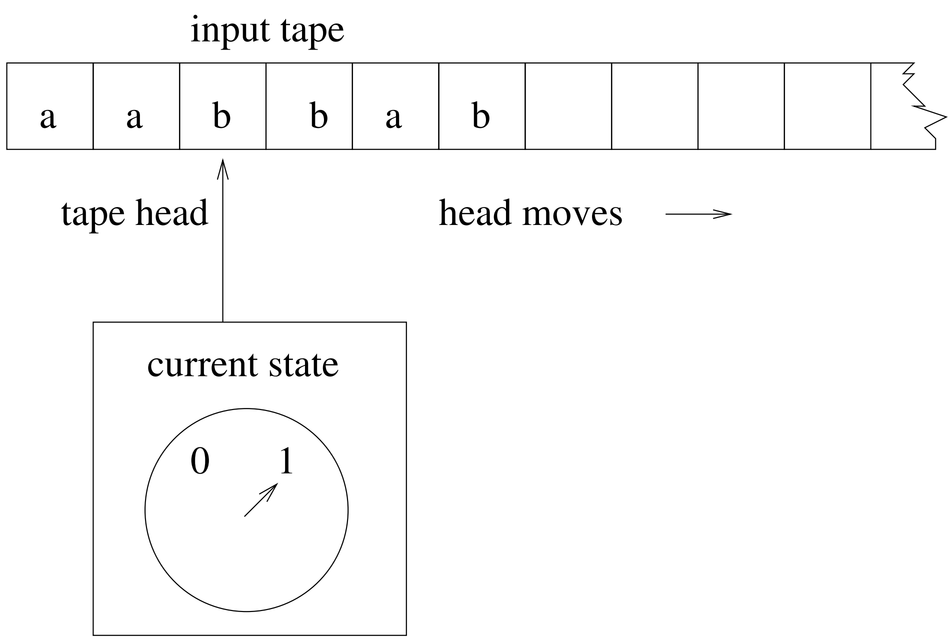
Section: Finite Automata

Deterministic Finite Accepter (or Automata)

A DFA=(Q,Σ,δ,q_0,F)



where

Q is finite set of states
Σ is tape (input) alphabet
q_0 is initial state
F $\subseteq$ Q is set of final states.
δ:Q$\times\Sigma\rightarrow$Q

Example: DFA that accepts even binary numbers.

Transition Diagram:

diagram on next page

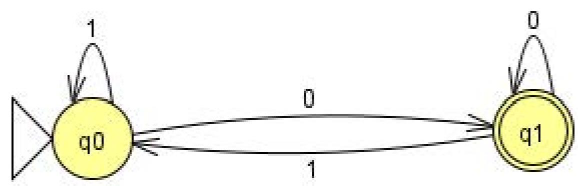
M=(Q,Σ,δ,q_0,F) = ({q_0, q_1}, {0,1}, δ, q_0, {q_1})

Tabular Format

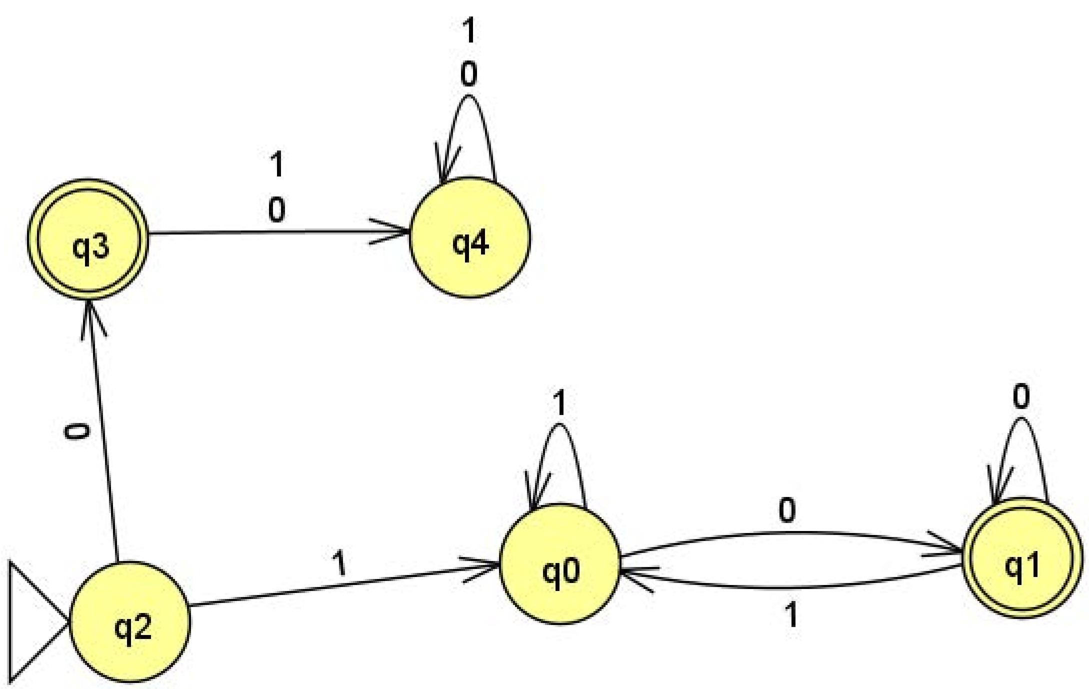
	0	1
q0	q_1	q_0
q1	q_1	q_0

Example of a move: δ**(q0,1)=** q_0

DFA for even binary numbers (allowing leading zeros)



DFA for even binary numbers, not allowing leading zeros, and showing a trap state (q4).



