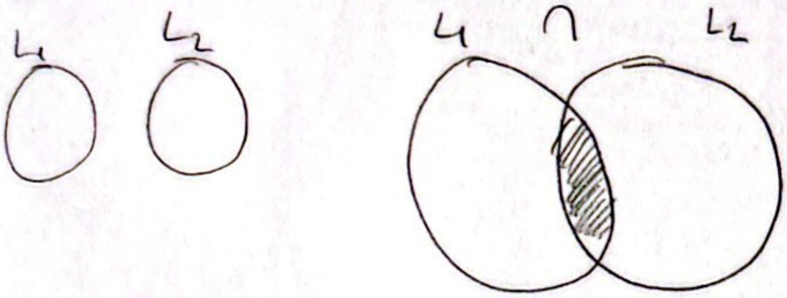
Theory Of Computation Homework I

Fatih Doğaç
1901042654

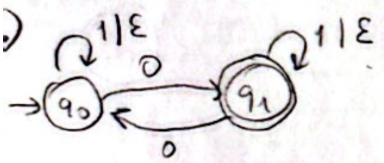
1) Regular Language: A language that can be expressed by some Finite Automata

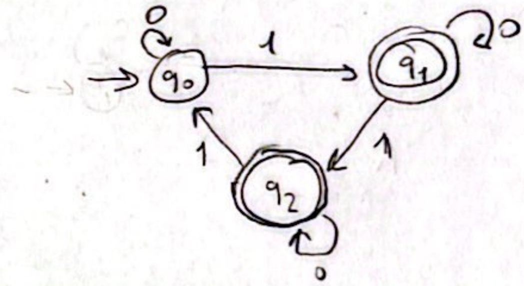
Say we have L_1 and L_2. And they are both regular languages. Let me use Venn schema for this question.



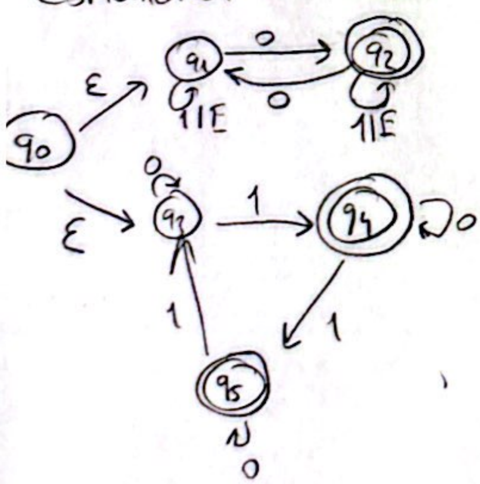
If L_1's all outputs can be expressed with a finite automata and L_2's too. Then $L_1 \cap L_2$ means they have some outputs in common. So this outputs are already defined as <u>regular</u>. and $L_1 \cap L_2$ has no other elements.

2) number of 0's odd $\cup$ number of 1's not a multiple of 3





Combination of these:



a) $M = \{Q, \Sigma, \delta, q_0, F\}$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

$\Sigma = \{0, 1\}$

$\delta = Q \times \Sigma \rightarrow P(Q)$

q_0 = Start state

$F = \{q_2, q_4, q_5\}$

c) No, M can recognize minimum 3 language because it is a union of 2 languages

- A
- Language with number of 0's is odd.
- Language with number of 1's is not a multiply of 3.

d) $S = 1000$:

$q_0 \times \varepsilon \rightarrow q_1$ $\quad$ 1000

$q_1 \times 1 \rightarrow q_1$ $\quad$ 1000

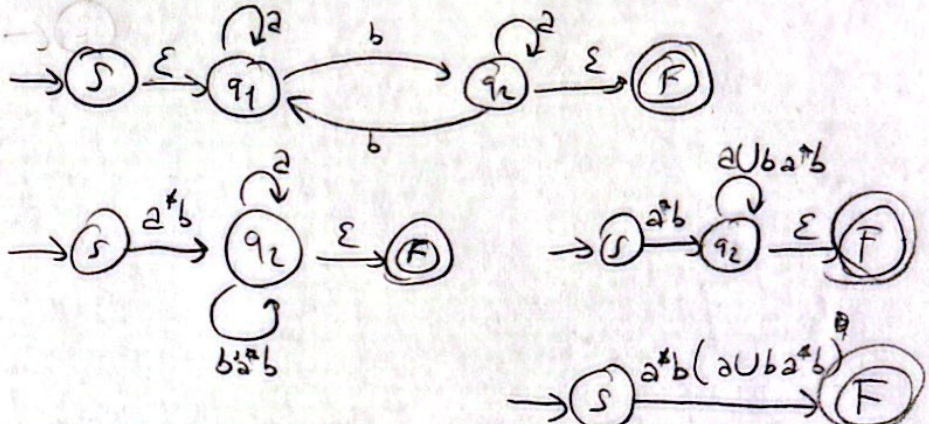
$q_1 \times 0 \rightarrow q_2$ $\quad$ 1000

