Simple closure operations on regular languages

Closure under complementation :

If $L = L(M)$ with $M = (Q, \sum, \delta, q_0, F)$ then

obviously $L^c = L(M')$ with $M' = (Q, \sum, \delta, q_0, F^c)$

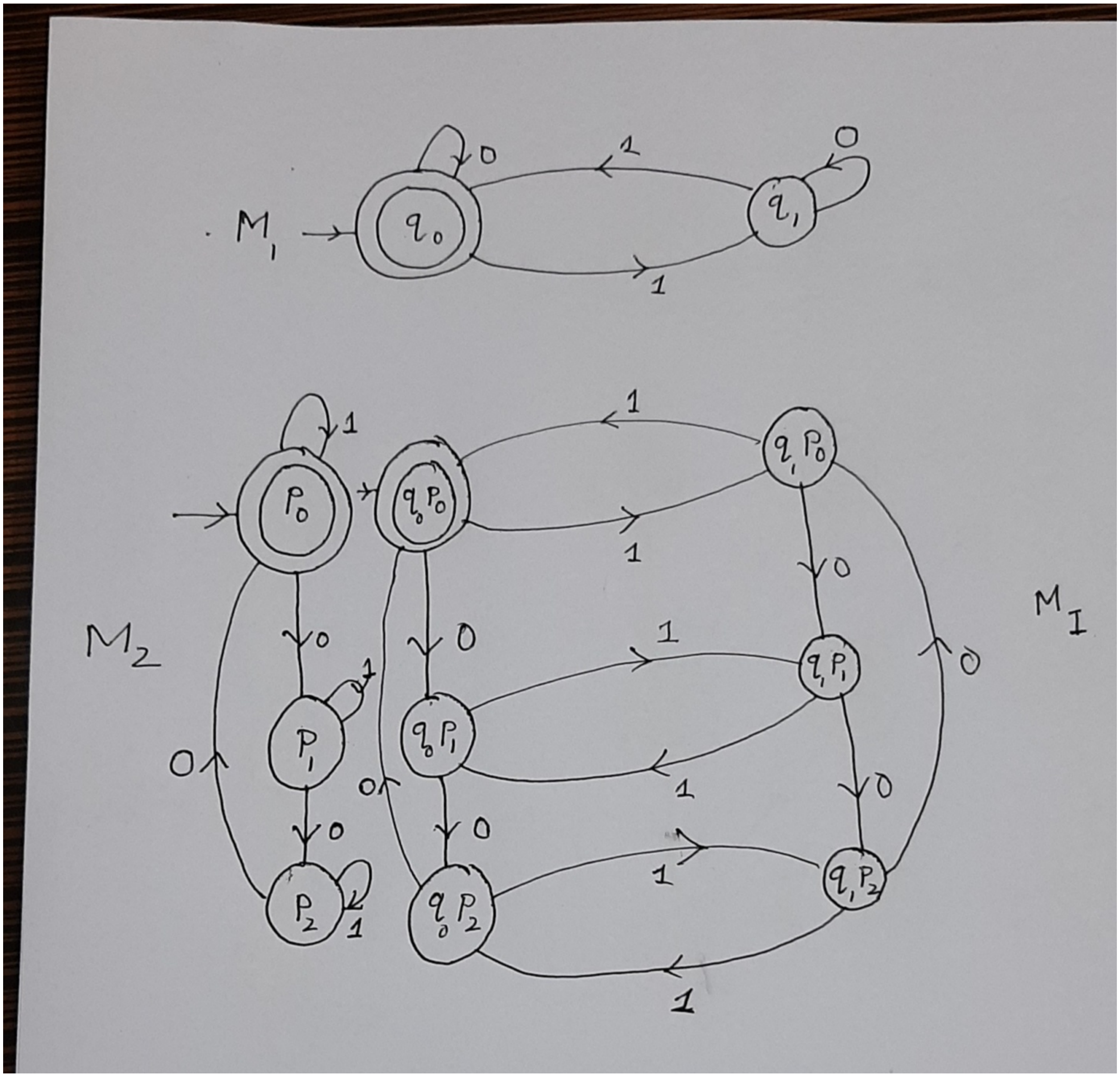
For union and intersection, we first note that if $\sum$ is a subset of $\sum'$, a DFA M over $\sum$ is equivalent to a DFA M' over $\sum'$ by introducing an error state q_e and defining $\delta'(q,a) = \delta(q,a)$ for $a \in \sum$,

$\delta'(q,a') = q_e$ for $a' \in \sum' - \sum$ and $\delta'(q_e,a') = q_e$ for $a' \in \sum'$.

