Nondeterministic Finite State Machines

Read K & S 2.2, 2.3
Read Supplementary Materials: Regular Languages and Finite State Machines: Proof of the Equivalence of Nondeterministic and Deterministic FSAs.
Do Homework 6.

Definition of a Nondeterministic Finite State Machine (NDFSM/NFA)

M = (K, Σ, Δ, s, F), where

K is a finite set of states
Σ is an alphabet
$s \in K$ is the initial state
$F \subseteq K$ is the set of final states, and
Δ is the transition *relation*. It is a finite subset of

$$(K \times (\Sigma \cup \{\varepsilon\})) \times K$$

i.e., each element of Δ contains:

a configuration (state, input symbol or ε), and a new state.

M accepts a string w if there exists *some path* along which w drives M to some element of F.

The language accepted by M, denoted L(M), is the set of all strings accepted by M, where computation is defined analogously to DFSMs.

A Nondeterministic FSA

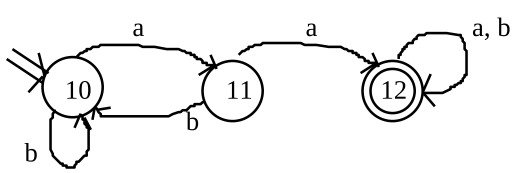
L= {w : there is a symbol $a_i \in \Sigma$ not appearing in w}

The idea is to guess (nondeterministically) which character will be the one that doesn't appear.

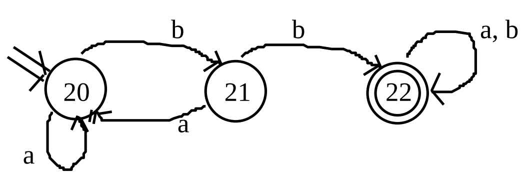
Another Nondeterministic FSA

L_1= {w : aa occurs in w}
L_2= {x : bb occurs in x}
L_3= {y : $\in L_1$ or L_2 }

M_1 =

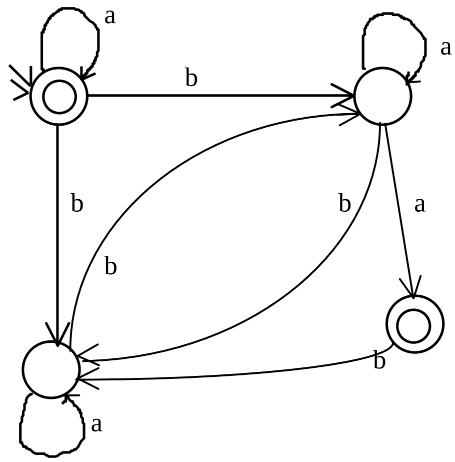


M_2=



M_3=

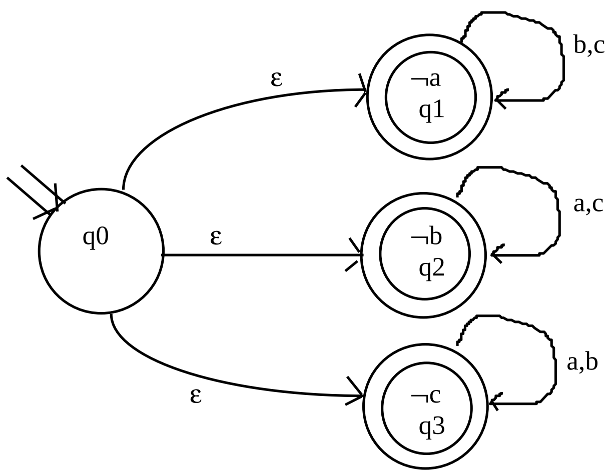
Analyzing Nondeterministic FSAs



Does this FSA accept: baaba
Remember: we just have to find one accepting path.