$q_2 \times 0 \rightarrow q_1$ $\quad$ 1000

$q_1 \times 0 \rightarrow q_2$ $\quad$ 1000

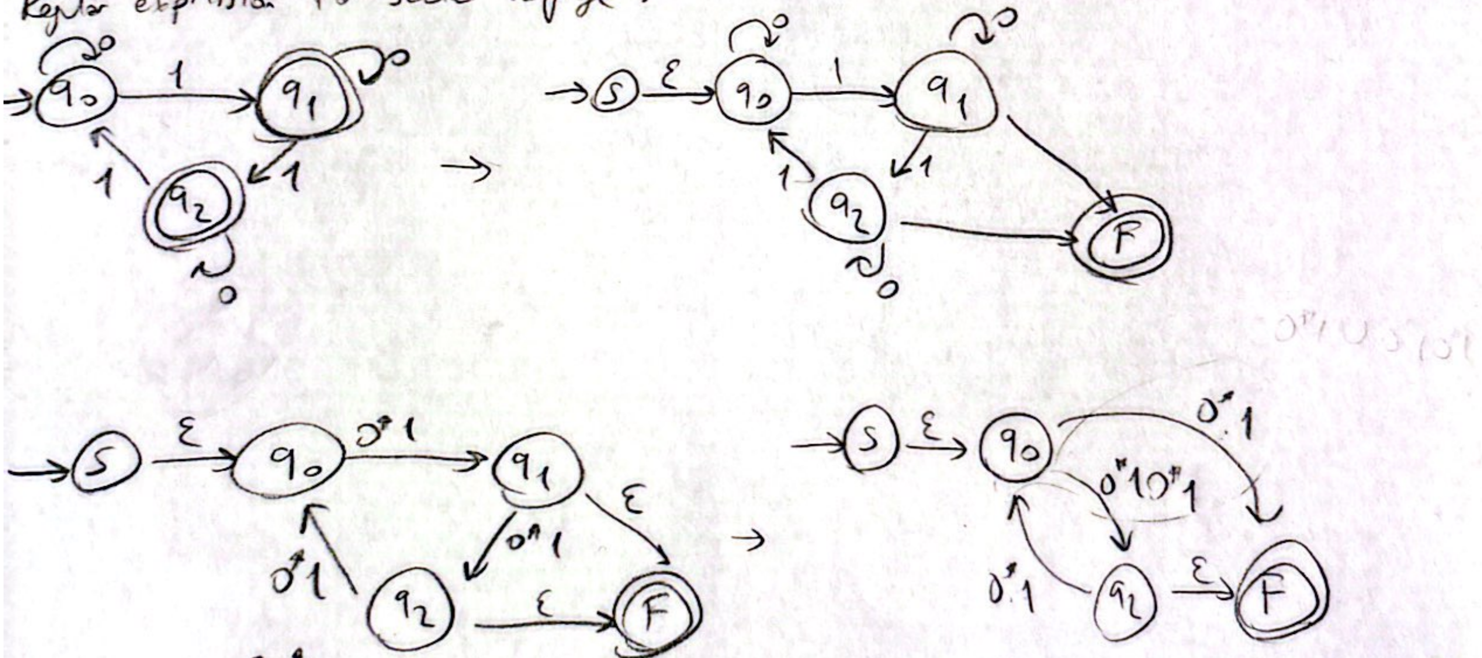
accept state

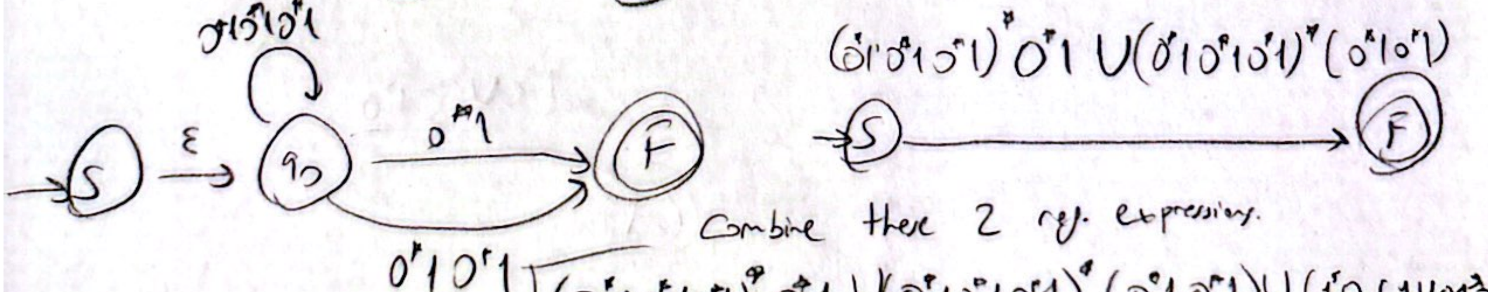
e) Regular expression for first language



$1^*0(1 \cup 01^*0)^*$

Regular expression for second language:





Combine these 2 reg. expressions.

$\boxed{(0^*10^*10^*1)^*0^*1 \cup (0^*10^*10^*1)^*(0^*10^*1) \cup (1^*0(1 \cup 01^*0)^*)}$

3) $S \to aSb \mid Ab$
$A \to aA \mid \varepsilon$

C_1

$S \to aSb \mid T$
$T \to aTb \mid \varepsilon$

C_2

a) Yes, they are proper CFG's. Because the 4 tuple can be defined for both of them.

For C_1:

$V = \{S, A\}$

$\Sigma = \{a, b, \varepsilon\}$

$R = \{S \to aSb,\ S \to Ab,\ A \to aA,\ A \to \varepsilon\}$

$S = S$

For C_2:

$V = \{S, T\}$

$\Sigma = \{a, b, \varepsilon\}$

$R = \{S \to aSb,\ S \to T,\ T \to aTb,\ T \to \varepsilon\}$

$S = S$

