

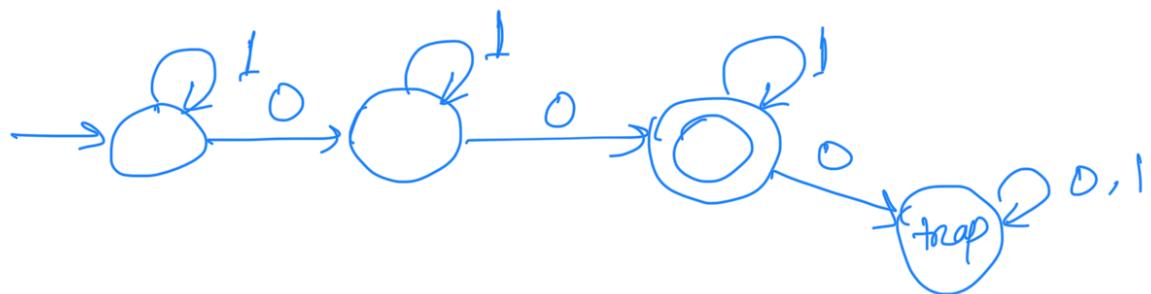
## Deterministic Finite Automata

- \* An automaton is a simple computing machine.
  - \* We want to know how the computation works. That's why we use simpler machine like DFA to understand the underline works properly.
  - \* In DFAs we only want to store the useful informations as the memory is lot less in DFA.
  - \* Symbol, alphabet, string, Language.
- \* We construct DFAs in such a manner, that it recognizes a set of strings with one/many properties.
- set of strings  $\Rightarrow$  Language.
- \* Each language has a set of alphabets.
  - \* Any language recognized by a DFA is called a regular language.
  - \* Think yourself as a DFA.
  - \* We will learn how to construct DFAs using examples.

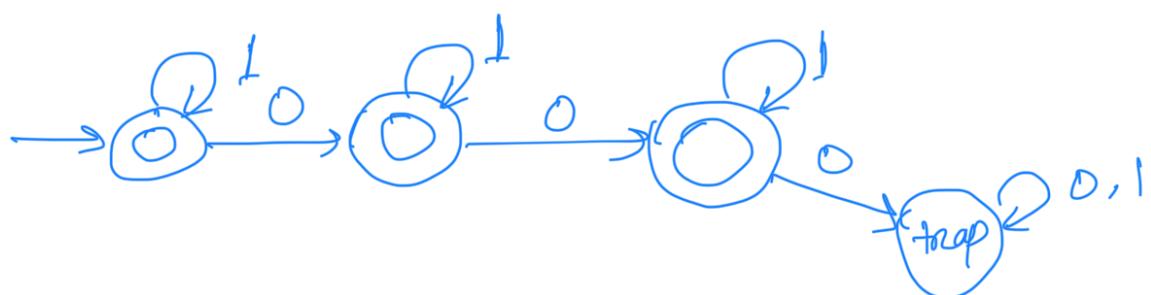
①  $L = \{ w \in \{0,1\}^*, w \text{ contains at least two } 0's \}$



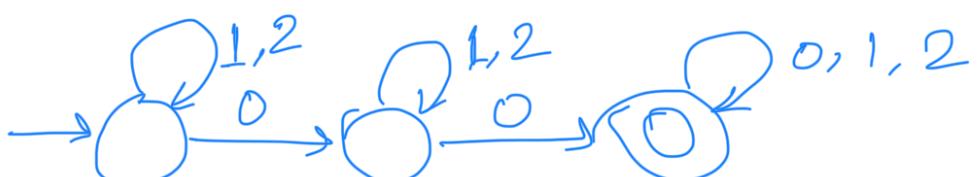
②  $L = \{ w \in \{0,1\}^*, w \text{ contains exactly two } 0's \}$



③  $L = \{ w \in \{0,1\}^*, w \text{ contains at most two } 0's \}$

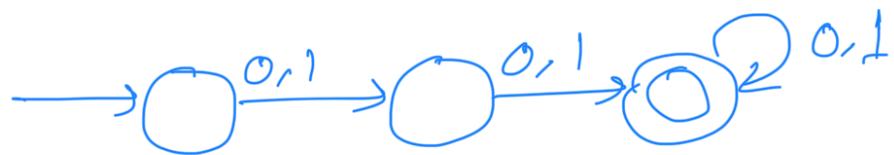


④  $L = \{ w \in \{0,1,2\}^*, w \text{ contains at least two } 0's \}$



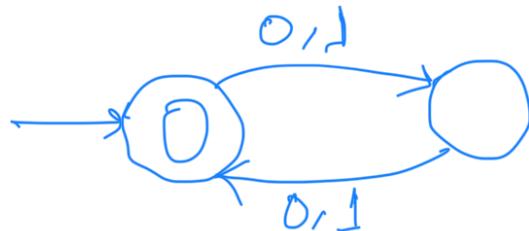
[check the difference]