Let $L1 = L(M1)$ and $L2 = L(M2)$ where

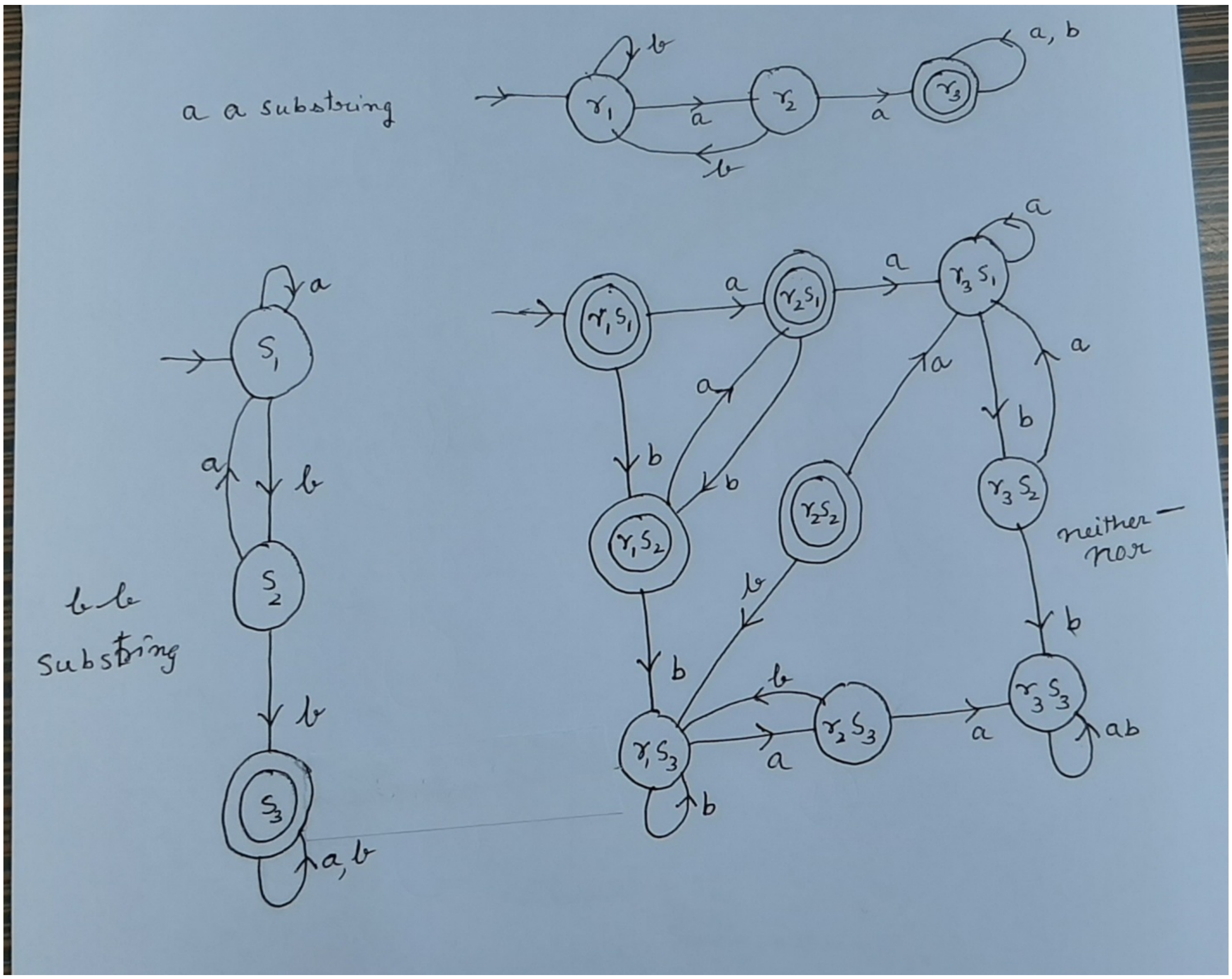
$M1 = (Q1, \sum1, \delta1, q01, F1)$ and $M2 = (Q2, \sum2, \delta2, q02, F2)$. Because of the above remarks we can take $\sum1 = \sum2 = \sum1 \cup \sum2 = \sum$. We will find MI and MU s.t. $L(MI) = L1 \cap L2$ and $L(MU) = L1 \cup L2$. For both $Q = Q1 \times Q2$, $\delta((q1, q2), a) = (\delta1(q1, a), \delta2(q2, a))$, $q_0 = (q01, q02)$. Also $FI = F1 \times F2$ and $FU = (F1 \times Q2) \cup (Q1 \times F2)$.

For example let L1 = no of 1's even and L2 = no of 0's is a multiple of 3. Then $L1 \cap L2$ is given by the DFA MI which can be constructed by



For L1 U L2 the 1st row and the 1st col are final.

Let L be the language over {a, b} where neither aa nor bb are substrings. Then taking L1 = aa is a substring and L2 = bb is a substring L = (L1 U L2)^c. Then we can construct a DFA for L1 U L2 as shown and take its complement as shown. In the DFA for L1 U L2, r1s3, r2s3, r3s3, r3s2 and r3s1 are final states. Hence in the DFA for L r1s1, r1s2, r2s1 and r2s2 are final states.

