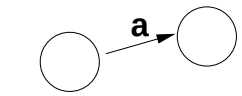
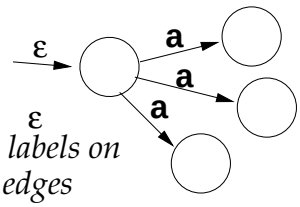
Nondeterministic Finite Automata

Nondeterminism allows several possible next states (no one state is determined)

Deterministic **S.D.** **Nondeterministic**



only 1 per **a** *in* Σ

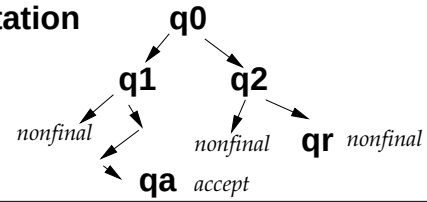


ε *labels on edges*

set of next states

accept, reject

Computation



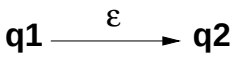
NFA vs. DFA

NFA are seemingly more powerful (NOT!)

NFA allow choice of next state

When choice, NFA may be more compact

Handling ε

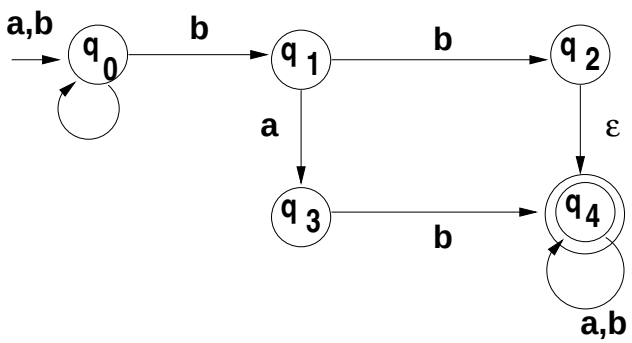


If in q1 reading input symbol a, and take ε edge, still reading a, but in state q2.

All ε edges must be included in execution tree.

Qu: Is every DFA an NFA?

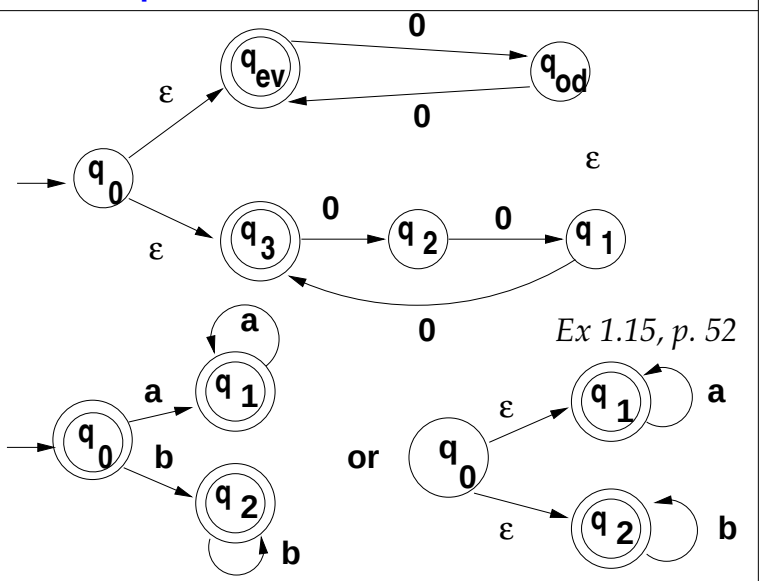
Example NFA



Execution on baabb

L(M) =

More Example NFA's



Formal Definition of Nondeterministic FA

A *NFA M* is a 5-tuple (Q, Σ, δ, q_0, F) where,

1. Q is a finite set of *states*
2. Σ is a finite set called the *alphabet*
3. δ : Q x $\Sigma_\varepsilon \rightarrow P$(Q) *(power set of Q, include ε)*
4. q_0 in Q, the *start state*
5. F $\subseteq$ Q, set of *accept (final) states*

We say that M *accepts w*

if there is a sequence of states r0, r1, ... rm in Q

and w = y1 y2 ... ym, yi in Σ_ε such that

1. r0 = q0
2. ri+1 element of δ(ri, yi+1)
3. rm is in F

5

Equivalence of NFA and DFA

Def: Two NFA M1, M2 are *equivalent* if L(M1) = L(M2).

Th.: For each NFA, there is an equivalent DFA.

Proof idea:
Start with NFA N. Want DFA M with L(N) = L(M).
We will simulate NFA with DFA.
*NFA has **set** of possible next states; DFA just has 1.*
*Solution: **state** of the DFA will be subset of states of NFA*

Proof: Let N = (Q,Σ, δ, q_0, F) be an NFA.