⑤  $L = \{w \in \{0,1\}^*, \text{ length of } w \text{ is at least two}\}$

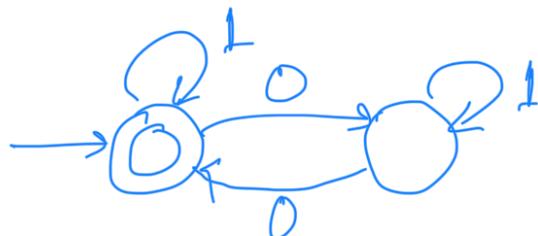


try to solve length of  $w$   
 ① exactly two  
 ② at most two

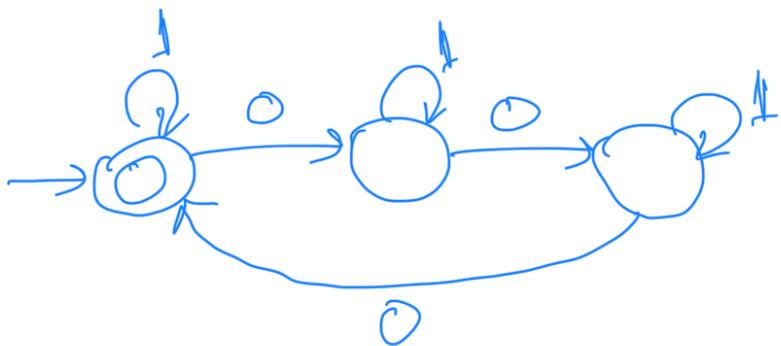
⑥  $L = \{w \in \{0,1\}^*, \text{ length of } w \text{ is even}\}$   
 $\text{multiple of two}$



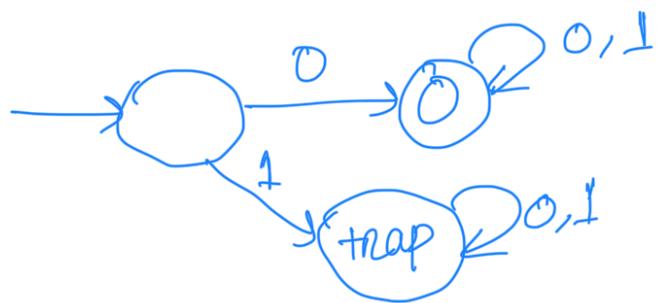
⑦  $L = \{w \in \{0,1\}^*, \text{ the count of } 0's \text{ in } w \text{ is even/multiple of two}\}$



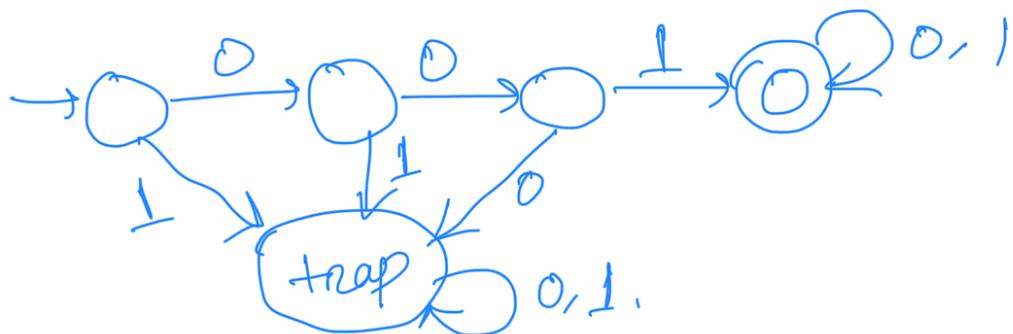
⑧  $L = \{w \in \{0,1\}^*, \text{ the count of } 0's \text{ in } w \text{ is a multiple of three}\}$



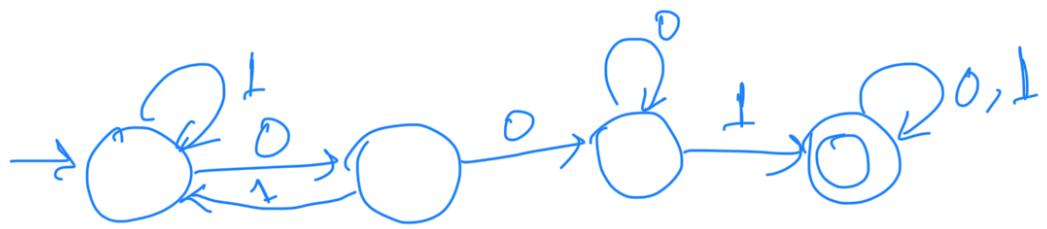
⑨  $L = \{w \in \{0,1\}^*, w \text{ starts with } 0\}$



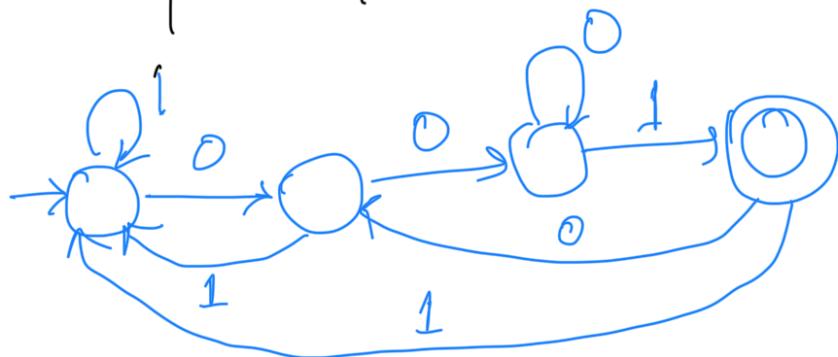
⑩  $L = \{w \in \{0,1\}^*, w \text{ starts with } 001\}$