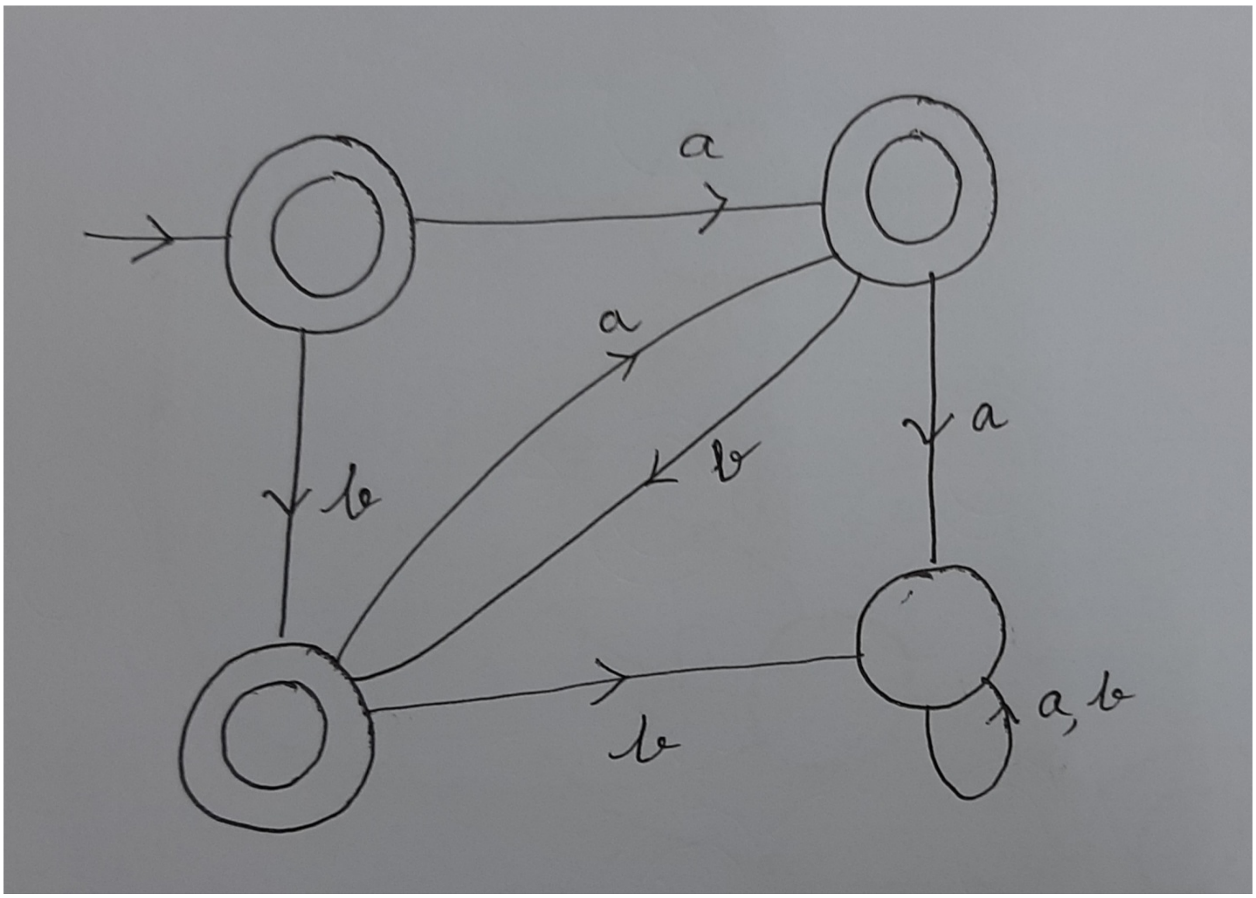


In any DFA ,

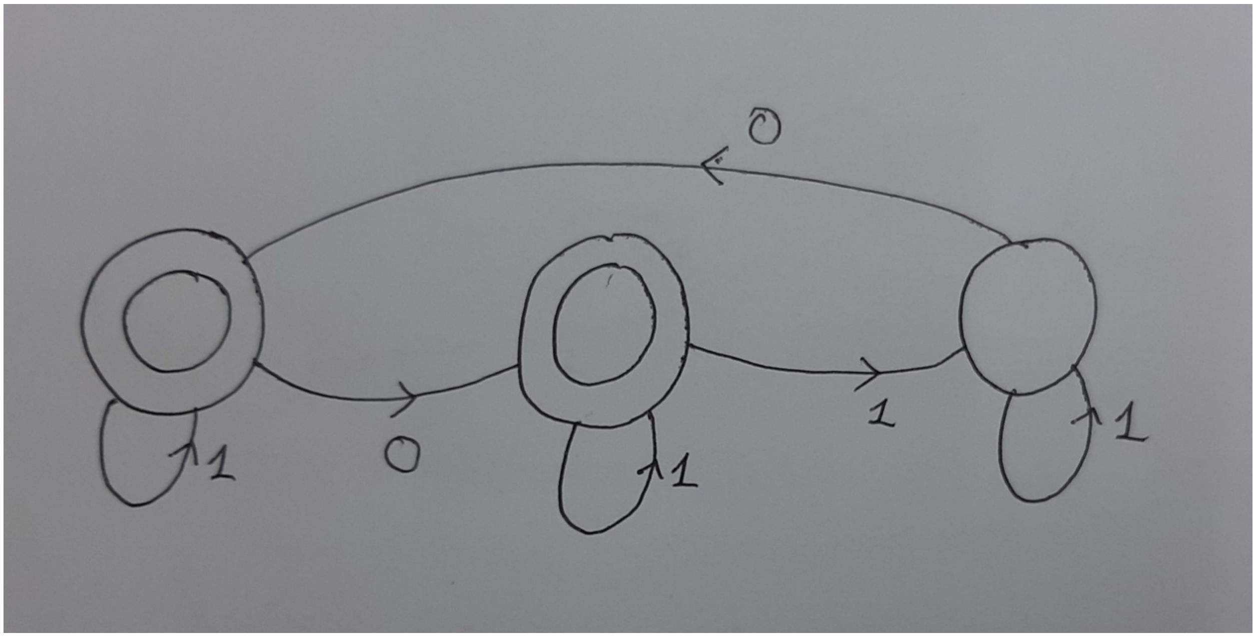
1) Unreachable states can be dropped.

