

CS 273, Fall 2006

Exam 2 2 November 2006

INSTRUCTIONS (read carefully)

- Fill in the following information giving name and ID.

NAME:

NETID:

- CLEARLY print your name and ID on every page.
- There are 7 problems, on pages numbered 2 through 9, plus a blank page numbered 10. Make sure you have a complete exam.
- The point value of each problem is indicated next to the problem, and below.
- The exam is designed for slightly over one hour, although you have two hours.
- It is probably wise to glance at all problems and point values before beginning, to best plan your time.
- This is a closed book exam. No notes of any kind are allowed. Do all work in the space provided, using the backs of sheets if necessary. See the proctor if you need more paper.

Problem	Possible	Score
1	9	
2	9	
3	8	
4	8	
5	8	
6	10	
7	8	
Total	60	

Problem 1: True/False (9 points)

Completely write out “True” if the statement is necessarily true. Otherwise, completely write “False”. For example, “ $x + y > x$ ” has the answer “False” assuming that y could be 0 or negative. But “If x and y are natural numbers, then $x + y \geq x$ ” has the answer “True”. You do not need to explain or prove your answers.

1. If a grammar G is in Chomsky-normal form, then each word in $L(G)$ has exactly one parse tree.
False
2. If a language L satisfies the conditions of the pumping lemma for regular languages, then L is regular.
False
3. Let $\Sigma = \{0, 1\}$. Let $L = \{0^n 1^n 0^m 1^m : n, m \geq 0\}$. L is context-free.
True
4. The context-free languages are closed under intersection.
False
5. The language $L = \{a^n a^n : n \geq 0\}$ is not regular.
False
6. The deterministic and non-deterministic versions of PDA’s are equivalent.
False
7. The language $L = \{a^n b^n c^n : a, b, c \geq 0 \text{ and } a, b, c \leq 1000\}$ is context-free.
Broken due to typo: both answers got full credit.
8. Let $\Sigma = \{a, b, c\}$. The language $\{w\#w : w \in \Sigma^*\}$ is context-free.
False
9. Let $\Sigma = \{a, b\}$. The language described by the regular expression a^*b^* is a context-free language.
True

Problem 2: Short answer (9 points)

The following questions require only short answers.

1. Given two context-free grammars G_1 and G_2 , with start symbols S_1 and S_2 , explain precisely how to construct a grammar G such that $L(G) = L(G_1) \circ L(G_2)$. (Recall that $A \circ B$ is the concatenation of the two languages A and B .)

Let $G_1 = (V_1, T_1, S_1, P_1)$ and $G_2 = (V_2, T_2, S_2, P_2)$. We can assume that $V_1 \cap V_2 = \emptyset$ (because otherwise we can relabel all the variables in V_2 to make this condition happen). Now define the new grammar to be $G = (V_1 \cup V_2 \cup \{S\}, T_1 \cup T_2, S, P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\})$. In the new grammar S_1 generates a typical string of $L(G_1)$ and S_2 does the same for $L(G_2)$. Finally $S \rightarrow S_1 S_2$ glues these two strings in order to make a string in $L(G_1) \circ L(G_2)$.

Many people have added the new rule $S \rightarrow S_1 | S_2$. Some people have tried to build a new PDA using the PDA's of languages $L(G_1)$ and $L(G_2)$. (No points is deducted if didn't mention to relabel V_2 .)

2. What is the difference between a Turing recognizable language and a Turing decidable language?

For every Turing Recognizable language L , there is a TM that accepts every string in L and doesn't accepts (which means rejects or loops till infinity) every string not in L .

For every Turing Decidable language L , there is a TM that accepts every string in L and rejects every string not in L .

Many people confuse the language with the machine that accepts it, for example "a decidable language accepts /rejects/loops ..."!!

Some people use insufficient terms to express themselves, like: "a Turing recognizable language can be recognized by a TM while a Turing decidable language can be decided by a TM"!!

