Amit Nijjar
A11489111

Pabasara Jayasena
A97059807

5/15/17

CSE 105 HW 5

Warmup: Given a regular language A with DFA M = {Q, Σ, δ, {q0}, F}, construct a DFA M' that recognizes A^R. M' = {Q ∪ {q0'}, Σ, δ', q0', {q0}} where:

1) δ' (q0' , ϵ) = F
2) δ' (q0' , a) = ∅ all a ∈ Σ
3) δ' (p, a) = {q | δ(q, a) = p} all q ∈ Q, a ∈ Σ

1. The empty string is the same thing as the empty string reversed
2. The null string is the same thing reversed
3. Every transition is reversed

Problem 1:

A. CFG D = ({S,T}; {0, 1}; R; S) with R = {S -> 0 | 1 | S | T, T -> 01}
    a. This grammar describes the language because it allows for any number of "0"'s and "1"'s but will always end with a "01"
B. D^R = {w^R | w ∈ D}
    a. CFG D' = ({A,B}; {0, 1}; R; A) with R = {A -> 10B; B -> 0 | 1 | ϵ | B}
        i. This grammar describes the language D^R because it the language of D starts with (0 U 1)* and must end with 01 and the CFG D' starts with 10 followed by (0 U 1)*
C. Given a CFL A with CFG M = (V,Σ,R,S), construct a CFG M' = (V,Σ,R',S') where:
    a. If ϵ ∈ M, then ϵ ∈ M'
        i. The empty string is the empty string reversed
    b. For every V -> Σ in M where V is a variable and Σ is a string of terminals and variables, add V -> Σ^R to M' where Σ^R is the reversal of Σ.
        i. M generates a word x if and only if M' generates a word x^R

Problem 2:

To prove that this grammar is ambiguous, we need to show that there is a string that has at least two different leftmost derivations.

String: “if bool then if bool then assnt else assnt”

First:

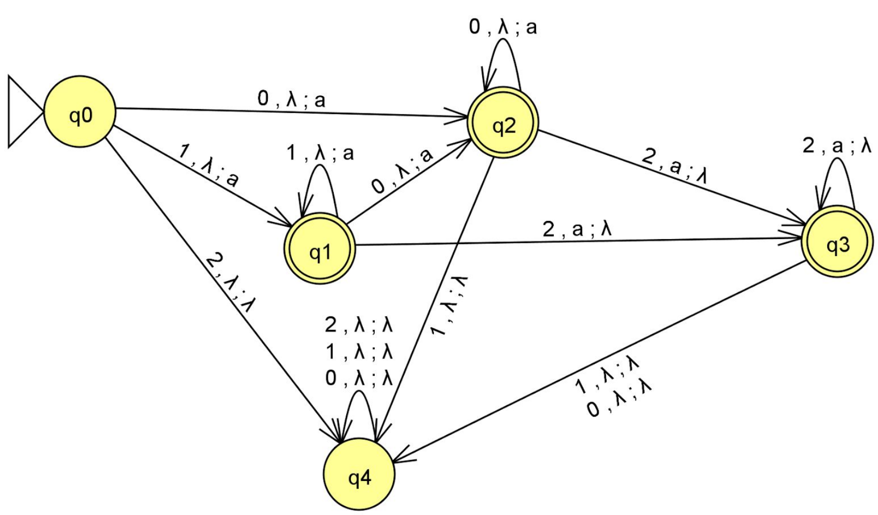
1) S -> C
    a) If bool then S
2) S -> B
    a) If bool then if bool then S else S
3) S -> A
    a) If bool then if bool then assnt else S
4) S -> A
    a) If bool then if bool then assnt else assnt

Second:

1) S -> B
    a) If bool then S else S
2) S -> C
    a) If bool then if bool then S else S
3) S -> A
    a) If bool then if bool then assnt else S
4) S -> A
    a) If bool then if bool then assnt else assnt

These two generated strings have two different leftmost derivations but both derive the same string.

Problem 3:



This PDA ensures the number of 2's is always less than or equal to the number of 0s and 1s together and that the 1s occur first, then 0s, then 2s. q4 is used as a dead state for the case that a two is read first in the input as a '2' cannot be the only character in a valid input as even if there were no 1's or 2's there a 2 is not valid since k would be greater than n+m. Similarly q2 functions like this, except it allows a transition the the accepting state if a '2' follows a '0' acting as the case where there is no 1s. Finally, q1 shows the path if there were 1s followed by 0s and then a 2.

Problem 4:

A. A string of length 3 that is accepted by M is: "ba_".
    a. This string is accepted by M because the machine moves from state q0 to q1 when reading b, stays in q1 from q1 when reading a, and moves to q1 to the accept state q_acc when reading the blank character
B. A string of length 3 that is not accepted by M is: "bbb"
    a. The string is not accepted by M because the machine moves from state q0 to q1 when reading b, moves from q1 to the reject state q_rej when reading in b, and stays in the reject state when reading in the last b
C. M halts on a state given all possible input strings. All of the paths move right on the tape. The machine either ends on an accept state or a reject state. Therefore, M is a decider.
D. The language of M is regular. The language of M is {w | w begins with "ba*_" followed by (a U b U _)*} The language of M can be written in set notation and can be represented by a regular expression. The language is context free because it can be represented with context free grammar.
    a. A DFA representing M is: ({q0, q1, q2, q3}, {a, b, _} , δ, q0, {q2}) with δ =
        i. (q0, a) -> q3
        ii. (q0, b) -> q1
        iii. (q0, _) -> q3
        iv. (q1, a) -> q1
        v. (q1, b) -> q3
        vi. (q1, _) -> q2
        vii. (q2, a) -> q2
        viii. (q2, b) -> q2
        ix. (q2, _) -> q2
        x. (q3, a) -> q3
        xi. (q3, b) -> q3
        xii. (q3, _) -> q3
    b. A CFG representing M is: ({A, B, C, D},{a, b, _}, R, A) with R = {A -> bB, B -> a | B | C, C -> _D, D -> a | b | _ | D }