CSE 322, Fall 2010

Nondeterministic
Finite State Machines

Concatenation

Defn: For any $X, Y \subseteq \Sigma^*$, define

$$X \bullet Y = \{ xy \mid x \in X \ \& \ y \in Y \}$$

Ex:

$X = \{ a, ab \}$

$Y = \{ \varepsilon, b, bb \}$

$X \bullet Y = \{ a, ab, abb, abbb \}$

$Y \bullet X = \{ a, ab, ba, bab, bba, bbab \}$

note $|X \bullet Y| \leq |X| \bullet |Y|$

Powers

$L^2 = L \cdot L$

$L^3 = L \cdot L \cdot L$

$\vdots$

$L^1 = L$

$L^0 = \{\varepsilon\}$

$$\forall n \geq 0 \quad L^n = \begin{cases} L \cdot L^{n-1} & \text{if } n \geq 1 \\ \{\varepsilon\} & \text{if } \cancel{k}\, n = 0 \end{cases}$$

Eg

$\Sigma^2 = \{ w \mid |w| = 2 \}$

$\Sigma^n = \{ w \mid |w| = n \}$

$(\Sigma \cup \{\varepsilon\})^n = \{ w \mid |w| \leq n \}$

$X, Y \subseteq \Sigma^*$

$X \cdot Y = \{x \cdot y \mid x \in X \;\&\; y \in Y\}$

Examples

$L_{oddparity} \cdot L_{oddparity} = L_{even}$.

$$X, Y \subseteq \Sigma^*$$

$$X \cdot Y = \{x \cdot y \mid x \in X \;\&\; y \in Y\}$$

Examples

$$L_{oddparity} \cdot L_{oddparity} = L_{even} - \{0\}^*$$

$$L_{odd\,parity} \cdot L_{even} = L_{odd}$$

$$X, Y \subseteq \Sigma^*$$

$$X \cdot Y = \{ x \cdot y \mid x \in X \ \& \ y \in Y \}$$

Examples

$$L_{oddparity} \cdot L_{oddparity} = L_{even} - \{0\}^*$$

$$L_{odd\,parity} \cdot L_{even} = L_{odd}$$

$$\underset{\substack{\uparrow \\ \text{infinite}}}{A} \cdot \underset{\substack{\nwarrow \\ \text{finite}}}{B} \stackrel{?}{=} \underset{\substack{\uparrow \\ \text{Finite}}}{C}$$

possible?

$$X, Y \subseteq \Sigma^*$$

$$X \cdot Y = \{x \cdot y \mid x \in X \,\&\, y \in Y\}$$

Examples

$$L_{oddparity} \cdot L_{oddparity} = L_{even} - \{0\}^*$$

$$L_{odd\ parity} \cdot L_{even} = L_{odd}$$

$$\underset{\text{infinite}}{A} \cdot \underset{\text{finite}}{B} \stackrel{?}{=} \underset{\text{Finite}}{C}$$

possible?

$$\Sigma^* \cdot \Phi = \phi$$

yes

$X, Y \subseteq \Sigma^*$

$X \cdot Y = \{x \cdot y \mid x \in X \;\&\; y \in Y\}$

Examples

$L_{oddparity} \cdot L_{oddparity} = L_{even} - \{0\}^*$

$L_{odd\,parity} \cdot L_{even} = L_{odd}$

$\underset{\substack{\uparrow \\ \text{infinite}}}{A} \cdot \underset{\substack{\nwarrow \\ \text{finite}}}{B} \overset{?}{=} \underset{\substack{\uparrow \\ \text{Finite}}}{C}$] possible?

$\Sigma^* \cdot \emptyset = \emptyset$] yes

$X \cdot Y \overset{?}{=} Y \cdot X$] always ?

$X, Y \subseteq \Sigma^*$

$X \cdot Y = \{x \cdot y \mid x \in X \,\&\, y \in Y\}$

Examples

$L_{oddparity} \cdot L_{oddparity} = L_{even} - \{0\}^*$

$L_{odd\ parity} \cdot L_{even} = L_{odd}$

$\underset{\substack{\uparrow \\ \text{infinite}}}{A} \cdot \underset{\substack{\nwarrow \\ \text{finite}}}{B} \overset{?}{=} \underset{\substack{\uparrow \\ \text{Finite}}}{C}$] possible?

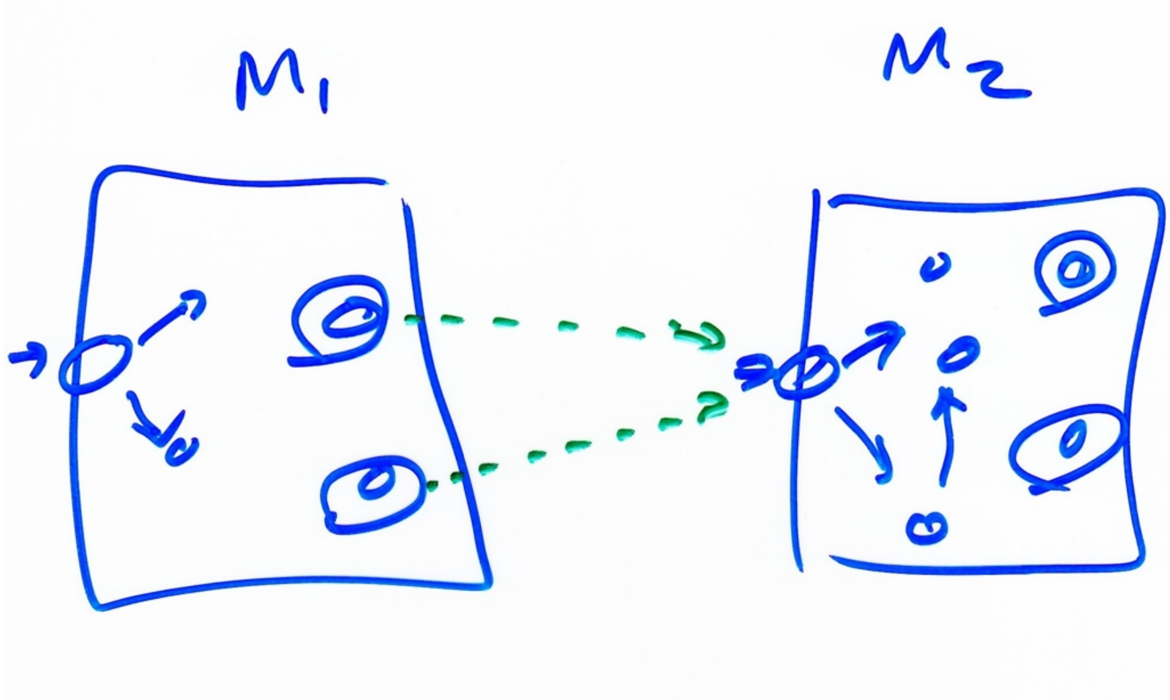
$\Sigma^* \cdot \emptyset = \emptyset$] yes

$X \cdot Y \overset{?}{=} Y \cdot X$] always true? ?

$\{0\} \cdot \{1\} \neq \{1\} \cdot \{0\}$] no

Q:

- Is the class of regular languages closed under concatenation?
- Again, for Java programs, say, it's not too hard to prove this.
- What about finite automata? Inability to back up the input tape is one issue...

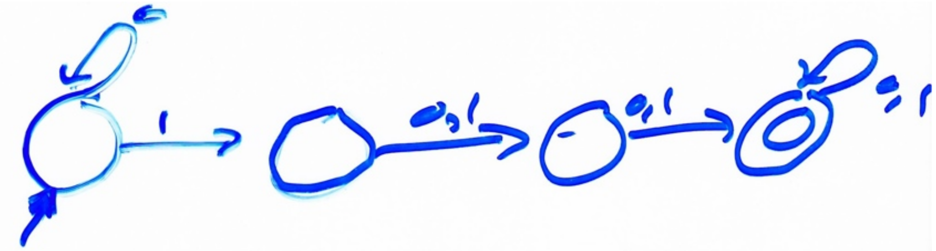


An idea for closure under concatenation, but not clear how to do it – may need to stay in M_1 for *several* visits to F before jumping to M_2.

E.g.:

{even parity} • {exactly 5 1's}

which 1 is 5th from end?



① DFA as a recognizer.

② . . . . . . generator

000110001

③ A different kind of generator:

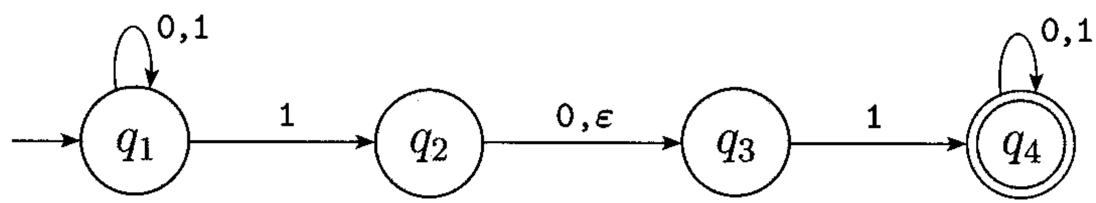
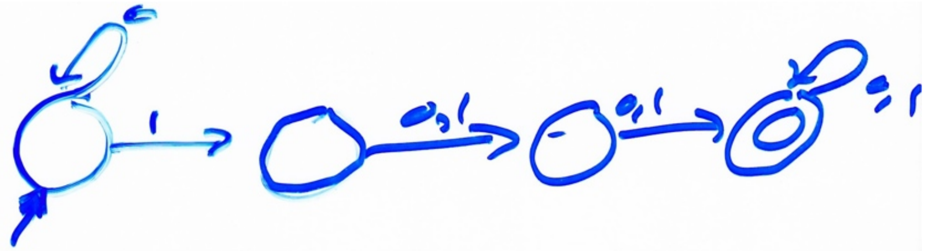


FIGURE 1.27

101, 11
0101, 011
00101, 0011
111101, 11111
1010, 0110
10100, 01101



① DFA as a recognizer.

