CMSC 330: Organization of Programming Languages

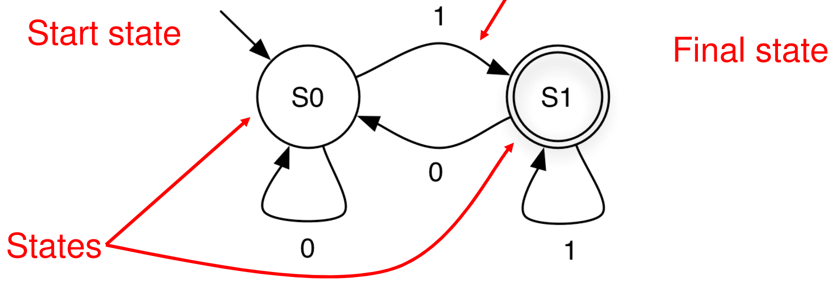
Finite Automata

Implementing Regular Expressions

- We can implement regular expressions by turning them into a *finite automaton*
  - a "machine" for recognizing a language

Example

Transition on 1



Start state

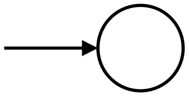
Final state

States

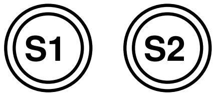
- Machine starts in *start* or *initial* state as current state
- Repeat until the end of the string is reached:
  - scan the next symbol s of the string
  - take transition edge from the current state labeled with s
- The string is *accepted* if the automaton is in a *final* or *accepting* state when the end of the string is reached

Finite Automata: States

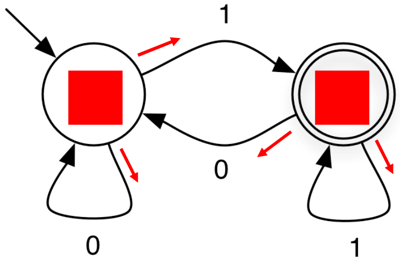
- Start state
  - A state that has an incoming transition from no other state
  - Can have only 1 start state



- Final state
  - State with double circle
  - Can have 0 or more final states



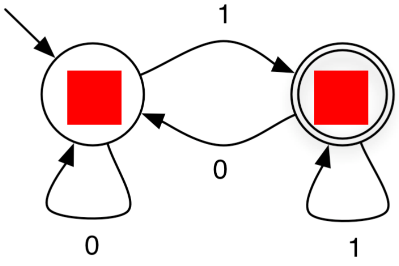
Example



0 0 1 0 1 1

accepted

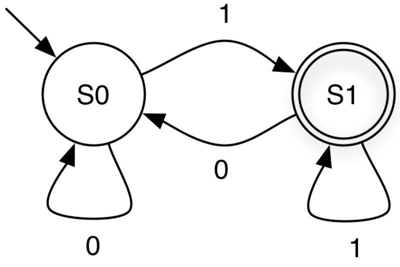
Example



0 0 1 0 1 0

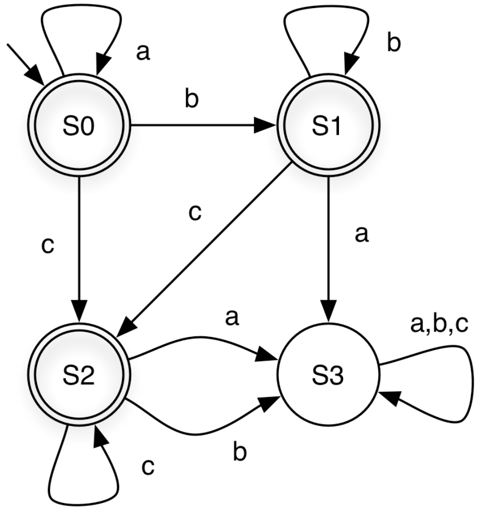
not accepted

What Language is This?