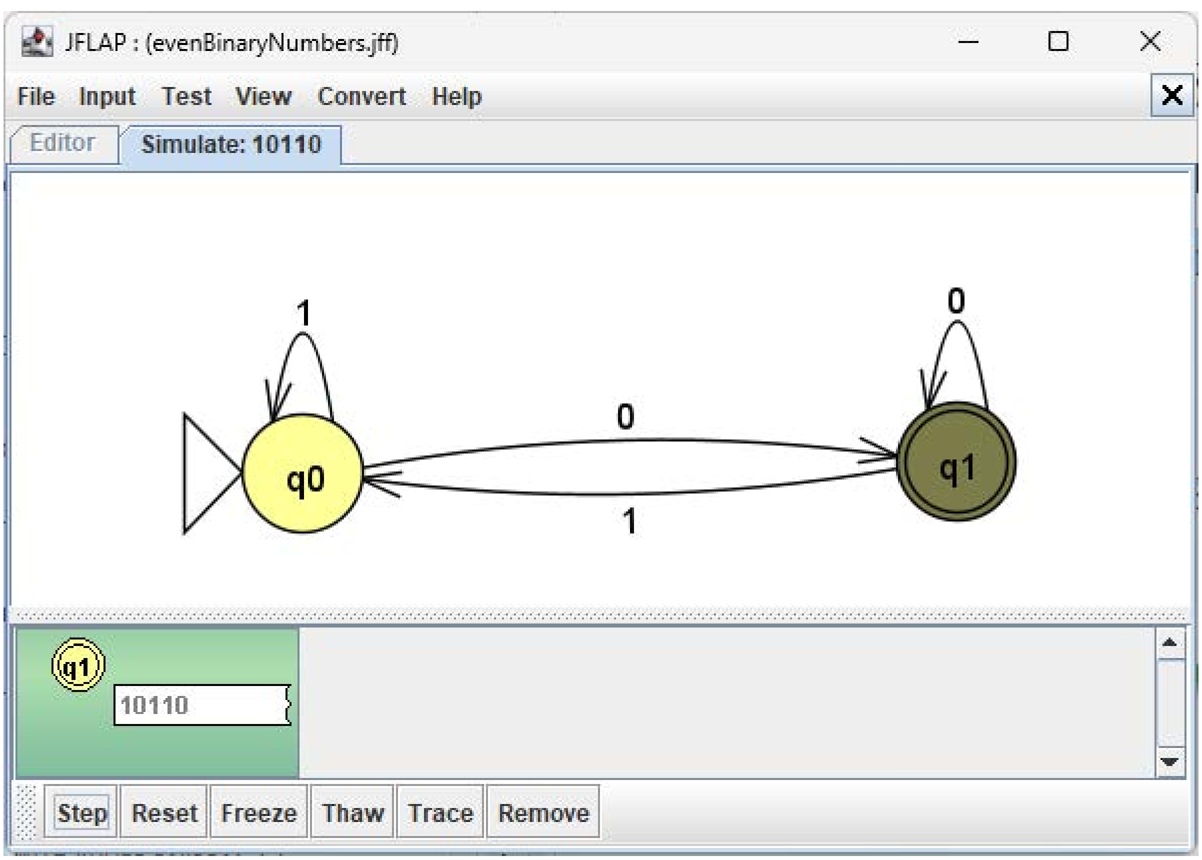
You do not have to show trap states in this course.

Algorithm for DFA:

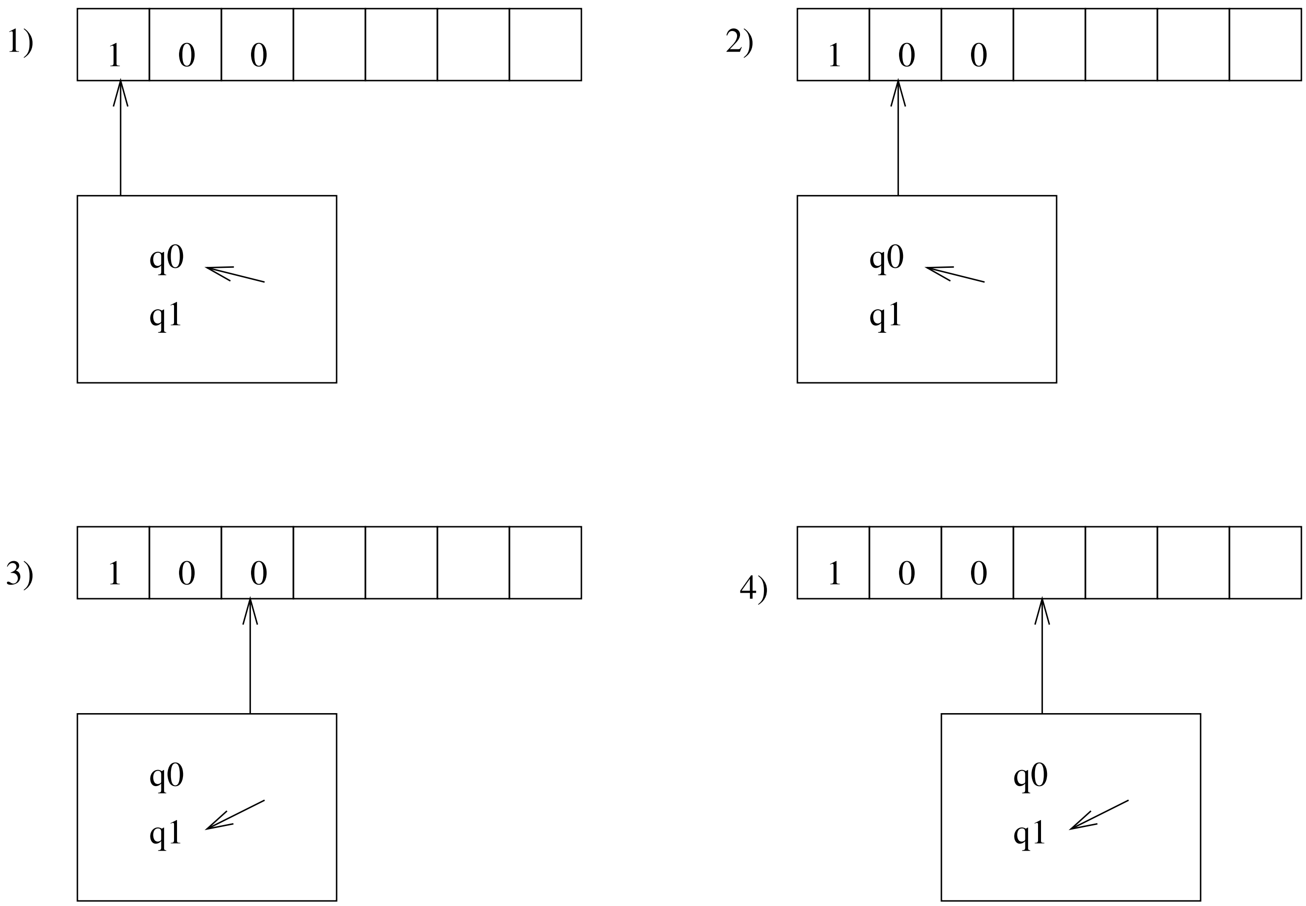
Start in start state with input on tape
q = current state
s = current symbol on tape
while (s != blank) do
  q = δ(q,s)
  s = next symbol to the right on tape
if q$\in$F then accept