They can both end. Because they have ε rules. And they both don't have terminal elements in the left section (non-terminal section.)

b) C_1 is not ambigous because the last 'b' on the string should be recognized by the rule $S \to Ab$.

C_2 is ambigous because $S \to aSb$ and $T \to aTb$ are basically the same

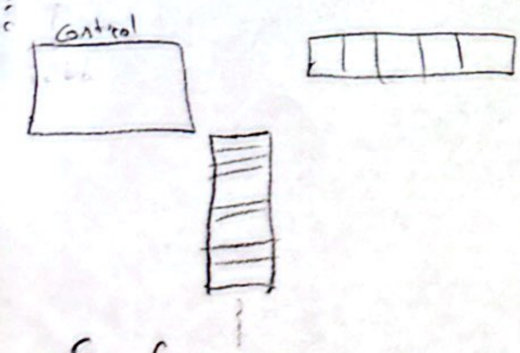
aabb

$S \to T \to aTb \to aaTbb \to aabb$

$S \to aSb \to aaSbb \to aaTbb \to aabb$

c) For C_1:

aaabbb   aabbb   a.... aabbb



For C_1, we can not define a pushdown automata. We cannot determine the number of 'a's in the input. Even we push 'a's to the stack, we cannot empty the stack. We don't have that kind of rule.

For C_2:

$L(C_2) = \{a^k b^k \mid k > 0\}$

1. read 'a's from the input and push into stack until reading 'b'
2. read 'b' from input and pop 'a's from stack
3. Enter accept state if the stack is empty