2) All states from which a final state cannot be reached, can be combined to form a single error state.

Therefore we get an equivalent DFA :

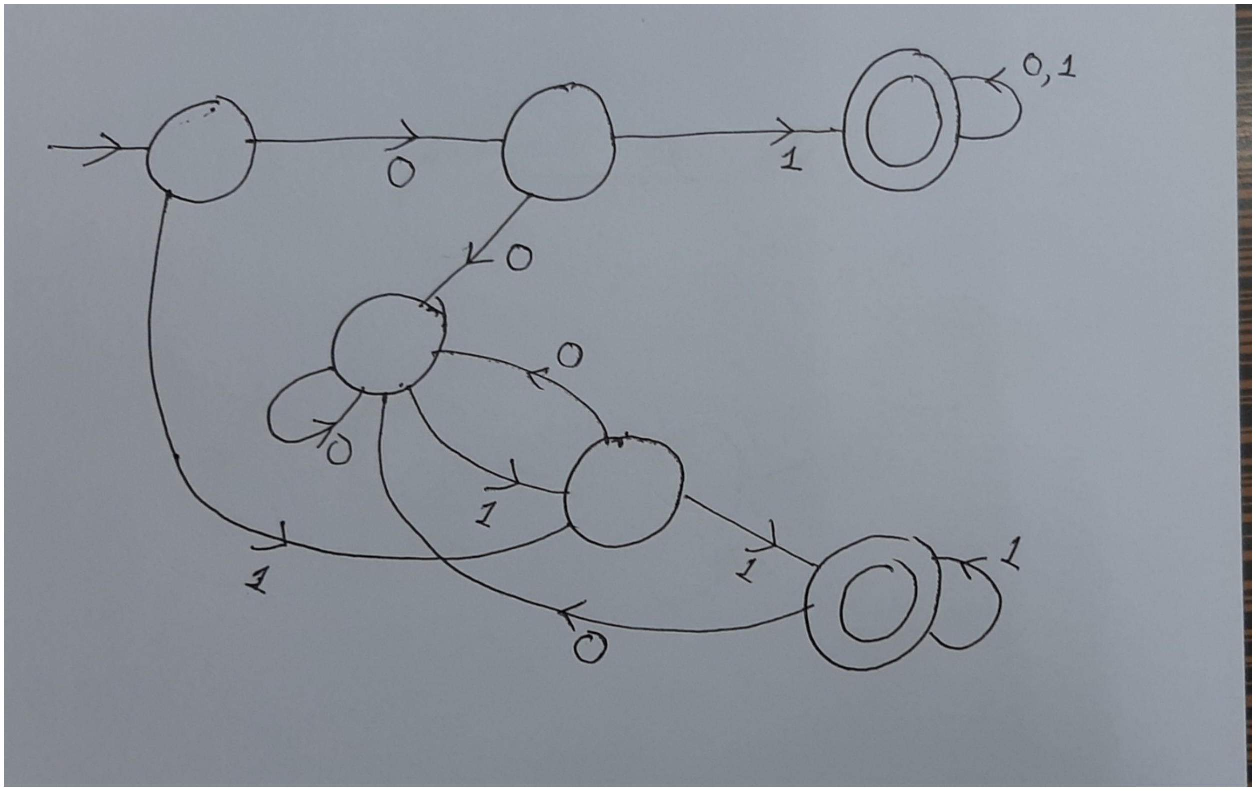


Cartesian product construction is tedious. Should not be used if the result can be obtained in a simpler way. For example No of 0’s is a multiple of 3 or no of 0’s is 1 mod 3.



