Fall 2020

Answers to Practice Final for December 9, 2020

This version `Sat Dec 5 13:52:14 PST 2020`

1. [5 points each] True or False. If the question is currently open, write "O" or "Open."

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

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

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

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

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

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

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

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

   (ix) **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.

   (x) **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.

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

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

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

   (xiv) **F** The complement of every context-free language is context-free.

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

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

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

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

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

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

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

(xxvi) **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.

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

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

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

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

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

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

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

(xxxiv) **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.)

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

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

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

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

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

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

(xli) **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.

These problem are only a sample of the many true/false questions I have given during the years I have taught this course. I will post a more complete list later.

2. [10 points] If there is an easy reduction from L_1 to L_2, then L_2 is at least as hard as L_1.

3. [15 points] Draw the state diagram for a minimal DFA that accepts the language described by the regular expression `a*a(b+ab)*`

4. [15 points] Write a regular expression for the language accepted by the NFA shown in Figure 1.

   $b^*(a + c + ba^*c)(a + c + bb^*(a + c) + bb^*ba^*c)^*$ This expression is not the only solution, and may not be the shortest one.

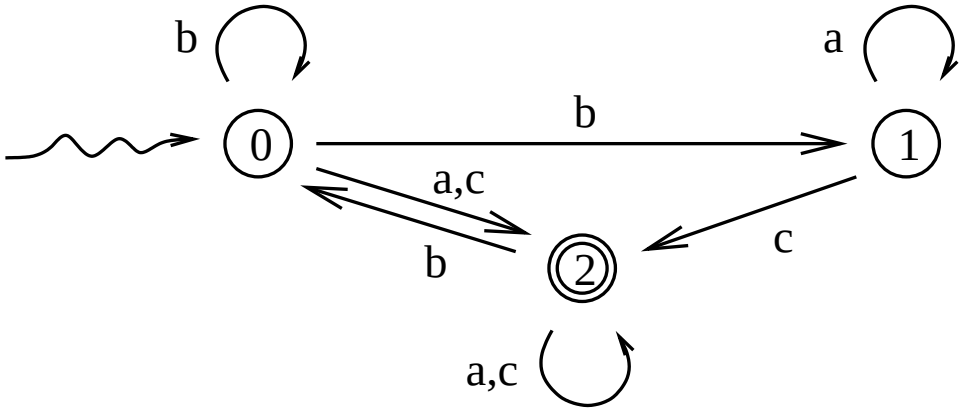


Figure 1: The NFA for Problems 4 and 18.

5. [20 points] Let L be the language of all binary numerals for positive integers equivalent to 2 modulo 3. Thus, for example, the binary numerals for 2, 5, 8, 11, 14, 17 ... are in L. We allow a binary numeral to have leading zeros; thus (for example) $001110 \in L$, since it is a binary numeral for 14. Draw a minimal DFA which accepts L.

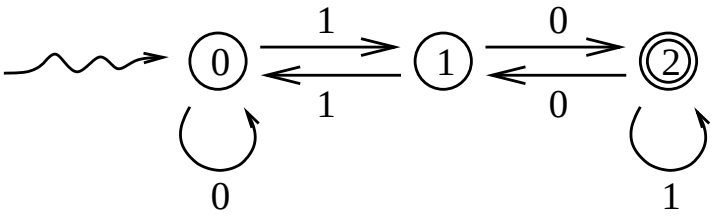


Figure 2: DFA which accepts the language of binary numerals equivalent to 2 modulo 3

6. [20 points] Design a PDA that accepts the language of all palindromes over the alphabet $\{a, b\}$.

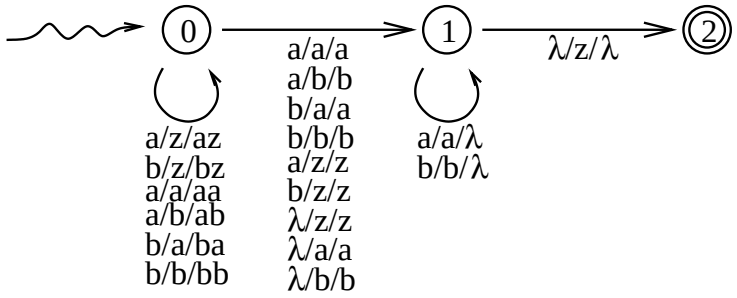


Figure 3: PDA which accepts by final state the palindromes over $\{\mathbf{a}, \mathbf{b}\}$

8. [10 points] Consider the context-free grammar with start symbol S and productions as follows:

   $S \to s$
   $S \to bLn$
   $S \to wS$
   $L \to \lambda$
   $L \to SL$

   Write a leftmost derivation of the string *bswsbwsnn*

   $S \Rightarrow bLn \Rightarrow bSLn \Rightarrow bsSLn \Rightarrow bswSLn \Rightarrow bswsLn \Rightarrow bswsSLn \Rightarrow bswsbLnLn \Rightarrow bswsbSLnLn \Rightarrow bswsbwSLnLn \Rightarrow bswsbwsLnLn \Rightarrow bswsbwsnLn \Rightarrow bswsbwsnn$

9. [5 points] What class of machines accepts the class of context free languages?

