COMP 3540 THEORY OF COMPUTATION -FALL 2020

NAME: KEERTHANA MADHAVAN

ASSIGNMENT 3

STUDENT NO: [redacted]

"Students of the University of Windsor pursue all endeavours with honour and integrity, and will not tolerate or engage in academic or personal dishonesty".

1) a) Given a DFA A of Language L, is L empty? The input to the algorithm is A = {Q, Σ, δ, q0, F}.

(DFA is a directed Graph)

Let's take Graph A, the language L is accepted by the DFA A is empty if none of the final states in A is reachable from the starting state q_0 to A. We we run Breadth First search algorithm on graph A, we get a list of all the vertices that are reachable at q_0. So if any final state, If yes then DFA A does not accept empty language, else it accepts empty language.

```
function checkEmpty {Q, Σ, δ, q0, F}
{  // we need to traverse through list starting at q0
   on function δ

   if (δ(q0, Σ) is in F)
   {  return "yes"
   }
   else
   {  nextState = δ(q0, Σ);
      checkEmpty { Q, Σ, δ, nextState, F);
   }
   return "No"
}
```

Since we need to check each vertices in List F, and runtime for BFS is $O(n^2)$ and the length of the list is n.
The time complexity is $O(n^2)$.

1.b) Given two DFAs, A and B, are the languages of A and B equivalent. The inputs to the algorithm are A and B.
We can use Ind

Input: DFA A and DFA B.

// A - Start States := Q [$q_0 \ldots n$]

B - Start States := R [$r_0 \ldots m$]

Product - Start States := $Q \times R$ [$(q_0, R_0) \to (n, m)$]

// theata δ A and δ B are transtion functions for DFA A and DFA B.

Transition states δ $([q,r], a) = [\delta A(q,a), \delta B(r,a)]$

Basis: if (q_0 is accepted and R_0 is not accepting state)

{

$q_0 \neq R_0$

Induction: If ($\delta(K, a) = r$ and $\delta(q,a) = s$) // p & q are distinguishable states.

{

$p \neq q$

}

Output: if($q_0 \neq r_0$)

{

return "Not Equivalent"

}

else

{ return "Equivalent"

}

Time Complexity is (On^2) because the algorith checks for all pairs of states in each induction to see if there are any distinguishable pairs. So each induction is like $O(n^2)$. Therefore, the time complexity of this algorithm is $O(n^2)$.

Q2. Prove that following languages are not regular. (Pumping lema).

a.) L = the language of balanced parentheses.
(Direct)

Let's assume L is regular, then by using pumping lemma of length n (pump), we can have word w in L of legth n. So $w = xyz$ and it must satisfy the following

i) $|xy| \leq n$

ii) $y > 0$

iii) $xy^k z \in L$ for all $k \geq 0$

With word w, we have left parenthesis n and right parenthesis n. Based on condition ii, we know that y is not empty. So string $xy^k z$ for all $k > 0$ must have more left parenthesis than right.
Therefore, the third condition is failed. because the string is unbalanced.

Therefore we prove that L is not regular.

2 b.) $L = \{ a^n b^{2n} \mid n \geq 1 \}$

(Contradiction)

Proof:

Let's assume L is regular.

let l be the pumping length. For every string w in L of length $\geq n$. we can have $w = xyz$ such that

i) $|xy| \leq l$

ii) $|y| > 0$

iii) For all $k \geq 0$, $xy^k z$ is in L

Now let w be in the string $a^l b^{2l}$. so $w = xyz$ by pumping lemma. We also know $|xy| \leq l$ and $|y| > 0$. so, y contains only zeros and it is also not empty.

Now let's assume the prime of w, $w' = xyyz$, so y will have more zeros compared to w but with same number of ones.

Thus, there is a contradiction. L is not a regular language.

3.) Given the following DPA.

a.) Draw the table of distinguishable/indistinguishable states.

H is a Dead state, so we can discard it.

We can use equivalance theorem to get the states.