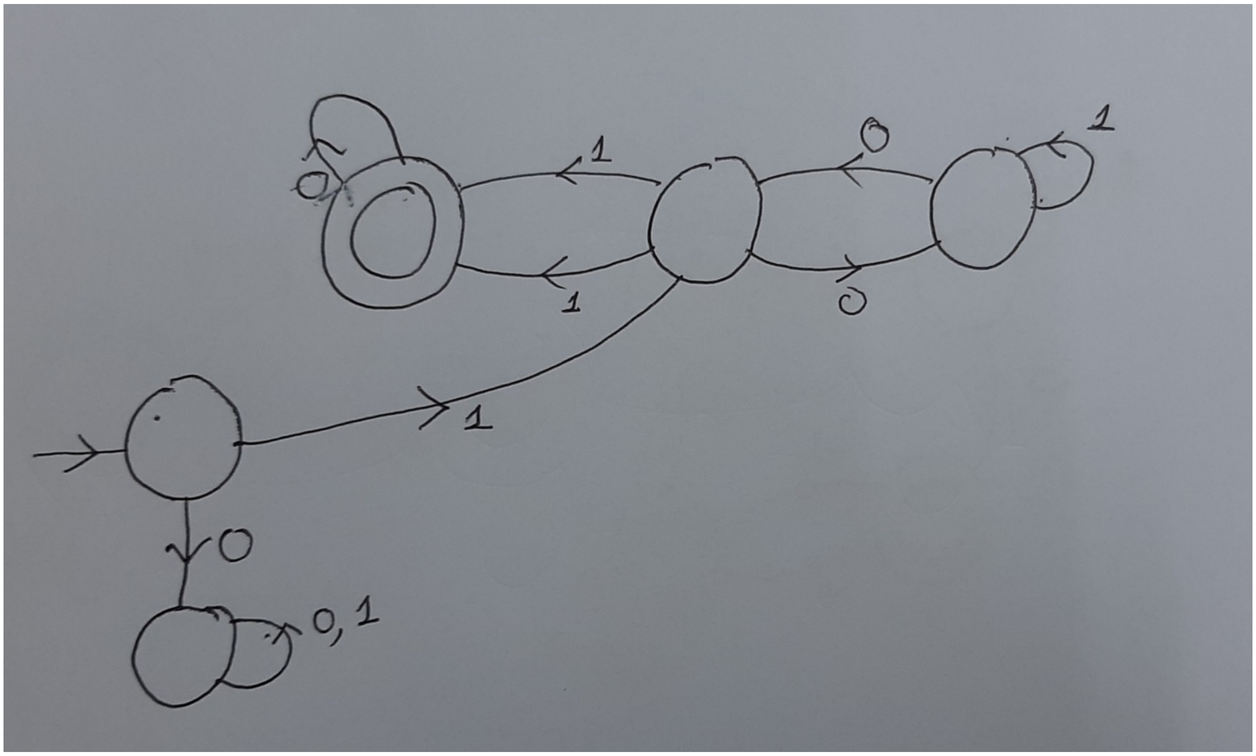
There is a mistake in the above diagram. The transition from the second to the third state should be labeled by 0.

Begin with 01 or end with 11.



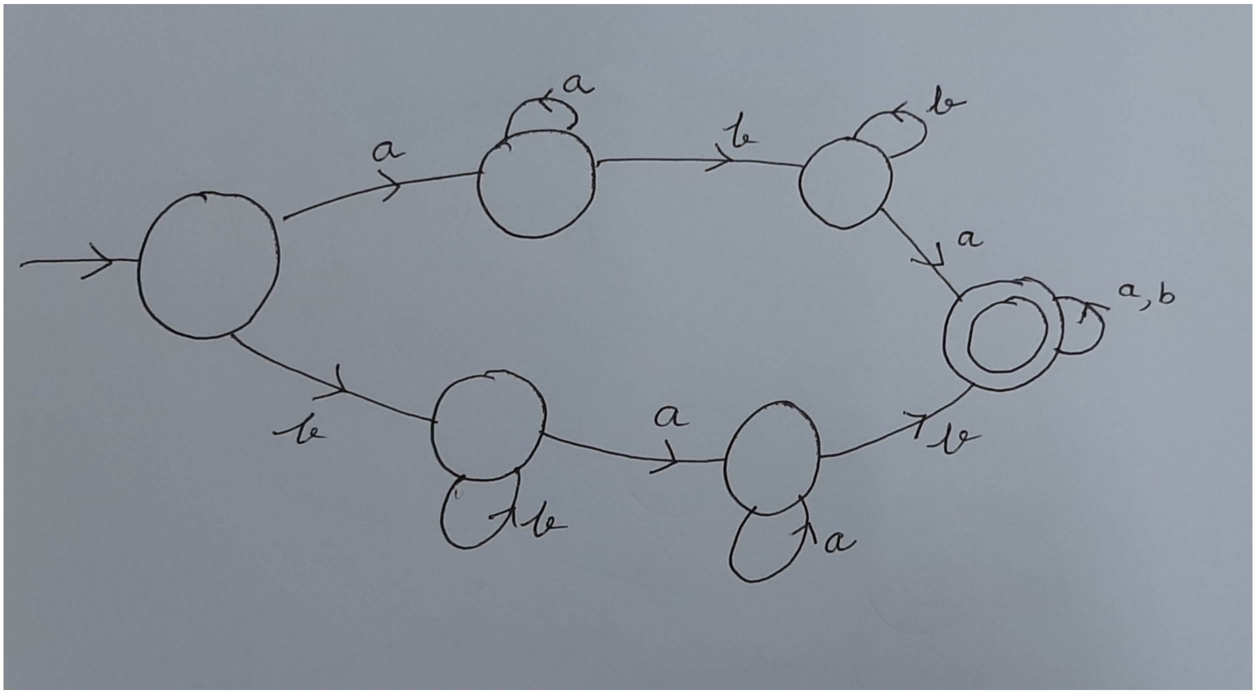
HW Begin with 01 and end with 11.