   Push down Automata (PDAs)

10. [5 points] What class of machines accepts the class of regular languages?

    Finite Automata, or DFAs, or NFAs.

11. [5 points] What class of machines accepts the class of recursively enumerable languages?

    Turing Machines

12. [10 points] What is the Church-Turing Thesis, and why is it important?

    Every computation that can be done by any machine can be done by a Turing machine. This is important because, if we can prove that no Turing machine can perform a certain computation, we know that no machine can perform that computation.

13. [10 points] What does it mean to say that a language can be recursively enumerated in *canonical order*? What is the class of languages that can be so enumerated?

    If u and v are strings, we say that u comes before v in the canonical the canonical order if either $|u| < |v|$, or $|u| = |v|$ and u comes before v in lexical [alphabetic] order.

    We say that a language L can be enumerated in canonical order if there is some machine with no input which outputs all the strings of L in canonical order, and does not output any other strings.

A language L can be enumerated in canonical order if and only if it is recursive [decidable].

14. [5 points] What does it mean to say that machines M_1 and M_2 are *equivalent*?

    If M_1 and M_2 are given the same input, they will give the same output.

15. Give definitions: [10 points each]

    (a) Give a definition of the language class $\mathcal{NP}$–TIME.

    Two definitions were given in class.

    i. A language L is in the class $\mathcal{NP}$–TIME if there is some non-deterministic machine which accepts L in polynomial time.

    ii. A language L is in the class $\mathcal{NP}$–TIME if there is some deterministic machine which, for each $w \in L$, verifies that $w \in L$ in polynomial time if given another string, called a witness or certificate for w.

    (II) Give the definition of a *polynomial time reduction* of a language L_1 to another language L_2.

    Let Σ_1, σ_2 be the alphabets of L_1 and L_2, respectively. A polynomial time reduction of L_1 to L_2 is a function $R : \Sigma_1^* \to \Sigma_2^*$ which is computable in polynomial time such that, for any $w \in \Sigma_1^*$, $w \in L_1$ if and only if $R(w) \in L_2$.

    (III) Give a definition of $\mathcal{NP}$-*complete* language. A language L is $calNP$-complete if

    i. L is in the class $\mathcal{NP}$–TIME

    ii. For any language L_2 in the class $\mathcal{NP}$-TIME, there is a polynomial time reduction of L_2 to L.

    (IV) Give a definition of a *decidable* language.

    We say that $L \subseteq \Sigma^*$ is decidable if there is a machine M such that, given $w \in \Sigma^*$, M halts with input w, and outputs 1 if $w \in L$ and outputs 0 if $w \notin L$.

16. [15 points] We say a binary string w over is *balanced* if w has the same number of 1's as 0's. Let L be the set of balanced binary strings. Give a context-free grammar for L.

    Here is a simple ambiguous context-free grammar for L.
    $S \to SS$
    $S \to aSb$
    $S \to bSa$
    $S \to \lambda$

    Here is an unambiguous grammar for L.
    $S \to aAbS$
    $S \to bBaS$
    $S \to \lambda$
    $A \to aAbA$
    $A \to \lambda$
    $B \to bBaB$

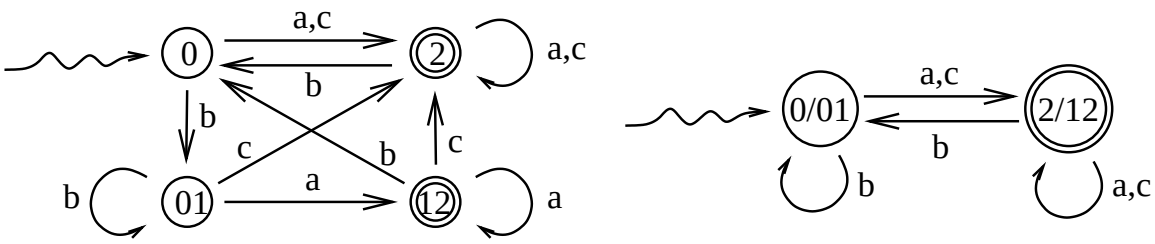
$B \to \lambda$

17. [10 points] Give a Chomsky normal form (CNF) grammar for the language of all palindromes of odd length over the alphabet $\{a, b\}$.