3. Let G be the following context-free grammar, where lowercase letters are terminals (i.e. $\Sigma = \{a, b\}$), uppercase letters are variables, S is the start symbol.

$$S \Rightarrow SS \mid (X) \mid \epsilon$$

$$X \Rightarrow XX \mid (X) \mid a \mid b \mid \epsilon$$

Give a precise, non-recursive definition of $L(G)$.

$L(G)$ is the language of all "fully balanced parenthesized expressions consisting of a and b (including trivial string ϵ)" (fully means that every a or b is inside at least one pair of parentheses).

Common Source of Error/Grading:

For the regular exam many people didn't notice that the string must be fully parenthesized (no point is deducted for this, just written warning on the paper).

Problem 3: Grammar construction (8 points)

Let $\Sigma = \{a, b, c\}$. Give a grammar that generates the language $L = \{a^i b^i c : i \geq 0\} \cup \{ab^i c^i : i \geq 0\}$. Carefully explain how your grammar works, and what each nonterminal does. (You don't need to give a formal proof that it does the right thing.)

$$S \rightarrow Ac|aB \quad A \rightarrow aAb|\epsilon \quad B \rightarrow bBc|\epsilon$$

Common Source of Error/Grading:

Some people used something similar to $A \rightarrow aA|Ab|\epsilon$, to generate $a^i b^i$.

Problem 4: PDA Building (8 points)

Let $\Sigma = \{0, 1\}$. Give a PDA for $L = \{x\#y^R : x, y \in \Sigma^* \text{ and } y \text{ is a prefix of } x\}$. Notice that y^R is the reversed version of the string y and that y is a prefix of x if there is some string w such that $x = yw$. Also, x and y can be equal: every string is a prefix of itself.

Present your PDA as a state diagram. Carefully explain how your PDA works, and the meaning of each state and/or transition. (You don't need to give a formal proof that it does the right thing.)

[see separate pdf file]

Problem 5: Pumping lemma (8 points)

Let $\Sigma = \{a, b\}$. Let $L = \{wbbw : w \in \Sigma^*\}$. Prove that L is not regular by filling in the missing parts of the following pumping lemma proof.

Suppose that L were regular. Let p be the constant given by the pumping lemma.

Consider the string $w_p = a^p b b a^p$

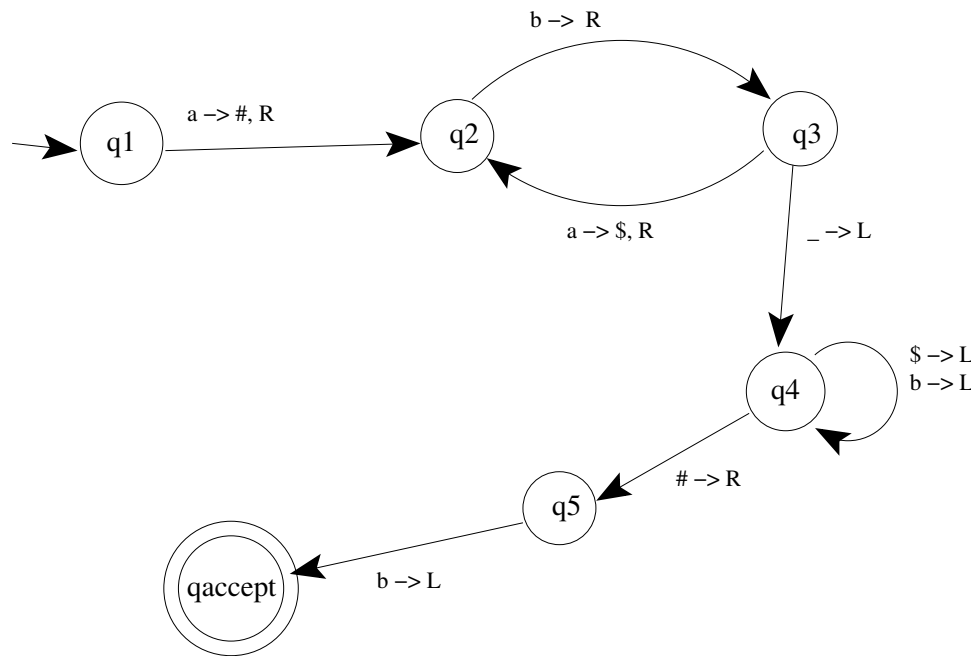
Because $|w_p| \geq p$, there must exist strings x, y , and z such that $w_p = xyz$, $|xy| \leq p$, $|y| > 0$, and $xy^i z \in L$ for every $i \geq 0$. Therefore, x, y, z must be of form $x = a^u$, $y = a^v$, and $z = a^{p-u-v} b b a^p$, for some $u, v \geq 0$, such that $u + v \leq p$ and $v > 0$.