Example of a trace: 11010

Did in JFLAP



Pictorial Example of a trace for 100:



Definition:

$$\delta^*(q, \lambda) = q$$

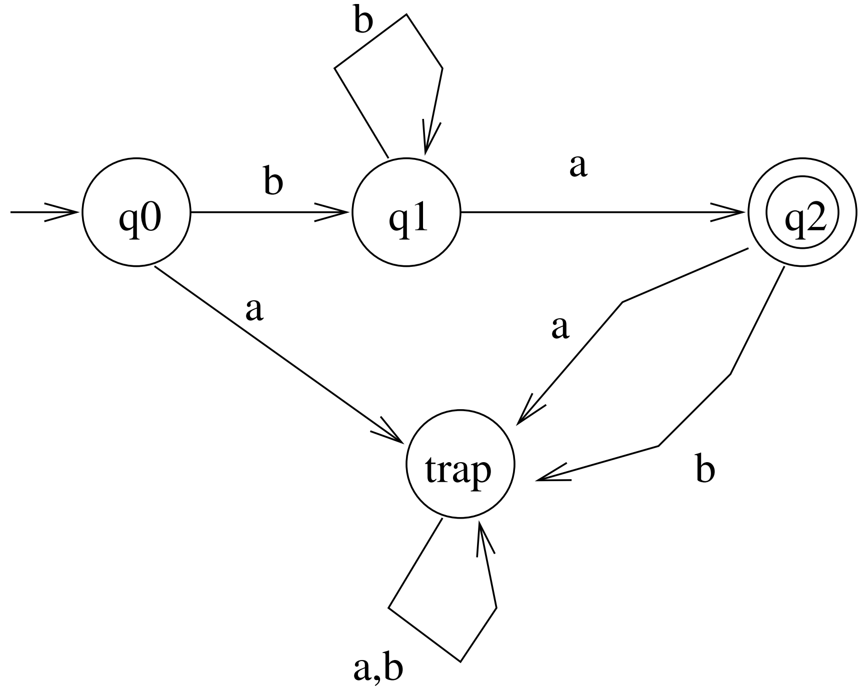
$$\delta^*(q, wa) = \delta(\delta^*(q, w), a)$$

Definition The language accepted by a DFA M=(Q,Σ,δ,q_0,F) is set of all strings on Σ accepted by M. Formally,