$S \to a$
$S \to b$
$S \to AC$
$S \to BD$
$C \to SA$
$D \to SB$
$A \to a$
$B \to b$

The grammar is unambiguous, although that was not a requirement. Unlike for the previous problem, I do not believe there is a simpler ambiguous answer.

18. [20 points] Construct a minimal DFA equivalent equivalent to the NFA shown in Figure 1.



The first figure is the DFA obtained by the powerset method. The second figure is obtained by minimization.

19. [10 points] Consider the context-free grammar G, with start symbol S and productions as follows:

$S \to s$

$S \to bLn$

$S \to iS$

$S \to iSeS$

$L \to \epsilon$

$L \to LS$

Prove that G is ambiguous by giving two different leftmost derivations for some string.

$S \Rightarrow iSeS \Rightarrow iiSeS \Rightarrow iiaeS \Rightarrow iiaea$

$S \Rightarrow iS \Rightarrow iiSeS \Rightarrow iiaeS \Rightarrow iiaea$

22. Every language we have discussed this semester falls into one of these categories.

(a) $\mathcal{NC}$.

(b) $\mathcal{P}$ but not known to be $\mathcal{NC}$.

(c) $\mathcal{NP}$ but not known to be $\mathcal{P}$ and not known to be $\mathcal{NP}$-complete.

(d) Co-$\mathcal{NP}$ but not known to be $\mathcal{P}$.

(e) Known to be $\mathcal{NP}$-complete.

(f) Known to be $\mathcal{P}$-space, but not known to be $\mathcal{NP}$.

(g) Known to be EXP-time, but not known to be $\mathcal{P}$-space.

(h) Known to be EXP-space, but not known to be EXP-time.

(i) Recursive, but not known to be EXP-space.

(j) RE (Recursively enumerable), but not recursive.

(k) Co-RE, but not recursive.

(l) Neither RE nor co-RE.

Identify which of the above categories each language or problem listed below belongs to. 5 points each.

(i) (j) The language consisting of all Pascal programs P such that P halts if given P as its input file.

(ii) (?) The language of all encodings of Turing Machines which fail to halt for at least one possible input string.

(iii) (e) The 0-1 Traveling Salesman Problem.

(iv) (k) The diagonal language.

(v) (e) L_{sat}, the set of satisfiable boolean expressions.

(vi) (e) The language described by the regular expression a^*b*.

(vii) (b) The Boolean circuit problem.

(viii) (b) Dynamic programming where all subproblems have Boolean solutions.

(ix) (a) The language $\{a^n b^n : n \geq 0\}$.

(x) (c) Factorization of a binary numeral.

(xi) _____ (a) Dynamic programming, where all subproblems have Boolean solutions, and each subproblem can be computed using only the previous subproblem.

(xii) (a) Multiplication of two binary numerals.

(xiii) (e) The 0-1 traveling salesman problem.

(xiv) (b) The restricted subset sum problem where the weight of each item is an integer which does not exceed the square of the number of items.

(xv) (?) The set of position of HEX from which the next player to move can force a win.

(xvi) (f) The set of configurations of RUSH HOUR which are solvable.

(xvii) (a) Factorization of a unary numeral.

(xviii) (j) The halting problem.

(xix) (k) The diagonal language.

(xx) (e) The clique problem.

(xxi) (b) Primality, where the input is written in binary.

(xxii) (a) The language generated by a given context-free grammar.

(xxiii) (a) The language of all monotone increasing sequences of arabic numerals for positive integers. (For example, "1,5,23,41,200,201" is a member of that language.)

(xxiv) (a) The language accepted by a given DFA.

(xxv) (?) The set of all positions from which black can force a win in a game of generalized checkers.

(xxvi) (?) The set of all configurations of the children's game "Boxes" from which the first player can force a win. (I used to play that game as a child, and I never did figure out an optimal strategy. I don't feel bad about that anymore, now that I know the complexity class of that problem.)

(xxvii) (a) The set of all configurations of the game "Nim" from which the first player can force a win.

(xxviii) (l) The set of all ordered pairs of positive numerals $(\langle n \rangle, \langle m \rangle)$ $m = \Sigma(n)$, where Σ is the busy beaver function.