Consider $xy^i z$ when $i = 0$. We have that $xy^0 z = xz = a^u a^{p-u-v} b b a^p = a^{p-v} b b a^p$. Clearly, by definition, $a^{p-v} b b a^p \notin L$, since $v > 0$.

Since $xy^0 z$ isn't in L , we have a contradiction. Therefore, L must not have been regular.

Problem 6: Turing machines (10 points)

Consider the following state diagram for a Turing Machine M . The input alphabet is $\{a, b\}$ and the diagram uses underscore to represent the special blank character.



(a) Give a run on the input string $abab$. That is, show the sequence of configurations starting with q_0abab and ending with an accepting configuration.

$q_1abab, \#q_2bab, \#bq_3ab, \#b\$q_2b, \#b\$bq_3, \#b\$q_4b, \#bq_4\$b, \#q_4b\$b, q_4\#b\$b, \#q_5b\$b, q_{accept}\#b\$b$

(b) What language does M recognize?

$$(ab)^+ = (ab)(ab)^*$$

(c) Describe the pattern of how M moves back-and-forth along the tape.

The tape head moves all the way to the right end of the input, then it moves all the way back left. It then moves right one space and then back left once more.

(d) Where does M leave the tape head when it halts in the accept state?

At the leftmost symbol on the tape, which is the $\#$.

(e) Describe how M modifies the contents of its tape. That is, what is the difference between the original tape contents and the tape contents when the Turing Machine halts in the accept state?

The first a is now a $\#$, and all other as are changed to $\$$. The bs are unchanged.

Problem 7: PDA modification (8 points)

If L is a context-free language and L' is a regular language, then $L \cap L'$ is context-free. To show this, let $M = (Q, \Sigma, \Gamma, \delta_M, q_0, F_M)$ be a PDA recognizing L and let $N = (P, \Sigma, \delta_N, p_0, F_N)$ be a DFA recognizing L' . We then need to construct a PDA M' recognizing $L \cap L'$.

To do this, set $M' = (Q \times P, \Sigma, \Gamma, \delta', (q_0, p_0), F')$.

(a) Express F' in terms of F_M and F_N . (2 points)

$$F_M \times F_N$$

(b) If c is a character in Σ (i.e. c is not ϵ), (q, p) is a pair of states from $Q \times P$, and t is a stack symbol in Γ_ϵ , what is the value of $\delta'((q, p), c, t)$? (4 points)

$$\{((q', p'), d) : (q', d) \in \delta_M(q, c, t) \text{ and } p' \in \delta_N(p, c)\}$$

Half credit was awarded for generally similar answers with significant type errors. Many people gave a single value rather than a set of values. Others assumed that δ_N returned a set rather than a single value. Many forgot that the the output value must include a new stack symbol. Taking the cross product of the two transition functions has two bugs: the DFA's transition function returns a single value (not a set) and this method produces outputs of the form $((q', d), p')$ which doesn't have the right order and internal structure.

(c) If (q, p) is a pair of states from $Q \times P$ and t is a stack symbol in Γ_ϵ , what is the value of $\delta'((q, p), \epsilon, t)$? (2 points)

$$\{((q', p), d) : (q', d) \in \delta_M(q, \epsilon, t)\}$$

Both points were awarded for the idea that the PDA makes a transition while the DFA maintains the same state. Nothing was deducted for mistakes (e.g. type errors) already penalized in part (b).

This was one of the harder questions on the exam, because it really pushed hard on how well you understand tuple notation. So many people good scores overall got a zero or half credit on this problem. Don't let this worry you, just try to understand the correct answer.