University of Nevada, Las Vegas Computer Science 456/656 Fall 2019

Answers to Assignment 8: Due December 6, 2019

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time. In the questions below, $\mathcal{P}$ and $\mathcal{NP}$ denote $\mathcal{P}$-TIME and $\mathcal{NP}$-TIME, respectively.

   (i) **T** Let L be the language over $\{a, b, c\}$ consisting of all strings which have more a's than b's and c's combined. (That is, $w \in L$ if and only if $\#_a(w) > \#_b(w) + \#_c(w)$.) There is some PDA that accepts L.

   (ii) **T** The language $\{a^p \mid p \text{ is prime}\}$ is in the class $\mathcal{P}$-TIME.

   (iii) **T** The dominating set problem is $\mathcal{NP}$-complete.

   (iv) **T** The bin packing problem is $\mathcal{NP}$-complete.

   (v) **O** The minimum spanning tree problem is $\mathcal{NP}$-complete.

   (vi) **F** The general grammar membership problem is $\mathcal{NP}$-complete.

   (vii) **O** The general sliding block problem is $\mathcal{NP}$-complete.

   (viii) **T** If L_1 is an undecidable language and $L_2 \subseteq L_1$ is decidable, then $L_1 \backslash L_2$ must be undecidable.

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

   (x) **F** If x and y are integers given as binary numerals, computation of the binary numeral for x^y can always be done in polylogarithmic time with polynomially many processors.

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

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

   (xiii) **F** It is decidable whether a context free gammar with terminal alphabet Σ generates Σ^*.

   (xiv) **T** If G_1 and G_2 are context-free grammars and $L(G_1) \neq L(G_2)$, there is a proof that $L(G_1) \neq L(G_2)$.

   (xv) **T** The Post correspondence problem is undecidable.

   (xvi) **T** The intersection of two recursively enumerable languages must be recursively enumerable.

   (xvii) **T** If x is a real number, and if the set of fractions whose valuea are less than x is decidable, then there must be a recursive function D where $D(n)$ is the n^{th} digit after the decimal point in the decimal expansion of x.

   (xviii) **T** There exists an algorithm which finds the prime factors of any positive integer, where the input is given as a binary numeral.

2. Which of these problems (not necessarily 0/1 problems) are **known** to be workable by polynomially many processors in polylogarithmic time? Answer **Y** for yes, **N** for no. For example, if a problem is known to be in $\mathcal{P}$–TIME but not known to be in $\mathcal{NC}$, the answer is **N**, since it is not known whether $\mathcal{P} = \mathcal{NC}$.

(a) **N** Compute the binary numeral for x^y, where x and y are given as binary numerals.

(b) **Y** Is a given array sorted?

(c) **N** The Boolean circuit problem.

(d) **Y** Sorting an array.

(e) **Y** Determining whether a given string is accepted by a given NFA.

(f) **Y** Finding a minimal spanning tree of a weighted graph. (Kruskal's algorithm and Primm's algorithm solve this problem in polynomial time.)

(g) **Y** Finding an optimal prefix-free binary code for a weighted alphabet. (Huffman's algorithm solves this problem in polynomial time.)

(h) **Y** Computing the first n Fibonacci numbers.

(i) **Y** Connectivity. Given a graph (not digraph) G, is G connected?

(j) **N** Boolean satisfiability. (Otherwise known as SAT.)

(k) **Y** 2-SAT.

(l) **Y** Context-free membership.

(m) **N** Context-sensitive membership.

3. Which of these functions are recursive? Answer **Y** if the function is recursive, **N** if it is not recursive.

(a) **Y** The Ackermann function.

(b) **N** The Busy Beaver function.

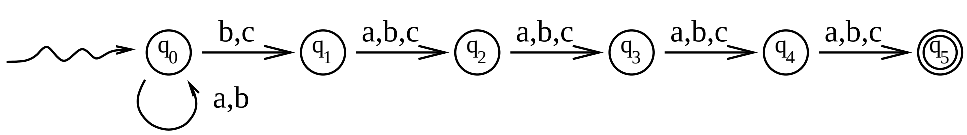
(c) **N** The number of members of L which have length exactly n, where L is some R.E. but undecidable language over the binary alphabet $\Sigma = \{0, 1\}$.

(d) **Y** The n^{th} decimal digit of π.