② . . . . . . generator

000110001

③ A different kind of generator:

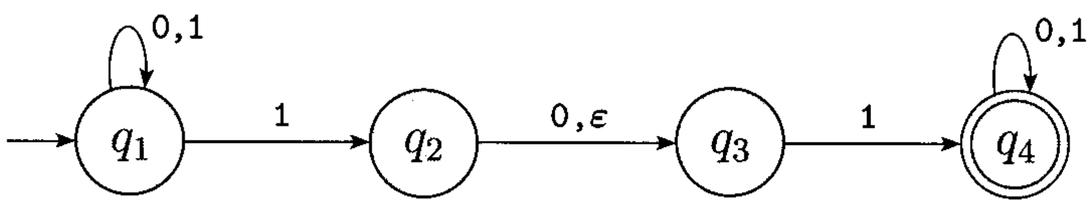


FIGURE 1.27

④ Q. What would it mean/how could we define an equivalent recognizer

A. Nondeterminism

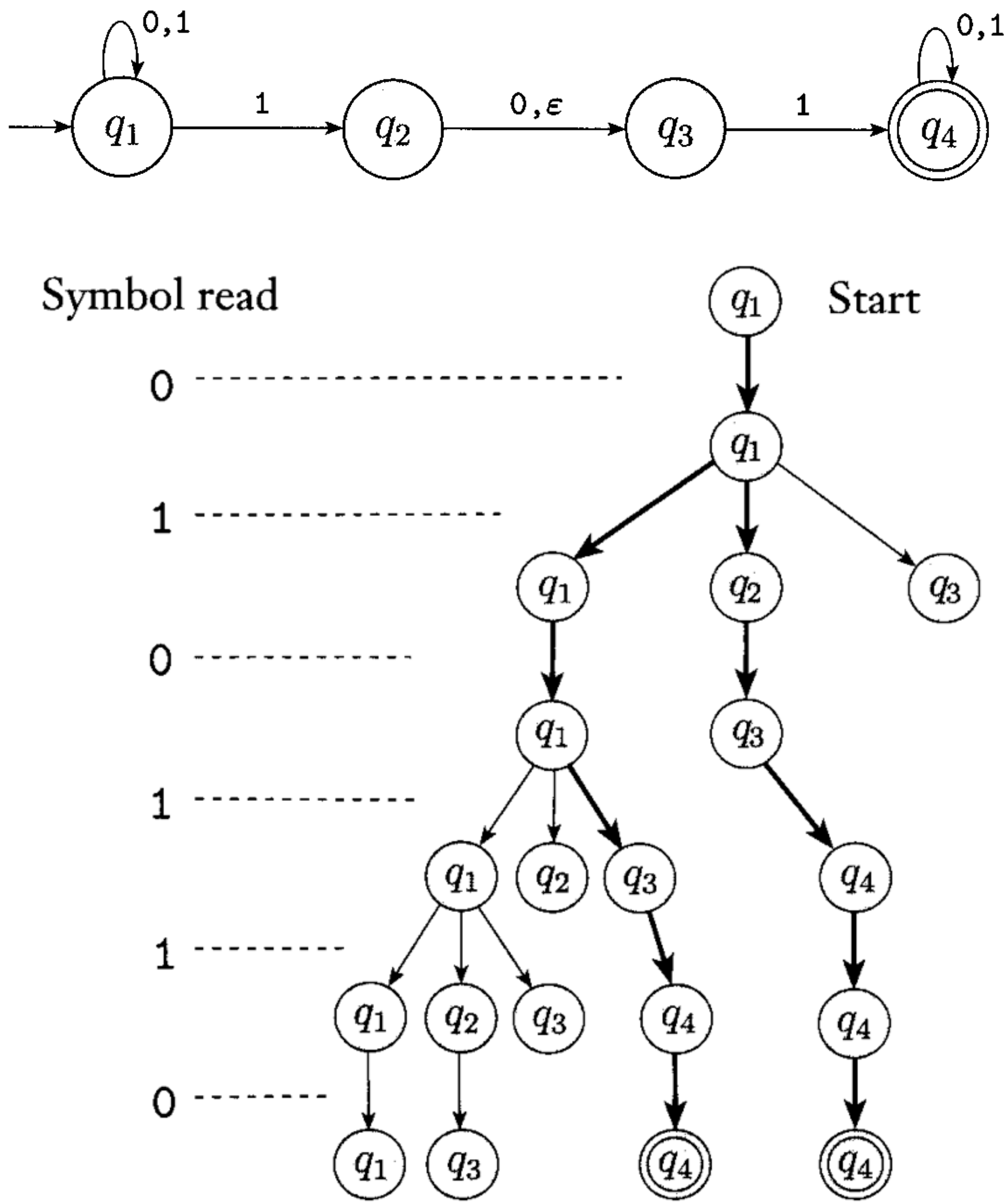


FIGURE 1.29

A nondeterministic finite state machine

$M = (Q, \Sigma, \delta, q_0, F)$

where

- Q is finite set (states)
- $q_0 \in Q$ start state
- Σ is a finite set (alphabet)
- $F \subseteq Q$ Final states / Accepting states

~~$\delta : Q \times \Sigma \to Q$ transition function~~

function

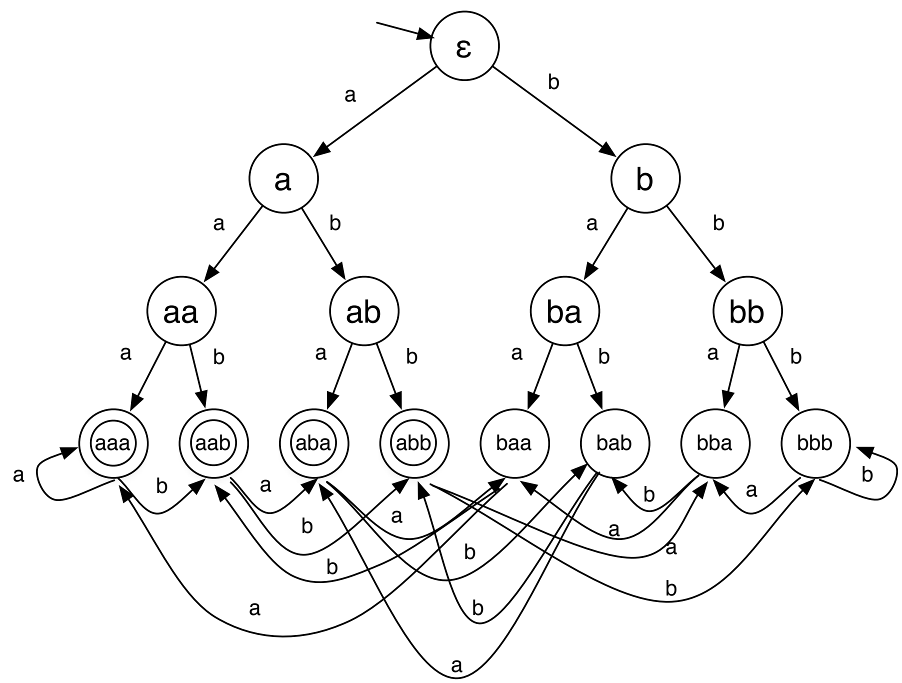
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$ transition function

E.g. for fig[2] M

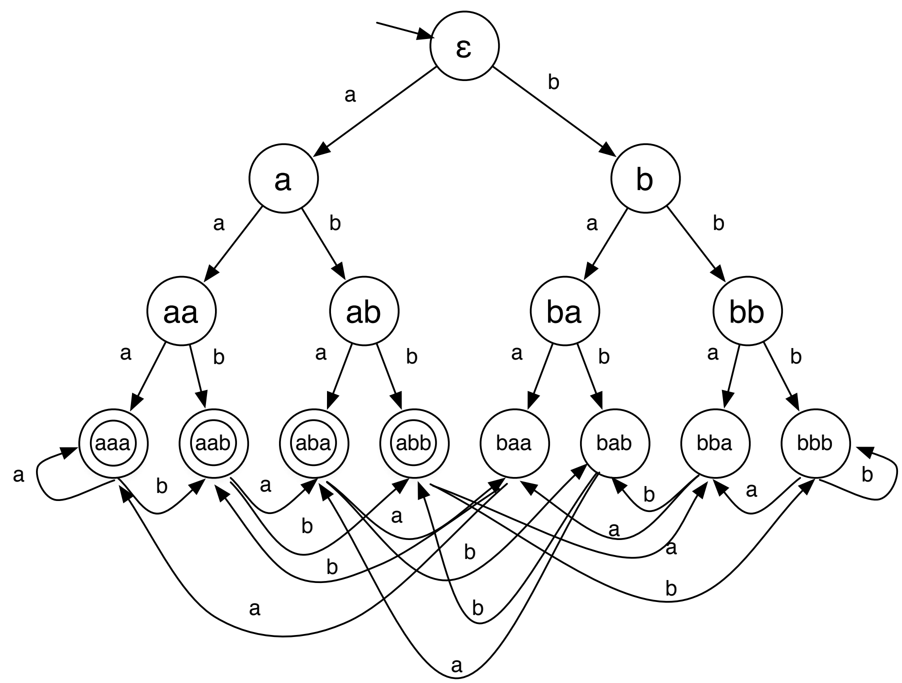
$\delta(q_1, 0) = \{q_1\}$
$\delta(q_1, 1) = \{q_1, q_2\}$
$\delta(q_2, 1) = \emptyset$

6.3
~~2.4~~

L = { w in {a,b}* l 3rd letter from the right end of w is "a" }



L = { w in {a,b}* l 3rd letter from the right end of w is "a" }



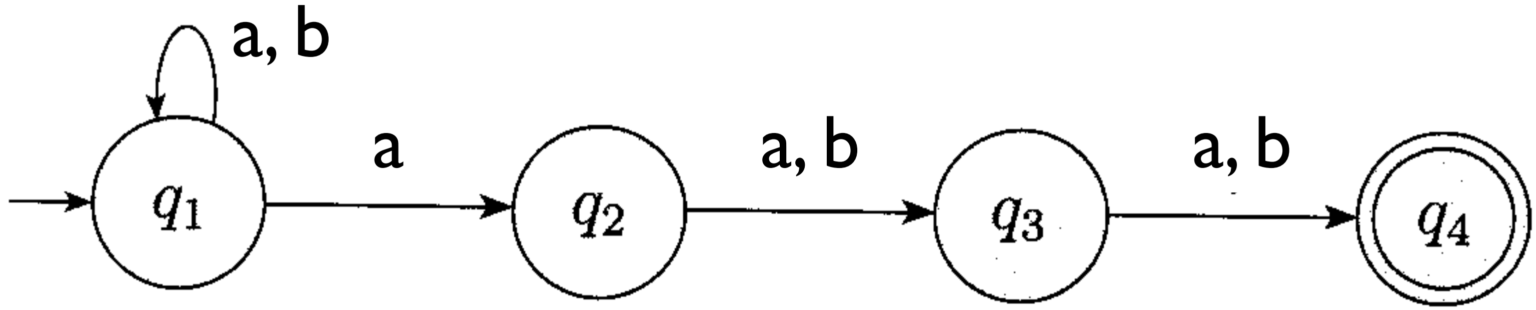


FIGURE **1.31**

DEFN ("~~is in~~ might be in state q")

M might ~~ends~~ in state q after reading $w \in \Sigma^*$ if

(1) $w = w_1 w_2 \cdots w_n$

where $w_i \in \Sigma \cup \{\varepsilon\}$

(2) $\exists$ state $r_0, r_1, r_2 \cdots r_n \in Q$

s.t. (a) $r_0 = q_0$

(b) $\forall\, 1 \leq i \leq n$

$r_i \in \delta(r_{i-1}, w_i)$ $\cancel{= r_i}$

(c) $r_n = q$

~~Fact: q is unique because δ is a function, basically~~

Defn

M accepts $w \in \Sigma^*$ $\Leftrightarrow$ ~~the~~ some state, q, reached by M after reading w is an accepting state, i.e., $q \in F$.

