

#5. (a) Use induction. Let  $P(n)$  be the proposition:  
 "If  $L_1, \dots, L_n$  are regular, then so is  $\bigcup_{i=1}^n L_i$ ".  
 Then  $P(2)$  is true by the Closure theorem.

Now suppose  $P(k)$  is true.

Let  $L_1, \dots, L_k, L_{k+1}$  be any  $k+1$  regular languages. Then

$$\bigcup_{i=1}^{k+1} L_i = \underbrace{\left( \bigcup_{i=1}^k L_i \right)}_{\substack{\text{reg.} \\ \text{by ind. hyp.}}} \cup \underbrace{L_{k+1}}_{\text{reg.}} \text{ is regular.}$$

So  $P(k) \Rightarrow P(k+1)$ . Hence  $P(k)$  is true for all  $k \geq 2$ . So the results follow.

(b) The situation for intersection is entirely similar.

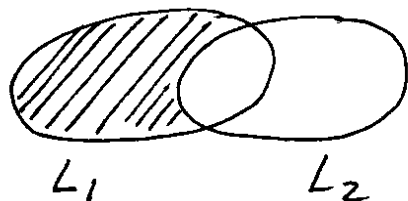
#6.  $S_1 \ominus S_2 = (S_1 - S_2) \cup (S_2 - S_1)$

Now if  $S_1$  &  $S_2$  are regular, then  $S_1 \ominus S_2$  will also be regular because  $(S_1 - S_2)$  &  $(S_2 - S_1)$  will be regular by the Closure theorem and so  $(S_1 - S_2) \cup (S_2 - S_1)$  will then be regular by the Closure theorem. Hence reg. lang. are closed under symmetric differences.

#7 Hint:  $\text{nor}(L_1, L_2) = \bar{L}_1 \cap \bar{L}_2$ .  
 Now use the Closure theorem.

#12 Yes. First observe that  $L_2 = (L_1 \cup L_2) - (L_1 - L_2)$   
 Now  $L_1 - L_2$  is finite, so  $L_1 - L_2$  is regular. And  $L_1 \cup L_2$  is given as regular.  $\therefore (L_1 \cup L_2) - (L_1 - L_2)$  is reg. by Clos. Thm.

#12 Hence  $L_2$  is regular.



$$L_2 = (L_1 \cup L_2) - (L_1 - L_2)$$

$L_1 - L_2 \subseteq L_1$ . So  
 $L_1 - L_2$  is finite.

#13 Let  $\Sigma$  be the alphabet on which  $L$  is based. Then

$$\begin{aligned} L_1 &= \{uv : u \in L \text{ \& } |v| = 2\} \\ &= L \cdot \Sigma \cdot \Sigma \end{aligned}$$

Now any alphabet is finite, so  $\Sigma$  is regular  
 Hence  $L \cdot \Sigma \cdot \Sigma$  is regular by the Closure Theorem. So  $L_1$  is regular

#14  $\{uv : u \in L, v \in L^R\} = L \cdot (L^R)$

Now use the 'closure theorem

#15. No. Let  $L_1 = \{a^n : n \geq 0\}$  and  $L_2 = \{a^p : p \text{ is prime}\}$ .

$$\begin{aligned} \text{Then } L_1 \cdot L_2 &= \{a^n : n \geq 0\} \cdot \{a^p : p \text{ is prime}\} \\ &= \{a^{n+2} : n \geq 0\} = \underline{a} \underline{a} \underline{a}^* \end{aligned}$$

So  $L_1$  &  $L_1 \cdot L_2$  are both regular. It will be show later that  $L_2$  is non-regular by using various means. (See supplementary problems also)

#26. See class notes.