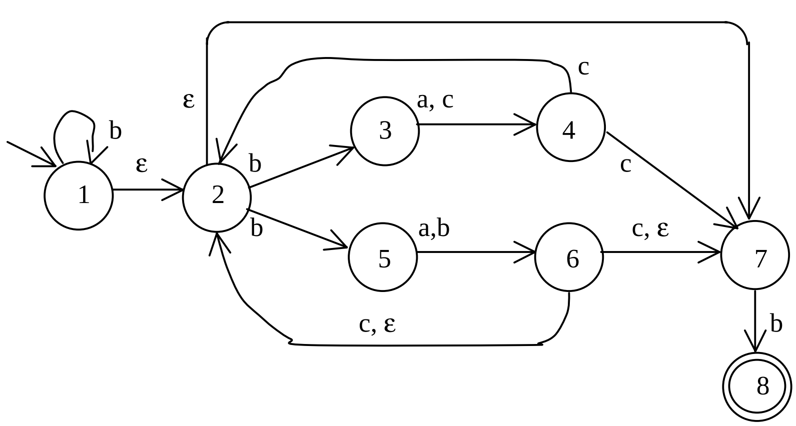
Nondeterministic and Deterministic FSAs

Clearly, {Languages accepted by a DFSA} ⊆ {Languages accepted by a NDFSA}
(Just treat δ as Δ)
More interestingly, **Theorem**: For each NDFSA, there is an equivalent DFSA.
Proof: By construction

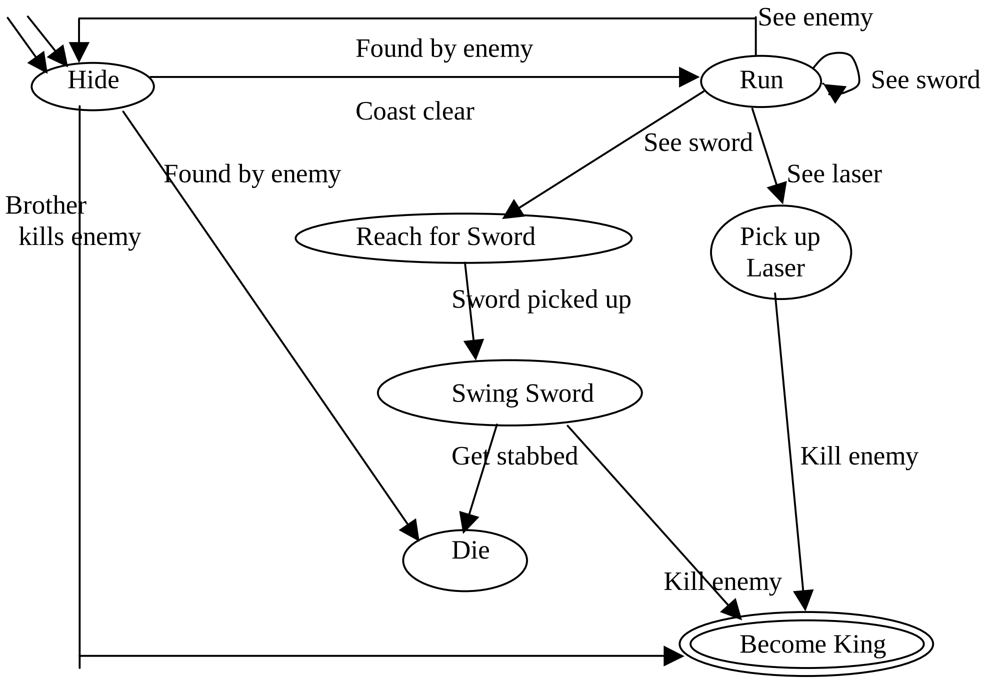


Another Nondeterministic Example

b* (b(a ∪ c)c ∪ b(a ∪ b) (c ∪ ε))* b



A "Real" Example



Dealing with ε Transitions

$E(q) = \{p \in K : (q,w) \vdash^*_M (p, w\}$. E(q) is the closure of {q} under the relation $\{(p,r) :$ there is a transition $(p, \varepsilon, r) \in \Delta\}$
An algorithm to compute E(q):

