Finite state machine

Capture patterns in behavior based on (limited) knowledge of what has happened in the past, and current input.

Abstract away details ….

- Input: an element from a finite set of symbols
- “Past”: string of input symbols
Terminology

- **Alphabet**: nonempty finite set of *symbols*
- **String over an alphabet**: finite sequence of symbols
- **Language over an alphabet**: set of strings
Deterministic finite automaton (DFA)

- Input: finite string over a fixed alphabet
- Output: "accept" or "reject"

Language of machine is set of strings it accepts (recognizes)
Deterministic finite automaton (DFA)

- Input: finite string over a fixed alphabet
- Output: "accept" or "reject"

Computation of the machine on an input string?

Each new symbol of input gives information about the string.
Deterministic finite automaton (DFA)

- Input: finite string over a fixed alphabet
- Output: "accept" or "reject"

Example input: 0001
What is the sequence of states followed on input 110?

A. q1, q1, q0  
B. q1, q1, q2  
C. q0, q1, q2, q3  
D. q0, q1, q1, q2  
E. None of the above
Deterministic finite automaton (DFA)

- Input: finite string over a fixed alphabet
- Output: "accept" or "reject"

Does this DFA accept the string 001000?

A. Yes
B. No
C. I don't know
Deterministic finite automaton (DFA)

- Input: finite string over a fixed alphabet
- Output: "accept" or "reject"

Does this DFA accept the **empty string**?

A. Yes
B. No
C. I don't know
Finite automaton

A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\) where:

1. \(Q\) is a finite set called the states
2. \(\Sigma\) is a finite set called the alphabet
3. \(\delta : Q \times \Sigma \to Q\) is the transition function
4. \(q_0 \in Q\) is the start state
5. \(F \subseteq Q\) is the set of accept states.

Sipser p. 35 Def 1.5

No circles and arrows, same information!
A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\) where

1. \(Q\) is a finite set called the states
2. \(\Sigma\) is a finite set called the alphabet
3. \(\delta : Q \times \Sigma \rightarrow Q\) is the transition function
4. \(q_0 \in Q\) is the start state
5. \(F \subseteq Q\) is the set of accept states.

Can there be more than one start state in a finite automaton?

A. Yes, because of line 4.
B. No, because of line 4.
C. I don't know
Finite automaton

A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\) where

1. \(Q\) is a finite set called the states
2. \(\Sigma\) is a finite set called the alphabet
3. \(\delta : Q \times \Sigma \rightarrow Q\) is the transition function
4. \(q_0 \in Q\) is the start state
5. \(F \subseteq Q\) is the set of accept states.

Can there be zero many accept states?

A. Yes, in which case the language is empty.
B. Yes, in which case the language is all strings.
C. No, because of line 5.
D. I don't know.
A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\) where:

1. \(Q\) is a finite set called the states
2. \(\Sigma\) is a finite set called the alphabet
3. \(\delta : Q \times \Sigma \rightarrow Q\) is the transition function
4. \(q_0 \in Q\) is the start state
5. \(F \subseteq Q\) is the set of accept states.

Can one state have two different transitions labelled with the same symbol going out of it?

A. Yes, because of line 1.
B. Yes, because of line 3.
C. No, because of line 1.
D. No, because of line 3.
E. I don't know.
Finite automaton

A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\) where

1. \(Q\) is a finite set called the states
2. \(\Sigma\) is a finite set called the alphabet
3. \(\delta : Q \times \Sigma \rightarrow Q\) is the transition function
4. \(q_0 \in Q\) is the start state
5. \(F \subseteq Q\) is the set of accept states.

How many outgoing arrows from each state?

A. May be different number at each state.
B. Must be 2.
C. Must be \(|Q|\).
D. Must be \(|\Sigma|\).
E. I don't know.
An example

What's the best description of the language recognized by this DFA?

A. Starts with b and ends with a or b
B. Starts with a and ends with a or b
C. a's followed by b's
D. More than one of the above
E. I don't know.
An example

\((\{q_1, q_2, q_3, q_4\}, \{a, b\}, \delta, q_1, \{q_4\})\)

This DFA recognizes the language of all strings of the form *a*s followed by *b*s

i.e. \(\{ a^n b^k | n,k \geq 1 \}\)
A language is **regular** if there is some finite automaton that recognizes **exactly** it.

If $A$ is the set of strings that DFA $M$ recognizes (accepts)

- We say $A$ is the **language** of $M$
- We write $L(M) = A$
- We conclude that $A$ is **regular**
An example

The language of all strings of the form a's followed by b's

i.e. \( \{ a^n b^k \mid n, k \geq 1 \} \)

is a regular language since it is recognized by this DFA.
Another example

What is the best description of the regular language recognized by this automaton?

A. $\{ a^n b^k | n,k \geq 1 \}$
B. $\{ a^n b^k | n \geq 1, k \geq 0 \}$
C. $\{ awb | w \in \{a,b\}^* \}$
D. $\{ aw | w \in \{a,b\}^* \}$
E. I don't know
Specifying an automaton

\( ( \{q1,q2,q3\}, \{a,b\}, \delta, q1, F ) \)

How can we represent \( \delta \) for this machine?

A. \( q1 \rightarrow b, q1 \rightarrow a, q2 \rightarrow a, q2 \rightarrow b, q3 \rightarrow a, \nu \).

B. \( \{ (q1,b,q1), (q1,a,q2), (q2,a,q2), (q2,b,q3), (q3,a,q3), (q3,b,q3) \} \)

C. \( \delta(b) = \text{same}, \delta(a) = \text{change} \)

D. No description other than the state diagram (circles & arrows) is possible.

E. I don't know.
Specifying an automaton

\[
( \{q_1,q_2,q_3\}, \{a,b\}, \delta, q_1, F )
\]

What state(s) should be in the set \(F\) so that the language of this machine is \(\{ w | \text{ab is a substring of w} \}\)?

A. \(F = \{q_2\}\)  
B. \(F = \{q_3\}\)  
C. \(F = \{q_1,q_2\}\)  
D. \(F = \{q_1,q_3\}\)  
E. I don't know.
Specifying an automaton

( \{q1,q2,q3\}, \{a,b\}, \delta, q1, F )

What state(s) should be in the set \( F \) so that the language of this machine is \( \{ w \mid \text{b's never occur after a's in } w \} \)?

A. \( F = \{q2\} \)
B. \( F = \{q3\} \)
C. \( F = \{q1,q2\} \)
D. \( F = \{q1,q3\} \)
E. I don't know.