Begin with 1 and binary multiple of 3



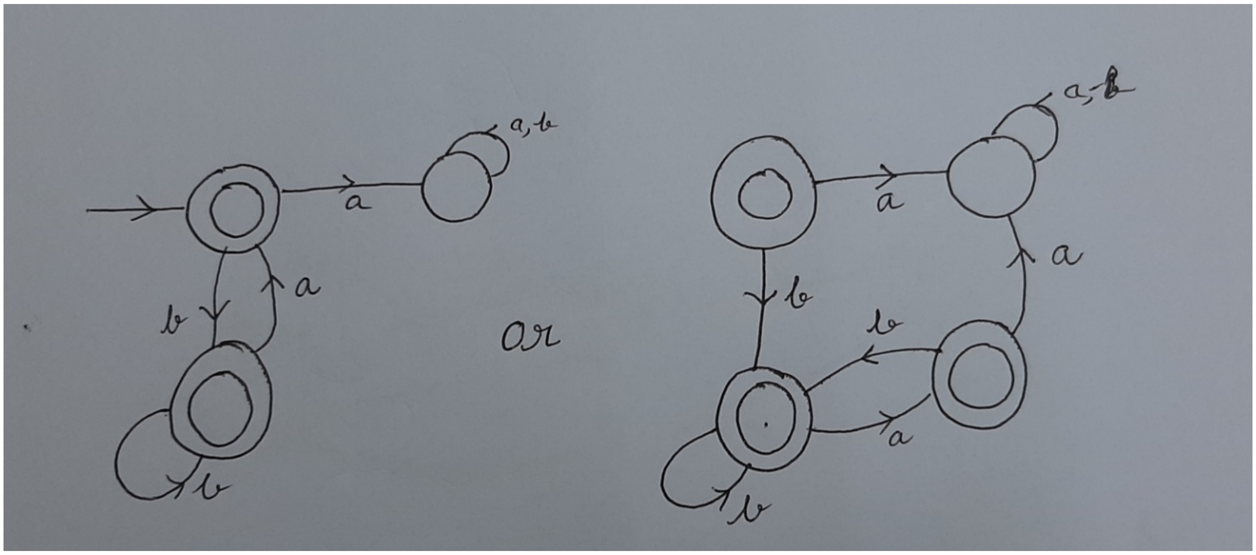
HW Begin with 1 or binary multiple of 3.

Both ab and ba as substring



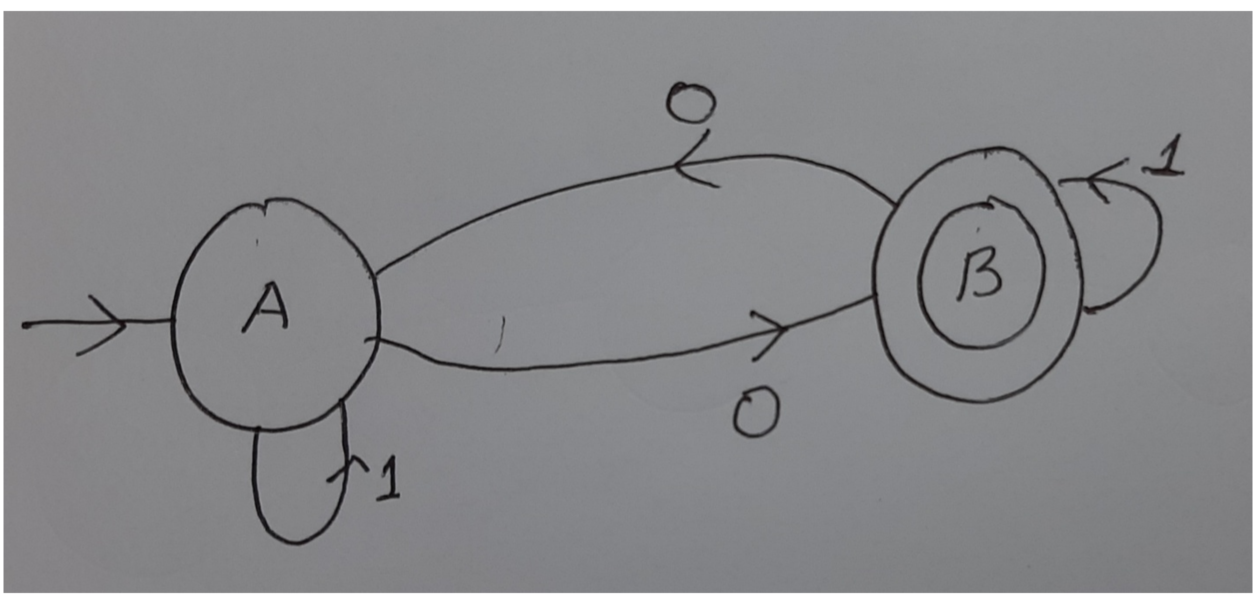
HW Neither ab nor ba as substring. Hint : Take the complement of either ab or ba as substring.

Each ‘a’ is immediately preceded by a ‘b’.



