

CS 273, Spring 2006

Exam 2 Solutions April 4, 2006

Problem 1: True/False (8 points)

1. A PDA with two stacks can accept the language $\{0^n 1^n 2^n : n \geq 0\}$.

Solution: TRUE. Push the 0s onto both stacks, and use the first to make sure the number of 1s matches the number of 0s, and the second to make sure the number of 2s is the same.

2. For any language L , there are infinitely many different grammars G such that $L(G) = L$.

Solution: TRUE. We could rename non-terminals, add useless non-terminals, etc., without affecting the language generated by the grammar.

3. If L is a CFL and R is a regular language, then $R - L$ is a CFL.

Solution: FALSE. We could choose $R = \Sigma^*$; this would then imply that CFLs are closed under complementation.

4. In the procedure for putting a grammar into Chomsky normal form (as given in class and in the book), you remove ϵ productions before removing unit productions.

Solution: TRUE. If you remove unit productions first, when the ϵ productions are later removed, more unit productions could be added.

5. If some word w in $L(G)$ has two different derivations, then G is ambiguous.

Solution: FALSE. If some w has two different *parse trees*, G is ambiguous.

6. If L is accepted by final state by a PDA with two states, then there is a PDA accepting L by null stack with two states.

Solution: TRUE. If L is accepted by final state by a PDA, it is context-free. All CFLs can be accepted by a single-state PDA accepting by null stack. A second state can then be added (a dummy state, if necessary).

7. If L is not context-free, then L^R is not context free either (where R is the reversal operator).

Solution: TRUE. If L^R were context-free, L would be too since Context Free Languages are closed under reversal.

8. A PDA M is a DPDA if the following two conditions hold:

- (a) for any state q , input symbol a , and stack symbol Z , there is at most one element in $\delta(q, a, Z)$, and
- (b) for any state q and stack symbol Z , there is at most one ϵ -transition.

Solution: FALSE. We also need the condition that, for any state q and stack-symbol Z , there is not both an ϵ -transition and a transition on any input character. (That is, the machine should never have to make a 'choice' between ϵ -transitions and regular transitions.)

Problem 2: Classification (7 points)

For each of the following languages L , answer “REGULAR” if L is necessarily regular, “CFL” if L is necessarily Context-Free but not necessarily regular, or “NEITHER” if L is not necessarily Context-free.

1. $L = \{1^n 1^m 0^n 0^m : n, m \geq 0\}$

Solution: CFL

2. $L = \{1^n 0^n 1^m 0^m : n, m \geq 0\}$

Solution: CFL

3. $L = \{1^n 0^m 1^n 0^m : n, m \geq 0\}$

Solution: Neither.

4. $L = \{x1y : x, y \in \{0, 1\}^n\}$

Solution: CFL

5. $L = \{w \in \{a, b, c\}^* : \text{the number of } a\text{'s in } w \text{ is less than the number of } b\text{'s in } w, \text{ or is less than the number of } c\text{'s in } w.\}$

Solution: CFL

6. $L = \{w \in \{a, b, c\}^* : \text{the number of } a\text{'s in } w \text{ is less than the number of } b\text{'s in } w, \text{ or is less than the number of } c\text{'s in } w, \text{ but not both.}\}$

Solution: Neither

7. $L = \{0^n 1^m 0^m : n + m = 3 \pmod{5}\}$

Solution: CFL

Problem 3: Reversed in parts (8 points)

Give a grammar that generates the language $\{w\#x\#y : w, x, y \in \{0, 1\}^*, xy = w^R\}$.

Solution:

$$S \rightarrow 1S1 \mid 0S0 \mid \mathbf{A}\#$$

$$\mathbf{A} \rightarrow 1\mathbf{A}1 \mid 0\mathbf{A}0 \mid \#$$

It is easy to verify that \mathbf{A} generates strings of the form $x^R\#x$, where $x \in \{0, 1\}^*$; and as such, \mathbf{S} generates strings of the form $y^R x^R \# x \# y$, where $x, y \in \{0, 1\}^*$. This is exactly what we want.

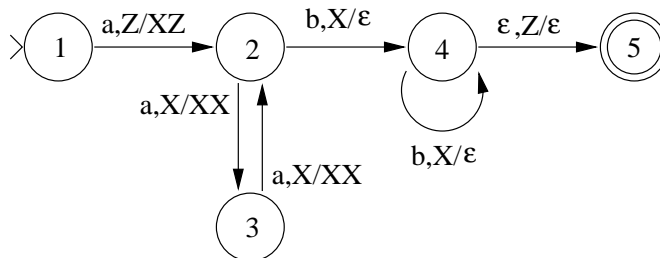
Comments: Most students did pretty well on this problem. Common mistakes include: (i) some solutions cannot generate the string ' $\#\#\#$ ', -2 pts; (ii) some students thought that $\epsilon \in L$, -1 pt; (iii) some students wrote too complicated grammars, -1 pt; (iv) some students used a reverse operator in their solutions; (v) some solutions could not even ensure $|w| = |x| + |y|$. The last two types of errors are serious; students who made them received 0-1 pts.

Problem 4: PDA Building (9 points)

Give a PDA for $L = \{a^{2n+1}b^{2n+1} : n \geq 0\}$ that in addition, meets the following specifications:

1. the stack alphabet is exactly $\{Z, X\}$. Thus, besides the initial stack symbol Z , you are allowed at most one additional stack symbol X .
2. The machine accepts by empty stack and final state.