⑪  $L = \{w \in \{0,1\}^*, w \text{ contains } 001 \text{ as a substring}\}$

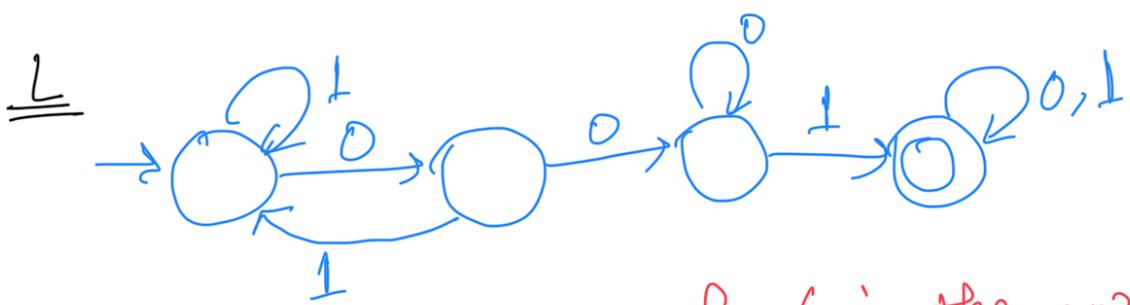


⑫  $L = \{ \omega \in \{0,1\}^*, \omega \text{ ends with } 001 \}$

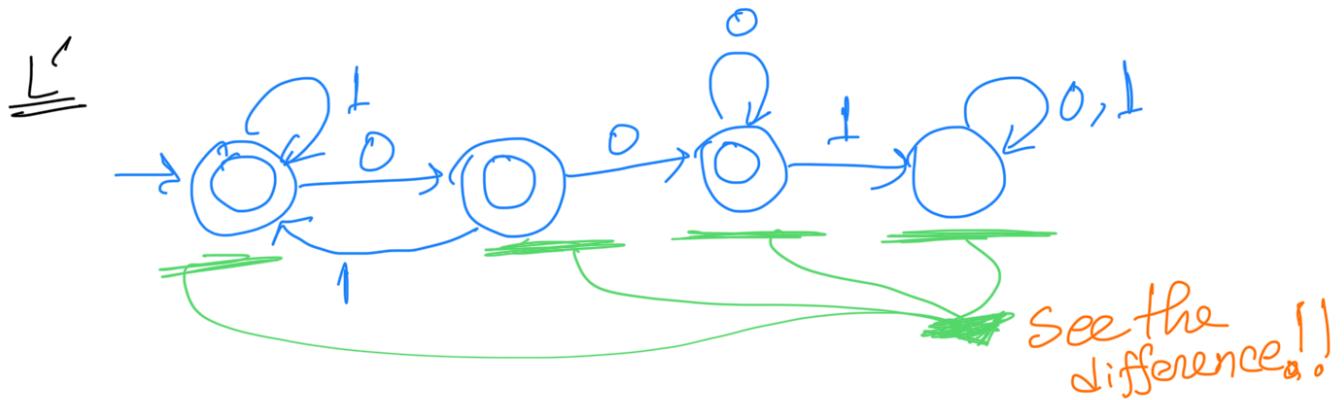


\* Sometimes some problems state the word "does not contain 001". This is the opposite of ⑪ example.

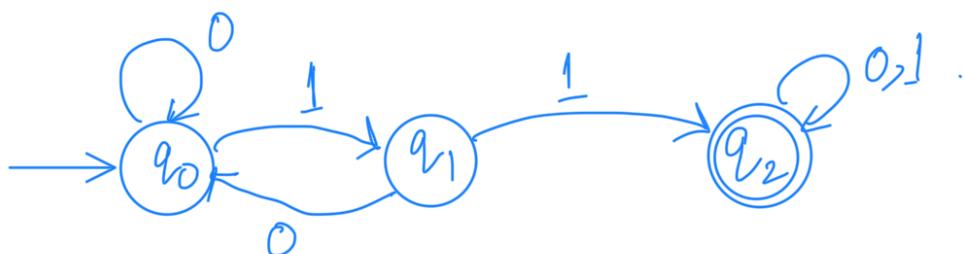
Let,  $L = \text{"contains 001 as a substring"}$   
 $L' = \text{"does not contain 001"}$



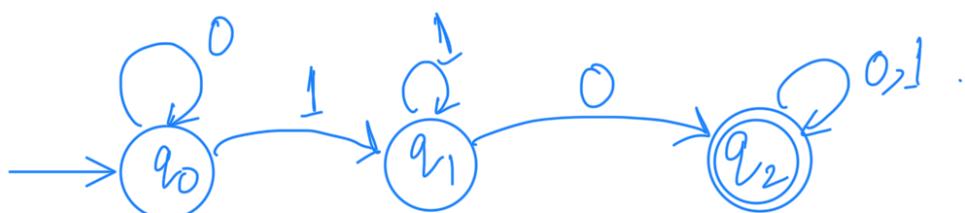
\* Accepting state of  $L'$  is the opposite of  $L$