- #1  $L_1 \subseteq L_2 \Leftrightarrow L_1 - L_2 = \emptyset$ . Since  $L_1$  &  $L_2$  are regular we can find dfa's  $M_1$  &  $M_2$  for  $L_1$  &  $L_2$ . Using  $M_1$  &  $M_2$  we can algorithmically get a dfa  $M$  for  $L_1 - L_2 = (\bar{L}_1 \cup L_2)^c$ . Now  $L_1 - L_2 = \emptyset \Leftrightarrow L(M) = \emptyset \Leftrightarrow$  there is no path from the initial state of  $M$  to an accepting state of  $M$  (and this can be checked algorithmically)
- #2 Since  $L$  is regular, we can find a dfa  $M$  such that  $L(M) = L$ . Now  $\lambda \in L \Leftrightarrow q_0 \in F(M)$  (i.e., if the initial state of  $M$  is an accepting state also). Since this can be algorithmically checked, there is an algorithm to tell if  $\lambda \in L$ .
- #5. Since  $L$  is reg., we can find a dfa  $M$  such that  $L(M) = L$ . Let  $M^R$  be the machine obtained by making all accepting states in  $M$  into initial states of  $M^R$  & all initial states in  $M$  into accepting states of  $M^R$ . Then it can be algorithmically checked by Thm 4.7 if  $L(M) = L(M^R)$ .  $L$  is palindromic  $\Leftrightarrow L = L^R \Leftrightarrow L(M) = L(M^R)$ . Oh! You also have to reverse the arrows in  $M$  to get  $M^R$ .
- #6. Make the machine  $M^R$  as above. Then  $L$  has a string  $w$  such that  $w^R \in L \Leftrightarrow w \in L(M) \& w^R \in L(M) \Leftrightarrow w \in L(M) \& w \in L(M^R) \Leftrightarrow w \in L(M) \cap L(M^R)$ . So check algorithmically if  $L(M) \cap L(M^R) \neq \emptyset$ .

#1 Hint: The idea is to look for the last occurrence of a repetition, in an accepting sequence. Following the proof on p. 119, look at the sequence

$$q_0, q_i, q_j, \dots, q_f.$$

At least one state must be repeated, and such a repetition must start no later than the  $n$ -th move from the end. From this the proof will follow.

#3(a) Let  $L = \{w \in \{a,b\}^* : n_a(w) = n_b(w)\}$ . Then  $L$  is infinite. Supp.  $L$  is regular. Let  $m$  be as in the Pumping Lemma. Choose  $u = a^m b^m$ . Then  $u = xyz$  as in the Lemma.

$|xy| \leq m \Rightarrow y$  consists only of  $a$ 's

$n_a(xy^2z) > m$  but  $n_b(xy^2z) = m$   
 $\therefore xy^2z \notin L$ . But  $xy^n z \in L$  for all  $n \geq 0$  by the lemma. Hence we have a contradiction. So  $L$  cannot be regular.

(b)  $L^* = L$ . So  $L^*$  is also non-regular.

#4 (a) Use the Pumping Lemma with  $u = a^m b^i a^{m+1}$

(b) If  $L$  is regular, so is  $\bar{L}$ . And if  $\bar{L}$  is regular so is  $\bar{L} \cap a^* b^* a^* = \{a^n b^l a^k : k = n+l\} = L_a$ . But  $L_a$  is not reg. by (a). So  $L$  is not regular.

#4 (c) Use the Pumping Lemma with  $\mu = a^m b^m a^m$

(d) Use the Pumping Lemma with  $\mu = a^m b^m$

(e) If  $L$  is regular, then  $\bar{L}$  will be regular. But  $\bar{L}$  is non-regular from Prob. #3. So  $L$  is not regular.

(f) Using the Pumping Lemma with  $\mu = a^m b a^m b$  will give you the result.

5. (a)  $L = \{a^p : p \text{ is prime}\}$       primes =  $\{2, 3, 5, 7, 11, \dots\}$

Apply the Pumping Lemma with  $\mu = a^p$  where  $p$  is a prime  $\geq m$ . Then

$$\begin{aligned} \text{So } \mu &= a^{|x|} \cdot a^{|y|} \cdot a^{|z|} \\ &= a^{|x|+|z|} \cdot a^{|y|} \end{aligned}$$

$\mu = xyz$  as in the PL

Now by the P.L.  $x y^n z = a^{|x|+|z|} \cdot a^{n|y|} \in L$

So

$(|x|+|z|) + k|y|$  will always be prime.

Let  $\alpha = |x|+|z|$  &  $\beta = |y|$ . Then  $\alpha + k\beta$  will always be prime. But

if  $\alpha = 0$  or  $1$ , we get a contradiction with  $k=0$ .  
And if  $\alpha \geq 2$ , we get a contradiction with  $k=\alpha$ ,  
because  $\alpha + \alpha\beta = \underbrace{\alpha}_{\geq 2} \cdot \underbrace{(1+\beta)}_{\geq 2}$  which is not prime

$\therefore L$  is not regular.

# 5(b)  $\bar{L}$  is not regular by 5(a). So  $L$  is not regular also

(c) Apply P.L. with  $\mu = a^{(m^2)}$

(d) Apply P.L. with  $\mu = 2^m$

# 14 False. Let  $L_1 = \{a^n b^n : n \geq 0\}$  and  $L_2 = \{a, b\}^* - L_1$ . Then  $L_1$  and  $L_2$  are both non-regular but  $L_1 \cup L_2 = (a \cup b)^*$  is regular.

