Submission number: 80229
Submission certificate: CD742165
Submission time: 2014-04-21 18:22:39 PST (GMT - 8:00)

Number of questions: 5
Positive points per question: 4.0
Negative points per question: 0.0
Your score: 20

1. Here is a grammar, whose variables and terminals are NOT named using the usual convention. Any of *R* through *Z* could be either a variable or terminal; it is your job to figure out which is which, and which could be the start symbol.

```
R → ST | UV
T → UV | W
V → XY | Z
X → YZ | T
```

We do have an important clue: There are no useless productions in this grammar; that is, each production is used in some derivation of some terminal string from the start symbol. Your job is to figure out which letters definitely represent variables, which definitely represent terminals, which could represent either a terminal or a nonterminal, and which could be the start symbol. Remember that the usual convention, which might imply that all these letters stand for either terminals or variables, does not apply here.

a) *U* must be a variable.

b) *S* must be a terminal.

c) *T* must be the start symbol.

d) *X* must be the start symbol.

Answer submitted: **b)**

You have answered the question correctly.

Question Explanation:

All the symbols appearing on the left side of some production must be variables. Thus, *R, T, V*, and *X* are definitely variables. But if any other symbol was a variable, then it could not derive any terminal string, since it has no productions. Since all the other symbols appear in the body of some production, we can be sure they are terminals, because if they were variables they would render that production useless. Since we are told there are no useless productions, we can be sure *S, U, W, Y*, and *Z* are terminals.

We can also deduce that *R* is the start symbol. Notice that *R* appears in no production body. If *R* were not the start symbol, then its productions would be useless. Again, we are told there are no useless productions, so we must conclude *R*, and only *R*, could be the start symbol.

2. G_1 is a context-free grammar with start symbol S_1, and no other nonterminals whose name begins with "S." Similarly, G_2 is a context-free grammar with start symbol S_2, and no other nonterminals whose name begins with "S." S_1 and S_2 appear on the right side of no productions. Also, no nonterminal appears in both G_1 and G_2.

We wish to combine the symbols and productions of G_1 and G_2 to form a new grammar G, whose language is the union of the languages of G_1 and G_2. The start symbol of G will be S. All productions and symbols of G_1 and G_2 will be symbols and productions of G. Which of the following sets of productions, added to those of G, is guaranteed to make L(G) be $L(G_1)$ [union] $L(G_2)$?

a) $S \rightarrow S_1S_2 \mid S_2S_1$

b) $S \rightarrow S_1S_3,\ S_3 \rightarrow S_2 \mid \varepsilon$

c) $S \rightarrow S_1,\ S_1 \rightarrow S_2,\ S_2 \rightarrow \varepsilon$

d) $S \rightarrow S_1 \mid S_3S_2,\ S_3 \rightarrow \varepsilon$

Answer submitted: **d)**

You have answered the question correctly.

Question Explanation:

Each of the choices involves only S, S_1, S_2, and in some cases other nonterminals whose names begin with "S" and that therefore are known not to appear in G_1 or G_2. As a result, we need only to look at the strings involving S_1 and S_2 only, that are derivable from S, using the new productions. In order for L(G) to be $L(G_1)$ [union] $L(G_2)$, it is necessary and sufficient that the strings involving only symbola S_1 and S_2 that are derived from S using only the additional productions be exactly the two strings $\{S_1, S_2\}$.

For example, adding $S \rightarrow S_1 \mid S_2$ obviously has this property. So does the set of productions

$S \rightarrow S_1$
$S_1 \rightarrow S_2$

Note that if S_1 could appear on the right side of productions of G_1, then this choice would not work --- derivations of G_2 could suddently appear in the middle of derivations that should be in G_1 only.

$S \rightarrow S_1 \mid S_3$
$S_3 \rightarrow S_2$

3. A *unit pair* (X,Y) for a context-free grammar is a pair where:

1. X and Y are variables (nonterminals) of the grammar.
2. There is a derivation X =>* Y that uses only unit productions (productions with a body that consists of exactly one occurrence of some variable, and nothing else).

For the following grammar:

```
S → A | B | 2
A → C0 | D
B → C1 | E
```

```
C → D | E | 3
D → E0 | S
E → D1 | S
```

Identify all the unit pairs. Then, select from the list below the pair that is NOT a unit pair.

a) (A,A)

b) (S,S)

c) (E,C)

d) (C,B)

Answer submitted: **c)**

You have answered the question correctly.