$Q: \{q_0, q_1, r\}$

$\Sigma: \{a, b, \varepsilon\}$

$\Gamma: \{A, Z\}$

$Q_0: q_0$

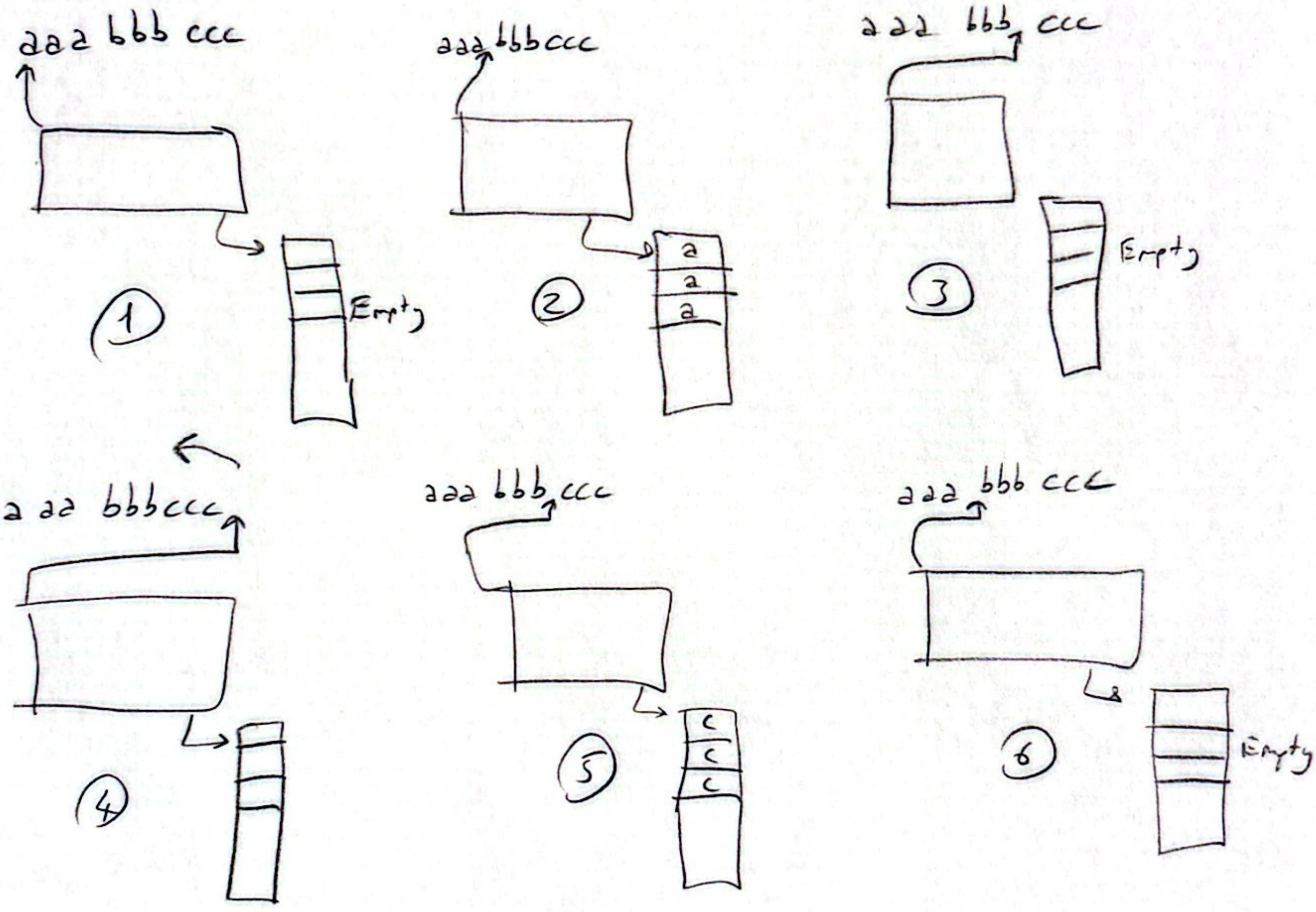
$Z: Z$

$F: \{q_1\}$

δ:
(q_0, a, Z, q_0, AZ)
(q_0, a, A, q_0, AA)
(q_0, b, A, q_1, Z)
(q_0, ϵ, Z, r, Z)
(q_1, b, A, q_1, A)
(q_1, b, A, r, Z)

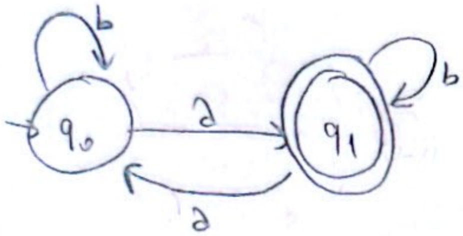
4) $a^i b^i c^i \mid i > 0$

1) NPDA reads a's and pushes a for each a it reads. Until it encounters a 'b'.
2) It reads b's and pops one a for each b it reads.
3) Then it moves the input to the end.
4) It is starting to read from right to left and it is reading c's.
5) For each c it reads it pushes a c to the stack.
6) Then it reads b's and for each b it reads, pops a 'c' from stack
7) When it encounters an 'a', if the stack is empty the string accepted.