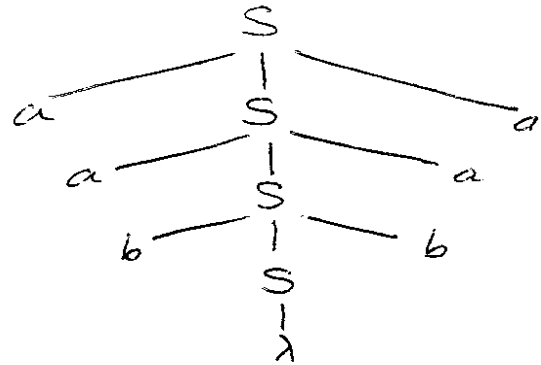
# 17 Yes.  $L = L_1 \cap (L_2^R)$ . Now use Closure Theorem.

# 21 Let  $L_n = \{a^n b^n\}$ . Then  $L_n$  is reg. for each  $n \geq 0$ . But  $L = \bigcup_{n=0}^{\infty} L_n = \{a^i b^i : i \geq 0\}$  is not regular.

# 23 No. Let  $L = \{a, b\}^* - \{a^n b^n : n \geq 0\}$ . Then  $L$  is a non-regular language. Let  $L_i = \{a, b\}^* - \{a^i b^i\}$  for  $i \geq 0$ . Then each  $L_i$  is regular but  $\bigcap_{n=0}^{\infty} L_i = L$  which is non-regular.

# 24 No. Let  $L_1 = \{a, b\}^*$  &  $L_2 = \{a^n b^n : n \geq 1\}$ . Then  $L_1$  is reg. &  $L_1 \cup L_2 = \{a, b\}^*$  is also regular. But  $L_2$  is not regular. (See #11 Sec. 4.1)

#2 The derivation tree for  $aabbaa$  is given on the right



#7 (a)  $S \rightarrow ASb | AAA$ ,  $A \rightarrow a | \lambda$

(b)  $S \rightarrow A | B$ ,  $A \rightarrow aAb | aA | \lambda$ ,  $B \rightarrow aBb | Bb | \lambda$   
 $\{a^n b^m : n \geq m\}$ ,  $\{a^n b^m : n \leq m-2\}$

(c)  $S \rightarrow A | Bb$ ,  $A \rightarrow aaAb | aA | a$ ,  $B \rightarrow DDBb | D$ ,  $D \rightarrow a | \lambda$   
 $\{a^n b^m : n \geq 2m+1\}$ ,  $\{a^n b^m : n \leq 2(m-1)+1\}$

(d)  $S \rightarrow aSbbB | \lambda$ ,  $B \rightarrow b | \lambda$ .

#8 (a)  $S \rightarrow A | B$ ,  $A \rightarrow Ac | D$ ,  $D \rightarrow aDb | \lambda$ ,  $n=m$   
 $B \rightarrow aB | E$ ,  $E \rightarrow GEc | \lambda$ ,  $G \rightarrow b | \lambda$ ,  $m \leq k$

(b)  $S \rightarrow A | B$ ,  $A \rightarrow Ac | D$ ,  $D \rightarrow aDb | \lambda$ ,  $n=m$   
 $B \rightarrow aB | E$ ,  $E \rightarrow bEc | F | G$ ,  $F \rightarrow bF | b$ ,  $m > k$   
 $G \rightarrow Gc | c$ ,  $m < k$

(c)  $S \rightarrow aSc | T$ ,  $T \rightarrow bTc | \lambda$

(d)  $S \rightarrow aSc | T$ ,  $T \rightarrow bTcc | \lambda$

- #13 (a)  $L^2$  is generated by  $S \rightarrow AA, A \rightarrow aAb/\lambda$   
 (b)  $L^k$  is generated by  $S \rightarrow \underbrace{AA \dots A}_{k \text{ times}}, A \rightarrow aAb/\lambda$   
 (c)  $L^*$  is generated by  $S \rightarrow AS/\lambda, A \rightarrow aAb/\lambda$

#20 Any derivation of  $aa.bbabb.a$  would have to start with

$$S \Rightarrow aaB \Rightarrow aaAa \Rightarrow aabBba \\ \Rightarrow aabAaba$$

and this can never lead to  $aa.bbabb.a$

#22  $S \rightarrow [S] \mid (S) \mid SS \mid \lambda$

#23  $S \rightarrow \underline{\lambda} \mid \underline{\phi} \mid \underline{a} \mid \underline{b} \mid (S+S) \mid (S.S) \mid (S^*)$

#24 Let  $V = \{X, Y\}$  and  $T = \{A, B, C, a, b, \dot{\rightarrow}\}$   
 The productions are given below:  
 $S \rightarrow X \dot{\rightarrow} Y, X \rightarrow A/B/C, Y \rightarrow A/B/C/a/b/YY$

" $\dot{\rightarrow}$ " denotes arrows from CFG's  
 $\rightarrow$  denotes arrows from our grammar.