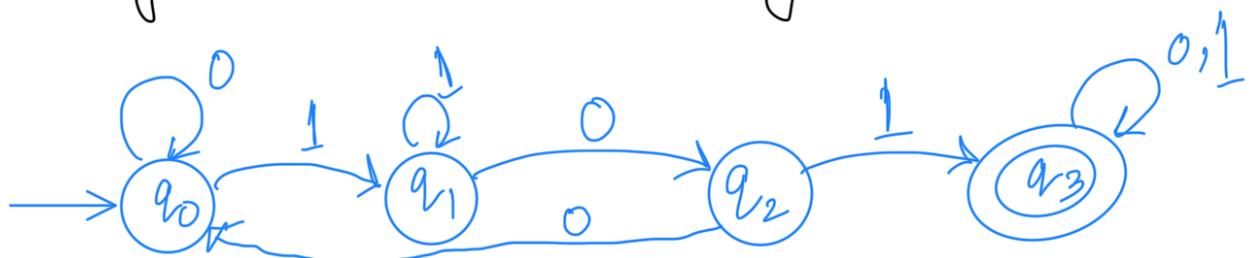
- (13) Construct a DFA that recognizes strings with 11 as substring.



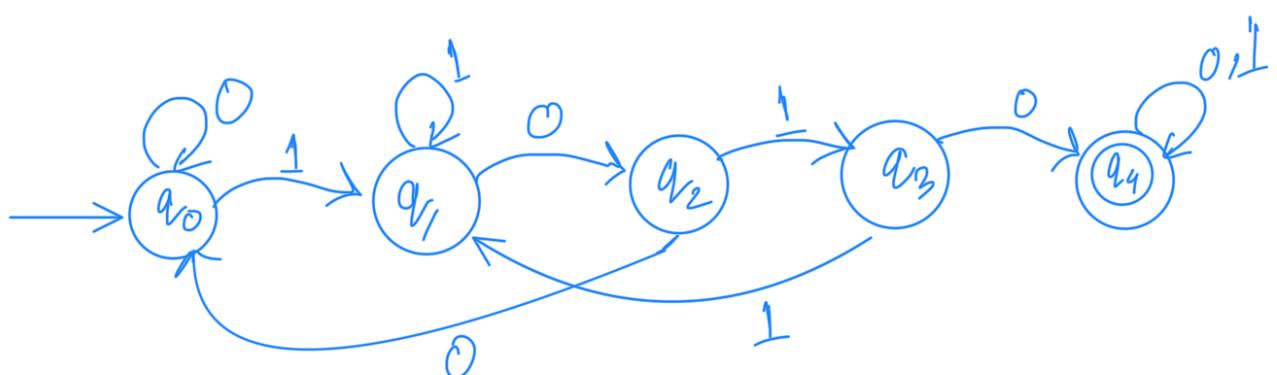
- (14) Construct a DFA that recognizes strings with 10 as substring.