Solution: One solution is the following:



The first three states ensure that a valid number of a s occur and put an X on the stack for each a . Once the first b is found, we transition to state 4 so that all remaining characters are b . For each b , an X is removed from the stack, which ensures that the number of a s and b s are equal. The last transition removes the initial stack symbol, which empties the stack, and goes to the final state.

The 9 points were divided into 3 parts, with 3 points for ensuring that the number of *as* and *bs* are equal, 3 for ensuring that this number is odd, and 3 for the general format of the PDA. Common ways to lose points from the last part were not marking the start state, not removing the initial symbol *Z* from the stack, or not allowing the number of *as* to be 1.

Problem 5: Find the Language (9 points)

Precisely describe, using correct set and mathematical notation where possible, the language generated by each grammar below.

1. $S \rightarrow aSa \mid bSb \mid aSb \mid bSa \mid a \mid b$ (3 points)

Solution: $L(G) = \{w \mid w \in \{a, b\}^{2k+1}, k \geq 0\}$
 (That is, w is an odd-length string over the alphabet $\{a, b\}$.)

2. $S \rightarrow aaSbb \mid aaaaCbbbb$ (3 points)
 $C \rightarrow cc \mid cCc$

Solution: $L(G) = \{a^{2i}c^{2j}b^{2i} \mid i \geq 2, j \geq 1\}$
 A common mistake was using $i \geq 1, j \geq 0$, or something similar. Half a point was taken off for each such mistake.

3. $S \rightarrow aSa \mid bSb \mid aSb \mid bSa \mid aTb \mid bTa$ (3 points)
 $T \rightarrow aTa \mid bTb \mid aTb \mid bTa \mid a \mid b \mid \epsilon$

Solution: $L(G) = \{w : w \neq w^R\}$ (That is, strings over the alphabet $\{a, b\}$ that are *not* palindromes)

Also acceptable:

$$L(G) = \{xwy \mid x, w, y \in \{a, b\}^*, |x| = |y|, x \neq y^R\}$$

$$L(G) = \{x \cdot c \cdot w \cdot d \cdot y \mid x, y \in \{a, b\}^m, w \in \{a, b\}^*, c, d \in \{a, b\}, c \neq d\}$$

Solutions that were correct, but unnecessarily complex (such as the one immediately above) lost up to half a point.

A few people tried to use the notation for regular expressions, sometimes mixed in with standard set notation. This does not work at all here, because the last two languages are not regular. Partial credit may have been awarded depending on how close the notation came to expressing the correct answer.

Problem 6: CNF Parse Trees (9 points)

Let G be a grammar in Chomsky Normal Form.

Prove by induction on n , that every parse tree for a word $w \in L(G)$ of length n must have $2n - 1$ internal nodes. (Internal nodes are those that are labeled with nonterminals - hence, nodes that are not leaves of the parse tree.)

Solution:

You can't successfully prove this by modifying a parse tree for a string of length $n + 1$ to make a parse tree for a string of length n , or vice versa. (Though partial credit was awarded for these approaches.) We've seen examples of languages which contain only even-length strings. Even when strings exist in consecutive lengths, parse trees for even-length strings can have a totally different structure from those for odd-length strings, as in the following grammar.

- $S \rightarrow AB|C$
- $A \rightarrow aa|AA$
- $B \rightarrow bb|BB$
- $C \rightarrow c|cCc$

Remember that the nodes in a parse tree have to follow the production rules of the grammar, so you can't do random surgery on a parse tree of the sort that you could use on a plain binary tree.

To successfully attack this problem, you must break up the tree at the root node, creating two subtrees which might both be longer than one character.

Grading key: -4 points for the above major problem setting up the induction ("divide at root"). -2 points for not stating an inductive hypothesis or starting inductive step with an inductive assumption. (-1 if present in fuzzy form.) -1 or -2 points for minor bugs, insufficient detail, and the like.

Base: $n = 1$. The parse tree for a word of length 1 consists of exactly two nodes: a variable node generating a terminal node, using a production of the form $A \rightarrow a$. This matches $2n - 1 = 1$. (Because CNF contains no ϵ -productions, strings of length zero are not possible, nor are more complex parse trees for length 1 words.)

Induction: Suppose that for every word of length k , $1 \leq k \leq n$, all parse trees contain $2k - 1$ internal nodes. Let w be a word of length $n + 1$ in $L(G)$ and pick an arbitrary parse tree for w (call it P). Notice that the length of w is at least 2.

Consider the root node R of P . Since the grammar is in CNF, R must either have a single terminal child, or two non-terminal children. If R had a single terminal child, the parse tree would yield a string of length 1. Since we know that $|w| \geq 2$, R must have two non-terminal children.

Let's call R 's two children A and B . Suppose that A yields a string x and B yields a string y . Let k be the length of x . The length of y is then $n + 1 - k$.

Since the grammar contains no ϵ -productions, both x and y have length ≥ 1 . Therefore, since their lengths sum to $n + 1$, both have length $< n + 1$, i.e. $\leq n$. Therefore, by the induction hypothesis, the tree rooted at A contains exactly $2k - 1$ internal nodes and the tree rooted at B contains exactly $2(n + 1 - k) - 1$ internal nodes.

The number of internal nodes in the whole tree R is the root node (one), plus the number in A ($2k - 1$) plus the number in B ($2(n + 1 - k) - 1$). $1 + (2k - 1) + 2(n + 1 - k) - 1 = 2n + 1 = 2(n + 1) - 1$.