

We introduce the simplest deterministic theoretical machines: Finite Automata.





- A finite automaton (FA) is the following 3 things:
  - a finite set of states, one of which is designated as the start state, and some (maybe none) of which are designated the final states (or accepting states)
  - 2. an alphabet  $\Sigma$  of input letters
  - 3. a finite set of transitions that indicate, for each state and letter of the input alphabet, the state to go to next

state \_\_\_\_\_→ state





- The language defined or accepted by a finite automaton is the set of words that end in a final state.
- If w is in the language defined by a finite automaton, then we also say that the finite automaton accepts w.





Example:  $\Sigma = \{a,b\}$ a bstates =  $\{x,y,z\}$ x y zstart state : xtransitions:final states:  $\{z\}$ z z

aaa: 
$$x \xrightarrow{a} y \xrightarrow{a} x \xrightarrow{a} y$$
 y is not final;  
aaa is not accepted  
aaba:  $x \xrightarrow{a} y \xrightarrow{a} x \xrightarrow{b} z \xrightarrow{a} z$  z is final;  
aaba is accepted



**Transition Diagram** 





Regular expression: (**a**+**b**)\***b**(**a**+**b**)\*

aaaabba?

bbaabbbb?





Dr. Nejib Zaguia

CSI3104-W11









 $(a+b)(a+b)^* = (a+b)^+$ 

(**a**+**b**)\*





Finite Automata that Accept No Words







The middle state is not a final state and all transitions that go into this state do not exit.





# Two Ways to Study Finite Automata

- 1. Starting with a finite automaton (FA), analyze it to determine the language it accepts.
- 2. Starting from a language, build an FA.





Example with 2 states: The language of all words with an even number of letters over the alphabet {a,b}:

- Two states: 1 even number, 2 odd number
- Start state: 1
- Final state: 1
- The transitions:









# From Languages to Finite Automata

- There is not necessarily a unique FA that accepts a given language.
- Is there always at least one FA:
  - that accepts each possible language?
  - that defines a language associated with a given regular expression?





- <u>Example</u>: Build an FA that accepts all words containing a triple letter (either aaa or bbb).
  - 1. Build an FA that accepts aaa
  - 2. Add a path that accepts bbb.
  - 3. Add paths for words that contain a's and b's before or after the aaa or bbb.



# From Finite Automata to Languages



This example shows that it is possible to characterize states by the purposes they serve.

- Does this FA accept the word ababa?
- The word babbb?
- 2 ways to get to state 4
- The only way to get to state 2 is by reading an input a.
- The only way to get to state 3 is by reading an input b.
- What language?

Dr. Nejib Zaguia CSI3104-W11



- The regular expression is not unique.
- Is there always at least one regular expression defining the language accepted by an FA?





- Regular expression: **baa**
- A "collecting bucket" state for all other words.







# Regular expression: baa + ab



h



Λ



 $(a+b)*a + \Lambda$ 

Words that do not end in b.

a

?



Words that end in a.

Dr. Nejib Zaguia

CSI3104-W11







The only letter that can change state is a. (b\*ab\*)(ab\*ab\*)\*
an odd number of a's



Words that contain a double a (a+b)\*aa(a+b)\*

- Start state: the previous letter (if there is one) was not an a.
- Middle state: we have just seen an a that was not preceded by an a.
- Final state: we have seen a double a.











# The Language EVEN-EVEN