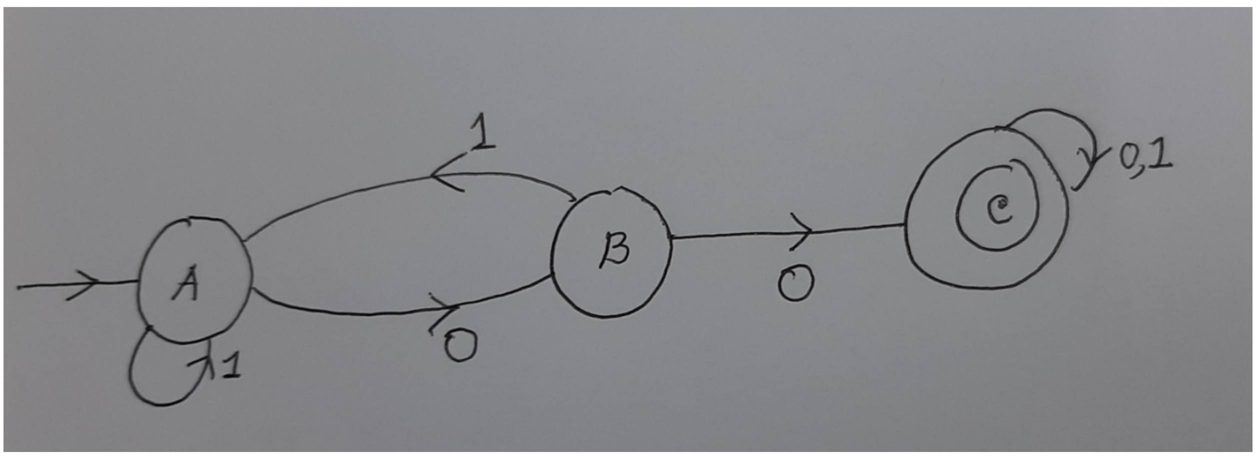
Describe language :