Defn

The language recognized by M, $L(M) = \{ w \in \Sigma^* \mid M \text{ accepts } w \}$.

Note

Every M recognizes exactly one language. Implicitly, it "recognizes" both strings it must accept and those it must reject.

??

Very important: note that "might be in a non-final state" does not imply "reject".

6-7

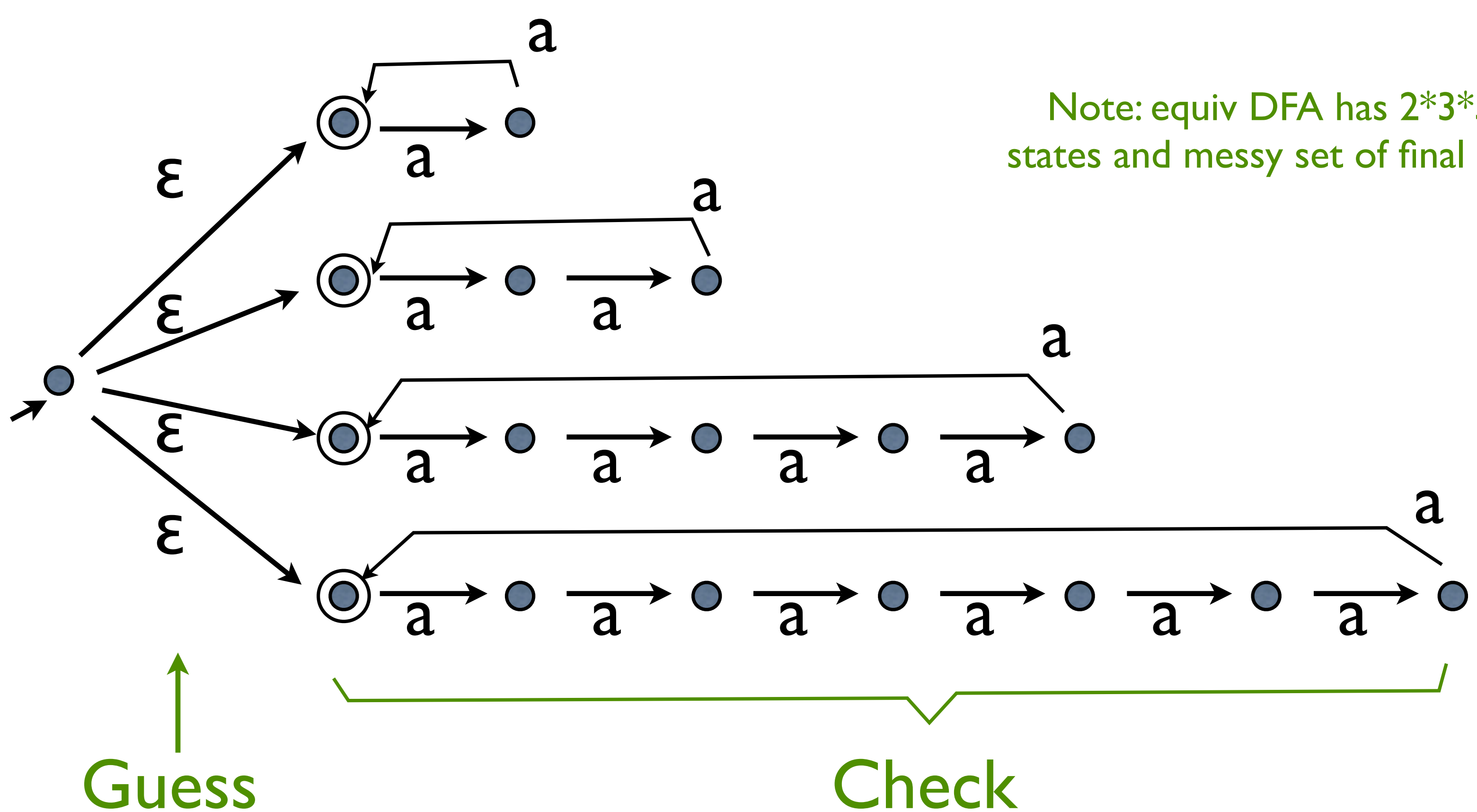
to show M on w:

Accepts–show one path ending in F

Rejects–show all paths fail to end in F

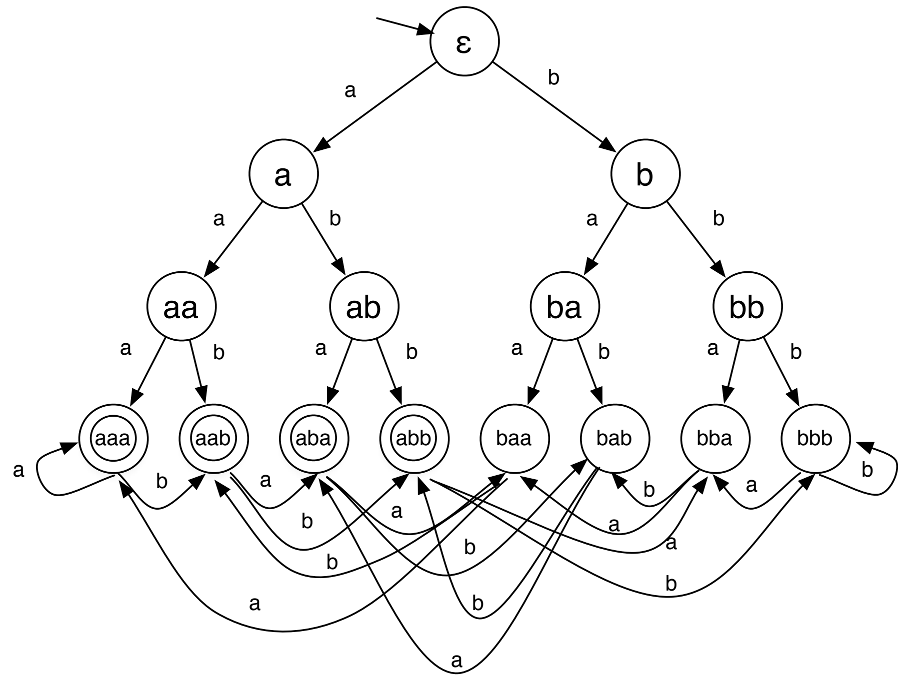
Example "guess & check":

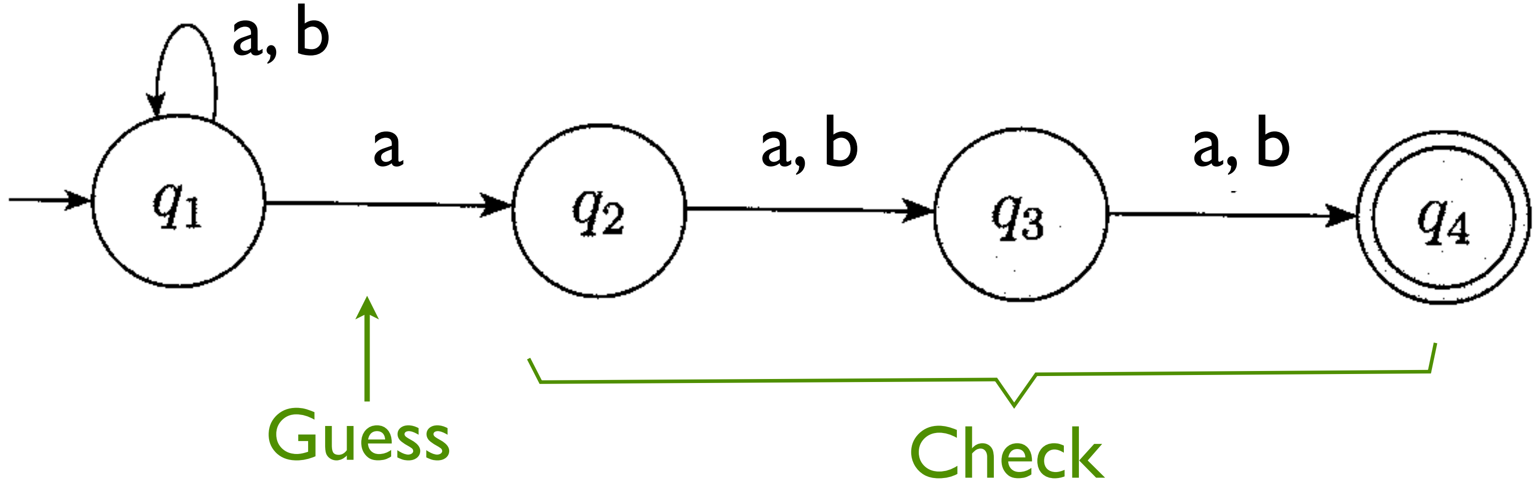
L = { a^n | n is a multiple of 2, 3, 5 or 7 }



Note: equiv DFA has 2*3*5*7 states and messy set of final states

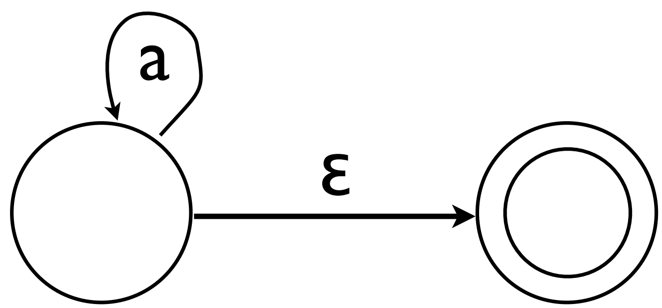
L = { w in {a,b}* l 3rd letter from the right end of w is "a" }





(Non-)Example

$L = \{ a^p \mid p \text{ is prime} \}$

M: 

Q: is M deterministic?

Q: Does M accept a^p for every prime p?

Q: does L(M) = L?

Q: but, doesn't it always guess right?