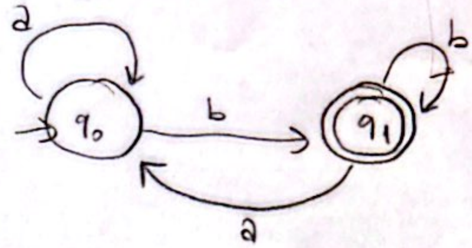


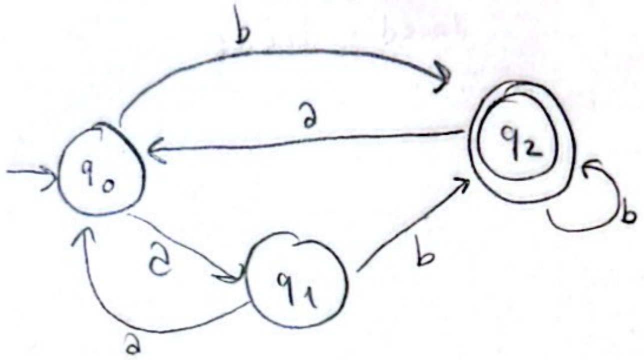
5-) a) $\Sigma = \{a, b\}$ aaa abaa

w has an odd number of a's



w ends with b





baaab
abaab
aaaaab

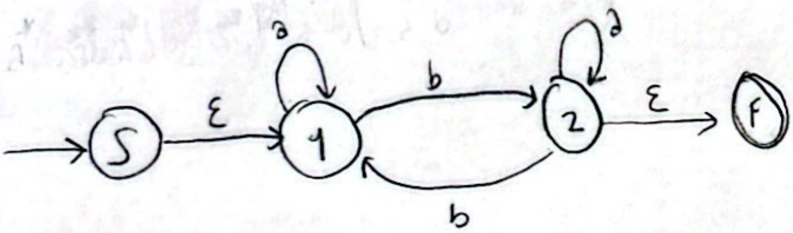
b)

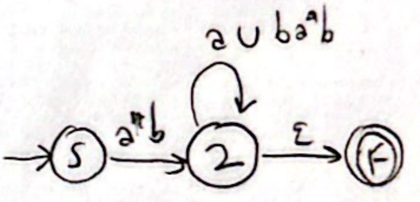
Part a:

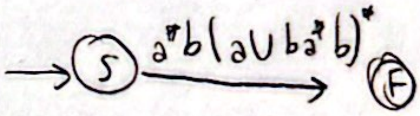
aabb

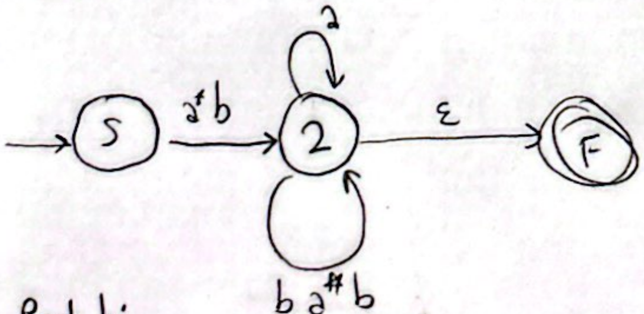
aab aa b| aa b aab

aab|aabaab









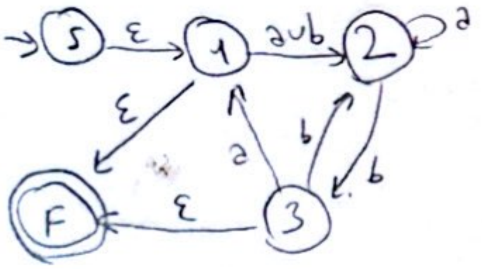
Merge 2 Loop = $\cup$

Merge 2 state = Concat, if a state has loop, put star.

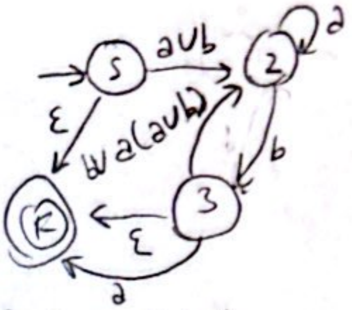
$a^*b(a \cup ba^*b)^*$

Part b:

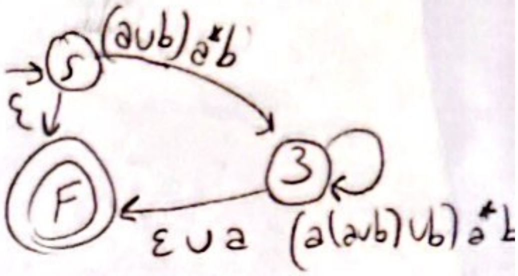
Part b:



Create one start state and one finish state



Destroy state 1.
From state 3 the only way is 1 to 2. If 1 is gone then concat the wanted inputs



$$\left(\left((a\cup b)a^*b\right)\left((a(a\cup b)\cup b)a^*b\right)^*(\varepsilon\cup a)\right)\cup\varepsilon$$

c) 1.46 part d:

pumping lemma: A is regular language. P pumping length. S string in A $|S| \geq P$

$S = xyz$

1) $xy^iz \in A$ for every $i \geq 0$
2) $|y| > 0$
3) $|xy| \leq P$

if one of them not satisfy language is <u>not regular</u>.