L(M)=$\{w \in \Sigma^* \mid \delta^*(q_0, w) \in F\}$

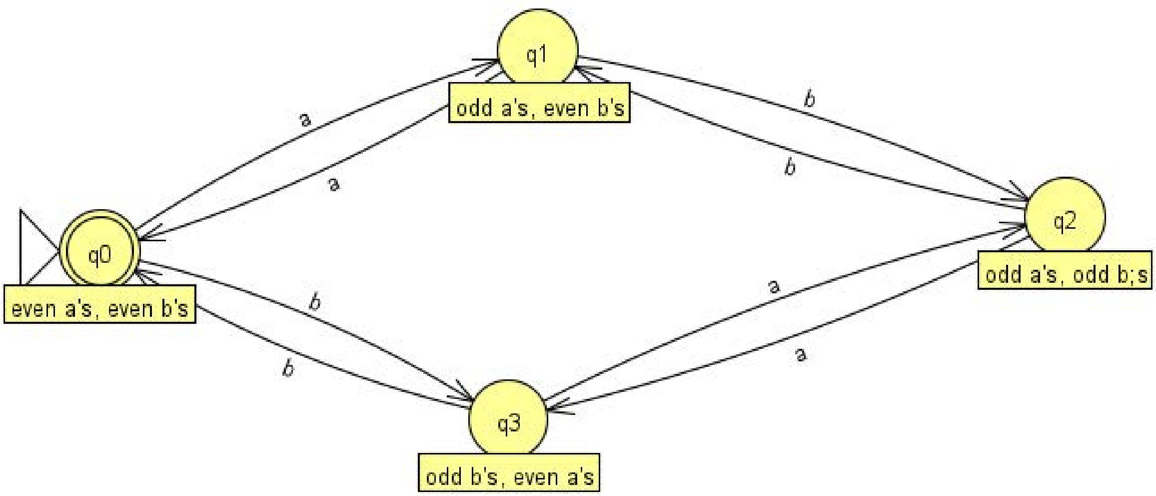
Trap State

Example: L(M) = $\{b^n a \mid n > 0\}$

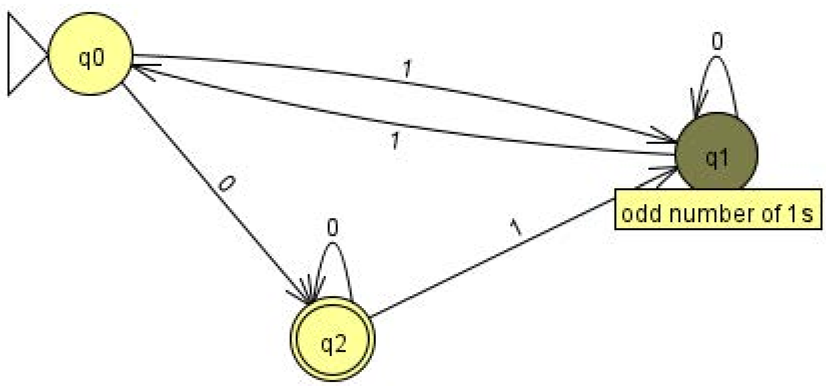


Example:

L $= \{w \in \Sigma^* \mid$ w has an even number of a's and an even number of b's$\}$



Example: DFA that accepts even binary numbers that have an even number of 1's.



Note this has leading zeros, how would you do it without the leading zeros?

Definition A language L is regular iff there exists DFA M s.t. L=L(M).

Stopped here!

Chapter 2.2

Nondeterministic Finite Automata (or Accepter)

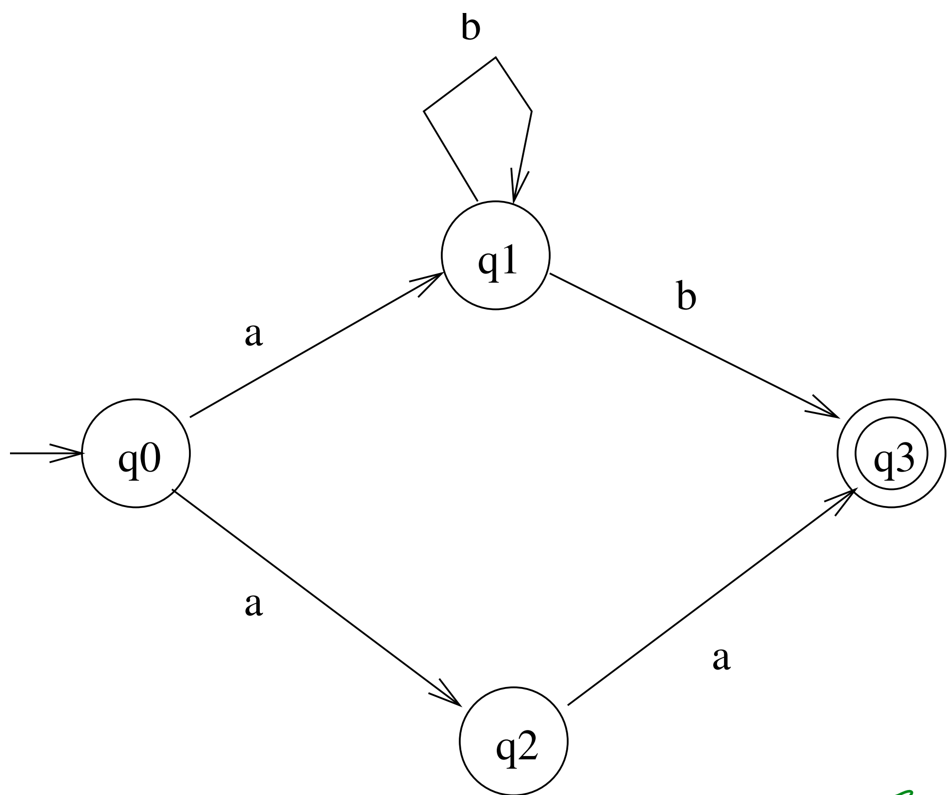
Definition

An NFA=(Q,Σ,δ,q_0,F)

where

Q is finite set of states
Σ is tape (input) alphabet
q_0 is initial state
F $\subseteq$ Q is set of final states.
δ:Q$\times(\Sigma \cup \{\lambda\}) \rightarrow 2^Q$

