CS154 and CS154N Final Examination
Wednesday, March 15, 2000, 12:15–3:15PM

Directions

- The exam is *open book*; any written materials may be used.
- Succinctness counts. You may receive as little as 0 points for an answer that, even if correct, is much more complex than the best answer. Use the spaces allotted as a guide to how long answers should be.
- If you are taking CS154N, answer questions 1 through 7. You have 2 hours for the exam. The total number of points for your questions is 100.
- If you are taking CS154, answer all 12 questions. You have 3 hours for the exam. The total number of points is 200.
- Do not forget to **sign the pledge** below.

I acknowledge and accept the honor code.

_____

Print your name here: _____

1	2	3	4
5	6	7	8
9	10	11	12

Problem 1: (14 points) Suppose there are four problems (languages) A, B, C, and D. Each of these languages may or may not be in the class $\mathcal{NP}$. However, we know the following about them:

i. There is a polynomial-time reduction from A to B.
ii. There is a polynomial-time reduction from B to C.
iii. There is a polynomial-time reduction from D to C.

Below are seven statements. Indicate whether each is:

- CERTAIN to be true, regardless of what problems A through D are, and regardless of the resolution of unknown relationships among complexity classes, of which "is $\mathcal{P} = \mathcal{NP}$?" is one example.
- MAYBE true, depending on what A through D are, and/or depending on the resolution of unknown relationships such as $\mathcal{P} = \mathcal{NP}$?
- NEVER true, regardless of what A through D are, and regardless of the resolution of unknown relationships such as $\mathcal{P} = \mathcal{NP}$?

a) If A is NP-complete then C is NP-complete. _____

b) A is NP-complete and C is in $\mathcal{P}$. _____

c) B is NP-complete and D is in $\mathcal{P}$. _____

d) If A is NP-complete and B is in $\mathcal{NP}$, then B is NP-complete. _____

e) If C is NP-complete, then D is in $\mathcal{NP}$. _____

f) C is in $\mathcal{P}$, and the complement of D is not in $\mathcal{P}$. _____

g) B is not in $\mathcal{P}$, and A is not in $\mathcal{NP}$. _____

Problem 2: (16 points) Suppose there are four problems (languages) A, B, C, and D. Each of these languages may or may not be recursively enumerable. However, we know the following about them:

i. There is a reduction (i.e., an algorithm, not necessarily polynomial-time) from A to B.
ii. There is a reduction from B to C.
iii. There is a reduction from D to C.

Below are eight statements. Indicate whether each is:

- CERTAIN to be true, regardless of what problems A through D are.
- MAYBE true, depending on what A through D are.
- NEVER true, regardless of what A through D are.

a) A is recursively enumerable but not recursive, and C is recursive. _____

b) A is not recursive and D is not recursively enumerable. _____

c) The complement of A is not recursively enumerable, but the complement of B is recursively enumerable. _____

d) The complement of B is not recursive, but the complement of C is recursive. _____

e) If A is recursive, then the complement of B is recursive. _____

f) If C is recursive, then the complement of D is recursive. _____

g) If C is recursively enumerable, then the union of B and D is recursively enumerable. _____

h) If C is recursively enumerable, then the intersection of B and D is recursively enumerable. _____

Problem 3: (20 points) We wish to prove that NT, the *nontautology problem* (given a Boolean expression E, does there exist a truth-assignment for the variables of E that makes E false?), is NP-complete. We can easily show that NT is in $\mathcal{NP}$; assume this part done.

a) Describe a polynomial-time reduction from SAT to NT.

b) Explain briefly why your reduction is correct.

c) We know that the existence of this reduction, plus the fact that SAT is known to be NP-complete, is enough to prove NT is NP-complete. However, if we met someone who didn't accept the theorem that the existence of the reduction is sufficient, how would you apply the "clincher," and prove that NT is NP-complete, given only that NT is in $\mathcal{NP}$, that SAT is NP-complete, and that this reduction exists?

Problem 4: (20 points) A two-dimensional Turing machine has a tape that is infinite in two directions, as shown in Fig. 1. That is, the tape, divided into squares, occupies the entire southeast quadrant of the plane. The input is placed at the left (west) end of the top (north) row, with the tape head initially at the northwest corner; all other cells are blank. Figure 1 shows the TM after it has been operating for a while. The head has moved, the current state is q, and some cells have been changed.