15 Construct a DFA that recognizes strings with 101 as substring.

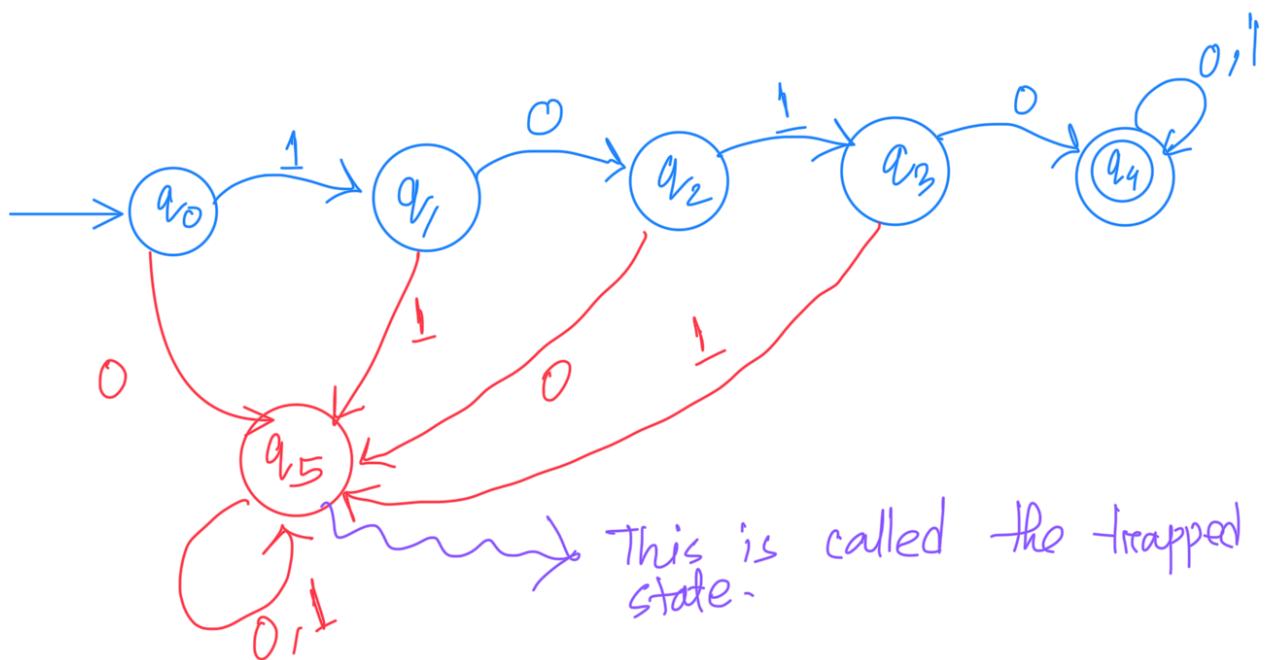


16 Construct a DFA that recognizes strings with 1010 as substrings.

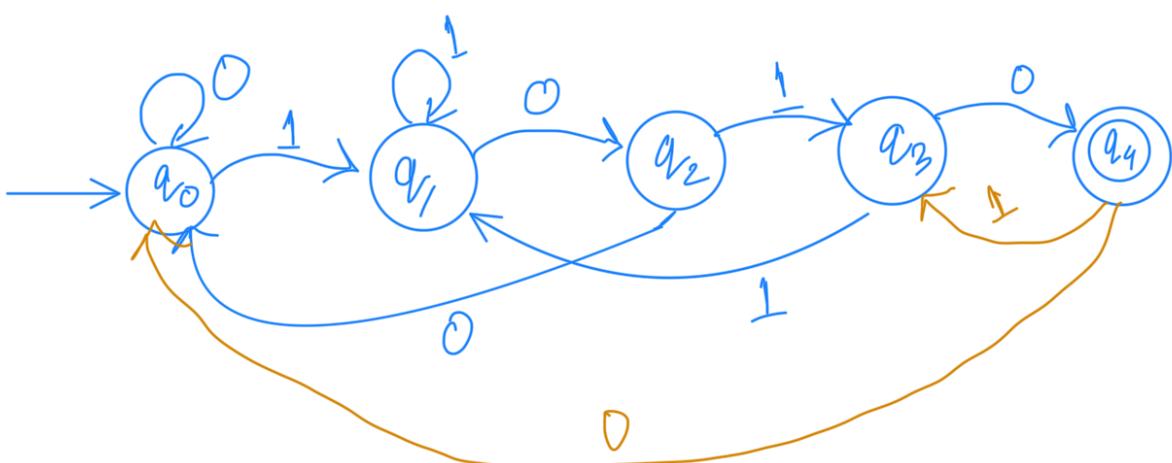


17 Construct a DFA that recognizes strings that start with 1010.

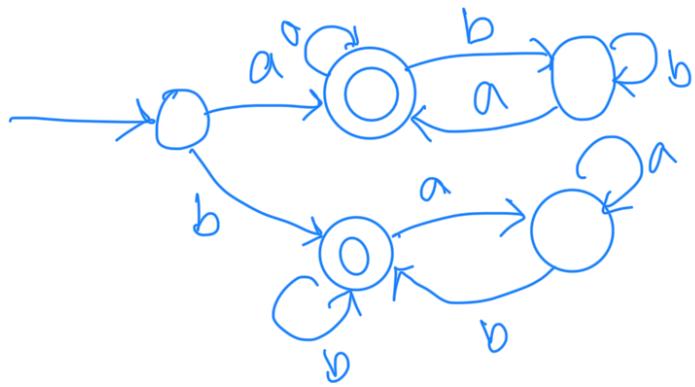
DFA = Deterministic Finite Automaton



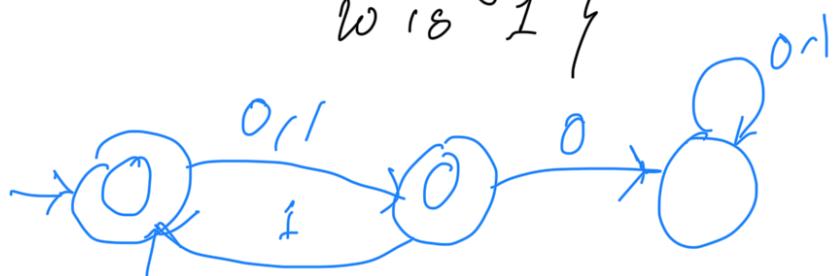
- 18 Construct a DFA that recognizes strings that end with 1010



- 19  $L = \{ w \in \{a, b\}^*, w \text{ starts and ends with same symbol} \}$



20  $L = \{w \in \{0,1\}^*: \text{every 2nd symbol in } w \text{ is 1}\}$



### Practice:

- i)  $L = \{w \in \{0,1\}^*, \text{ number of 0's in } w \text{ is not a multiple of 3}\}$
- ii)  $L = \{w \in \{0,1\}^*: w \text{ contains exactly one } 00\}$
- iii)  $L = \{w \in \{0,1\}^*, \text{ every 3rd symbol in } w \text{ is 1}\}$
- iv)  $L = \{w \in \{a,b\}^*, w \text{ contains atleast two "ab"s}\}$
- v)  $L = \{w \in \{0,1\}^*, \text{ An odd number of 0's follow the last 1}\}$

## Regular Operations:

DFA recognises regular languages. The operations that can be applied upon some regular languages and the resulting languages still be regular languages are called regular operations.