Nondeterminism: How

- View it as a *generator* of a language
- View it as a *recognizer* of a language
  - "build the tree"
  - explore all paths
  - guess-and-check

Nondeterminism: Why

- *Specifications*: say, clearly & concisely, *what*, not *how*
- Precise, and often *con*cise specification
  - "do A or B, but I don't yet know/don't want clutter of saying which"
  - Sometimes *exponentially* more concise - "3rd letter from end"
- Natural model of incompletely specified/partially known systems
  - if correct wrt a partial spec, then correct wrt *any* implementation consistent with that spec
  - "is state 'reactor boiling / control rods out' unreachable, even allowing for unknown behavior of subsystem X"?

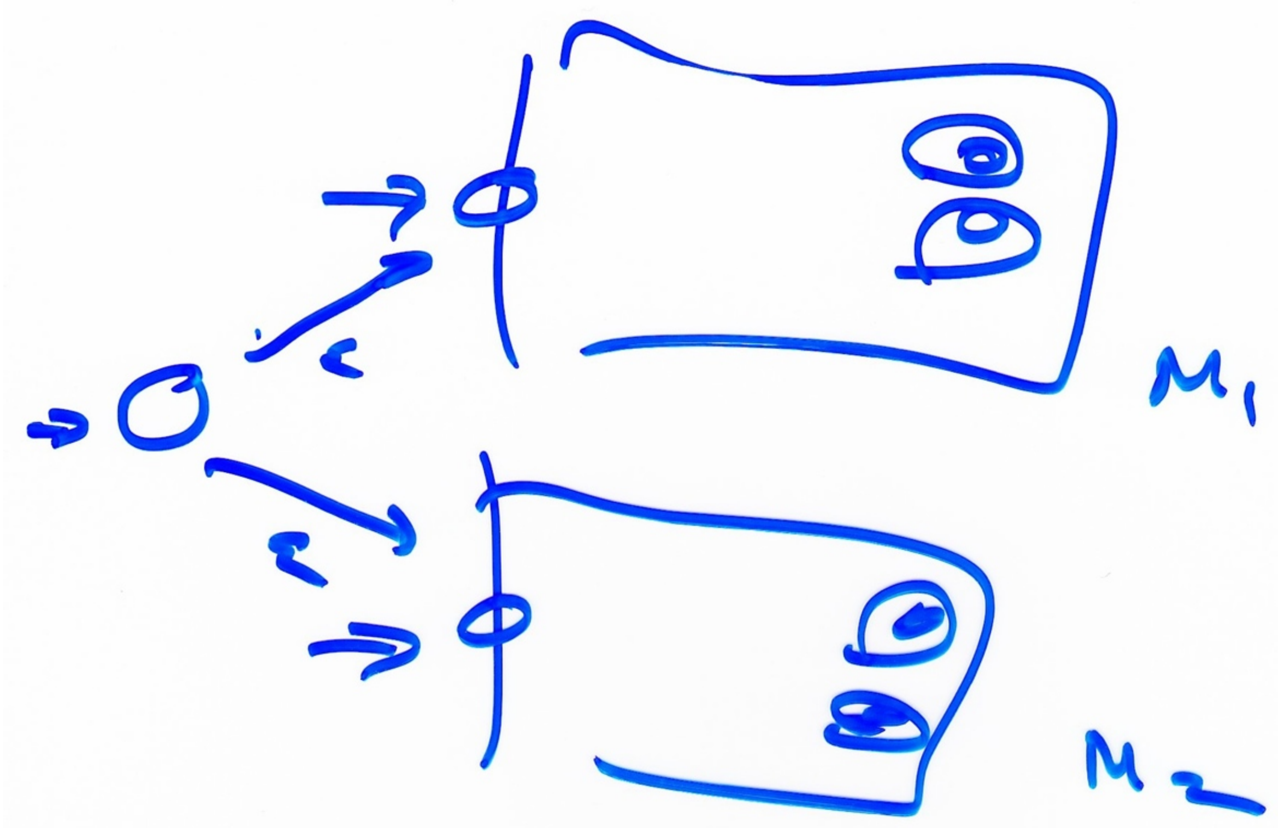
Kleene Star

- Defn: $L^* = \bigcup_{n \geq 0} L^n$
- Examples

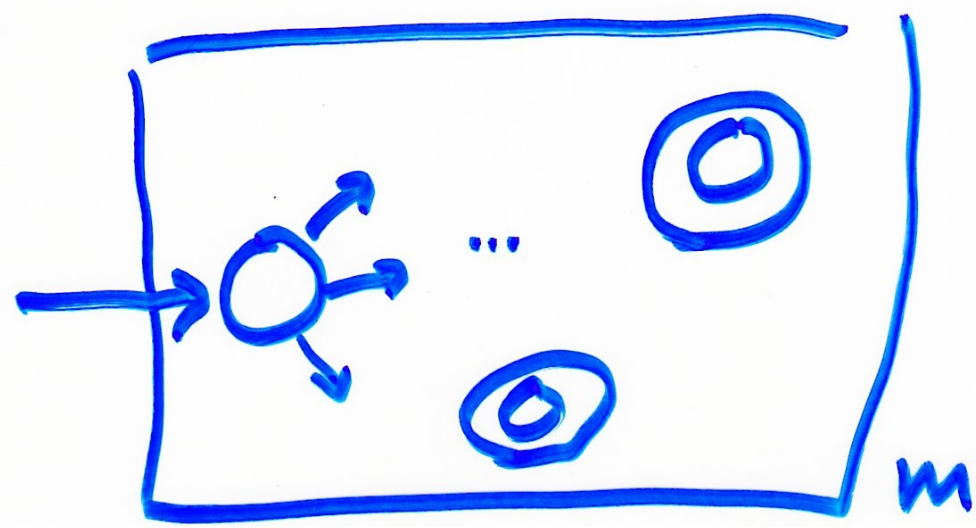
  i) Σ^* : a simple special case

  ii) $L = \{ a^p b \mid p \text{ is prime}\}$
  $L^* = \{\varepsilon\} \cup \{a^{p_1} b\ a^{p_2} b \ldots b\ a^{p_k} b \mid k \geq 1,$
  and each p_i is prime$\}$

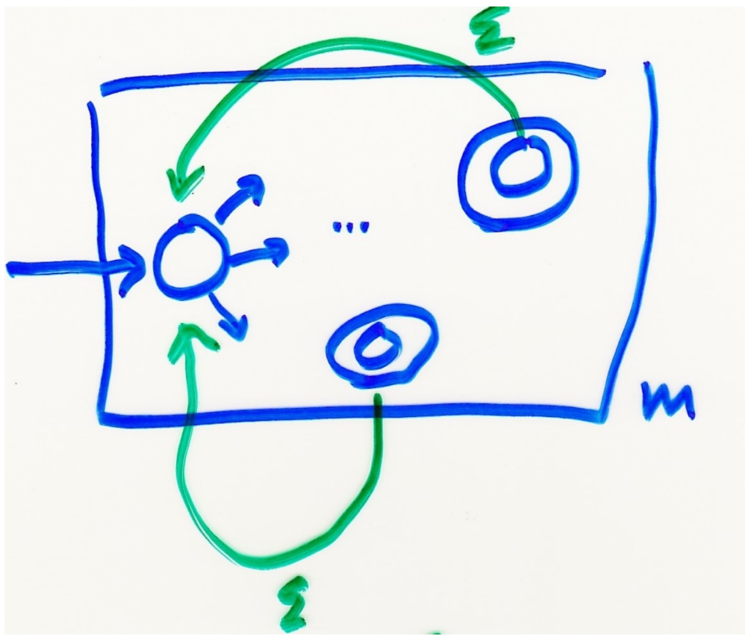
Closure under union



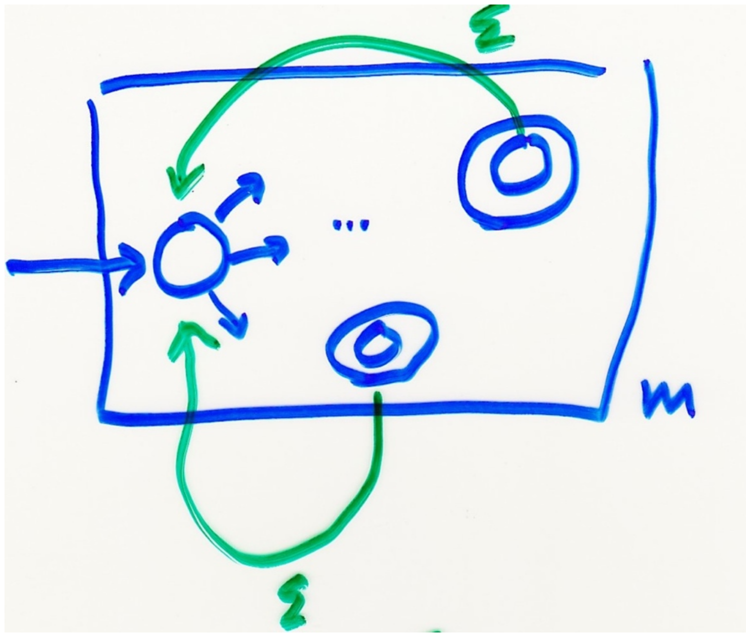
Given NFA M, can build one for L(M)*?



Given NFA M, can build one for $L(M)^*$?

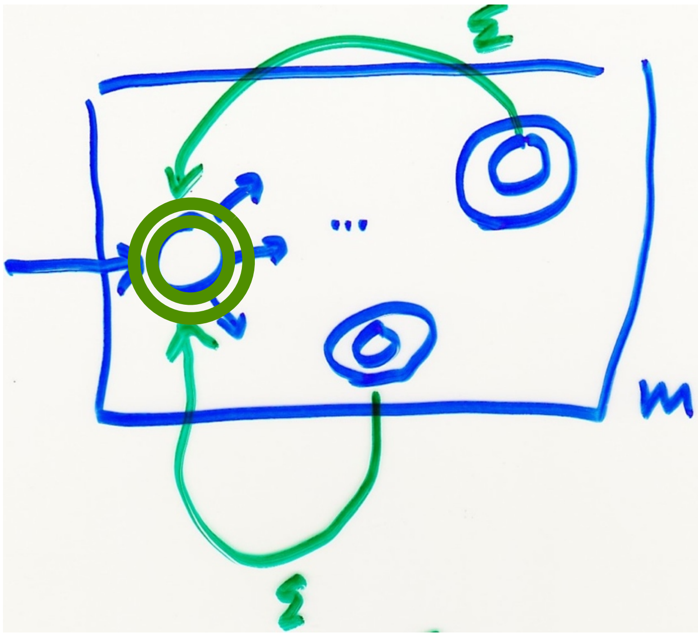


Given NFA M, can build one for L(M)*?

