Discrete Maths

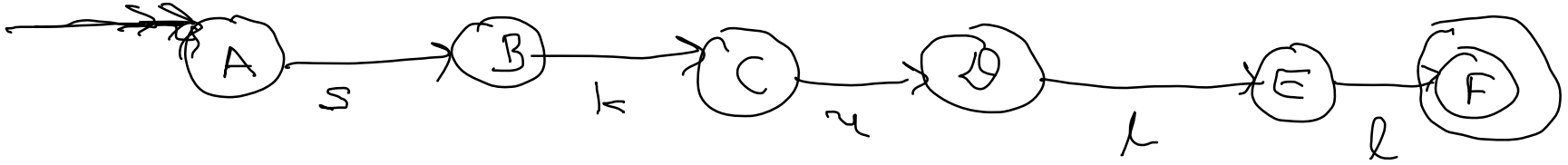
While we learn
grammar & FSM,
keep thinking of
memory,
is it involved?
If yes,
- , in which form?
If No, Why not
Isn't it needed?

FSM as language recognizer

Example

And a finite state machine/Automata that accepts a language

$L = \{skull\}$



○ represents state

(A) is initial state in this example → Entry point main fn

◎ indicates "Accepted" state

Will "sdull" is accepted by current given fsm?

~~state~~ A $\xrightarrow{S}$ B

B $\xrightarrow{d}$ There is no transition defined

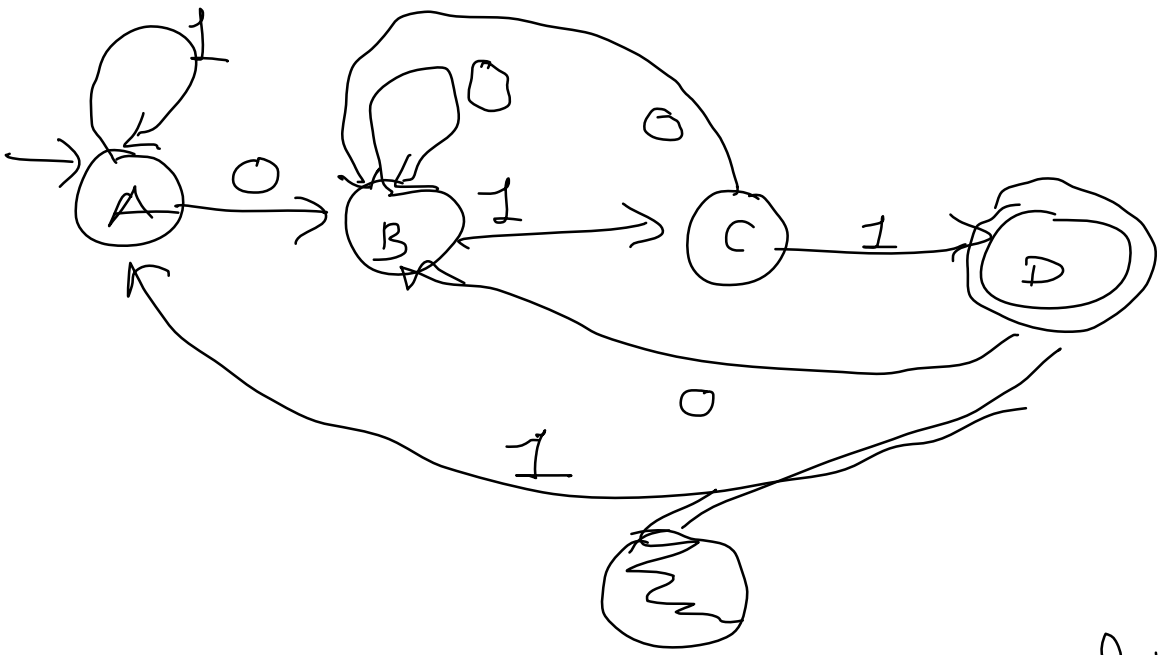
and hence, "sdull" is NOT accepted by this FSM.

Try writing grammar

Also, instead of diagram write in tabular form FSM

Example:

L = { All binary sequences that ends with 011 }



states themselves kind of memory.

Tabular form

Present state / Input current	0	1
A	B	A
B	B	C
C	B	D
D	B	A

A in initial state

D is ~~accepting~~ / Final state

Keep thinking

- Recursive Vs Nonrecursive
- Ambiguous Vs Unambiguous
- Deterministic Vs Nondeterministic
- Empty Vs Nonempty

Ex. $L = \{ a^i b^{2i} \mid i \geq 1 \}$

$Term = \{ a, b \}$ terminal

$i \in$ Natural

$Non_{Terminal} = \{ S \}$

① $S \rightarrow a S bb$

② $S \rightarrow abb$

① & ② are production rules.

derivation ~~algorithm~~

aabb

abb

aaabbb

Observation of language

- all a's must be before any of b
- number of a's are half of number b's

OR

number of b's are double than number of a's

- empty string, not accepted bcoz $i \geq 1$. not zero.

~~derivation~~

ex ① input abb

$S \rightarrow abb$ rule ②

Yes, input belongs to L.

ex derivation

② a $\notin L$

③ b $\notin L$

④ ab $\notin L$

⑤ aabbbb $\rightarrow$ a abbbb

$S \rightarrow aSbb$ (R1) $\rightarrow$ a abbbb (R2) $\in L$

✓ pattern matching

✓ searching.

```
func ( int i )
{
    if (i==1)
        print("abb")
    else {
        print("a")
        func(i-1)
        print("bb")
    }
}
```

i.e.
i = 2 ⇒
a ( ) bb