We have 3 regular operations.

- Union
- Concatenate
- Star

$A, B \Rightarrow$  regular languages

Union :  $A \cup B = \{w \mid w \in A \text{ or } w \in B\}$

Concatenate :  $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$

Star :  $A^* = \{x_1 x_2 \dots x_k \mid k \geq 0 \text{ and } x_i \in A\}$

$$= A^0 U A^1 U A^2 U A^3 U \dots$$

$$= \{ \epsilon \} \cup \{x_1 \in A\} \cup \{x_1 \in A \circ A\} \cup \{x_1 \in A \circ A \circ A\} \cup \dots$$

### Union :

Suppose DFA  $M_A$  recognizes language  $A$ .

"  $M_B$  recognizes language  $B$ .

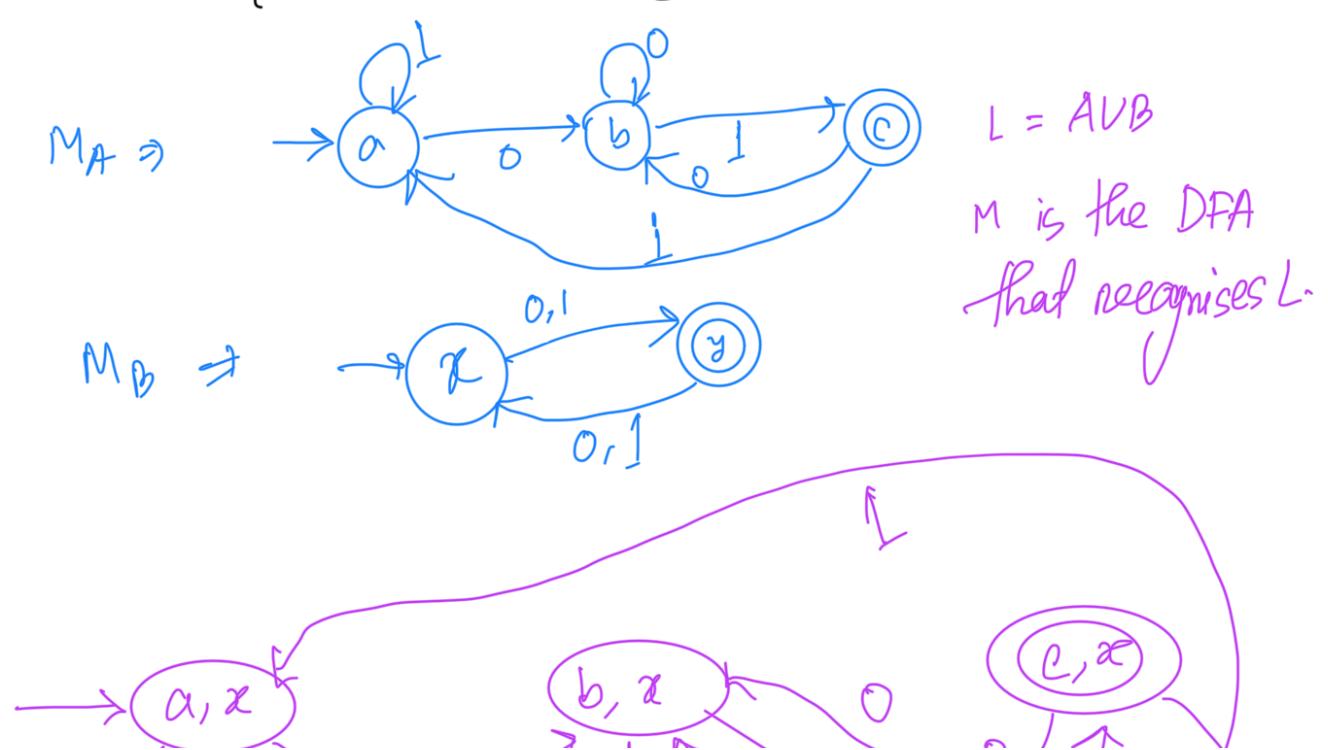
If string  $s$  is in language  $A \cup B$ , then we cannot pass  $s$  through  $M_A$  and rewind it and again pass it through  $M_B$ . It's because after reading each alphabet from  $s$ , we forget that and don't store  $s$ . So we need to pass  $s$  in  $M_A$  and  $M_B$  simultaneously. The  $\uparrow$  DFA  $M$  that recognizes  $A \cup B$  is the cartesian product of the states.