(e) **N** The n^{th} decimal digit of Chaitin's constant.

(f) **Y** The n^{th} decimal digit of a given real algebraic number, *i.e.* a real number which is a root of a given polynomial with integral coefficients.

4. Give an NFA with six states whose equivalent minimal DFA has 64 states (including the dead state).



5. Fill each the blank with one of the following three words: preorder, postorder, level order.

(a) A recursive-descent parser determines the internal nodes of the parse tree in **preorder.**

(b) An LALR parser determines the internal nodes of the parse tree in **postorder.**

6. A context-free grammar is *linear* if the right side of each production has at most one variable. Find a linear grammar which generates a language which is not regular.

$S \to aSa$
$S \to bSb$

7. Give an example of an unambiguous context-free grammar G, such that $L(G)$ is not accepted by any DPDA.

   Same answer as Problem 6.

8. Give an example of an ambiguous context-free grammar G such that there is a DPDA which finds a parse tree of every $w \in L(G)$.

   $S \to SS$
   $S \to \lambda$

9. Given an example of an undecidable $\mathcal{NP}$-complete problem.

   There is no such thing.

10. Give an example of two undecidable languages whose intersection is decidable.

    HALT and $\overline{\text{HALT}}$. The interstion is $\emptyset$.

11. Find two undecidable languages whose union is decidable.

    HALT and $\overline{\text{HALT}}$. The union is Σ^*.

12. Prove that L^R is regular for any regular language L. (Hint: NFA)

    Let M be an NFA for L which has exactly one start state. Let M^R have the same states as R, but with the start statge and the final state exchanged, and with all arrows reversed. Then M^R accepts L^R.

13. Give a polynomial time reduction of 4-SAT to 3-SAT.

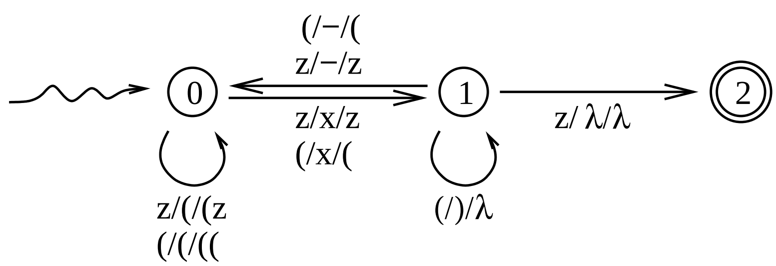
    For problems 13 and 14, see http://web.cs.unlv.edu/larmore/Courses/CSC477/F19/Assignments/2SatP.pdf

14. Give a polynomial time reduction of 3-SAT to 2-SAT.

    Existence of such a reduction would imply that $\mathcal{P} = \mathcal{NP}$, since 2SAT is $\mathcal{P}$-TIME.

15. Construct a PDA which accepts the algebraic language whose grammar is given below, where E is the start symbol. The grammar is unambiguous. This is a rather hard problem. In the compiler class, it gets much worse. (If I leave out the second production, the problem is much easier.)

    $E \to E - T$
    $E \to T$
    $T \to (E)$
    $T \to x$



16. Give a proof that the factoring problem is both $\mathcal{NP}$ and co-$\mathcal{NP}$. (The factoring problem is, given a binary numeral for a number n and another binary numeral for a number a, determine whether there is divisor d of n which is less than a but more than 1.)

    If d exists, then d is a certificate for the factoring problem, hence the problem in $\mathcal{NP}$. If d does not exist, then a certificate is the prime factorization of n, which can be verified by multiplying the factors, and by verifying that each factor is prime in $\mathcal{P}$ time, and then verifying that the smallest of those is at least a.

17. Prove that if $\mathcal{NP} = \mathcal{P}$, no trapdoor function exists.

    If $\mathcal{P} = \mathcal{NP}$, then the inverse of any function that can be computed in polynomial time can be computed in polynomial time.

18. Find a general grammar which generates $\left\{w \in \{a,b\}^* \mid \#_a(w) + \#_b(w) = 2^n, n \geq 0\right\}$ (I can do it with 5 variables and 8 productions. Can you do better?)

    Variables L, R, D, A, X.

    $S \to LAR$
    $L \to LD$
    $DA \to AAD$
    $AR \to \lambda$
    $L \to X$
    $XA \to aX \mid bX$
    $XR \to \lambda$