Question Explanation:

The cycle of unit-productions S → A → D → S says that any pair involving only S, A, and D is a unit pair. Similarly, the cycle S → B → E → S tells us that any pair involving S, B, and E is a unit pair. But since S is on both cycles, there are derivations involving only unit productions from any of S, A, B, D, and E to any of these five variables. Thus, any combination of these five variables is a unit pair.

Further, the productions C → D | E tells us that any pair (C,X) is a unit pair if X is any of the six variables. The only pairs that are not unit pairs are (X,C) for X = S, A, B, D, or E.

4. The intersection of two CFL's need not be a CFL. Identify in the list below a pair of CFL's such that their intersection is not a CFL.

a) $L_1=\{aca^nb^jc^ic \mid n>0, i>0, j>0\}$
$L_2=\{aca^ja^nb^ic^i \mid n>0, i>0, j>0\}$

b) $L_1=\{aba^nb^nc^i \mid n>0, i>0\}$
$L_2=\{aba^nb^jc^i \mid n>0, i>0, j>0\}$

c) $L_1=\{aba^nb^nc^iba \mid n>0, i>0\}$
$L_2=\{aba^nb^ic^iba \mid n>0, i>0\}$

d) $L_1=\{aba^nb^nc^nba \mid n>0, i>0\}$
$L_2=\{aba^nb^ic^jba \mid n>0, i>0, j>0\}$

Answer submitted: **c)**

You have answered the question correctly.

Question Explanation:

The incorrect choices fall in two categories: either one of the two given languages is not a CFL, or one is a CFL and the other is a regular language, in which case we know that their intersection is a CFL (see Section 7.3.4 on p. 291).

For the four correct choices the intersections of the two given languages are:

1. $\{a^nb^nc^n \mid n> 0\}$.
2. $\{aba^nb^nc^n \mid n> 0\}$.
3. $\{aba^nb^nc^nba \mid n> 0\}$.
4. $\{a^nb^nc^n \mid n> 0\}$.

In all cases we can prove the language not to be context-free by using the pumping lemma on a word $a^n b^n c^n$ for sufficiently large n. If we pump any pair of strings (of length much smaller than n) then the balance on the number of a's, b's and c's will be ruined.

5. A *linear grammar* is a context-free grammar in which no production body has more than one occurrence of one variable. For example, $A \rightarrow 0B1$ or $A \rightarrow 001$ could be productions of a linear grammar, but $A \rightarrow BB$ or $A \rightarrow A0B$ could not. A *linear language* is a language that has at least one linear grammar.

The following statement is false: "The concatenation of two linear languages is a linear language." To prove it we use a counterexample: We give two linear languages L_1 and L_2 and show that their concatenation is not a linear language. Which of the following can serve as a counterexample?

a) $L_1 = \{w|w=(aaa)^n(ab)^n$, where n is a positive integer$\}$
$L_2=\{w|w=(ab)^n b^n a^n$, where n is a positive integer$\}$

b) $L_1 = \{w|w=aaac^{n-1}(ab)$, where n is a positive integer$\}$
$L_2=\{w|w=c(aaa)^n$, where n is a positive integer$\}$

c) $L_1 = \{w|w=(aa)^n a^n b^n$, where n is a positive integer$\}$
$L_2=\{w|w=c(aaa)^n(ab)^n$, where n is a positive integer$\}$

d) $L_1 = \{w|w=(aaa)^n(ab)^n$, where n is a positive integer$\}$
$L_2=\{w|w=(ab)^n$, where n is a positive integer$\}$

Answer submitted: **c)**

You have answered the question correctly.

Question Explanation:

The correct answer should have all properties:

1. L_1 and L_2 are both linear languages.
2. Their concatenation is provably not a linear language.

To prove the first property we need a linear grammar that generates the language.

To prove the second property, there is a variant of the pumping lemma (Section 7.2, p.279) that applies only to linear languages. For every linear language there is a constant *n* such that if *z* is in the language and of length at least *n*, then we can write $z = uvwxy$ such that $|uy| \leq n$, $|w| \leq n$, and for all *i*, uv^iwx^iy is in the language. Note that unlike the general pumping lemma for CFL's, here it is *v* and/or *x* that are allowed to be long, and the other components must be short. Can you prove this lemma, making use of the fact that for a linear grammar, the path leading to *w* in Fig. 7.6 (p. 282) must have all the variables in the entire parse tree?

However, in order to show that a choice is incorrect, we need to prove either that one of the languages is not linear or that their concatenation is linear.