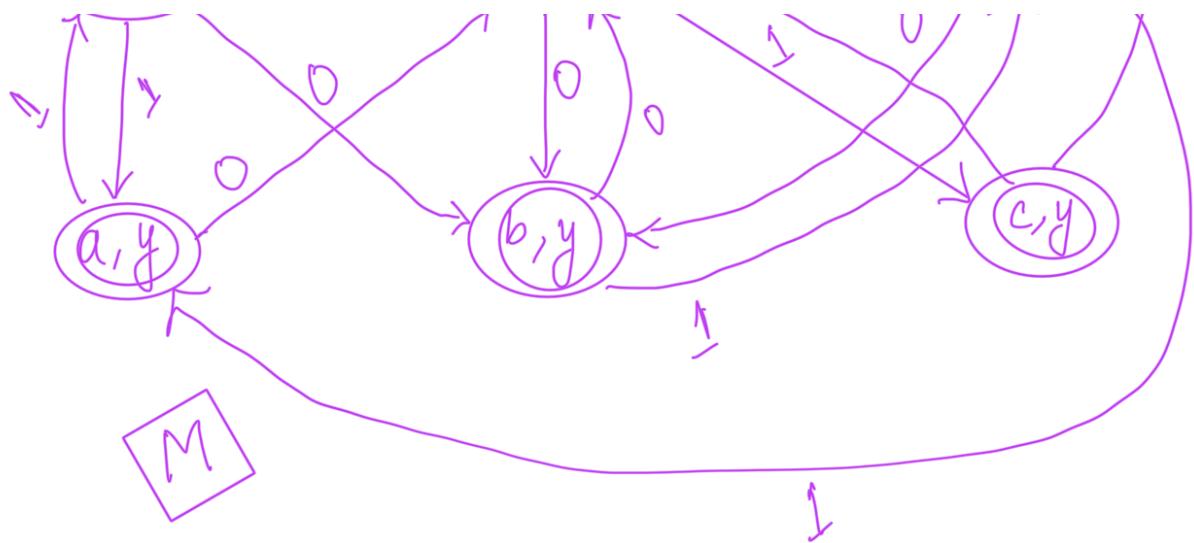
\* See an example to fully understand it.

### Example:

$$A = \{ w \in \{0,1\}^*: w \text{ ends with } 01 \}$$

$$B = \{ w \in \{0,1\}^*: \text{Length of } w \text{ is odd} \}$$

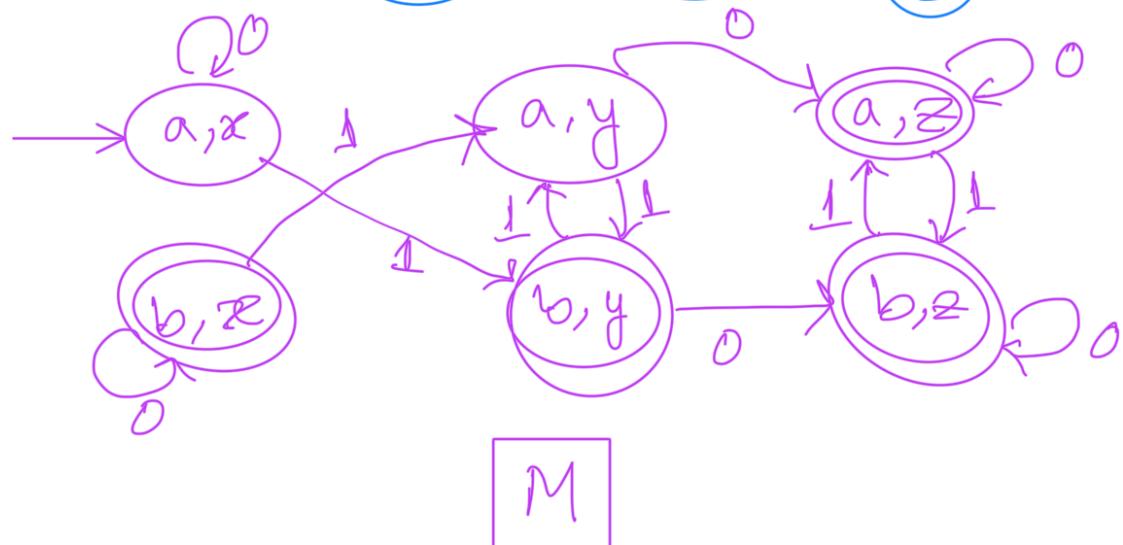
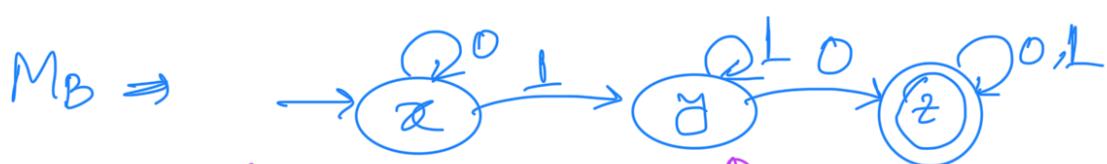




### Example 7:

$A =$  strings with odd number of 1's

$B =$  " " 10 as substring.



start state  $\Rightarrow$  that contains both a and x.  
 $(a,x)$

(---)

Accepting state  $\Rightarrow$  that contains b or  $\geq$  one both,  
 $(b,x, b,y, a,z, b,z)$