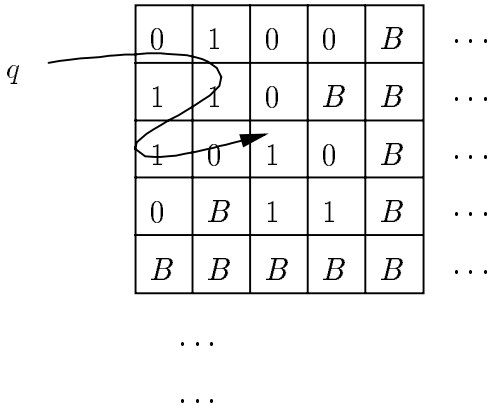

Figure 1

One move of this TM is determined by the state and tape symbol scanned. In one move, it can change state, write a new tape symbol on the tape scanned, and move one cell in any of the four directions: north, east, south, and west. In this problem, you will outline (part of) a proof that any language accepted by a two-dimensional TM is also accepted by some one-dimensional (ordinary) TM. You may construct a multitape TM if you like, but the number of tapes must be fixed, independent of how many rows of its two-dimensional tape the simulated two-dimensional TM uses.

a) One tape of the one-dimensional TM will represent the entire tape of the two-dimensional TM. Describe informally but clearly how you would represent a two-dimensional tape on a one-dimensional tape.

b) As an example, show how you would encode the tape of Fig. 1.

c) Describe, informally but clearly, how you would simulate a move of the two-dimensional TM in which the head moved south (down). You may use a fixed number of scratch tapes if you like.

Problem 5: (10 points) The following instance of Post's Correspondence Problem:

Index	A	B
1	10	101
2	101	011
3	110	100

Has no solution.

a) Give a brief but clear argument showing that there is no solution.

b) If we add a fourth pair, there can be solutions. Choose one pair to add so that (i) there is a solution, and (ii) that solution is as short as possible.

Problem 6: (10 points) The Turing machine $(\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, B\}, \delta, q_0, B)$ has the following three transitions, and no others:

$\delta(q_0, 0) = (q_1, 1, R)$
$\delta(q_1, 1) = (q_2, 0, L)$
$\delta(q_2, 1) = (q_0, 1, R)$

Starting with initial ID $q_0 0110$, show the entire sequence of ID's entered by this TM until it halts.

Problem 7: (10 points) It is known that there is a polynomial algorithm to decide whether the language of a CFG is empty. However, the following is an apparent "proof" that the problem of emptiness of a CFL (hereafter *ECFL*) is NP-complete:

We shall reduce the problem SAT to ECFL. Given an instance E of SAT, if E is satisfiable, produce the CFG $(\{S\}, \{0\}, \{S \to S\}, S)$. If E is not satisfiable, produce instead the CFG $(\{S\}, \{0\}, \{S \to 0\}, S)$. Thus, our reduction turns any satisfiable expression into a CFG with an empty language, and any nonsatisfiable expression into a CFG with a nonempty language.

Briefly and clearly explain the fallacy in this "proof," i.e., why does this "proof" not show that ECFG is NP-complete? Note: you may assume it is understood that ECFL is in $\mathcal{NP}$; the fallacy does not involve our omission of the "in $\mathcal{NP}$" part of an NP-completeness proof.

END OF THE EXAM FOR CS154N STUDENTS
CS154 STUDENTS CONTINUE ON THE NEXT PAGE

Problem 8: (30 points) For each of the following pairs of language descriptions, choose the correct relationship from among the following four choices:

(1) $\subset$ (2): language (1) is a proper subset of language (2).

(1) $\supset$ (2): language (1) is a proper superset of language (2).

(1) = (2): languages (1) and (2) are the same.

(1) ? (2): languages (1) and (2) have no containment relationship; i.e., there are strings in (1) that are not in (2), and there are strings in (2) that are not in (1).

Score = 6 $\times$ number right minus 2 $\times$ number wrong, so guessing is outcome-neutral. Assume the usual conventions regarding the types of symbols (e.g., $A, B, \ldots$ are variables).

a)

(1): The language of the CFG with productions $S \to 0S1 \mid 1S0 \mid \epsilon$

(2): The language of regular expression $(\mathbf{0} + \mathbf{1})^*$.

---

b)

(1): The language (accepted by final state) of PDA $(\{p, q\}, \{0, 1\}, \{X, Z\}, \delta, q, Z, \{p\})$ with δ defined by the rules $\delta(q, 0, Z) = \{(q, XZ)\}$, $\delta(q, 0, X) = \{(q, XX)\}$, $\delta(q, 1, X) = \{(p, \epsilon)\}$, and $\delta(p, 1, X) = \{(p, \epsilon)\}$.

(2): The language of the CFG with productions $S \to 0S1 \mid 0S \mid \epsilon$.