Example

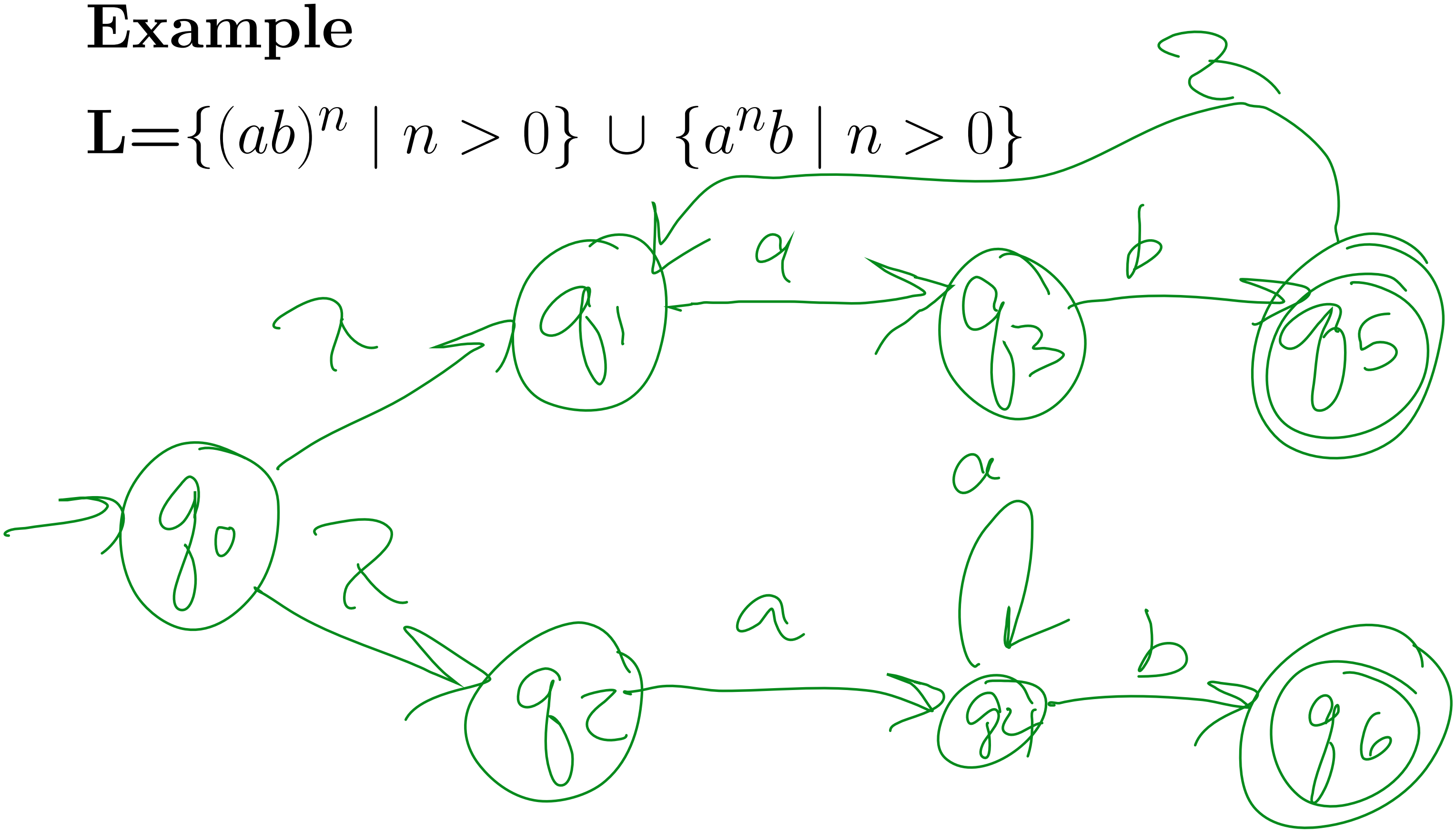


Note: In this example $\delta(q_0, a) = \{q_1, q_2\}$

L= $\{aa\} \cup \{ab^n \mid n \geq 0\}$

Example

$\mathbf{L}=\{(ab)^n \mid n > 0\} \cup \{a^n b \mid n > 0\}$



abab ∈ L b, λ ∉ L

Definition $q_j \in \delta^*(q_i, w)$ **if and only if there is a walk from** q_i **to** q_j **labeled** w**.**

Example From previous example:

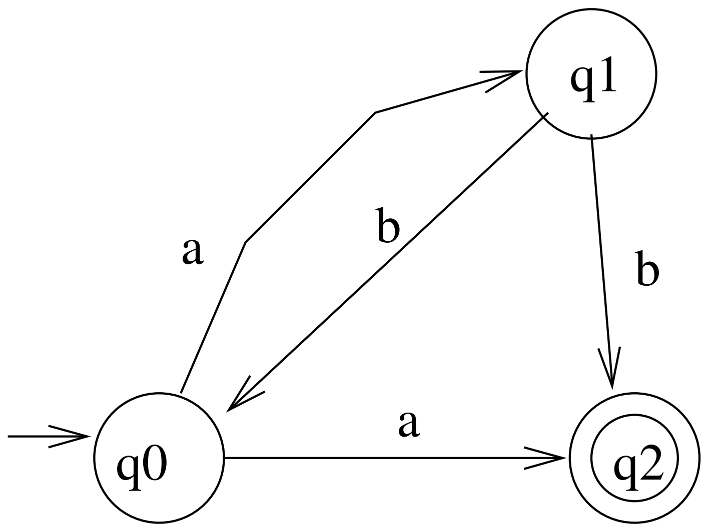
$\delta^*(q_0, ab)=$ {q1, q5, q6}

$\delta^*(q_0, aba)=$ {q3}

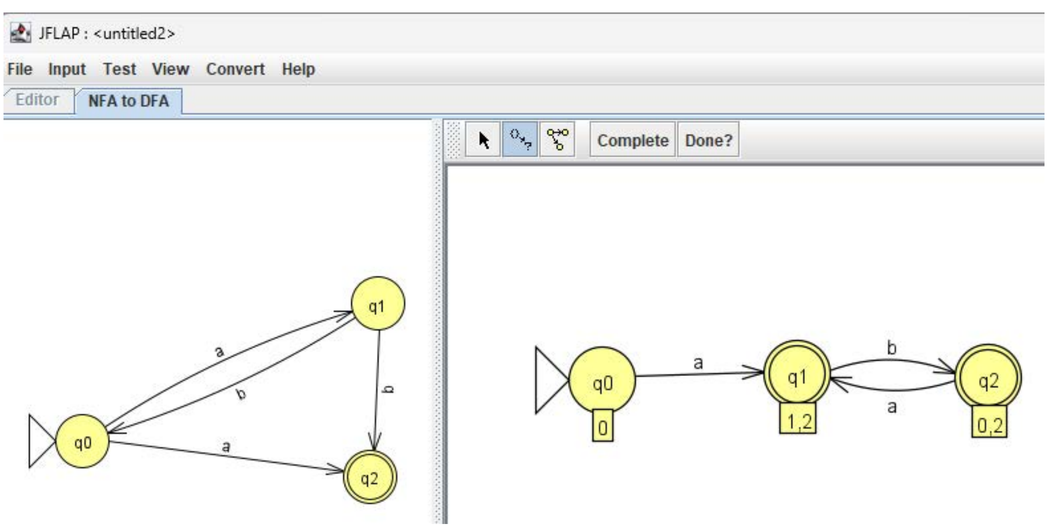
Definition: For an NFA M,
L(M)=$\{w \in \Sigma^* \mid \delta^*(q_0, w) \cap F \neq \emptyset\}$