Case 1: N has no ε transitions.

M = (Q', Σ, δ', q'_0 , F ')

1. Q' = *P*(**Q**)
2. δ'(R, a) = $\bigcup_{r \text{ in } R} \delta$(r,a)
3. $q'_0 = \{q_0\}$
4. F' = { R in Q' | R $\cap$ F = ϕ }

6

Equivalence of NFA and DFA

Case 2: N has ε transitions.

Proof Idea: What needs to change? Have to account for all ε moves after read an input symbol.

Proof: For R a subset of Q, define

E(R) = {q | q can be reached from some state in R by following only ε edges}

Change the construction in Case 1 as follows:

2. δ'(R, a) = { q in Q | q in E(δ(r,a)) for some r in R}

3. q'_0 = {E(q_0) }

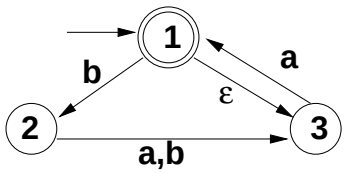
M keeps track of exactly the subset of states N would be in, and accepts when N accepts.

7

Equivalence of NFA and DFA

Cor.: A language is regular iff some NFA recognizes it.

Example: NFA, construct DFA *(Ex. 1.21 p. 57)*



Get lots of states....can we do better?

States that can't be reached from start states that have no incoming edges.....More later!

8

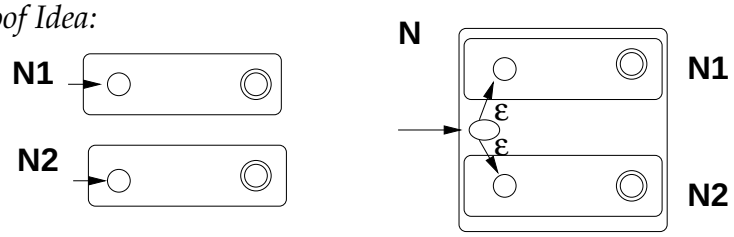
Back to: Regular languages are closed under ...

Th. The class of reg. lang. are closed under $\cup$

Let N1 = $(Q1, \Sigma, \delta_1, q1, F1)$,
let N2 = $(Q2, \Sigma, \delta_2, q2, F2)$ be NFA's.

Construct NFA N such that L(N) = L(N1) $\cup$ L(N2)

Proof Idea:



Proof: We define NFA N as follows....

9

Regular languages are closed under Union

N = $(Q, \Sigma, \delta, q0, F)$

1. Q = {q0} $\cup$ Q1 $\cup$ Q2 *(q0 new, Q1 and Q2 disjoint)*

2. q0 is start

3. F = F1 $\cup$ F2

4.

$$\delta(q, a) = \begin{cases} \delta_1(q, a) & \text{if } q \text{ in } Q1 \\ \delta_2(q, a) & \text{if } q \text{ in } Q2 \\ \{q1, q2\} & \text{if } q = q0 \text{ and } a = \varepsilon \\ \phi & \text{if } q = q0 \text{ and } a = \varepsilon \end{cases}$$

10

Regular languages are closed under Star

Proof Idea:

Proof: Define N = (Q, Σ, δ , q0 , F) from N1.

1. **Q = {q0} ∪ Q1 q0 is new**
2. **q0 is start**
3. **F = {q0} ∪ F1**
4. **δ(q, a) =**

$$\begin{cases} \delta_1(q,a) & \text{if } q \text{ in } Q1,\ q \text{ not in } F1 \\ \delta_1(q,a) & \text{if } q \text{ in } F1,\ a \text{ not } = \varepsilon \\ \delta_1(q,a) \cup \{q1\} & \text{if } q \text{ in } F1 \text{ and } a = \varepsilon \\ \{q1\} & \text{if } q = q0,\ a = \varepsilon \\ \phi & \text{if } q = q0,\ a \text{ not } = \varepsilon \end{cases}$$

Regular languages are closed under Concatenation

Proof Idea: Start out in N1, and in any final state of N1, can switch to N2; accept if wind up in final state in N2.

Proof: Define N = (Q, Σ, δ , q1 , F) from N1 and N2

1. **Q = Q1 ∪ Q2**
2. **q1 is start (same as start of N1)**
3. **F = F2**
4. **δ(q, a) =**

$$\begin{cases} \delta_1(q,a) & \text{if } q \text{ in } Q1,\ q \text{ not in } F1 \\ \delta_1(q,a) & \text{if } q \text{ in } F1,\ a \text{ not } = \varepsilon \\ \delta_1(q,a) \cup \{q2\} & \text{if } q \text{ in } F1 \text{ and } a = \varepsilon \\ \delta_2(q,a) & \text{if } q \text{ in } Q2 \end{cases}$$