(xxix) (?) Primality of a unary numeral.

(xxx) (k) The context-free grammar equivalence problem.

(xxxi) (e) The independent set problem.

23. [30 points] The grammar below is an alternative unambiguous CF grammar for the Dyck language, and is parsed by the given LALR parser. Write a computation of the parser for the input string *aabb*.

24. [30 points] The grammar below is an alternative unambiguous CF grammar for the Dyck language.

1 $S \to a_2 S_3 b_4 S_5$

2 $S \to \lambda$

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

The following ambiguous grammar, with start symbol E, generates certain expressions. The parser illustrated below is intended to parse a string in accordance with C++ rules. However, the action table has one error. (Wrong!) Identify that error, and fix it.

1 $E \to x_2$

2 $E \to E -_3 E_4$

3 $E \to -_5 E_6$

4 $E \to (_7 E_8)_9$

	x	$-$	(	)	\$	S
0	$s2$	$s5$	$s7$			1
1		$s3$			halt	
2		$r1$		$r1$	$r1$	
3	$s2$	$s5$	$s7$			4
4		$r2$		$r2$	$r2$	
5	$s2$	$s5$	$s7$			6
6		$r3$		$r3$	$r3$	
7	$s2$	$s5$	$s7$			8
8		$s3$		$s9$		
9		$r4$		$r4$	$r4$	

There were more errors. I hope I caught them all.

25. For each of the following languages, state whether the language is regular, context-free but not regular, context-sensitive but not context-free, or not context-sensitive.

    [5 points each]

    (a) **Regular.** The set of all strings over the alphabet $\{a, b\}$ of the form $a^n b^m$.

    (b) **Context-free, not regular.** The set of all strings over the alphabet $\{a, b\}$ of the form $a^n b^n$.

    (c) **Context-sensitive, not context-free.** The set of all strings over the alphabet $\{a, b, c\}$ of the form $a^n b^n c^n$.

    (d) **Context-free, not regular.** The set of all strings over the alphabet $\{a, b, c\}$ which are **not** of the form $a^n b^n c^n$.

    (e) **Not context-sensitive.** The set of all strings over the alphabet $\{a\}$ of the form $a^{n^2 : n \geq 1}$.

26. [15 points] Draw a minimal DFA which accepts the language L over the binary alphabet $\Sigma = \{a, b, c\}$ consisting of all strings which contain either *aba* or *caa* as a substring.

27. [10 points] State the pumping lemma for regular languages accurately. If you have all the right words but in the wrong order, that means you truly do not understand the lemma, and you might get no partial credit at all.

28. These are reduction problems. I could give one of them on the test. The proof should be very informal.

    (a) Find a $\mathcal{P}$-time reduction of 3-CNF-SAT to the independent set problem.

29. [20 points] Prove that every recursively enumerable language is accepted by some Turing machine.

    Let L be a recursively enumerable language; let M be a machine that enumerates L. By the Church Turing thesis, we only need show that some machine accepts L. We let our machine be a program which reads a string w and halts if and only if $w \in L$. Let $w_1, w_2, \ldots$ be the strings of L in the order they are enumerated by M.

    Here is the program:

Read w
For $i = 1, 2, \ldots$
  If $(w = w_i)$ HALT

30. [20 points] Prove that every language accepted by a Turing machine is recursively enumerable.

Let M be a Turing machine that accepts a language L over an alphabet Σ. Let $w_1, w_2, \ldots$ be an enumeration of Σ^* in canonical order. (That enumeration is easy to compute.) The following program runs forever, enumerating L.

For $(t = 1, 2, \ldots)$ // infinite loop
  For $(i = 1, 2, \ldots t)$ // finite loop
    If (M accepts w_i within t steps)
      WRITE w_i.

If w_i is written, then M clearly accepts w_i, hence $w_i \in L$. Conversely, if $w_i \in L$, then M accepts L within t steps for some t, and hence will be written during the t^{th} iteration of the outer loop.

31. [20 points] Give a context-sensitive grammar for the language $\{a^n b^n c^n : n \geq 1\}$

Previous answer was perhaps incorrect.

32. [20 points] Give a context-sensitive grammar for the language $\{a^n b^n c^n d^n : n \geq 1\}$

Previous answer was perhaps incorrect.

33. [20 points] Give a general grammar for the language $\left\{a^{2^n}\right\}$

$S \to L1R$
$L \to LD$
$D1 \to 11D$
$DR \to R$
$L \to \lambda$
$R \to \lambda$

34. [20 points] Prove that the halting problem is undecidable.