2.3 NFA vs. DFA: Which is more powerful?

Example:



Convert NFA to DFA:



Theorem Given an NFA $M_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$, then there exists a DFA $M_D = (Q_D, \Sigma, \delta_D, q_0, F_D)$ such that $L(M_N) = L(M_D)$.

Proof:

We need to define M_D based on M_N.

$Q_D = 2^{Q_N}$

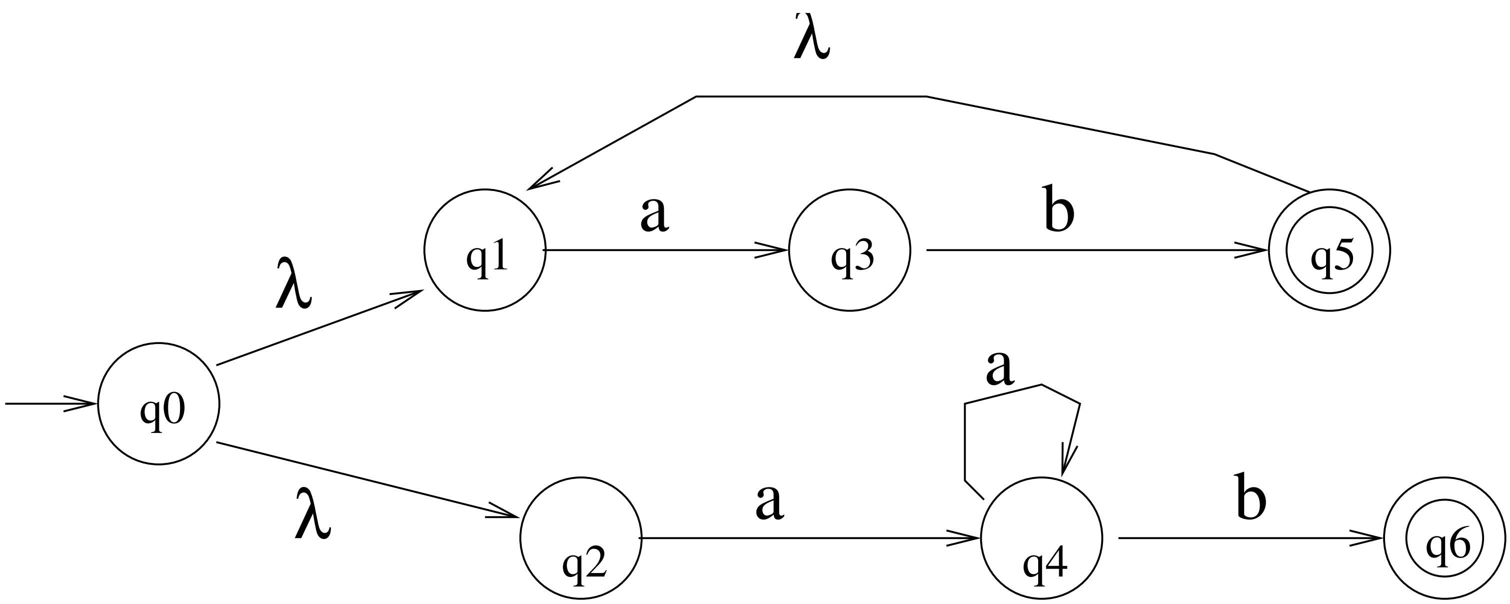
$F_D = \{ Q \in Q_D \mid \exists q_i \in Q \text{ with } q_i \in F_N \}$