	0	1
- > A	B	A
*B	A	B



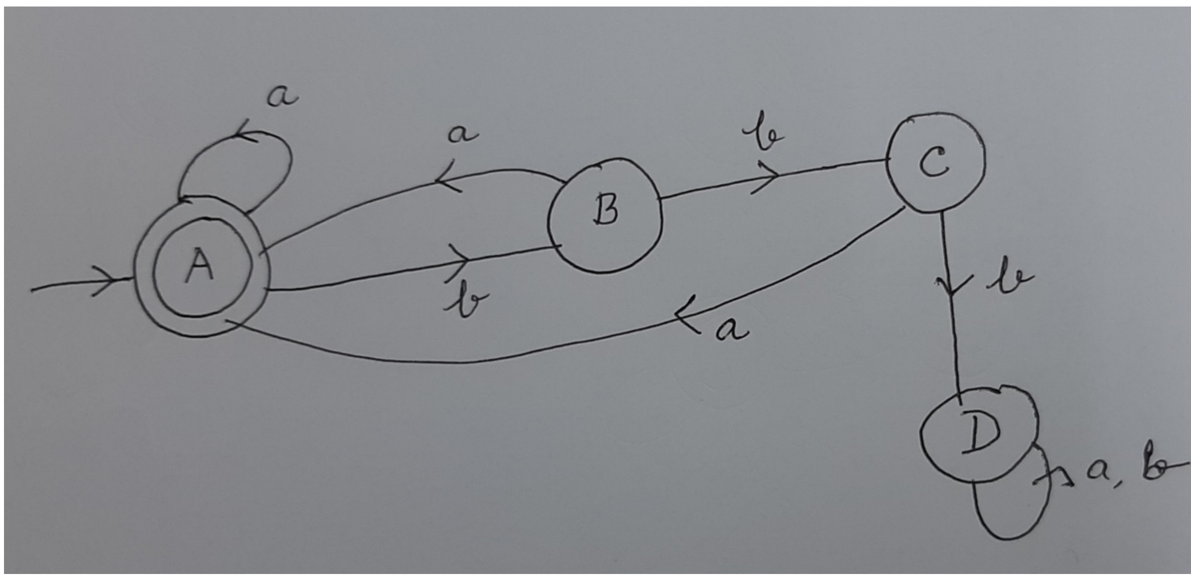
Odd 0's

	0	1
->A	B	A
B	C	A
* C	C	C



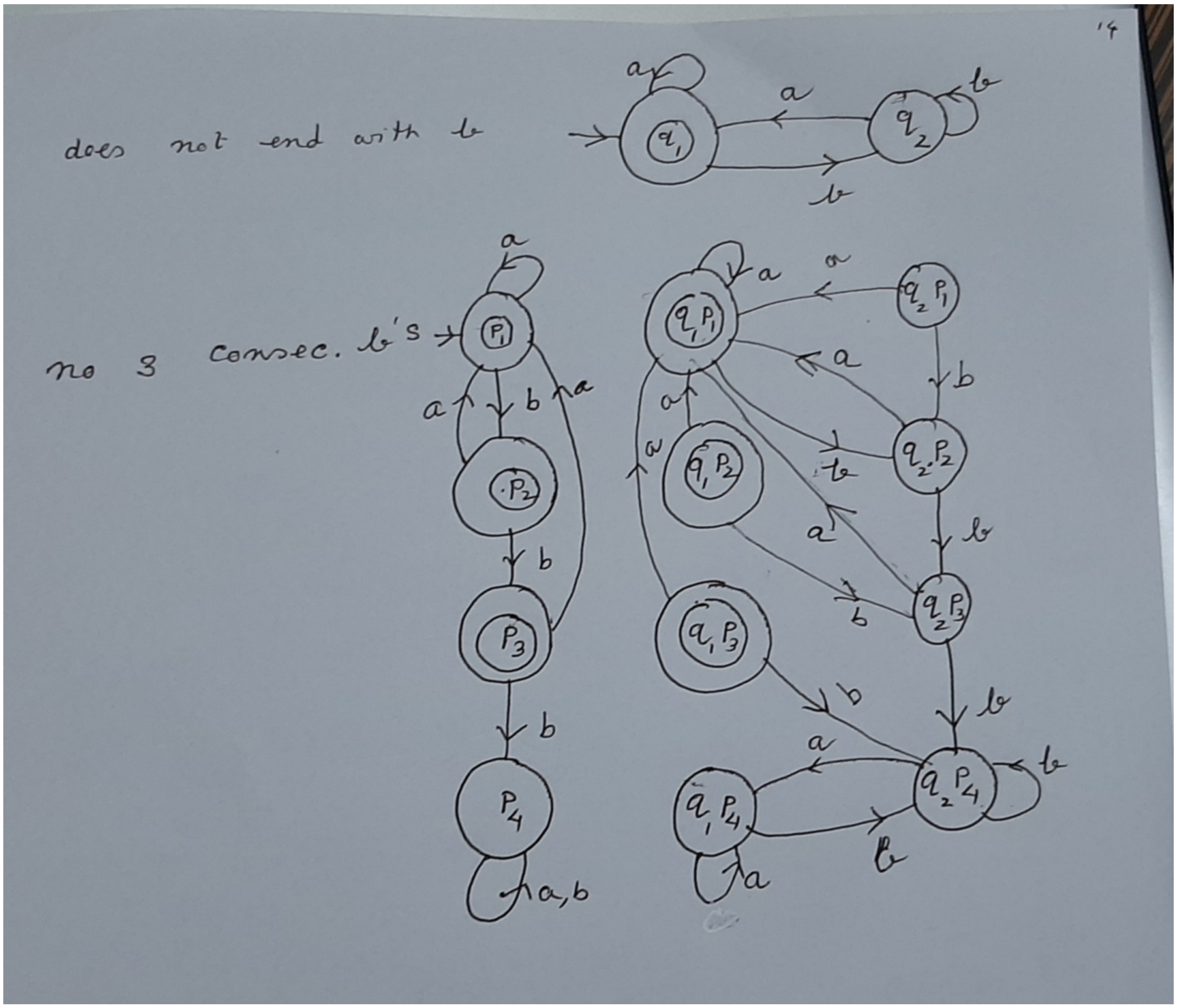
Contains 00

	a	b
- >* A	A	B
B	A	C
C	A	D
D	D	D

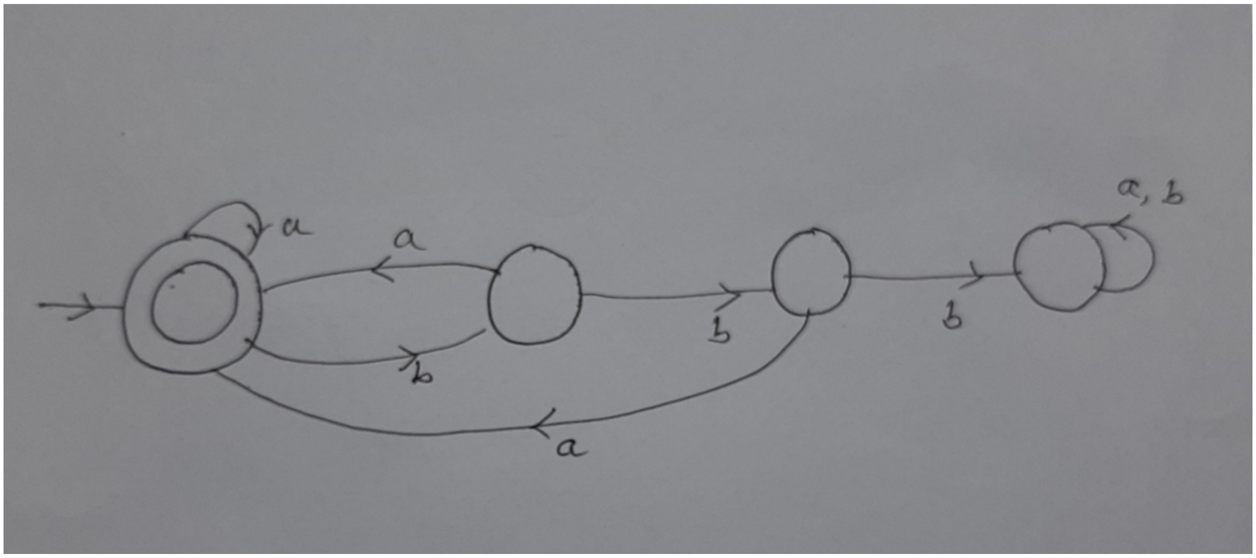


Does not end with b and cannot contain three consecutive b’s

This can be understood as :



This can be simplified to


a
a
b
b
b
a
a, b