---

c)

(1): The language of the CFG with productions $S \to AS \mid SB \mid \epsilon$, $A \to 0$, and $B \to 1$

(2): The language of regular expression $\mathbf{0}^*\mathbf{1}^*$

---

d)

(1): The language of regular expression $(\mathbf{0} + \mathbf{1})^*\mathbf{11}(\mathbf{0} + \mathbf{1})^*$

(2): The language of regular expression $(\mathbf{0}^*\mathbf{1}^*\mathbf{11})^*\mathbf{0}^*\mathbf{110}^*\mathbf{1}^*$

---

e)

(1): The NFA $(\{p, q\}, \{0, 1\}, \delta, q, \{q\})$ with transitions $\delta(q, 0) = \{p\}$, $\delta(q, 1) = \emptyset$, $\delta(p, 0) = \{p, q\}$, and $\delta(p, 1) = \{p\}$

(2): The CFG with productions $S \to AS \mid \epsilon$, $A \to 0B$, and $B \to 0B \mid 1B \mid 0$

Problem 9: (30 points) For each of the following languages, choose the correct category from the following four choices:

REG: The language is regular.

CF-CF: The language is context-free but not regular, and the complement of the language is also context-free.

CF-NON: The language is context-free, but its complement is not context-free.

NON: The language is not context-free.

Score = 6 $\times$ number right minus 2 $\times$ number wrong, so guessing is outcome-neutral.

a) The set of strings of 0's and 1's such that in no prefix does the number of 0's exceed the number of 1's by more than two, nor does the number of 1's exceed the number of 0's by more than two.

---

b) The set of strings of 0's and 1's such that the number of 1's is exactly twice the number of 0's.

---

c) The set of strings of 0's, 1's, and 2's, such that 0 is *not* the most frequently occurring symbol in the string.

---

d) The set of strings of 0's, 1's, and 2's such that the number of 0's is greater than the sum of the numbers of 1's and 2's.

---

e) The set of strings of 0's, 1's, and 2's with at least 100 of each of the three symbols.

Problem 10: (10 points) Here is a grammar G:

$S \to AS \mid b$
$A \to AA \mid a \mid b$
$B \to BB \mid a$

In the space below, draw the table that you get by applying the CYK algorithm to this grammar and the string $abaaa$.

Problem 11: (10 points) The language L consisting of all strings of 0's whose length is a prime is not a CFL. We can prove this fact using the pumping lemma for CFL's. Begin the proof by assuming that L is context free, and that therefore there is some pumping-lemma constant n for L.

a) What string do you choose for z?

---

b) Suppose that the "adversary" chooses $z = uvwxy$, such that $|vwx| \leq n$ and $vx \neq \epsilon$. What value of i do you choose to create a string uv^iwx^iy that is not in L?

---

c) Explain briefly but clearly why uv^iwx^iy is not in L.

Problem 12: (20 points) Let us consider functions $f(L) = M$, where L and M are languages over the alphabet $\{0, 1\}$. We say the function f is *nice* if whenever M is regular, L is regular. For example, the function $f(L) = L^R$ is nice, because if L^R is regular, then L must be regular. In proof, we know that the regular languages are closed under reversal. If L^R is regular, then $(L^R)^R$, which is L, is also regular. As another example, the function f that replaces all 1's by 0's (and leaves the 0's as they are) is not nice. For instance, let $L = \{0^n 1^n \mid n \geq 1\}$. Then $f(L)$ is the language of regular expression $(\mathbf{00})^*$. If f were nice, then since $f(L)$ is regular we could conclude that L is regular, which it isn't.

Your problem is to determine whether each of the following four functions f is nice. In each case, give either a brief proof or a counterexample. In proofs, you may assume that the regular sets are closed under any operation whose closure appears in the reader, the class notes, or the homeworks. For counterexamples, it is sufficient to (correctly) claim a particular language not to be regular; you do not have to prove that fact.

a) $f(L) = L \cup L(\mathbf{0}^*)$; that is, f adds to its argument language L all strings of 0's.

b) $f(L)$ is the language formed from L by changing every 0 to 1 and every 1 to 0 (simultaneously). For instance, if $L = \{001, 10\}$, then $f(L) = \{110, 01\}$.

c) $f(L)$ is the language formed by appending 11 to the end of every string in L. For instance, $f(\{\epsilon, 10, 1\}) = \{11, 1011, 111\}$.

d) $f(L)$ is the language formed from L by throwing away ϵ, and deleting the first positions of all other strings. For instance, $f(\{\epsilon, 010, 110, 1\}) = \{10, \epsilon\}$.