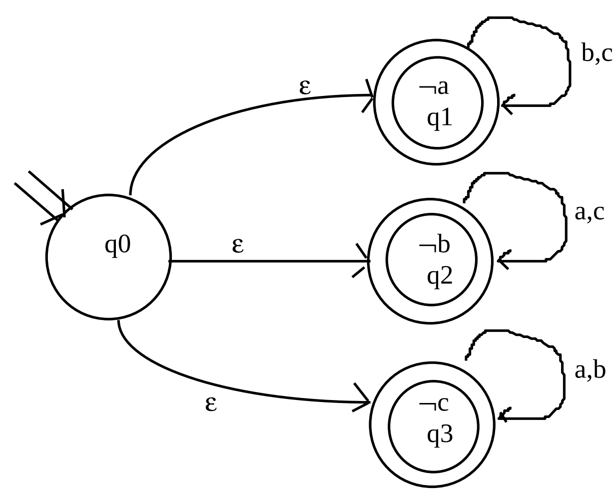
Defining the Deterministic FSA

Given a NDFSA $M = (K, \Sigma, \Delta, s, F)$,
we construct $M' = (K', \Sigma, \delta', s', F')$, where
$K' = 2^K$
$s' = E(s)$
$F' = \{Q \subseteq K : Q \cap F \neq \emptyset\}$
$\delta'(Q, a) = \cup\{E(p) : p \in K$ and $(q, a, p) \in \Delta$ for some $q \in Q\}$

Example: computing δ' for the missing letter machine

s' = {q0, q1, q2, q3}

δ' = { ({q0, q1, q2, q3}, a, {q2, q3}),
({q0, q1, q2, q3}, b, {q1, q3}),
({q0, q1, q2, q3}, c, {q1, q2}),
({q1, q2}, a, {q2}), ({q1, q2}, b, {q1}), ({q1, q2}, c, {q1, q2})
({q1, q3}, a, {q3}), ({q1, q3}, b, {q1, q3}), ({q1, q3}, c, {q1})
({q2, q3}, a, {q2, q3}), ({q2, q3}, b, {q3}), ({q2, q3}, c, {q2})
({q1}, b, {q1}), ({q1}, c, {q1})
({q2}, a, {q2}), ({q2}, c, {q2})
({q3}, a, {q3}), ({q3}, b, {q3}) }

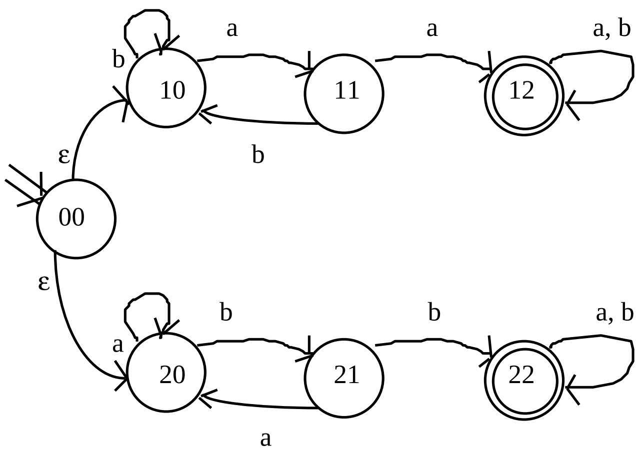


An Algorithm for Constructing the Deterministic FSA

1. Compute the E(q)s:
2. Compute s' = E(s)
3. Compute δ':
   $\delta'(Q, a) = \cup\{E(p) : p \in K \text{ and } (q, a, p) \in \Delta \text{ for some } q \in Q\}$
4. Compute K' = a subset of 2^K
5. Compute $F' = \{Q \in K' : Q \cap F \neq \varnothing\}$