$\delta_D : Q_D \times \Sigma \rightarrow Q_D$

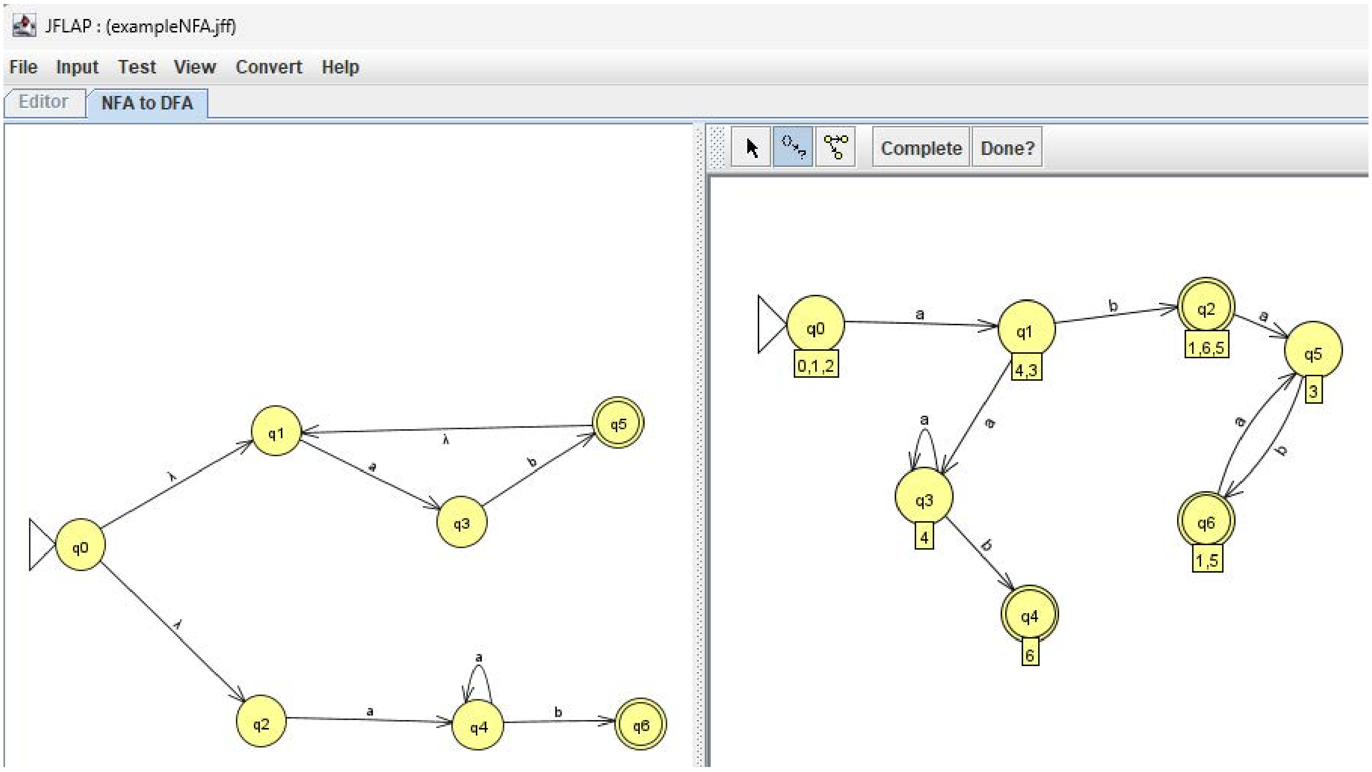
Algorithm to construct M_D

1. start state is $\{q_0\} \cup$ closure(q_0) ← include any states reachable by λ
2. While can add an edge
   (a) Choose a state A=$\{q_i, q_j, ...q_k\}$ with missing edge for $a \in \Sigma$
   (b) Compute B = $\delta^*(q_i, a) \cup \delta^*(q_j, a) \cup \ldots \cup \delta^*(q_k, a)$
   (c) Add state B if it doesn't exist
   (d) add edge from A to B with label a
3. Identify final states
4. if $\lambda \in L(M_N)$ then make the start state final.

Example:



Convert NFA to DFA:



Properties and Proving - Problem 1

Consider the property Replace_one_a_with_b or R1awb for short. If L is a regular, prove R1awb(L) is regular.

The property R1awb applied to a language L replaces one a in each string with a b. If a string does not have an a, then the string is not in R1awb(L).

Example 1: Consider L=$\{aaab, bbaa\}$

R1awb(L)= {baab, abab, aabb, bbba, bbab}

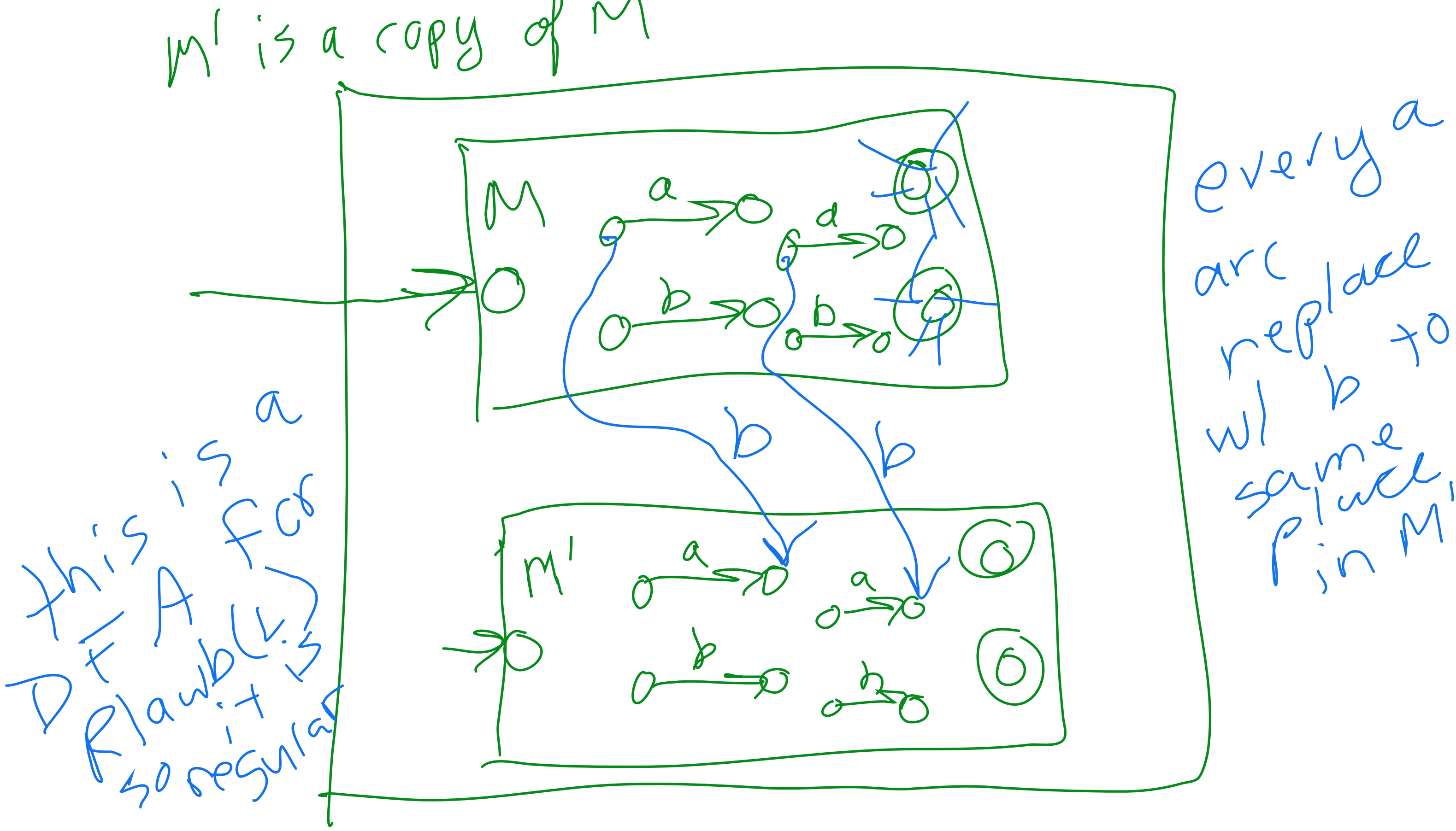
Example 2: Consider $\Sigma = \{a, b\}$, L = $\{w \in \Sigma^* \mid$ w has an even number of a's and an even number of b's$\}$