$P_0 = \{A, B, C, E, F, G\}\{D\}$

$P_1 = \{A, B, F, G\}\ \{C, E\}\ \{D\}$

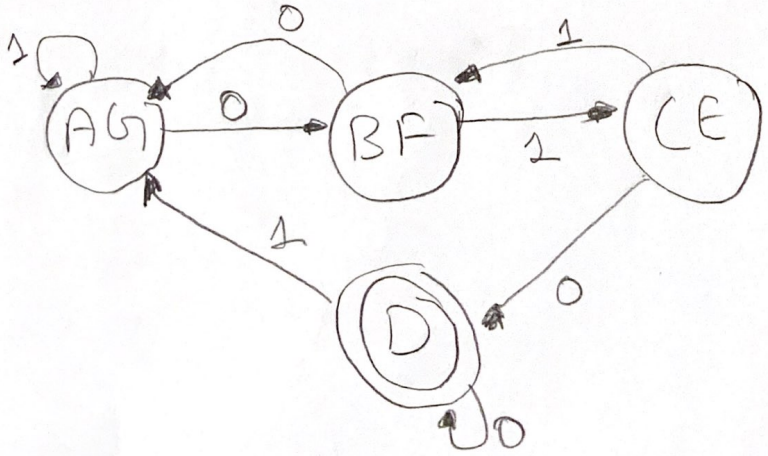
$P_2 = \{A, G\}\ \{B, F\}\ \{C, E\}\ \{D\}$

$P_3 = \{A, G\}\ \{B, F\}\ \{C, E\}\ \{D\}$

Table:

	0	1
→ AG	BF	AG
BF	AG	CE
CE	D	BF
* D	D	AG

b.) Construct the minimum-state equivalent DFA.



Q4.) Suppose that h is a homomorphism from Alphabet $\{0,1,2\}$ to alphabet $\{a,b\}$ defined by $h(0) = a$, $h(1) = ab$, and $h(2) = ba$.

a.) What is $h(0120)$ and $h(21120)$

i) $h(0120) = aabbaa$

ii) $h(21120) = baababbaa$

	0	1
→A	B	A
B	A	C
C	D	B
*D	D	A
E	D	F
F	G	E
G	F	G
H	G	D

b.) If L is the language $L(01^*2)$, what is $h(L)$?

$h(L) = \{a\,(ab)^*\,ba\}$

c.) If L is the language ~~$L(01^*2)$~~ $L(01+12)$, what is $h(L)$?

$h(L) = \{a + abba\}$

d.) If L is the language $\{ababa\}$, what is $h^{-1}(L)$?

$h^{-1}(L) = \{022, 102, 110\}$

e.) If L is the langage $L(acba)^*$, what is $h^{-1}(L)$?

$h^{-1}(L) = \{02^*\}$

5.) Suppose that L, M and N are regular languages

a) Prove (formally) that $\Sigma^* - [(\emptyset \cup L^R)^* \cap (\overline{NM^*L})]^*$ is regular.

Let's prove $\Sigma^* - [(\emptyset \cup L^R)^* \cap (\overline{NM^*L})]^*$ is a regular language:

Prove $(\emptyset \cup L^R)$

We need to prove $\emptyset \cup L^R$ to be regular, since L is regular, then L^R is also regular (Closure under reversal property). By Property of union, $\emptyset \cup L^R = L^R$ and by Kleen closure property $(\emptyset \cup L^R)^*$ is also regular.

Prove $(\overline{NM^*L})$ to be regular language. since N, M and L are infact regular laguages. We can say that M^* is also regular by Kleene closure property. Finally by using concatentation, we can say NM^*L is regular by the concatenation closure property. so $\overline{NM^*L}$ is also regular by close under complenentation

Now, $[(\emptyset \cup L^R)^* \cap (\overline{NM^*L})]^*$ is also regular by Closure under intersection and Kleene closure property.

Finally, we can use complenentation, $\Sigma^* - L$ is also regular.

Thus $\Sigma^* - [(\emptyset \cup L^R)^* \cap (NM^*L)]^*$ is regular.

5b) If L is described by regular expression $(00^*1+01^*0)^* + \varepsilon + (11^*0+\varepsilon)^*$, find L^R. Show all the steps..

$L = (00^*1 + 01^*0)^* + \varepsilon + (11^*0 + \varepsilon)^*$

$L^R = [(00^*1 + 01^*0)^* + \varepsilon + (11^*0 + \varepsilon)^*]^R$

$= [(11^*0+\varepsilon)^*]^R + (\varepsilon)^R + [(00^*1 + 01^*0)^*]^R$

$= [(11^*0)^R + (\varepsilon)^R]^* + \varepsilon + [(00^*1)^R + (01^*0)^R]^*$

$= [(0)^R(1^*)^R 1^R + \varepsilon]^* + \varepsilon + [1^R(0^*)^R 0^R + 0^R(1^*)^R 0^R]^*$

$= [0(1^R)^* 1 + \varepsilon]^* + \varepsilon \quad [1(0^R)^* 0 + 0(1^R)^* 0]^*$

$= (01^*1 + \varepsilon)^* + \varepsilon + (10^*0 + 01^*0)^*$