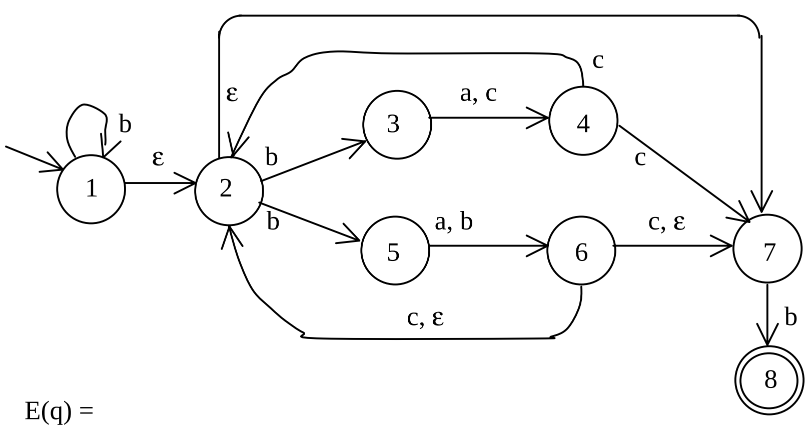
An Example - The Or Machine

$L_1 = \{w : aa \text{ occurs in } w\}$
$L_2 = \{x : bb \text{ occurs in } x\}$
$L_3 = \{y : \in L_1 \text{ or } L_2\}$



Another Example

$b^* (b(a \cup c)c \cup b(a \cup b) (c \cup \varepsilon))^* b$



E(q) =

δ' =

Sometimes the Number of States Grows Exponentially

Example: The missing letter machine, with $|\Sigma| = n$

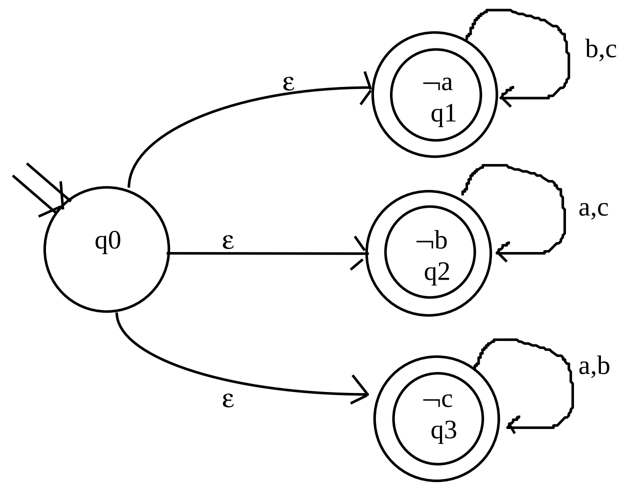
No. of states after 0 chars: 1

No. of new states after 1 char: $\binom{n}{n-1}$ = n

No. of new states after 2 chars: $\binom{n}{n-2}$ = n(n-1)/2

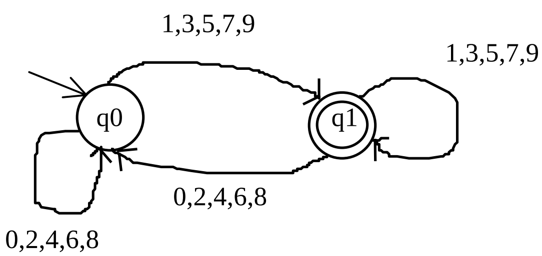
No. of new states after 3 chars: $\binom{n}{n-3}$ = n(n-1)(n-2)/6

Total number of states after n chars: 2^n



What If The Original FSA is Deterministic?

M=



1. Compute the E(q)s:
2. s' = E(q0) =
3. Compute δ'
   ({q0}, odd, {q1})
   ({q0}, even, {q0})
   ({q1}, odd, {q1})
   ({q1}, even, {q0})
4. K' = {{q0}, {q1}}
5. F' = {{q1}}

M' = M

The real meaning of "determinism"

A FSA is **deterministic** if, for each input and state, there is at most one possible transition.

DFSAs are always deterministic. Why?

NFSAs can be deterministic (even with ε-transitions and implicit dead states), but the formalism allows nondeterminism, in general.

Determinism implies uniquely defined machine behavior.