## Cross Product

Similarly, we can use AND or NOT just like Union (OR), ~~T~~ Intersection

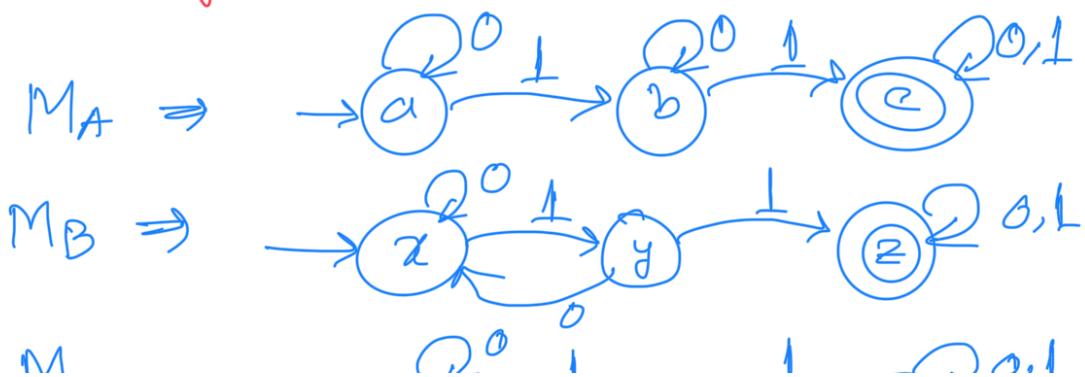
Example 8: Construct a DFA that recognizes strings with at least two 1's but no consecutive 1's.

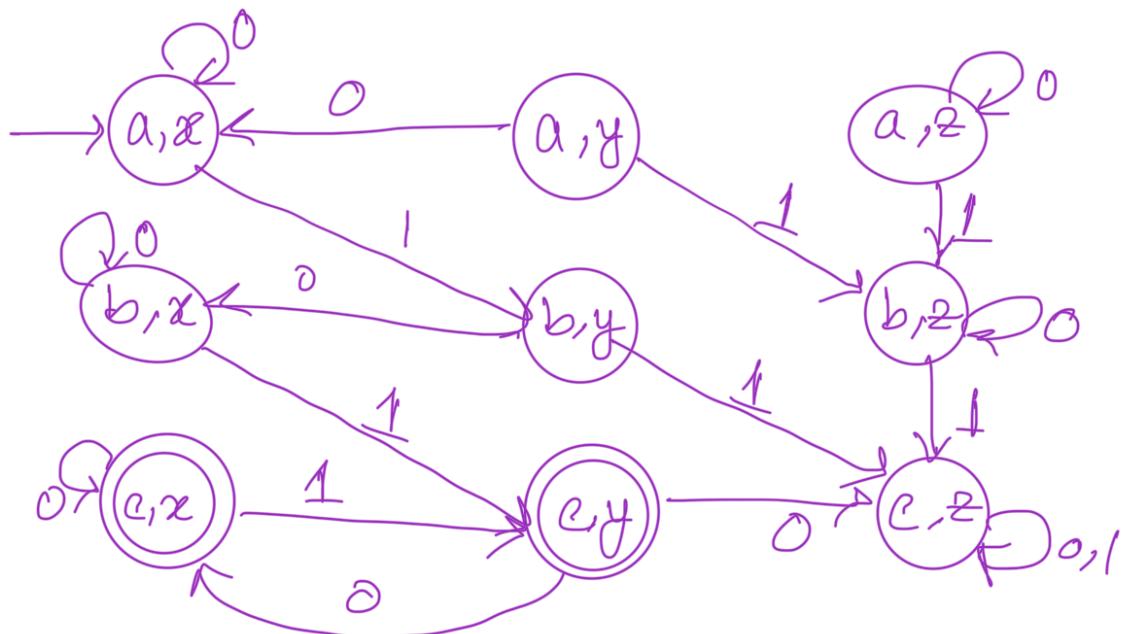
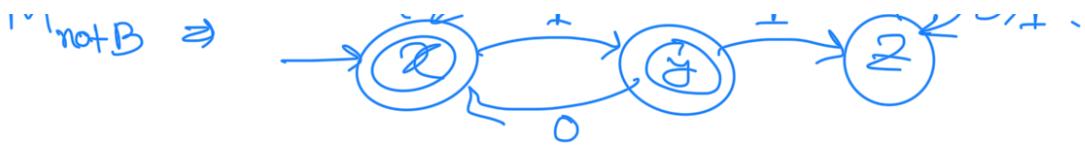
A = strings with at least two 1's

B = consecutive 1's.

NotB = no consecutive 1's

M recognizes (A and (notB))





starting state  $\Rightarrow$   $(a,x)$

Accepting state  $\Rightarrow$  that contains both { $c$  and  
{ $x$  or  $y$ }}  $\Rightarrow$   $(c,x)$   $(c,y)$

## Concatenation:

$$A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$$

The main problem is that we don't know  
which part of input string  $s$  to

Want to show any two many  $\Rightarrow$  to  
get  $x, y$  where  $x \in A$  and  $y \in B$ .

For example:

$$A = \{ w \mid w \text{ ends with } 10 \}$$

$$B = \{ w \mid \text{length odd} \}$$

Suppose input is 001101100.

We can divide it in two ways.

001101100  $\Rightarrow$  if we do that, then  
the input is not in the  
language that represents  
 $A \oplus B$ .  $x \in A$   
 $y \notin B$

But this is in fact in the language.

$$001101100 \Rightarrow x = 00110110, x \in A  
y = 0, y \in B$$

We can not explain this with the  
help of DFA.

We need NFA.