- All strings over {0, 1} that end in 1
- What is a regular expression for this language?
  - (0|1)*1

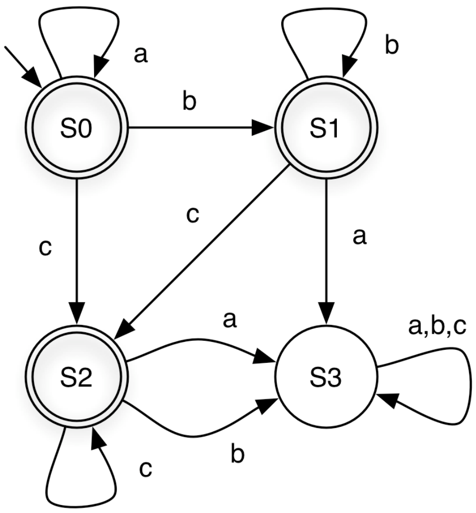
Another Example



string	state at end	accepts ?
aabcc	S2	Y
acc	S2	Y
bbc	S2	Y
aabbb	S1	Y
aa	S0	Y
ε	S0	Y
acba	S3	N

(a,b,c notation shorthand for three self loops)

Another Example (cont'd)

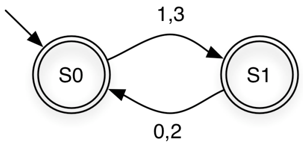


What language does this DFA accept? a*b*c*

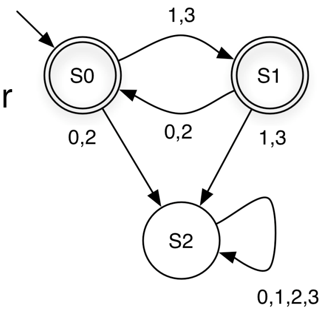
S3 is a *dead state* – a nonfinal state with no transition to another state

Shorthand Notation

- If a transition is omitted, assume it goes to a dead state that is not shown



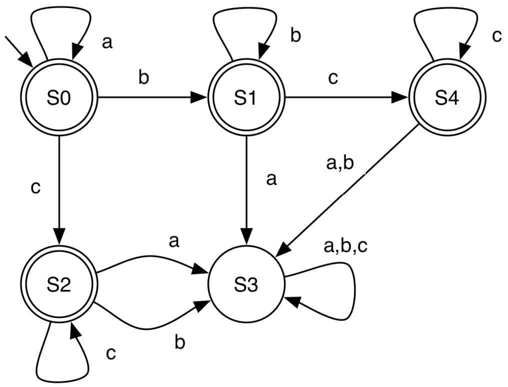
is short for



Language?

Strings over {0,1,2,3} with alternating even and odd digits, beginning with odd digit

What Language Does This DFA Accept?

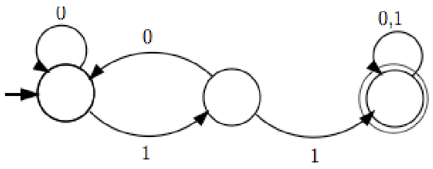


a*b*c* again, so DFAs are not unique

Practice

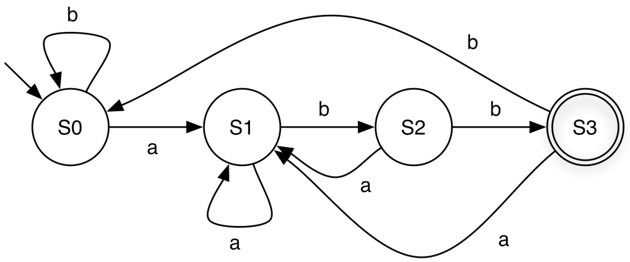
Give the English descriptions and the DFA or regular expression of the following languages:

- ((0|1)(0|1)(0|1)(0|1)(0|1))*
  - All strings of binary digits with length a multiple of 5
- (01)*|(10)*|(01)*0|(10)*1
  - All alternating binary digit strings



All binary strings containing the substring "11"

Example DFA



- Language?
  - S0 = “Haven’t seen anything yet” OR “Seen zero or more b’s” OR “Last three symbols seen were abbb”
  - S1 = “Last symbol seen was an a”
  - S2 = “Last two symbols seen were ab”
  - S3 = “Last three symbols seen were abb”
- (a|b)*abb

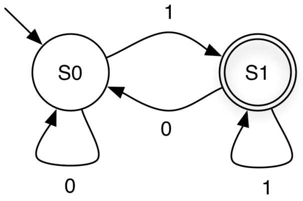
Formal Definition

- A *deterministic finite automaton (DFA)* is a 5-tuple $(\Sigma, Q, q_0, F, \delta)$ where
  - Σ is an alphabet
    - the strings recognized by the DFA are over this set
  - Q is a nonempty set of states
  - $q_0 \in \in Q$ is the start state
  - $F \subseteq Q$ is the set of final states
    - how many can there be?
  - $\delta : Q \times \Sigma \to Q$ specifies the DFA's transitions
    - what's this definition saying that δ is?