#25 Hint: Just keep applying the appropriate productions to get the rightmost needed letter to get the rightmost derivation. Same thing for the leftmost derivation except you look for leftmost needed letter.



#1  $S \rightarrow aA/b, A \rightarrow aB, B \rightarrow aB/b$

#2  $S \rightarrow aA, A \rightarrow aAB/b, B \rightarrow b$

#3 Hint: If  $G$  is an s-grammar, then each string  $\varphi$  in  $L(G)$  will have a unique leftmost derivation.

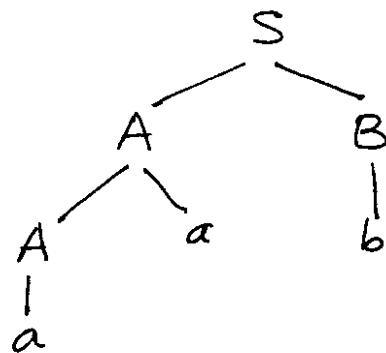
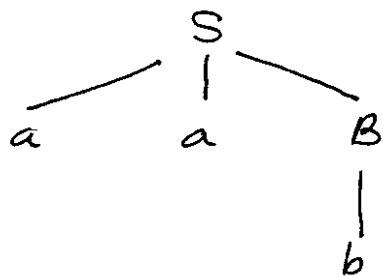
#5  $A \rightarrow \underbrace{ax}, \quad x \in V^*$   
           $|V| \text{ choices} \quad |T| \text{ choices}$

Maximum size of  $|P| = |V| \cdot |T|$

#6 The string  $aab$  has two left-most derivations. So the grammar is ambiguous.

1.  $S \Rightarrow aAB \Rightarrow aab$

2.  $S \Rightarrow AB \Rightarrow AaB \Rightarrow aab \Rightarrow aab$



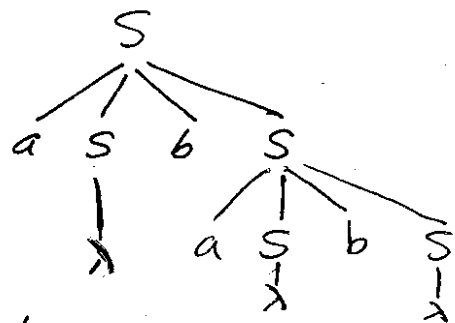
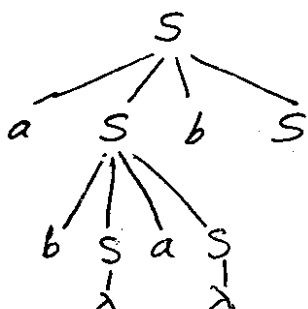
These are the two different derivation trees.

9. If  $L$  is a regular language then we can find a DFA which accepts  $L$ . Now if we convert this DFA into a RLG there will never be a choice of productions because the DFA was deterministic. So we will get an unambiguous RLG for  $L$ . This means  $L$  is not inherently ambiguous.

11. Yes. Let  $G$  be the grammar with productions  $S \rightarrow aA, S \rightarrow ab, A \rightarrow b$ . Then  $ab$  has two leftmost derivations in  $G$ . So  $G$  is ambiguous.

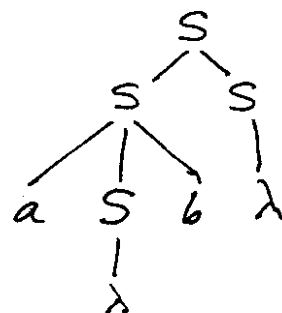
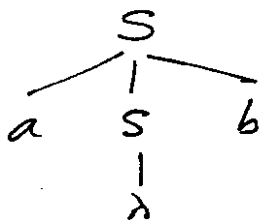
$G: S \rightarrow aSbS / bSaS / \lambda$

12(a) Consider the string  $w = abab$



We have two derivation trees. So grammar is ambiguous.

(b) Consider  $w = ab$ .



$G: S \rightarrow aSb / SS / \lambda$

$\therefore G$  is ambiguous