$\{wtw \mid w,t \in \{0,1\}^+\}$

$P = 5$ $P > 0$

$S = 0^P1^P$ 0000011111

3 cases:

case 1: $\underbrace{000}_{x}\underbrace{00\,11}_{y}\underbrace{111}_{z}$

case 2: $\underbrace{0}_{x}\underbrace{0}_{y}\underbrace{00\,0\,11111}_{z}$

case 3: $\underbrace{00000\,1}_{x}\underbrace{111}_{y}\underbrace{1}_{z}$

case 1: xy^iz

$\underbrace{000}_{x}\underbrace{0011\,0011}_{y^2}\underbrace{111}_{z} \notin 0^k1^k$ X

$|xy| \leq P \Rightarrow 7 \leq 5$ X

case 2:

$\underbrace{0}_{x}\underbrace{00000}_{y^2}\underbrace{00\,11111}_{z} \notin 0^k1^k$ X

$|xy| \leq P \Rightarrow 4 \leq 5$ ✓

case 3: $\underbrace{000001}_{x}\underbrace{111111}_{y^2}\underbrace{1}_{z} \notin 0^k1^k$ X

Hence, This language is not regular.

Scanned with CamScanner

) 1.54

$F = \{ a^i b^j c^k \mid i,j,k \geq 0$ and if $i=1$, then $j=k \}$

a) $S = a b^p c^p$ $p \geq 0$ $|S| > p$

$\underbrace{a}_{x}\underbrace{b^p}_{y}\underbrace{c^p}_{z}$ $xy^0z \Rightarrow xz \notin F$ Hence, F is not regular

b) $a^i b^j c^k$ $p=2$

$a^p b^j c^k$ $\underbrace{\varepsilon}_{x}\underbrace{aa}_{y}\underbrace{b^j c^k}_{z}$ $aaaaaa b^j c^k$

$xy^iz \in F$ ✓

$|xy| \leq p$ $|y| > 0$ ✓ 3 conditions of pumping lemma is satisfied

$2 \leq 2$ ✓

c) Pumping lemma is used to prove that a language is not regular.
It cannot be used to prove that a Language is regular

2] 1.55

c) $001 \cup 0^* 1^*$

1) $xy^iz \in A \ \forall i \geq 0$
2) $|y| > 0$
3) $|xy| \leq p$

string 1 is in the language.

$\underbrace{\varepsilon}_{x}\underbrace{1}_{y}\underbrace{\varepsilon}_{z}$

1) $xy^iz \in$ this language ✓
2) $|y| = 1 > 0$ ✓
3) $|xy| \leq p$ $1 \leq 1$ ✓

It satisfies the conditions. So min pumping length is 1.

h) $10(11^*0)^*0$ 100 10 10.0. 10 10 1010... 0 101100 1011.... 0 .. 0

$|s|=3$ can't be pumped because there is no true dividing xyz to pump

For $|s|=5$ $s = \underbrace{10}_{x}\underbrace{10}_{y}\underbrace{0}_{z}$ 1) $xy^iz \in L$ ✓ 2) $|y| > 1$ ✓ 3) $|xy| \leq p$ $4 \leq 5$ ✓

MPL for h is 5. 10100

j) Σ^*

Σ stands for alphabet so $=(\varepsilon|0|1)^*$

$\underbrace{\varepsilon}_{x}\underbrace{(\varepsilon|0|1)}_{y}\underbrace{\varepsilon}_{z}$

1) $xy^iz \in L$ ✓ 2) $|y|>0$ ✓ 3) $|xy| \leq P$

$1 \leq 1$ ✓

MPL $= 1$

f) 2.4 part e

$\{w \mid w = w^R$, that is w is a palindrome$\}$ $\Sigma = \{0,1\}$

$S \to F \mid T$

ε string is also a palindrome.

$F \to 0F0 \mid 1F1 \mid T$

$T \to \varepsilon$