Our First Example, Formally

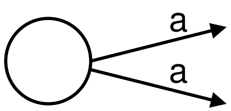
- $\Sigma = \{0, 1\}$
- $Q = \{S0, S1\}$
- $q_0 = S0$
- $F = \{S1\}$
-

δ	0	1
S0	S0	S1
S1	S0	S1

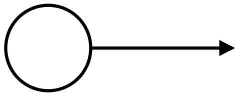


DFA Requirements

- Can't have more than one transition leaving a state on the same symbol
  - I.e., the transition function must be a valid function



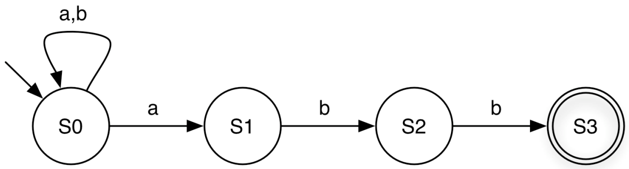
- Can't have transitions with empty labels
  - Each transition must be labeled by a single alphabet symbol



Nondeterministic Finite Automata (NFA)

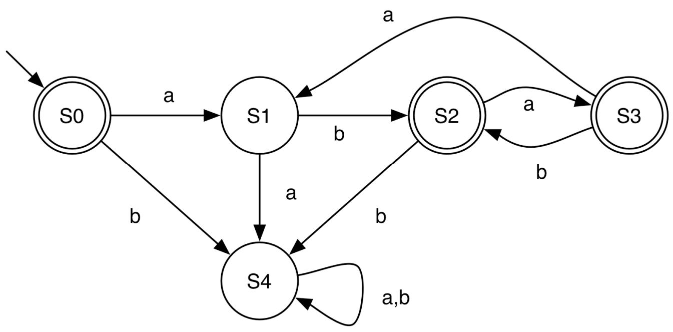
- An *NFA* is a 5-tuple $(\Sigma, Q, q_0, F, \delta)$ where
  - Σ is an alphabet
  - Q is a nonempty set of states
  - $q_0 \in Q$ is the start state
  - $F \subseteq Q$ is the set of final states
  - $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ specifies the NFA's transitions
- An NFA accepts a string s if there is *at least one* path from its start to final state on s

NFA for (a|b)*abb



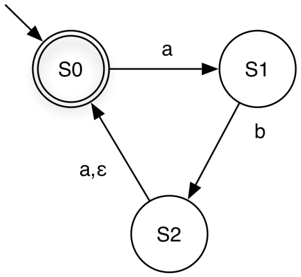
- ba
  - Has paths to either S0 or S1
  - neither is final, so rejected
- babaabb
  - Has paths to different states
  - one leads to S3, so accepted

Another example DFA



- Language?
- (ab|aba)*

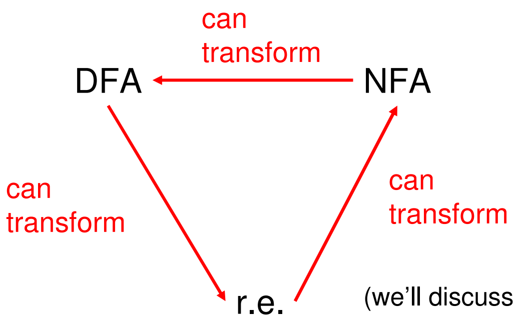
NFA for (ab|aba)*



- aba
  - has paths to states S0, S1
- ababa
  - has paths to S0, S1
  - need to use ε-transition

Relating R.E.'s to DFAs and NFAs

- Regular expressions, NFAs, and DFAs accept the same languages!



(we'll discuss this next)