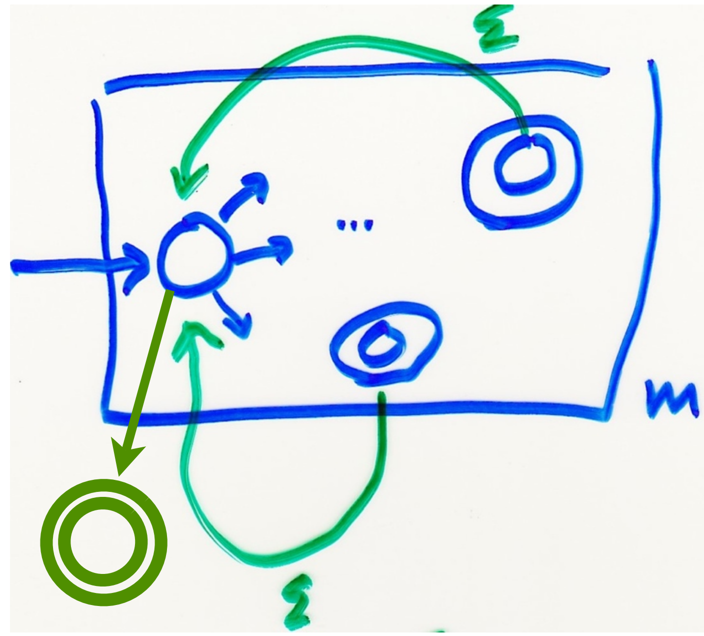


No
(may reject ε)

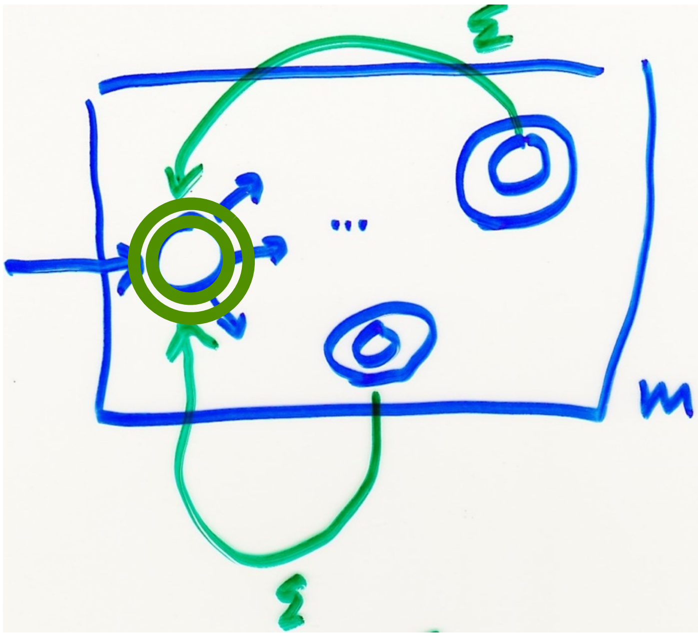
Given NFA M, can build one for L(M)*?



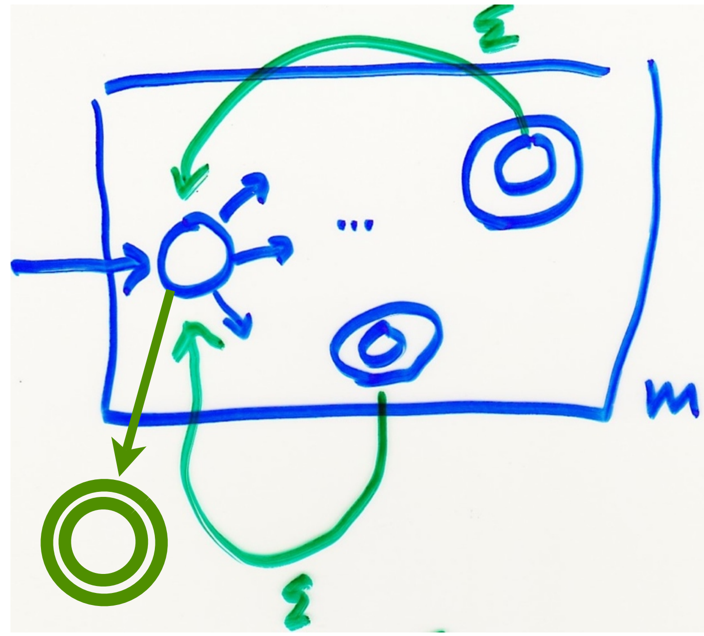
or



Given NFA M, can build one for L(M)*?

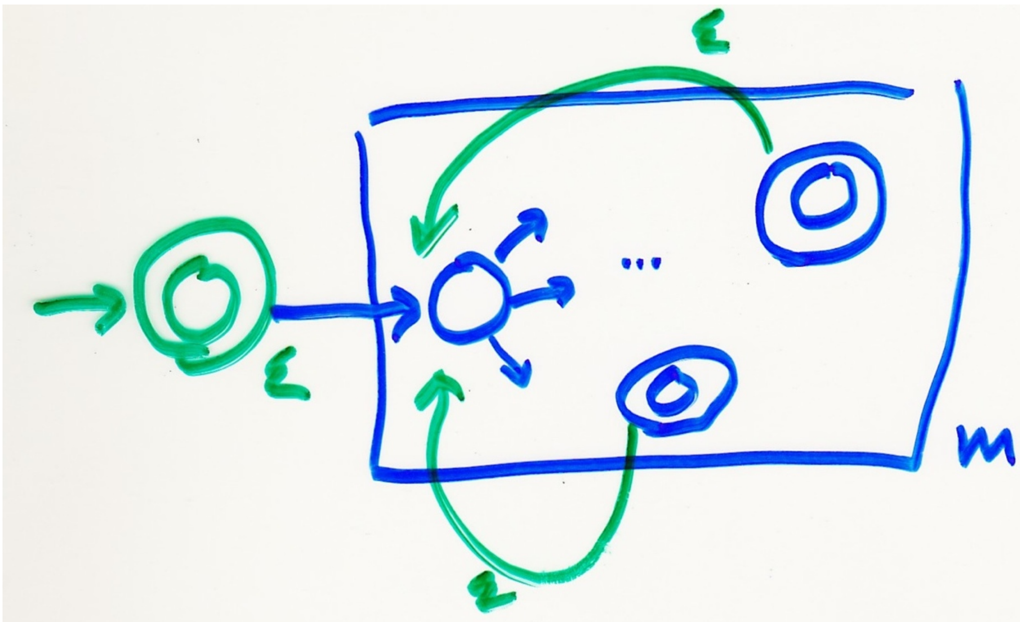


or

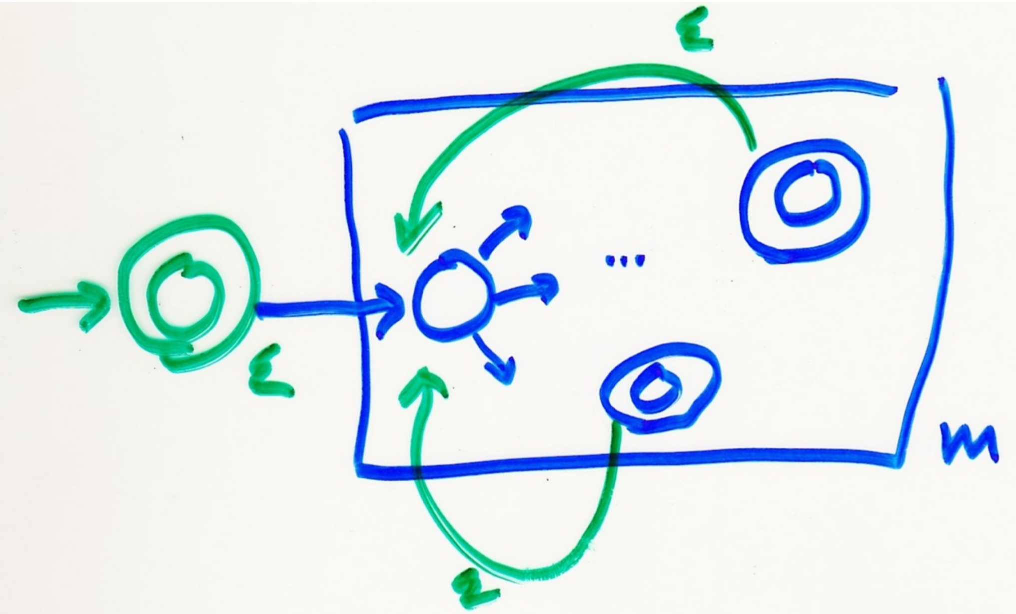


No, may accept extra stuff (if M can loop back to start before reaching F)

Given NFA M, can build one for L(M)*?



Given NFA M, can build one for $L(M)^*$?

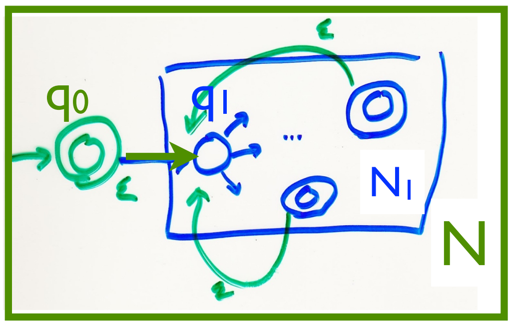


Yes!

Closure under *

General strategy: such proofs are usually *constructive*, i.e., given a (generic) NFA N_1, we *construct* a "new" NFA, N. In this case:

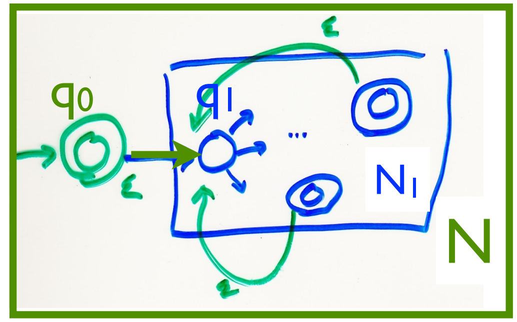
[Notation changed slightly to match Thm 1.49 in Sipser; see it for careful description of N vs N_1]



N_1, "Old": blue
N, "New": blue + green

Then prove the correctness of the construction, i.e., that L(N) = $(L(N_1))^*$. Proof idea: connect computation trace(s) of "old" NFA to ones in "new" NFA, where a "trace" means, recalling the definition of "M could be in state q after reading w," the/a sequence of states/transitions/edges M follows/could follow on some input.

Closure under *, II

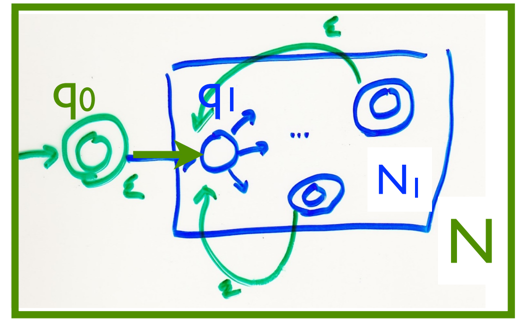


For the correctness proof, there are usually 2 directions, namely:
$(L(N_1))^* \subseteq L(N)$ and $L(N) \subseteq (L(N_1))^*$

1) $(L(N_1))^* \subseteq L(N)$, or, equivalently, given any $k \geq 0$ and any k strings x_1, x_2, ..., x_k, each in $L(N_1)$, show that their concatenation $x_1 \bullet x_2 \bullet \ldots \bullet x_k = x$ is in $L(N)$. For this direction, let $r_{i0}, r_{i1}, r_{i2}, \ldots, r_{in_i}$ be an accepting trace (in N_1) for x_i, $1 \leq i \leq k$. Note $q_1 = r_{i0}$, (why?) and $r_{in_i} \in F$ (why?) The key idea is that you can glue these together using the new start state and the new ε transitions (green state/arrows) to build an accepting trace *in N* for *x*. Namely: $q_0, r_{10}, r_{11}, r_{12}, \ldots, r_{1n_1}, \; r_{20}, r_{21}, r_{22}, \ldots, r_{2n_2}, \ldots, \; r_{k0}, \ldots, r_{kn_k} \in F$. This is a valid accepting trace in N since all transitions in that sequence are either transitions of N_1, hence in N, or are ε transitions from a final state of N_1 to N_1's start state $q_1 = r_{10} = r_{20} = \ldots$, hence again in N. $\therefore x \in L(N)$.

Trace really should be $r_{i0}, a_{i0}, r_{i1}, a_{i1}, \ldots$ i.e. alternately $\in Q, \in \Sigma \cup \{\varepsilon\}$, but slides are small & I'm being lazy.

Closure under *, III



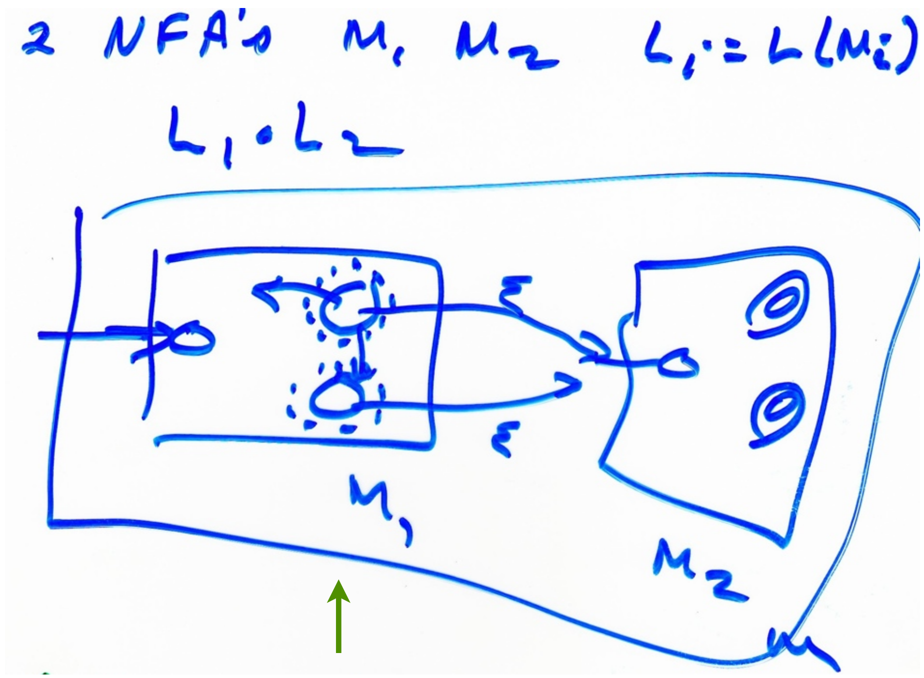
2) $L(N) \subseteq (L(N_1))^*$, or equivalently, given any x in L(N), show that it can be broken into $k \geq 0$ substrings $x_1, x_2, ..., x_k$, (i.e., $x = x_1 \bullet x_2 \bullet ... \bullet x_k$) so that each is in $L(N_1)$. For this direction, suppose $q_0 = r_0, r_1, r_2, ..., r_n \in F$ is an accepting trace (in N) for x. Note that $r_1 = q_1$, since the only transition leaving q_0 goes to q_1 (and is labeled ε). Let x_1 be the concatenation of all edge labels up to (but *ex*cluding) the next green edge (i.e., an ε-move from a final state back to q_1). Note that $x_1 \in L(N_1)$, since the *in*cluded transitions are all present in N_1 and run from its start state to a final state, so they are an accepting trace in N_1. Similarly, let x_2 be the concatenation of all edge labels up to the next green edge, ..., and x_k those after the last green edge. By the same reasoning, each $x_i \in L(N_1)$, for each $1 \leq i \leq k$. Finally, note that $x = x_1 \bullet x_2 \bullet ... \bullet x_k$ since the excluded transitions are all ε-moves. $\therefore x \in (L(N_1))^*$ QED

Closure under *, Leftovers

There are a few points in the proof above that I deliberately didn't address. I strongly suggest that you think about them and see if you can fill in missing details and/or explain why they actually *are* covered, even if not explicitly mentioned. I suggest you *write* it (but no need to turn it in).

- Are $x = \varepsilon$ / $k = 0$ correctly handled, or do you need to say more?
- Is it a problem if N_1's start state is a final state?
- Is it a problem if N_1 includes ε-moves from (some or all states in) F to q_1?
- Is there anything else I omitted?

Closure under Concatenation



Final states of M_1
no longer final

NFA == DFA, or not?

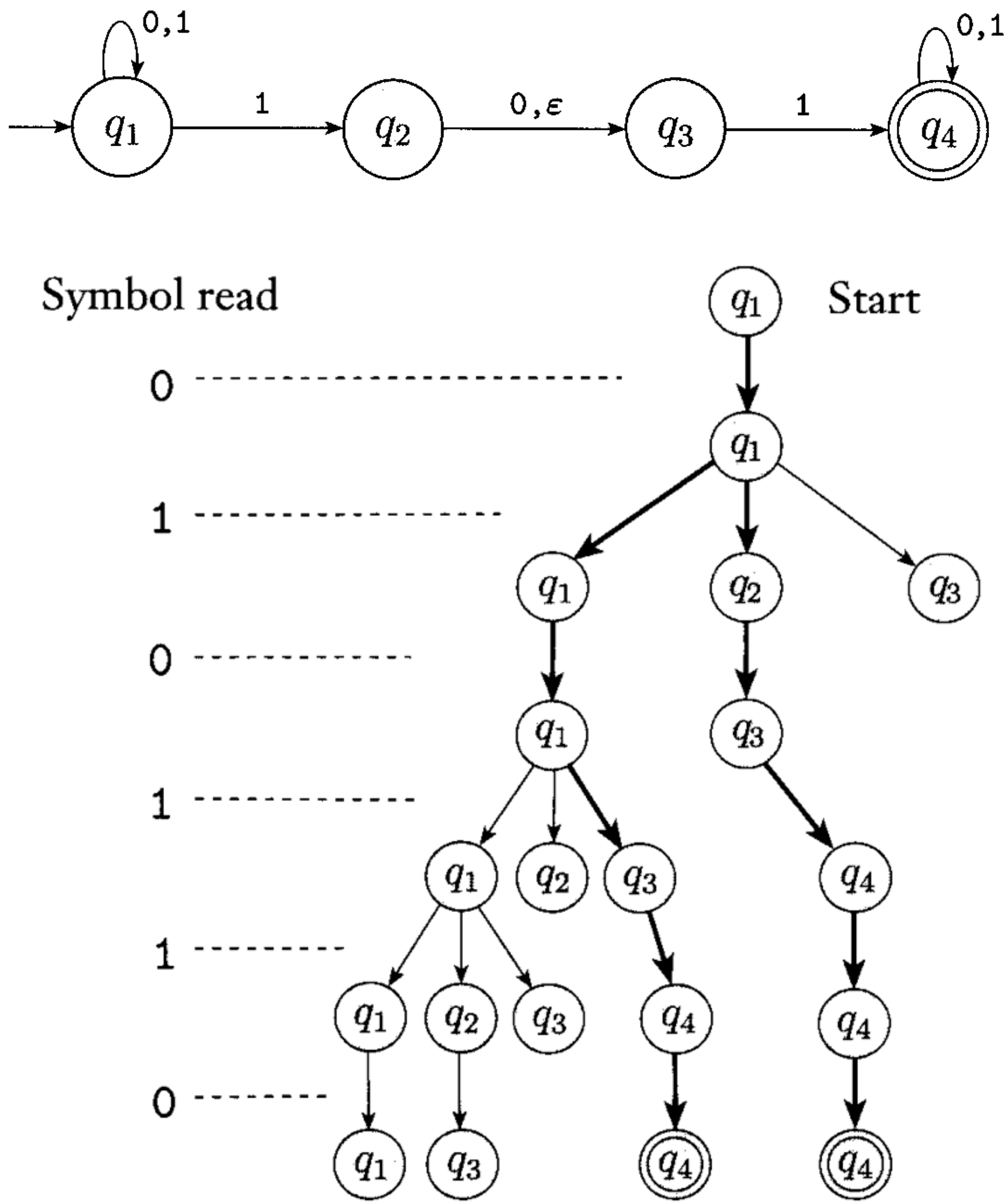


FIGURE **1.29**

Defn

M_1 & M_2 **equivalent** if $L(M_1) = L(M_2)$

Theorem 1.39

$\forall$ nfa N $\exists$ equivalent dfa M

given $N = (Q, \Sigma, \delta, q_0, F)$

build $M = (Q', \Sigma, \delta', q_0', F')$

(warm up: no ε-moves)

$$Q' = 2^Q$$

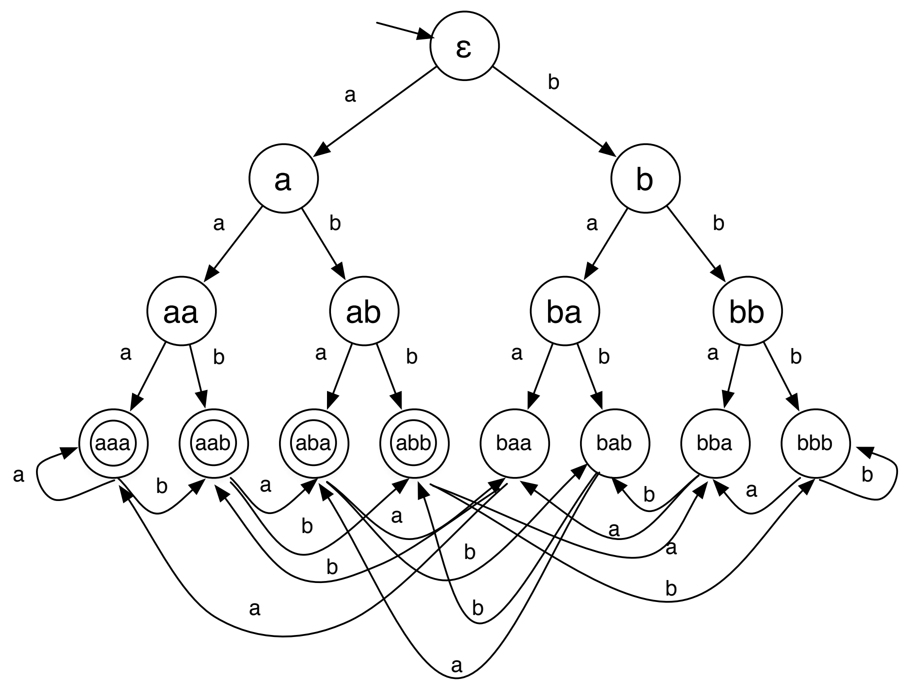
$$q_0' = \{q_0\}$$

$$F' = \{R \subseteq Q \mid R \cap F \neq \emptyset\}$$

$\forall a \in \Sigma, \forall R \subseteq Q$:

$$\delta'(R, a) = \bigcup_{r \in R} \delta(r, a)$$

L = { w in {a,b}* | 3rd letter from the right end of w is "a" }



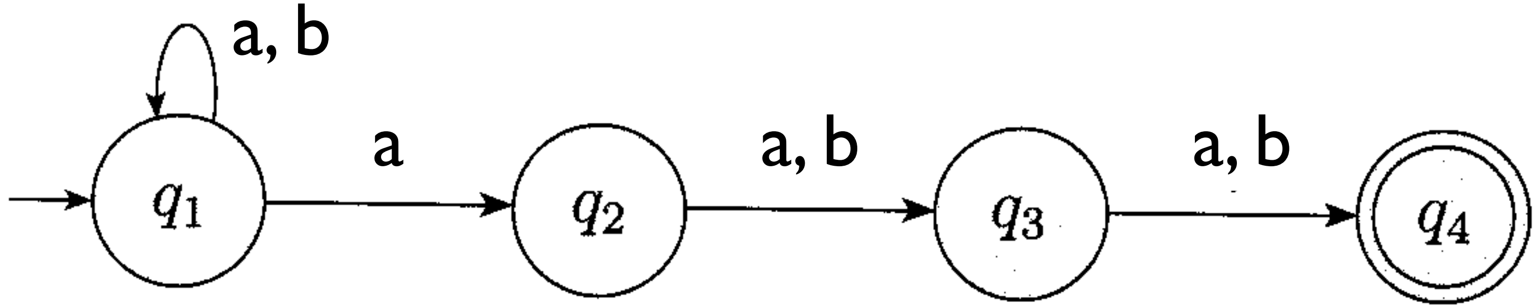
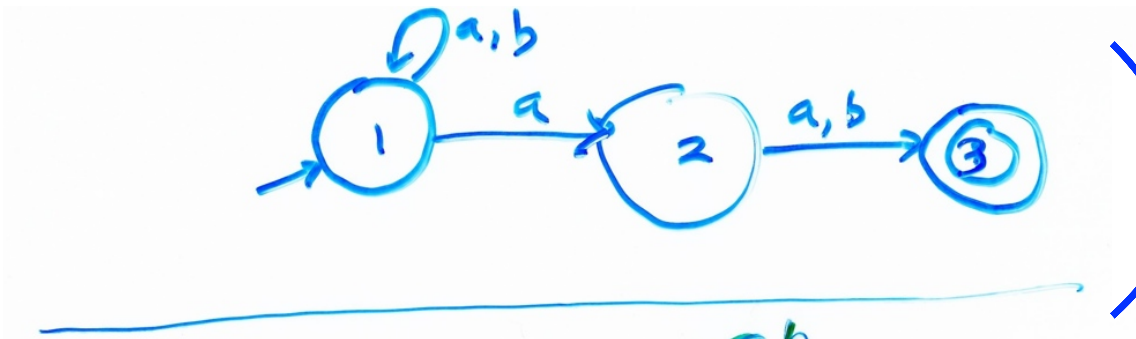
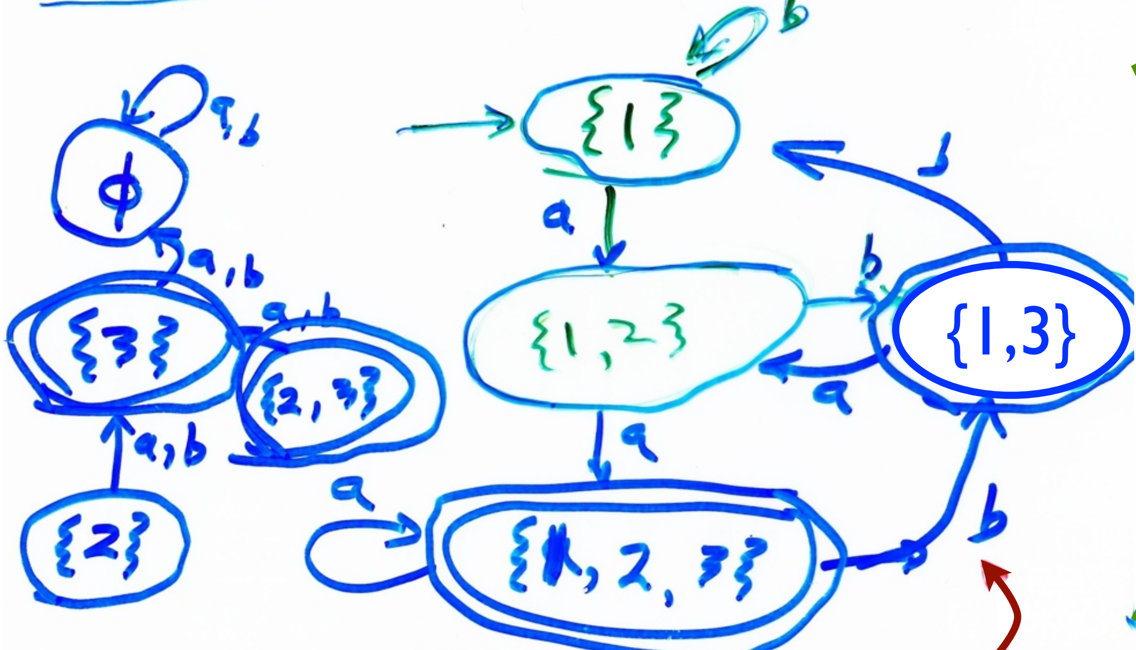


FIGURE 1.31



A 3-state NFA

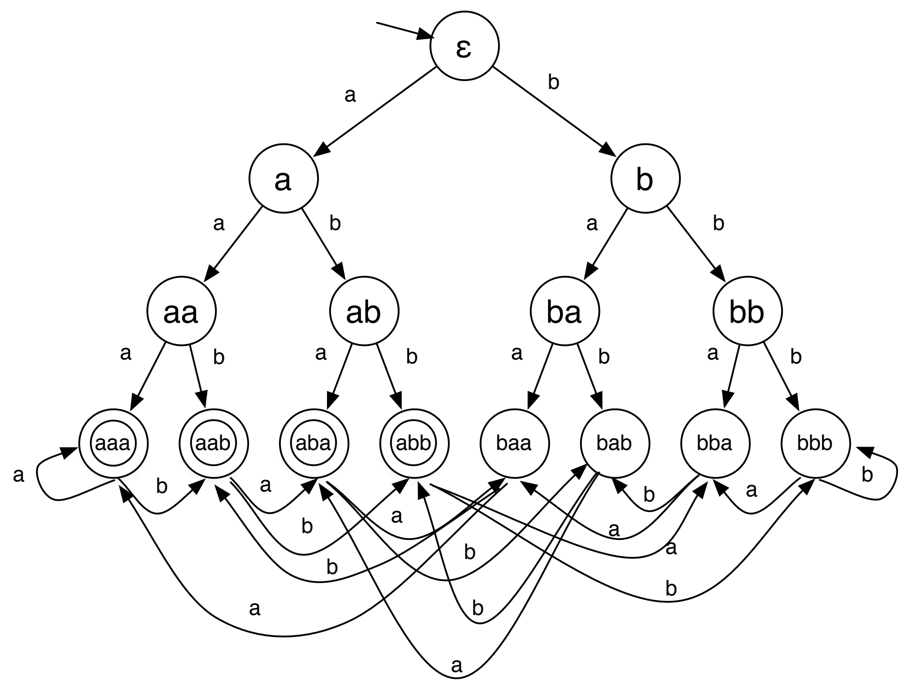


The equivalent 2^3 = 8-state DFA, built as in Theorem 1.39

(4 states on left are not reachable from start state but *are* part of the DFA.)

$\forall a \in \Sigma, \forall R \subseteq Q:$ $\delta'(R,a) = \bigcup_{r \in R} \delta(r,a)$	An example transition: $\delta'(\{1,2,3\}, b) = \delta(1,b) \cup \delta(2,b) \cup \delta(3,b) = \{1\} \cup \{3\} \cup \varnothing = \{1,3\}$

L = { w in {a,b}* | 3rd letter from the right end of w is "a" }



Exercise: apply the construction to the NFA below.
Note: You will *not* get the DFA above (but it will be equivalent).

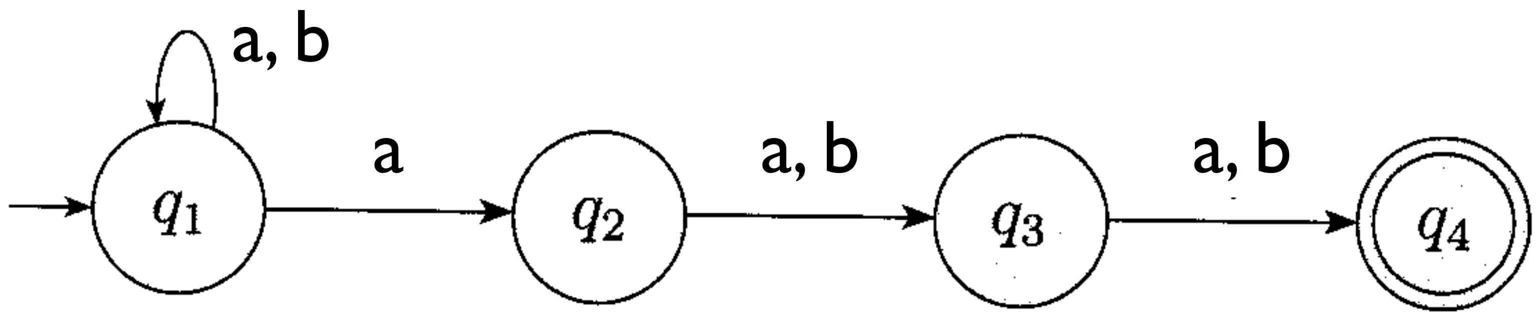


FIGURE 1.31

Simulation of NFAs by DFAs: Notes on the Proof of Theorem 1.39

W. L. Ruzzo 15 Oct 10

The text's assertion that the construction given in the proof of Theorem 1.39 (1st ed: 1.19) is "obviously correct" is a little breezy. Here is an outline of a somewhat more formal correctness proof. I will only handle the case where the NFA has *no* ϵ-transitions. Notation is as in the book.

For any $x \in \Sigma^*$, define

$$\begin{aligned} Q_{N,x} &= \{r \in Q \mid N \text{ could be in state } r \text{ after reading } x\}, \text{ and} \\ Q_{M,x} &= \text{ the state } R \in Q' \text{ that } M \text{ would be in after reading } x. \end{aligned}$$

The key idea in the proof is that these two sets are identical, i.e., that the single state of the DFA faithfully reflects the complete range of possible states of the NFA. The proof is by induction on $|x|$.

BASIS: ($|x| = 0$.) Obviously $x = \epsilon$. Then

$$Q_{N,\epsilon} = \{q_0\} = q_0' = Q_{M,\epsilon}.$$

The first and third equalities follow from the definitions of "moves" for NFAs and DFAs, respectively, and the middle equality follows from the construction of M.

INDUCTION: ($|x| = n > 0$.) Suppose $Q_{N,y} = Q_{M,y}$ for all strings $y \in \Sigma^*$ with $|y| < n$, and let $x \in \Sigma^*$ be an arbitrary string with $|x| = n > 0$. Since x is not empty, there must be some $y \in \Sigma^*$ and some $a \in \Sigma$ such that $x = ya$. For any $r \in Q$,

$$\begin{aligned} & N \text{ could be in state } r \text{ after reading } x = ya && (1) \\ \Leftrightarrow \quad & \text{there is some } r' \in Q \text{ such that } N \text{ could be in } r' \text{ after reading } y \text{ and } r \in \delta(r', a) && (2) \\ \Leftrightarrow \quad & r \in \bigcup \delta(r', a) && (3) \end{aligned}$$

reflects the complete range of possible states of the NFA. The proof is by induction on $|x|$.

BASIS: ($|x| = 0$.) Obviously $x = \epsilon$. Then

$$Q_{N,\epsilon} = \{q_0\} = q_0' = Q_{M,\epsilon}.$$

The first and third equalities follow from the definitions of "moves" for NFAs and DFAs, respectively, and the middle equality follows from the construction of M.

INDUCTION: ($|x| = n > 0$.) Suppose $Q_{N,y} = Q_{M,y}$ for all strings $y \in \Sigma^*$ with $|y| < n$, and let $x \in \Sigma^*$ be an arbitrary string with $|x| = n > 0$. Since x is not empty, there must be some $y \in \Sigma^*$ and some $a \in \Sigma$ such that $x = ya$. For any $r \in Q$,

$$\begin{aligned}
&N \text{ could be in state } r \text{ after reading } x = ya && (1)\\
\Leftrightarrow\quad & \text{there is some } r' \in Q \text{ such that } N \text{ could be in } r' \text{ after reading } y \text{ and } r \in \delta(r', a) && (2)\\
\Leftrightarrow\quad & r \in \bigcup_{r' \in Q_{N,y}} \delta(r', a) && (3)\\
\Leftrightarrow\quad & r \in \delta'(Q_{N,y}, a) && (4)\\
\Leftrightarrow\quad & r \in \delta'(Q_{M,y}, a) && (5)\\
\Leftrightarrow\quad & r \in Q_{M,x} && (6)
\end{aligned}$$

The equivalence of (1) and (2) follows from the definition of "moves" for NFAs: the last step must be a move from some state reached after reading y. The equivalence of (2) and (3) is just set theory. The equivalence of (3) and (4) follows from the definition of δ'. The equivalence of (4) and (5) follows from the induction hypothesis. The equivalence of (5) and (6) follows from the definition of "moves" for DFAs.

Given the equivalence established above, it's easy to see that $L(N) = L(M)$, since N accepts x if and only if it can reach a final state after reading x, which will be true if and only if $Q_{N,x}$ contains a final state, which happens if and only if $Q_{M,x} \in F'$.

Defn

M_1 & M_2 **equivalent** if $L(M_1) = L(M_2)$

Theorem 1.39

$\forall$ nfa N $\exists$ equivalent dfa M

given $N = (Q, \Sigma, \delta, q_0, F)$

build $M = (Q', \Sigma, \delta', q_0', F')$

(warm up: no ε-moves)

$$Q' = 2^Q$$

$$q_0' = \{q_0\}$$

$$F' = \{R \subseteq Q \mid R \cap F \neq \emptyset\}$$

$\forall a \in \Sigma, \forall R \subseteq Q:$

$$\delta'(R, a) = \bigcup_{r \in R} \delta(r, a)$$

No ε-moves

Defn

M_1 & M_2 **equivalent** if $L(M_1) = L(M_2)$

Theorem 1.39

$\forall$ nfa N $\exists$ equivalent dfa M

given $N = (Q, \Sigma, \delta, q_0, F)$

build $M = (Q', \Sigma, \delta', q_0', F')$

~~(warm up: no ε-moves)~~ → Full version: with ε-moves

$Q' = 2^Q$

$q_0' = E(\{q_0\})$

$F' = \{R \subseteq Q \mid R \cap F \neq \emptyset\}$

$\forall a \in \Sigma, \forall R \subseteq Q$:

$\delta'(R, a) = \bigcup_{r \in R} E\left(\delta(r, a)\right)$

$\forall R \subseteq Q$

$E(R) = \{q \mid q$ reachable by 0 or more ε-moves from some $r \in R\}$

~~No ε-moves~~

Yes, ε-moves.

NB: do ε-moves *before* start, *after* other moves, not both before & after each move.

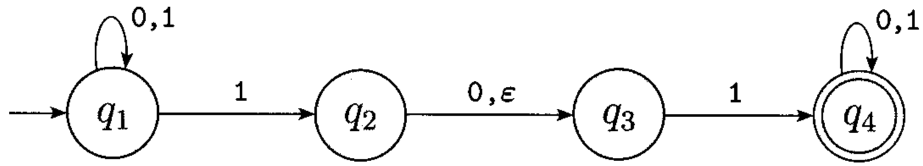
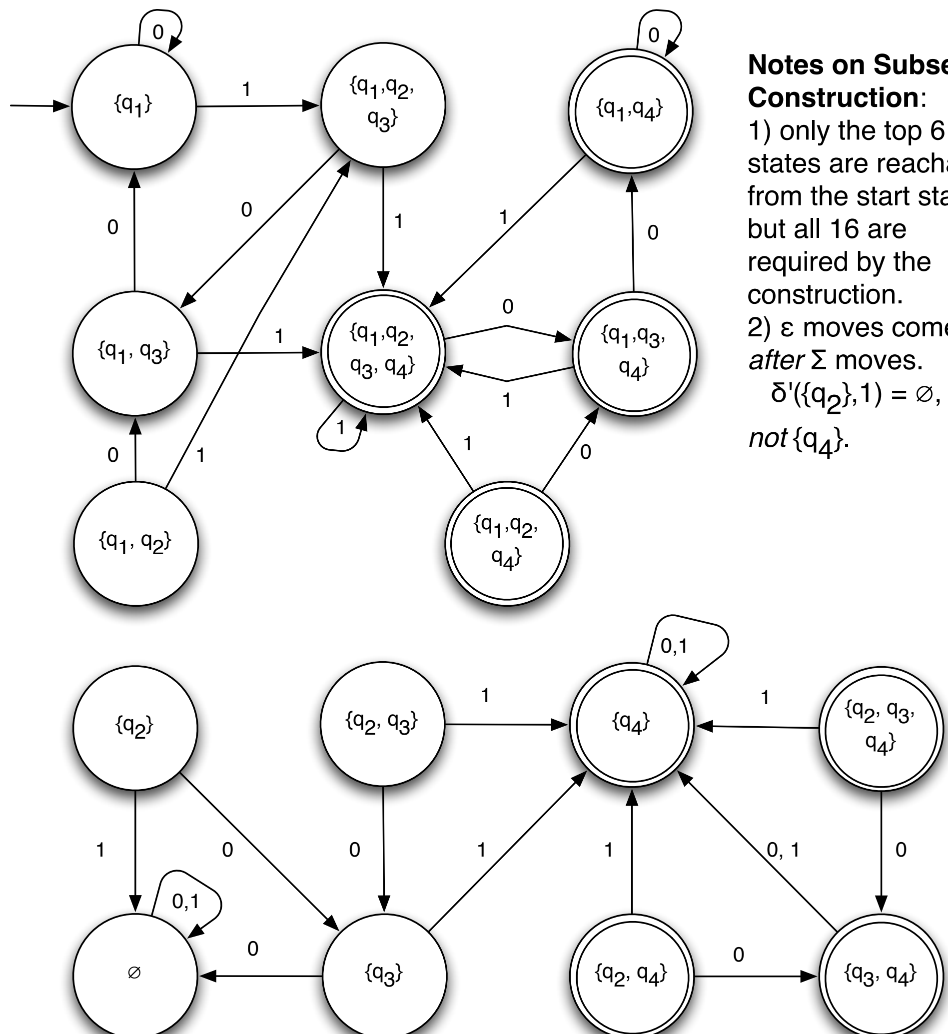


FIGURE 1.27



Notes on Subset Construction:

1) only the top 6 states are reachable from the start state, but all 16 are required by the construction.

2) ε moves come *after* Σ moves. E.g., $\delta'(\{q_2\},1) = \varnothing$, *not* $\{q_4\}$.