R1awb(L)= {w ∈ Σ* | w has an odd number of a's and an odd number of b's}

Proof:

assume L is regular

⇒ DFA M



See handout on January 15 on our calendar page on how to write this up as a proof.

STOPPED HERE

Properties and Proving - Problem 2

Consider the property Truncate_all_preceeding_b's or TruncPreb for short. If L is a regular, prove TruncPreb(L) is regular.

The property TruncPreb applied to a language L removes all preceeding b's in each string. If a string does not have an preceeding b, then the string is the same in TruncPreb(L).

Example 1: Consider L=$\{aaab, bbaa\}$

TruncPreb(L)=

Example 2: Consider L = $\{(bba)^n \mid n > 0\}$

TruncPreb(L)=

Proof:

Minimizing Number of states in DFA

Why?

Algorithm

- **Identify states that are indistinguishable**

  These states form a new state

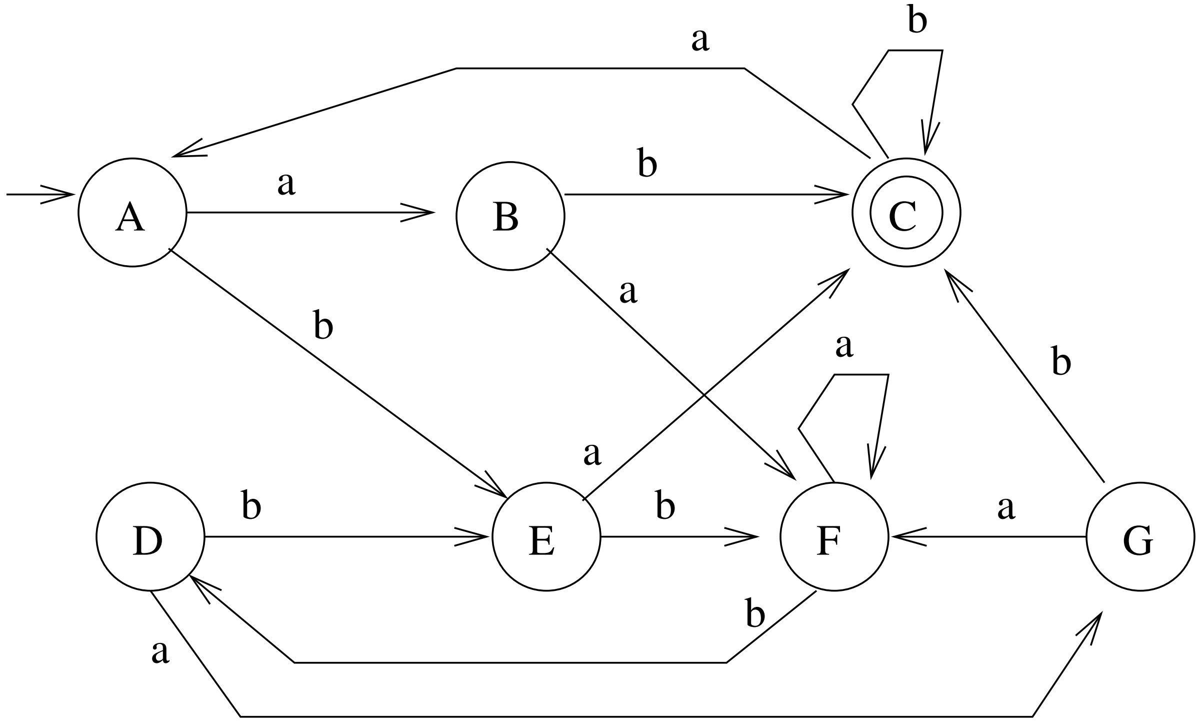
Definition Two states p and q are indistinquishable if for all $w \in \Sigma^*$

$$\delta^*(q, w) \in F \Rightarrow \delta^*(p, w) \in F$$
$$\delta^*(p, w) \notin F \Rightarrow \delta^*(q, w) \notin F$$

Definition Two states p and q are distinquishable if $\exists\ w \in \Sigma^*$ **s.t.**

$\delta^*(q, w) \in F \Rightarrow \delta^*(p, w) \notin F$ **OR**
$\delta^*(q, w) \notin F \Rightarrow \delta^*(p, w) \in F$

Example:



Example: