Automata and Languages

Prof. Mohamed Hamada

Software Engineering Lab.
The University of Aizu
Japan

Today's Topics

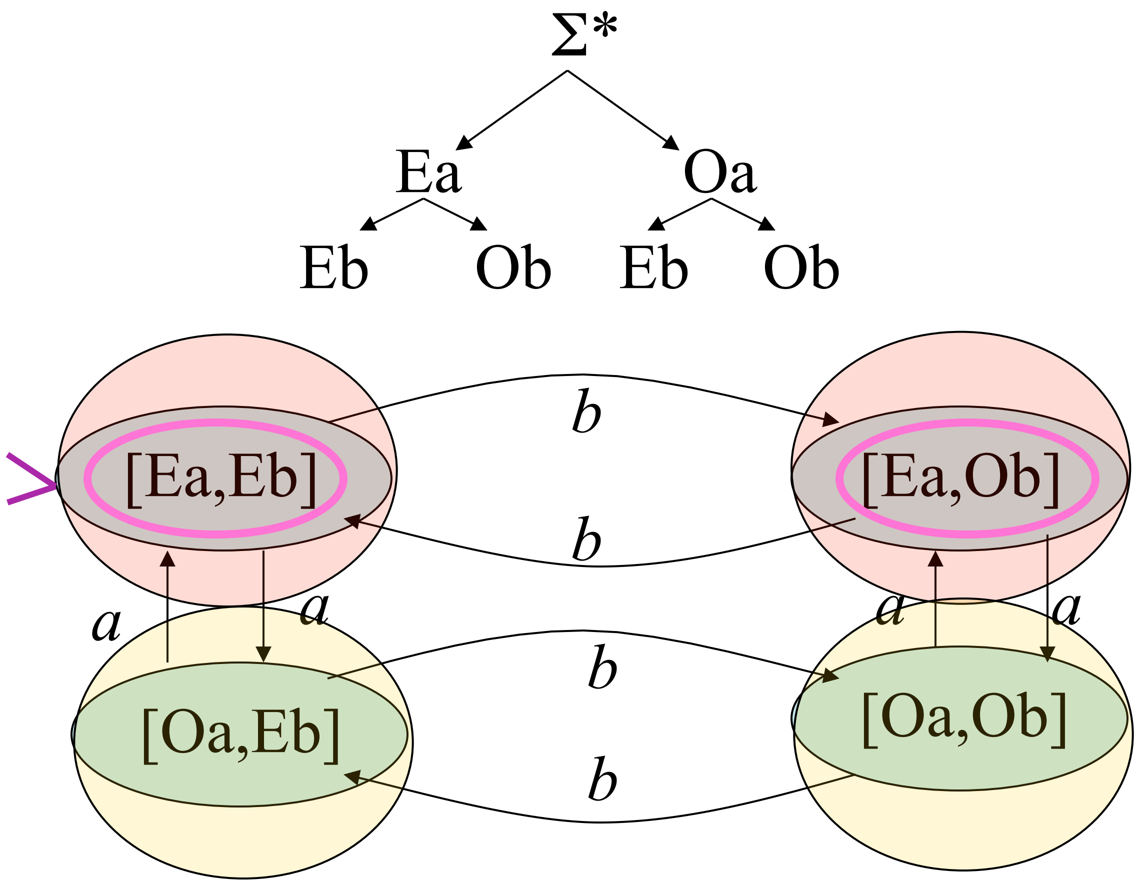
- DFA Minimization
- Examples
- Minimization Algorithms

DFA Minimization

DFA Minimization

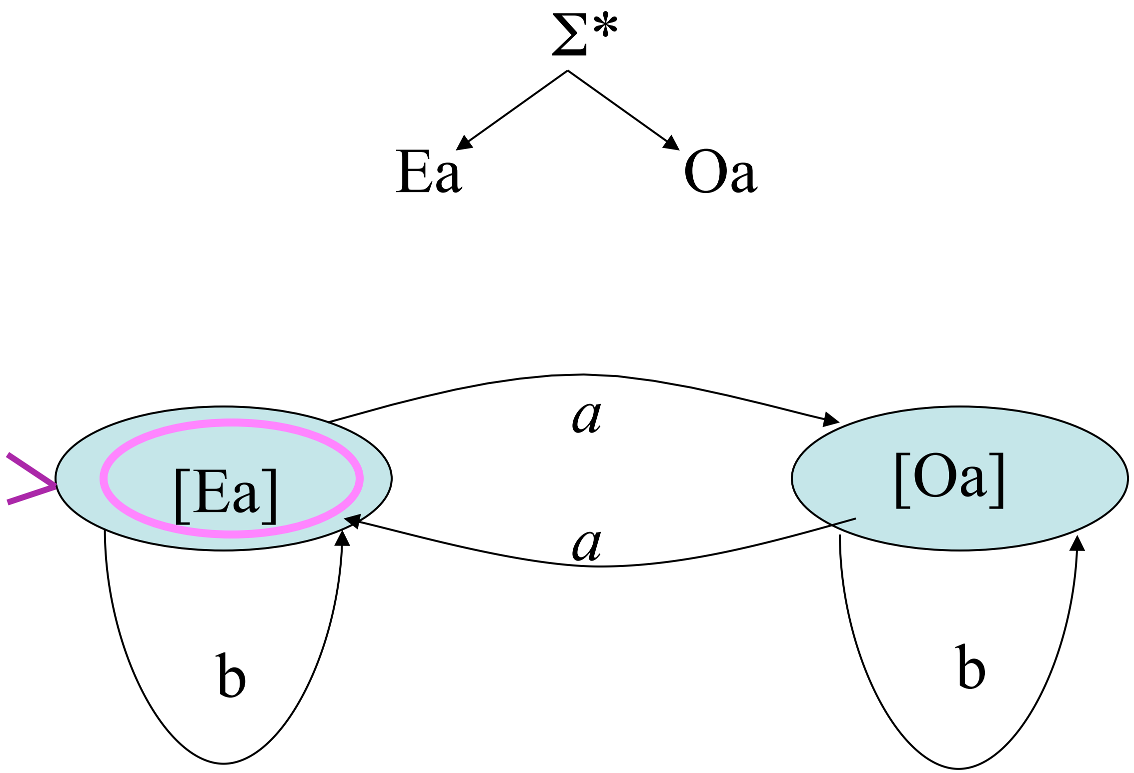
Example

Strings over {*a*,*b*} with even number of *a*' s



Example

Strings over {*a*,*b*} with even number of *a*' s



DFA Minimization

Observation

- The states among the state sets {[Ea,Eb], [Ea,Ob]} and {[Oa,Eb], [Oa,Ob]} differ on aspect immaterial for the problem at hand.
- Why not collapse these state sets into one state each, to get a smaller DFA?

Definition

Equivalent or Indistinguishable States

- Recall that a DFA state summarizes the substring consumed so far (that is, the past history).
- Two states q_i and q_j are *equivalent* or (*indistinguishable*), if, when started in these states, every string causes the machine to either end up in a final state for both or end up in a non-accepting state for both.

Two states q_i and q_k are ***equivalent*** (or ***indistinguishable***), **if for all strings $w \in \Sigma^*$**

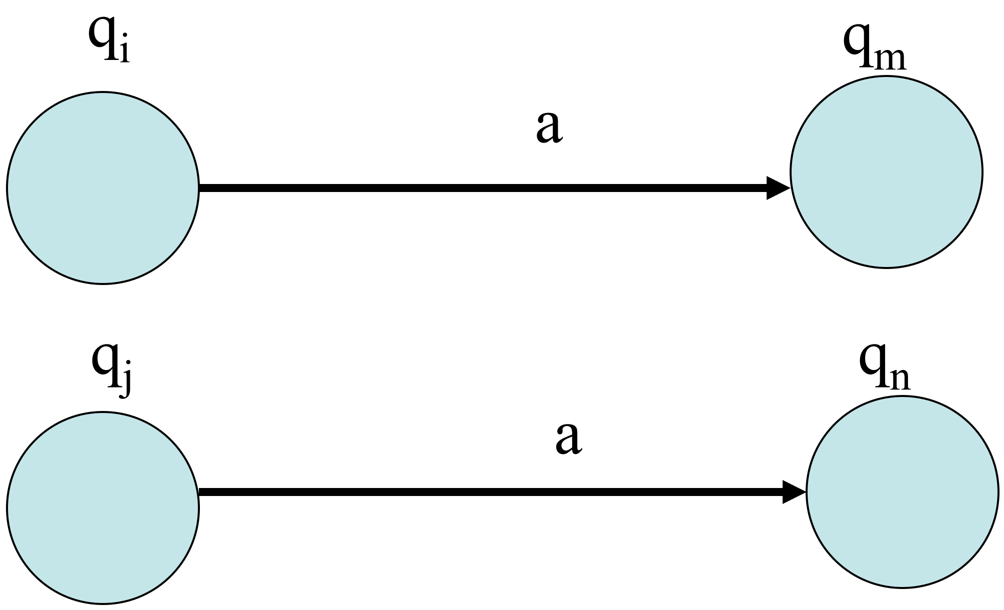
$$\delta(q_i, w) \in F \Leftrightarrow \delta(q_k, w) \in F$$

Two states q_i and q_k are ***distinguishable***, **if for some string $w \in \Sigma^*$**

$$\delta(q_i, w) \in F \Leftrightarrow \delta(q_k, w) \notin F$$

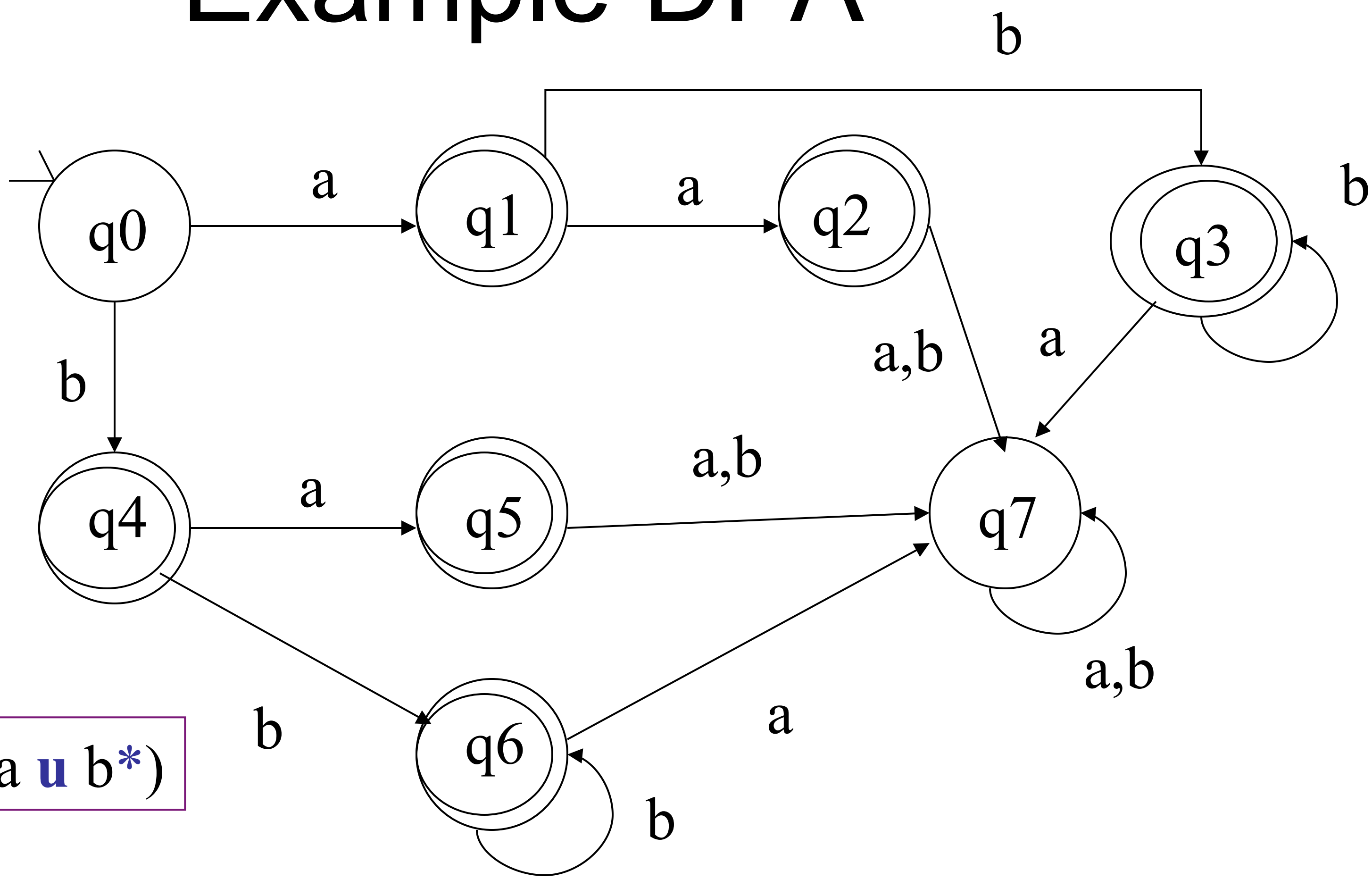
DFA Minimization

Main Idea



If q_m and q_n are distinguishable, then so are q_i and q_j

Example DFA



(a **u** b)(a **u** b*)

Example

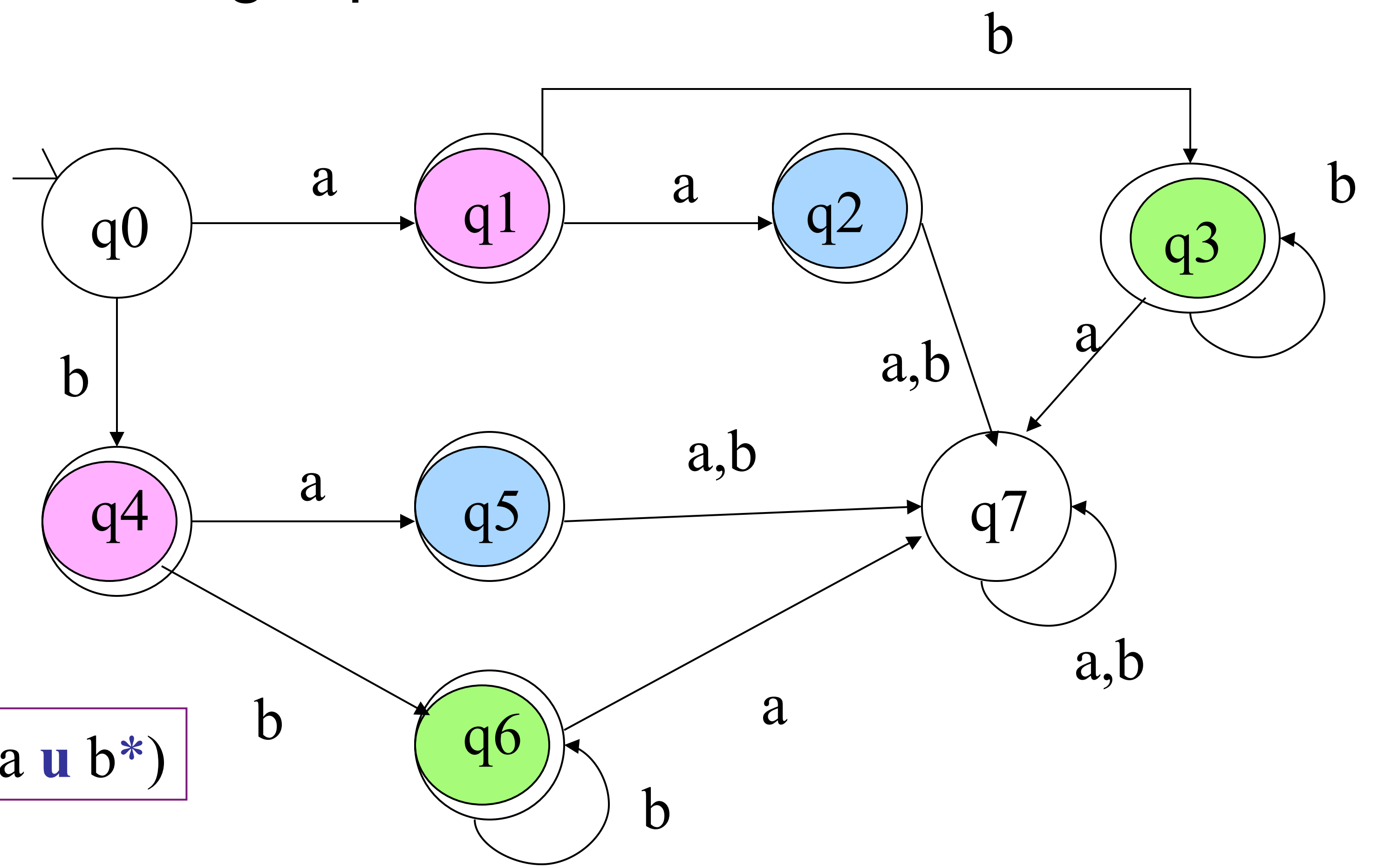
Refinement of State Partitions

- { {q0,q7} , {q1,q2,q3,q4,q5,q6} }
- { {q0},{q7}, {q1,q2,q3,q4,q5,q6} }
    - On any transition
- { {q0},{q7}, {q1,q2,q3,q4,q5,q6} }
- { {q0},{q7}, {q1,q4}, {q2,q3,q5,q6} }
    - On “a” transition
- { {q0},{q7}, {q1,q4}, {q2,q5},{q3,q6} }
    - On “b” transition

DFA Minimization

Example

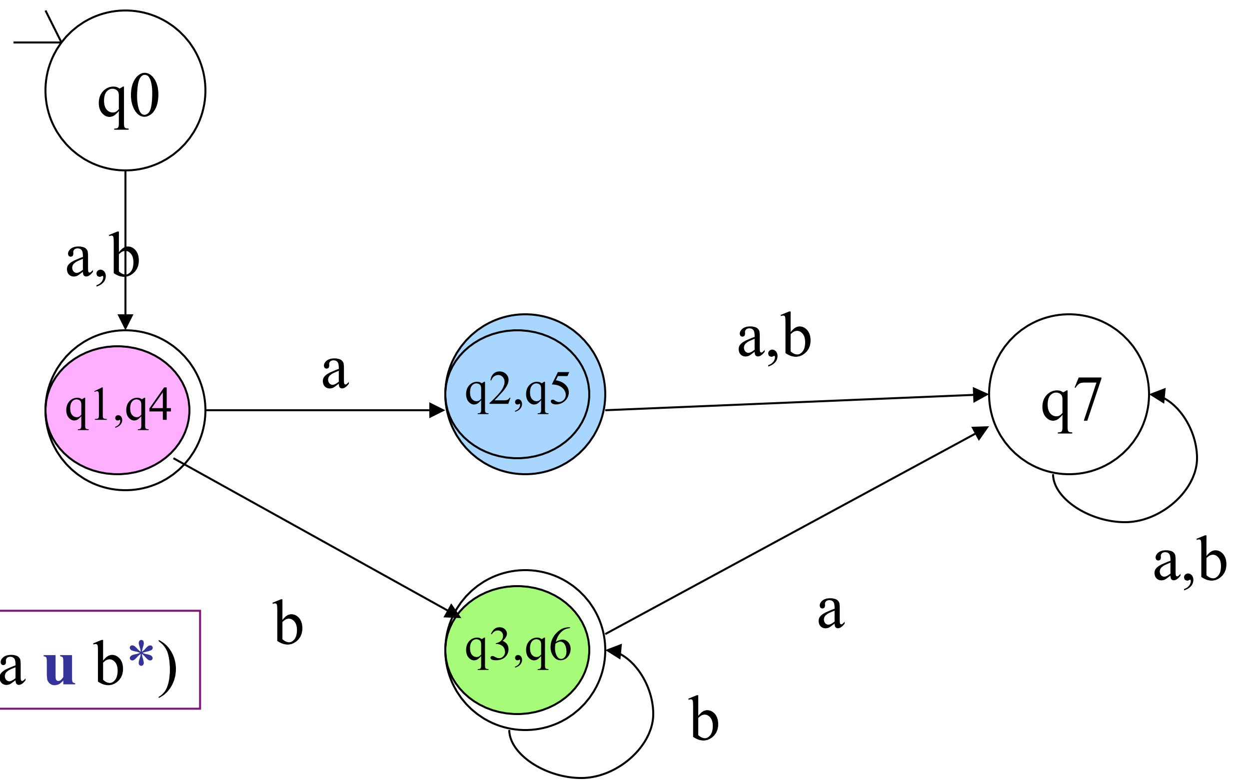
Showing equivalent states in DFA



(a **u** b)(a **u** b*)

Example

Minimum DFA



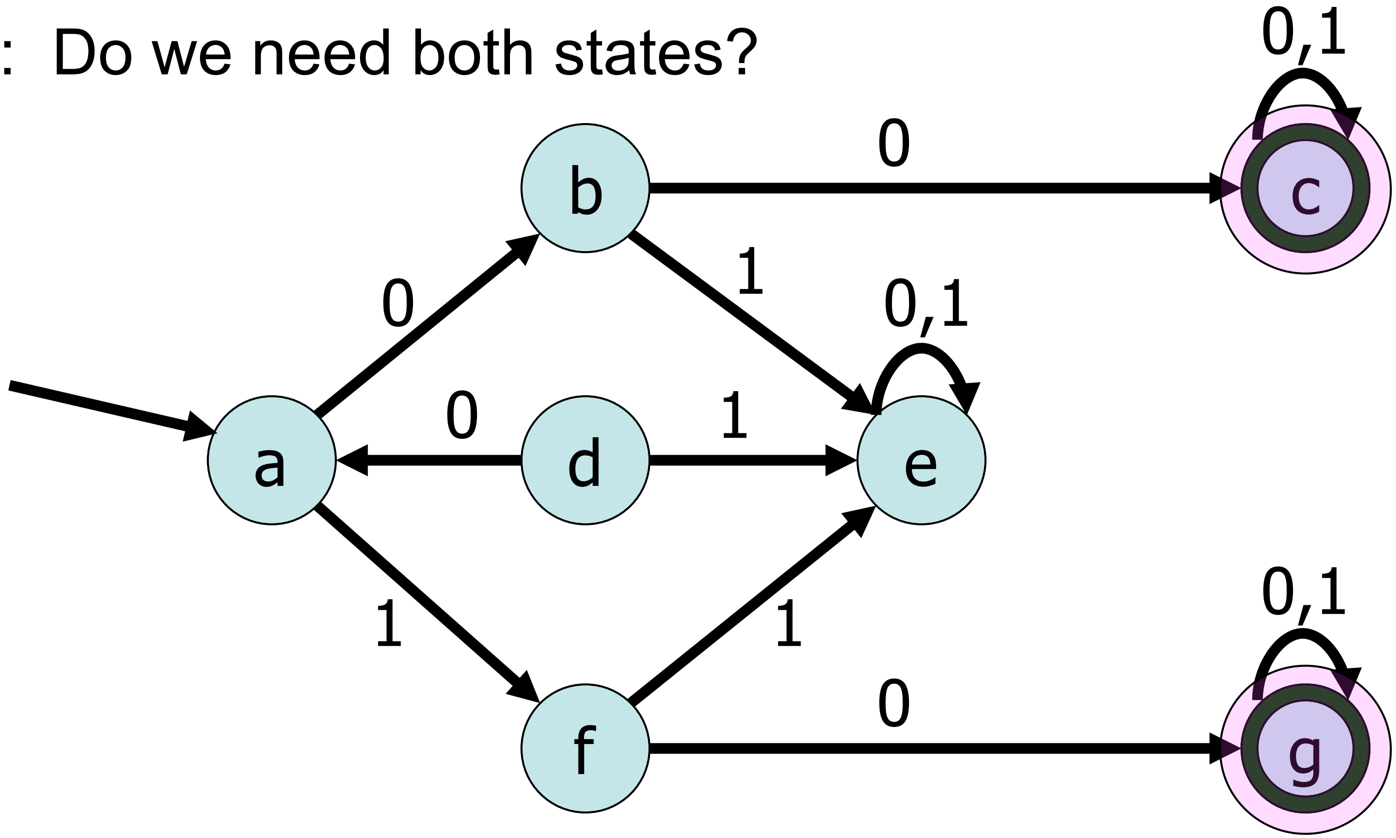
(a **u** b)(a **u** b*)

Example

Equivalent States

Consider the accept states c and g. They are both sinks meaning that any string which ever reaches them is guaranteed to be accepted later.

Q: Do we need both states?

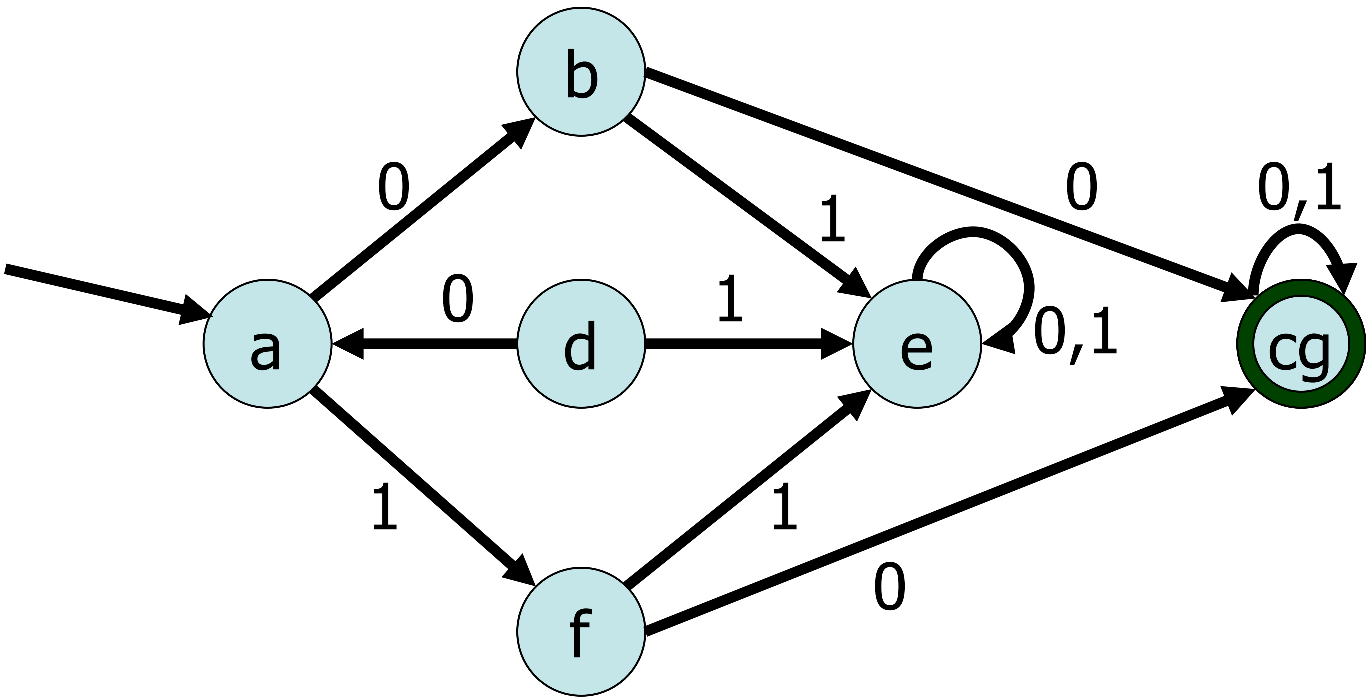


Example

Joining Equivalent States

A: No, they can be unified as illustrated below.

Q: Can any other states be unified because any subsequent string suffixes produce identical results?



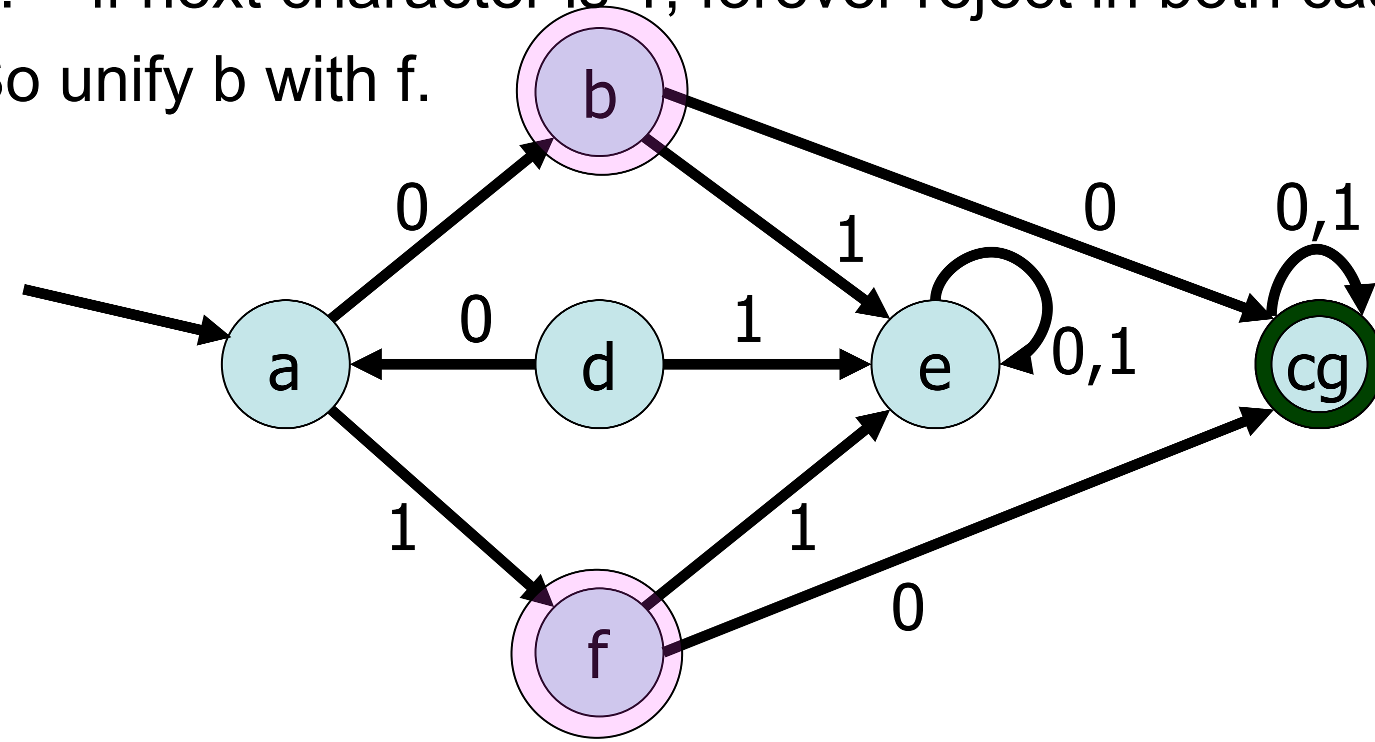
Example

Equivalent States

A: Yes, b and f. Notice that if you're in b or f then:

1. if string ends, reject in both cases
2. if next character is 0, forever accept in both cases
3. if next character is 1, forever reject in both cases

So unify b with f.

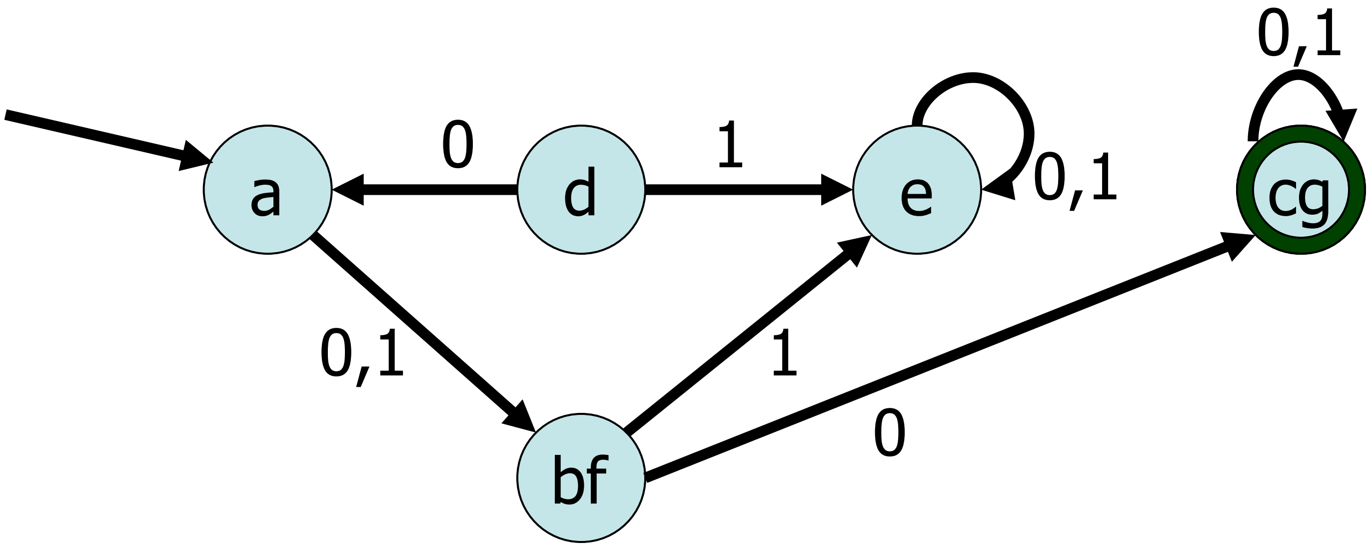


Example

Joining Equivalent States

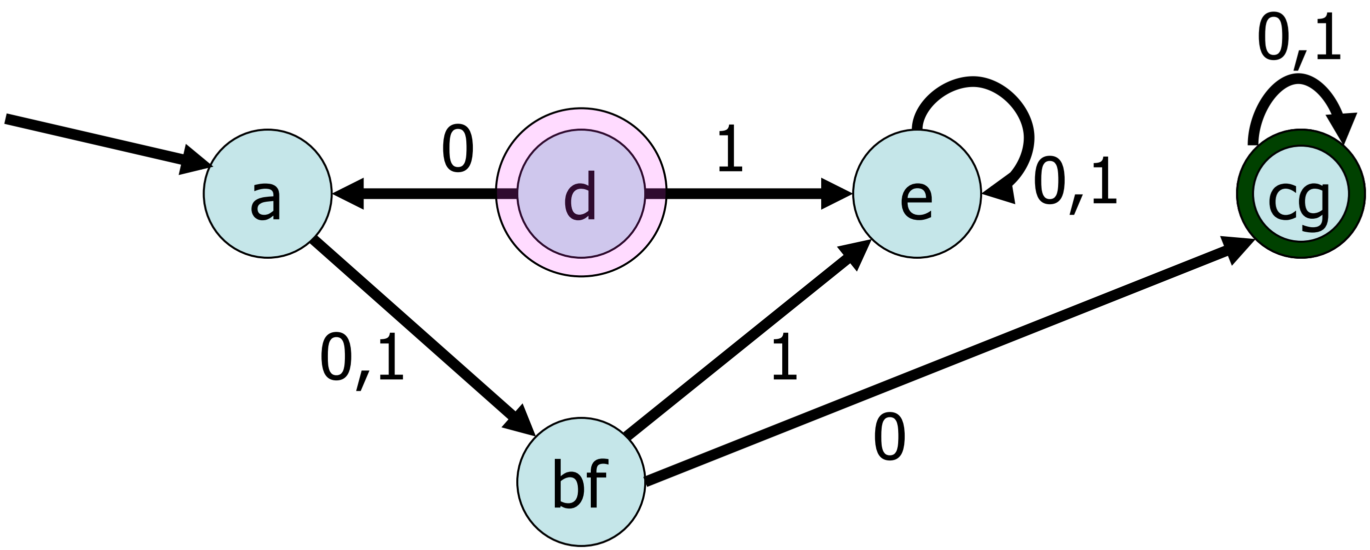
Intuitively two states are equivalent if all subsequent behavior from those states is the same.

Q: Come up with a formal characterization of state equivalence.



Finishing the Example

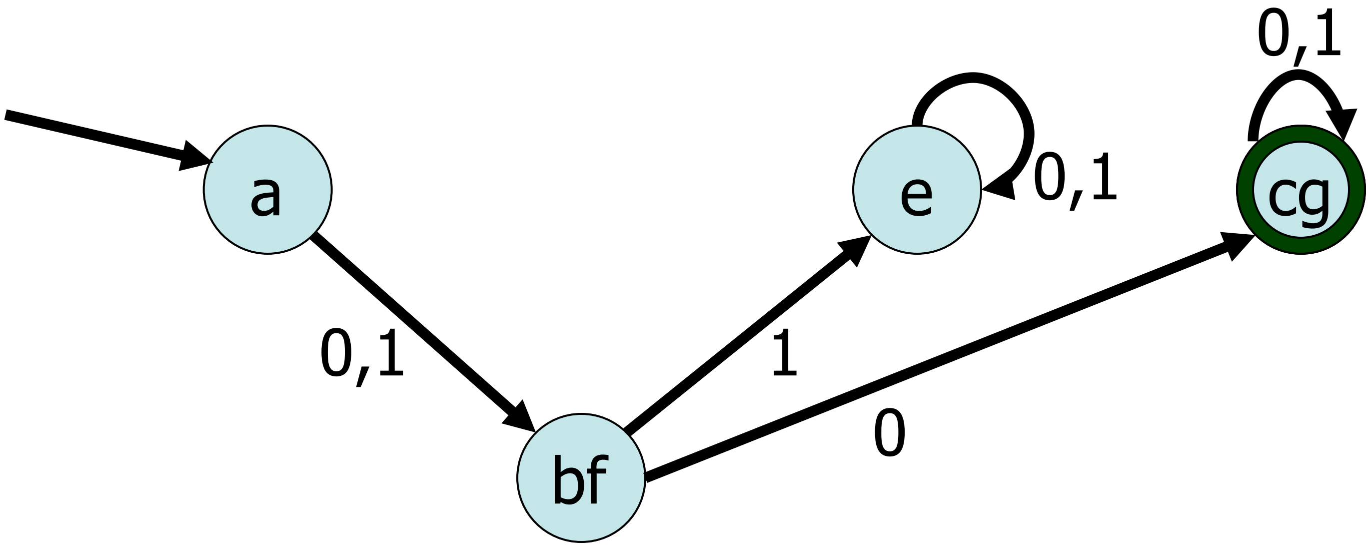
Q: Any other ways to simplify the automaton?



Useless States

A: Get rid of d.

Getting rid of unreachable ***useless states*** doesn't affect the accepted language.



DFA Minimization

Definition

An automaton is ***irreducible*** if

- it contains no useless states, and
- no two distinct states are equivalent.

Goals of the Minimization Algorithm

- The goal of minimization algorithm is to create irreducible automata from arbitrary ones.

- The algorithm actually produces smallest possible DFA for the given language, hence the name "minimization".

DFA Minimization

Minimization Algorithm

First Part: Partition

DFA minimize(DFA $(Q, \Sigma, \delta, q_0, F)$)

remove any state q unreachable from q_0

Partition $P = \{F, Q - F\}$

boolean Consistent = false

while(Consistent == false)

Consistent = true

for(every Set $S \in P$, symbol $a \in \Sigma$, Set $T \in P$)

Set temp = $\{q \in T \mid \delta(q,a) \in S\}$

if (temp != Ø && temp != T)

Consistent = false

$P = (P - T) \cup \{\text{temp}, T - \text{temp}\}$

return defineMinimizor($(Q, \Sigma, \delta, q_0, F)$, P)

DFA Minimization

Minimization Algorithm

Second Part: Minimization

DFA defineMinimizor (DFA (Q, Σ, δ, q_0, F), Partition P)

Set $Q' = P$

State q'_0 = the set in P which contains q_0

$F' = \{ S \in P \mid S \subseteq F \}$

for (each $S \in P$, $a \in \Sigma$)

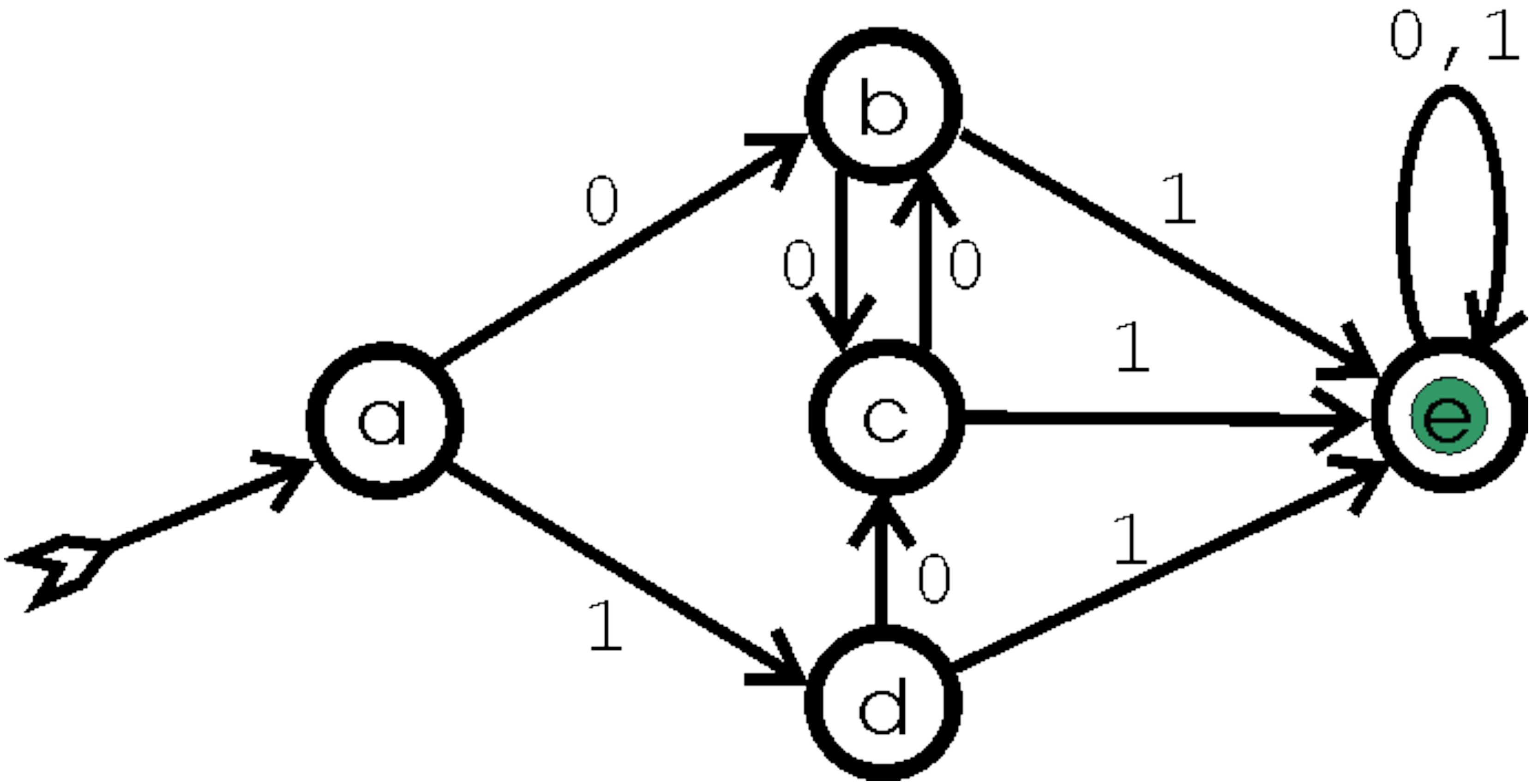
define $\delta'(S,a)$ = the set $T \in P$ which contains the states $\delta'(S,a)$

return (Q', Σ, δ', q'_0, F')

Example

Minimization Example

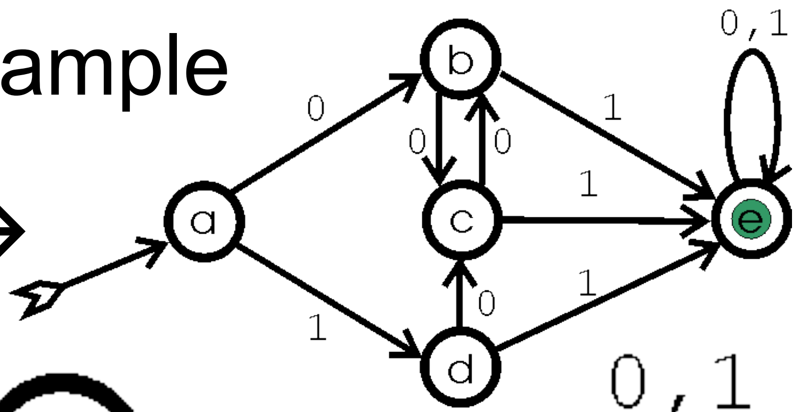
Start with a DFA

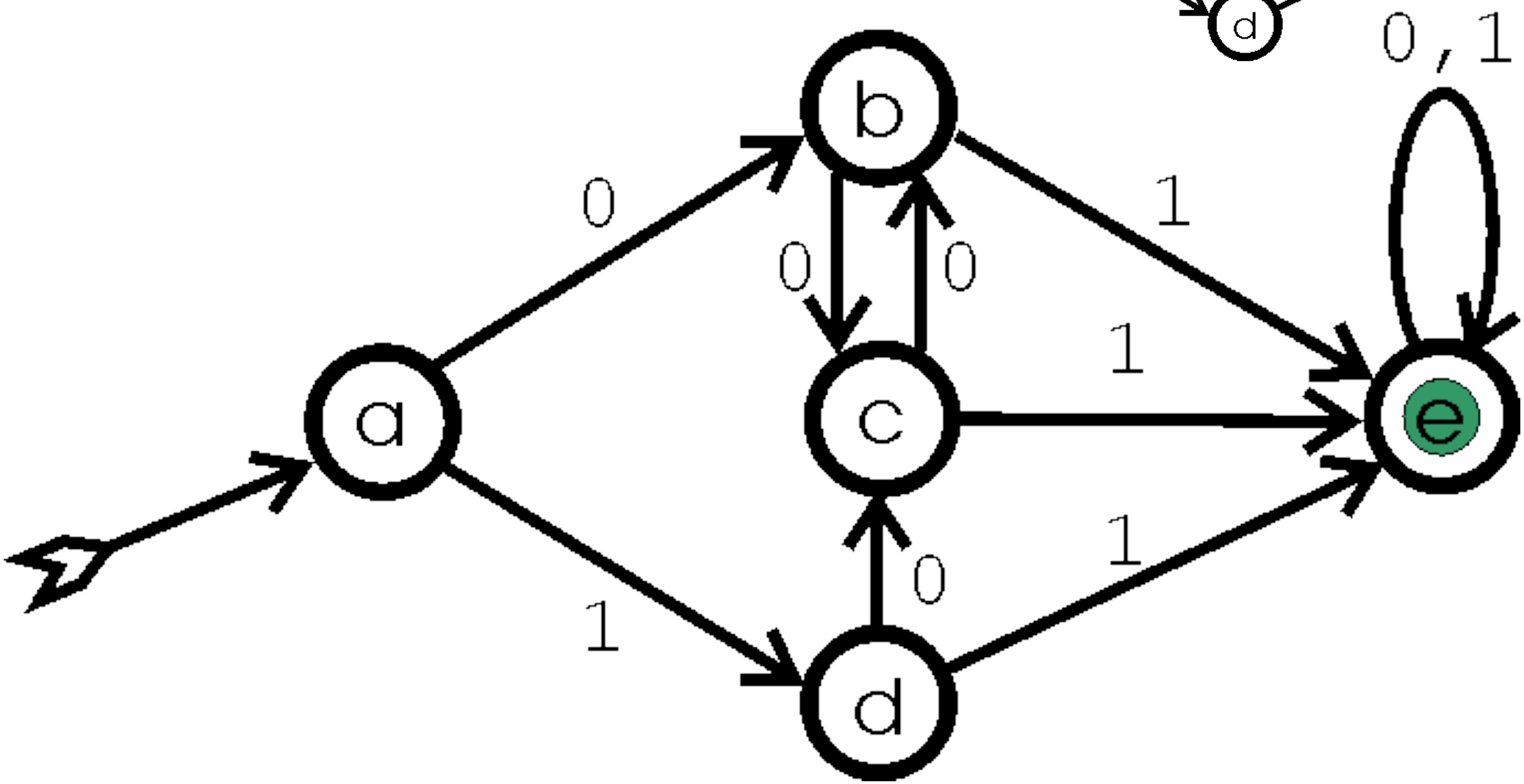


Example

Minimization Example

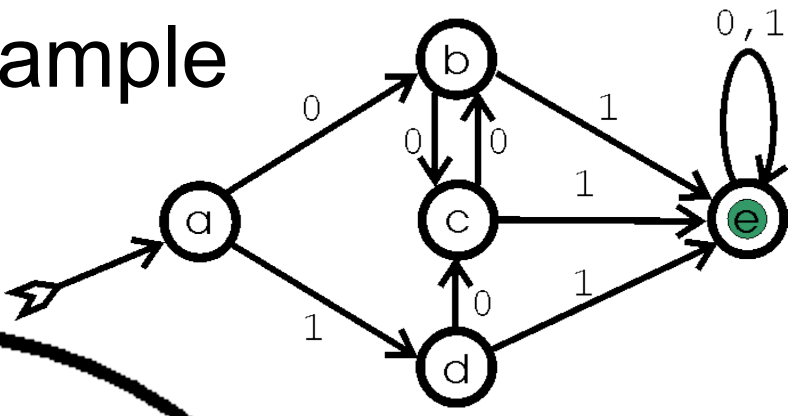
Miniature version →





Example

Minimization Example

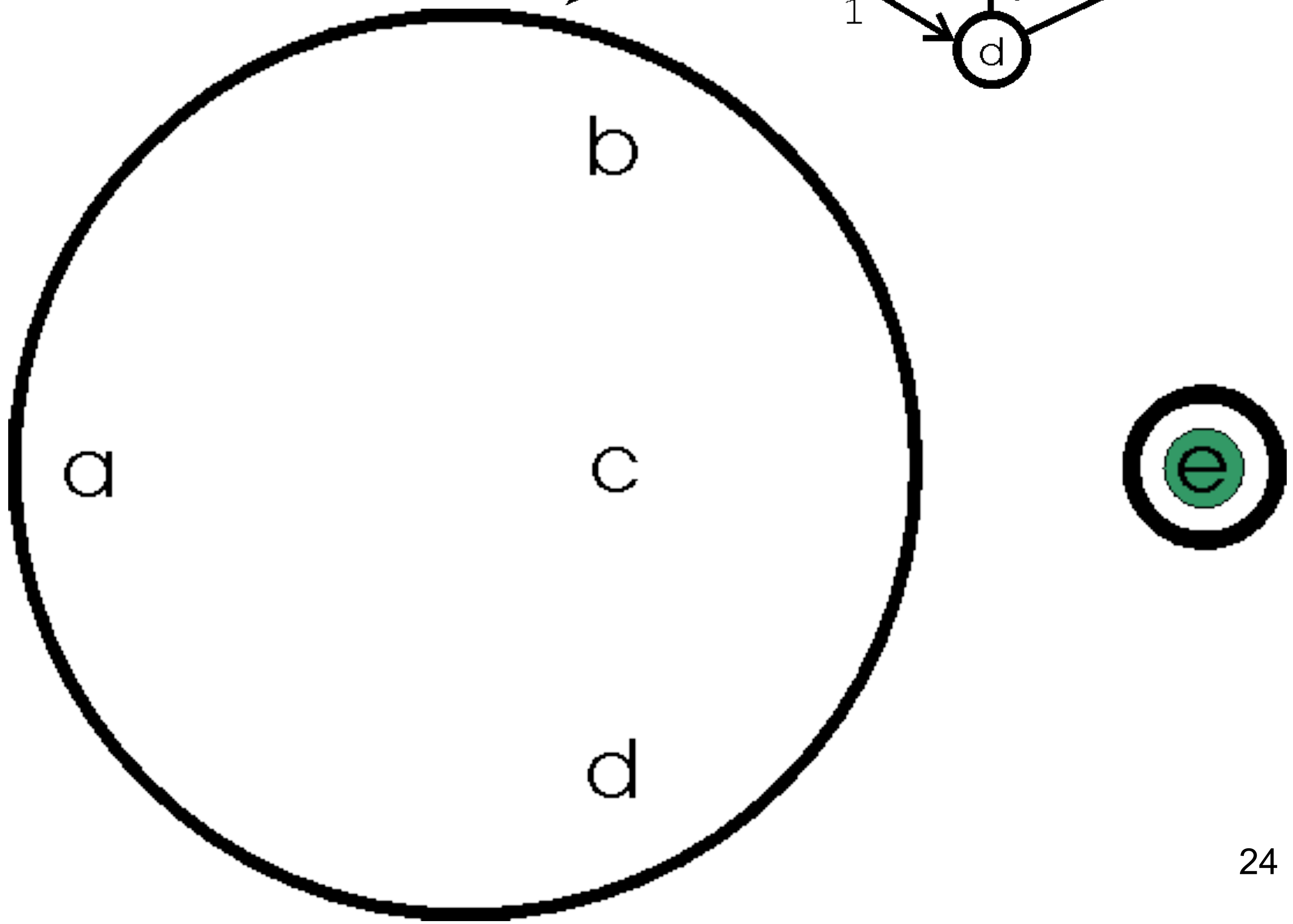


Split into two parts.

ACCEPT

vs.

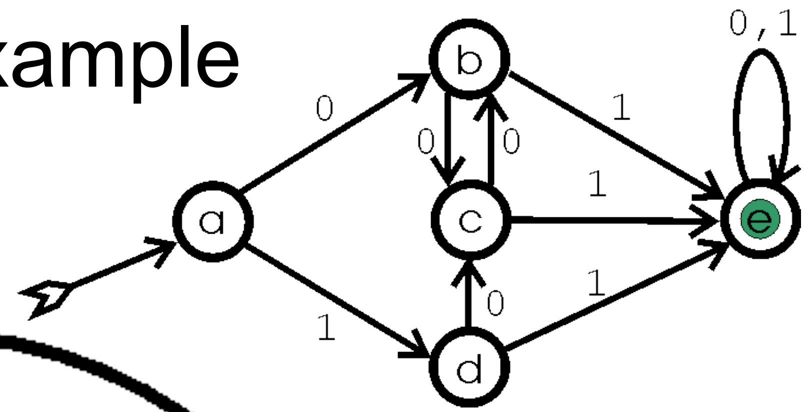
REJECT

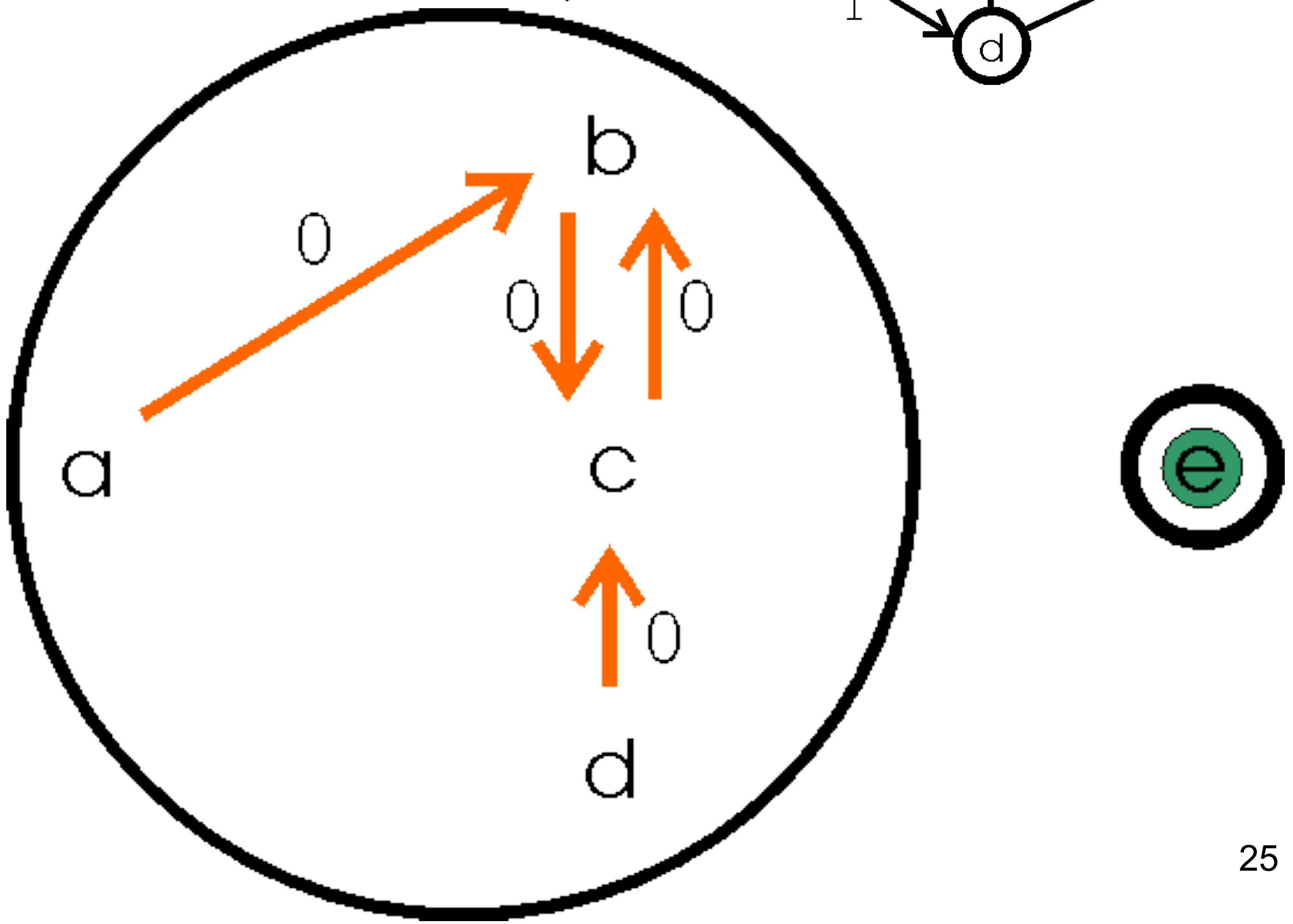


Example

Minimization Example

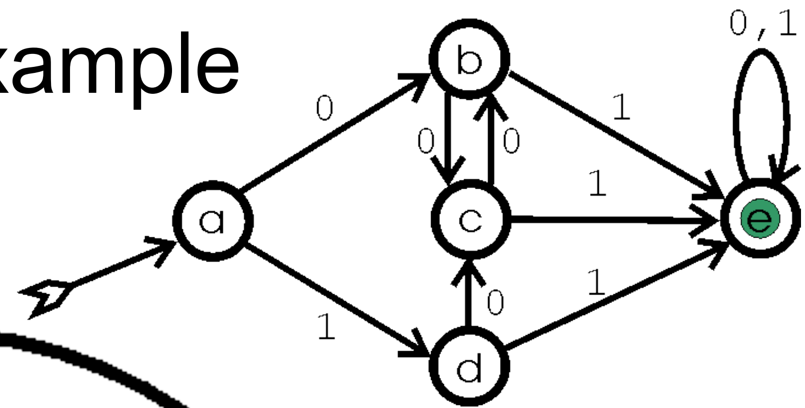
0-label doesn't split



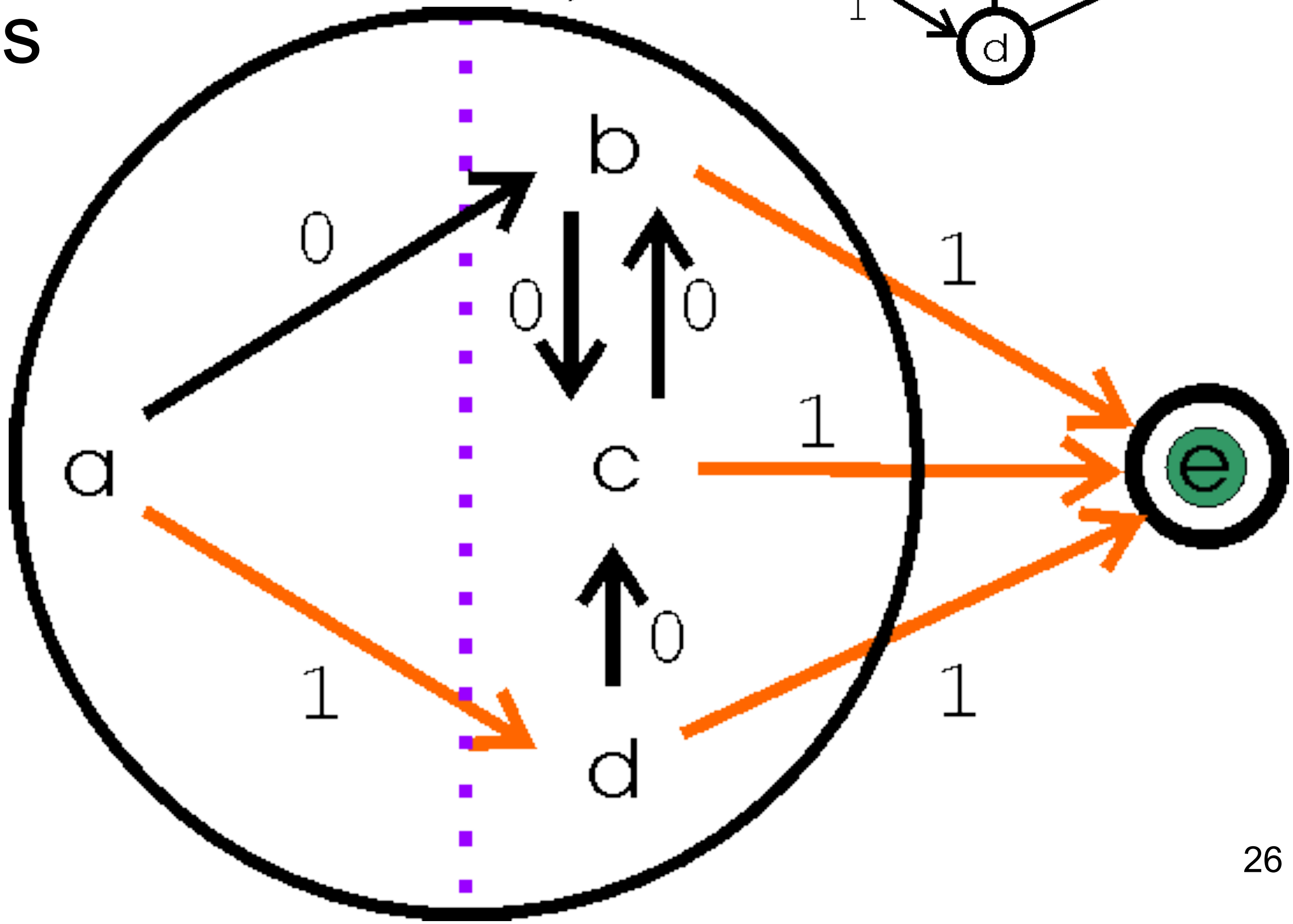


Example

Minimization Example

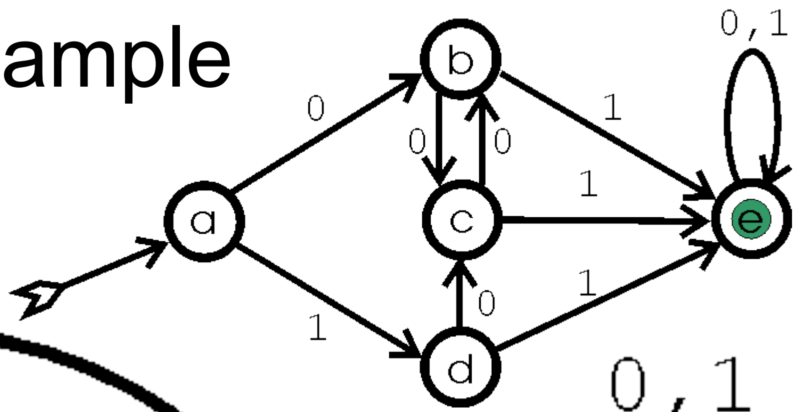


1-label splits up REJECT's

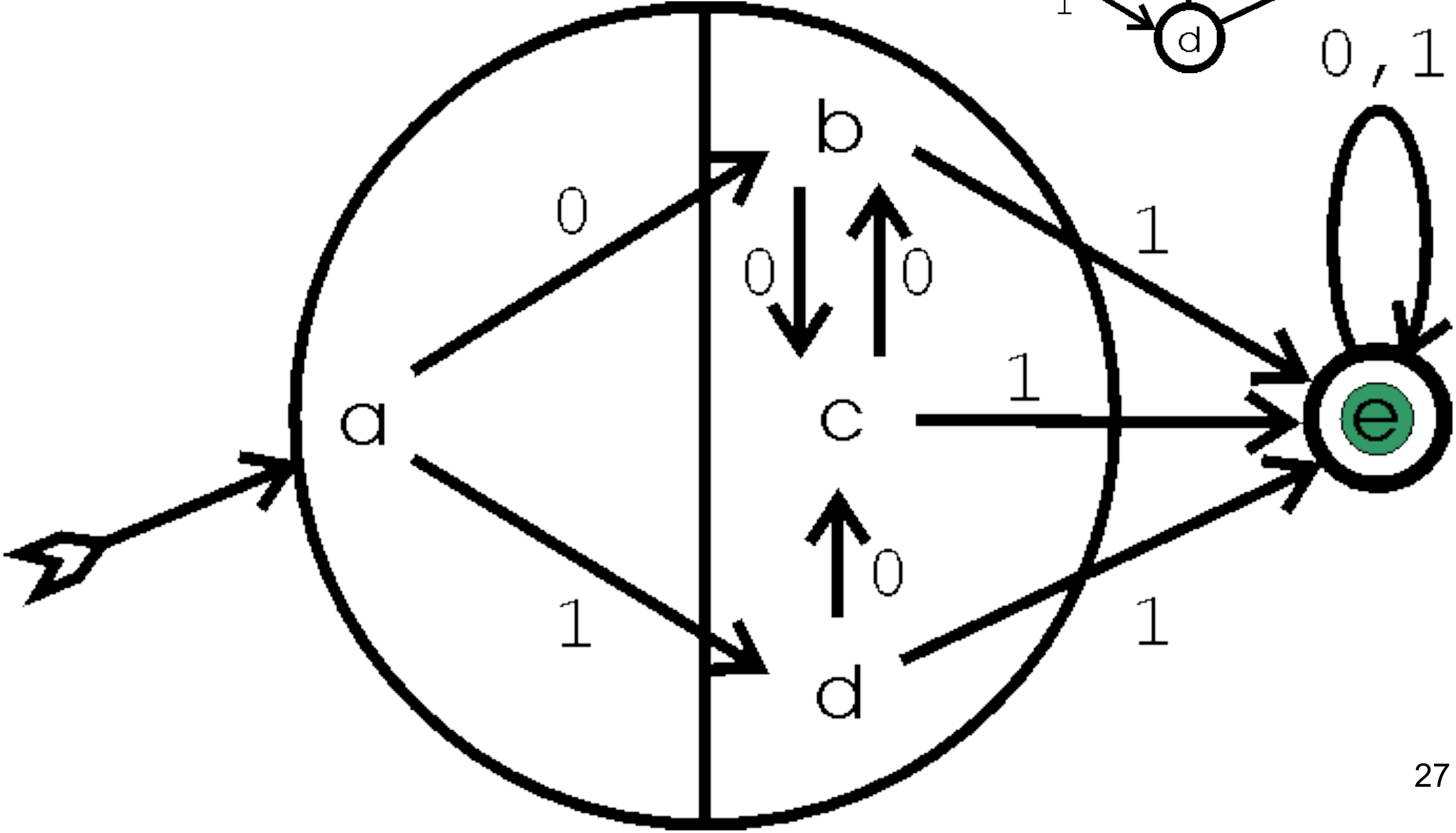


Example

Minimization Example



No further splits. HALT!

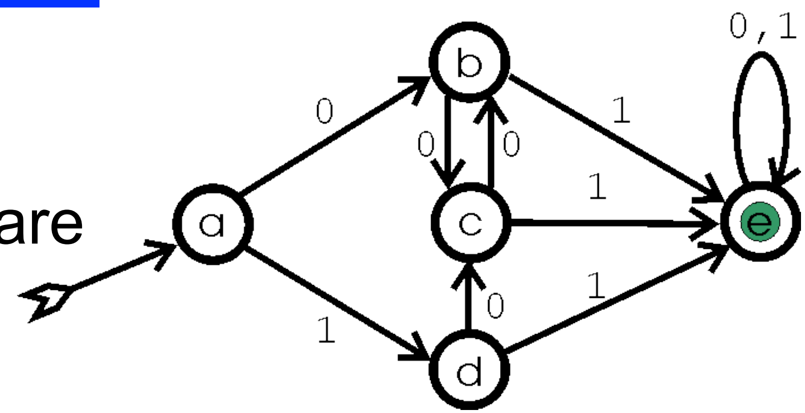


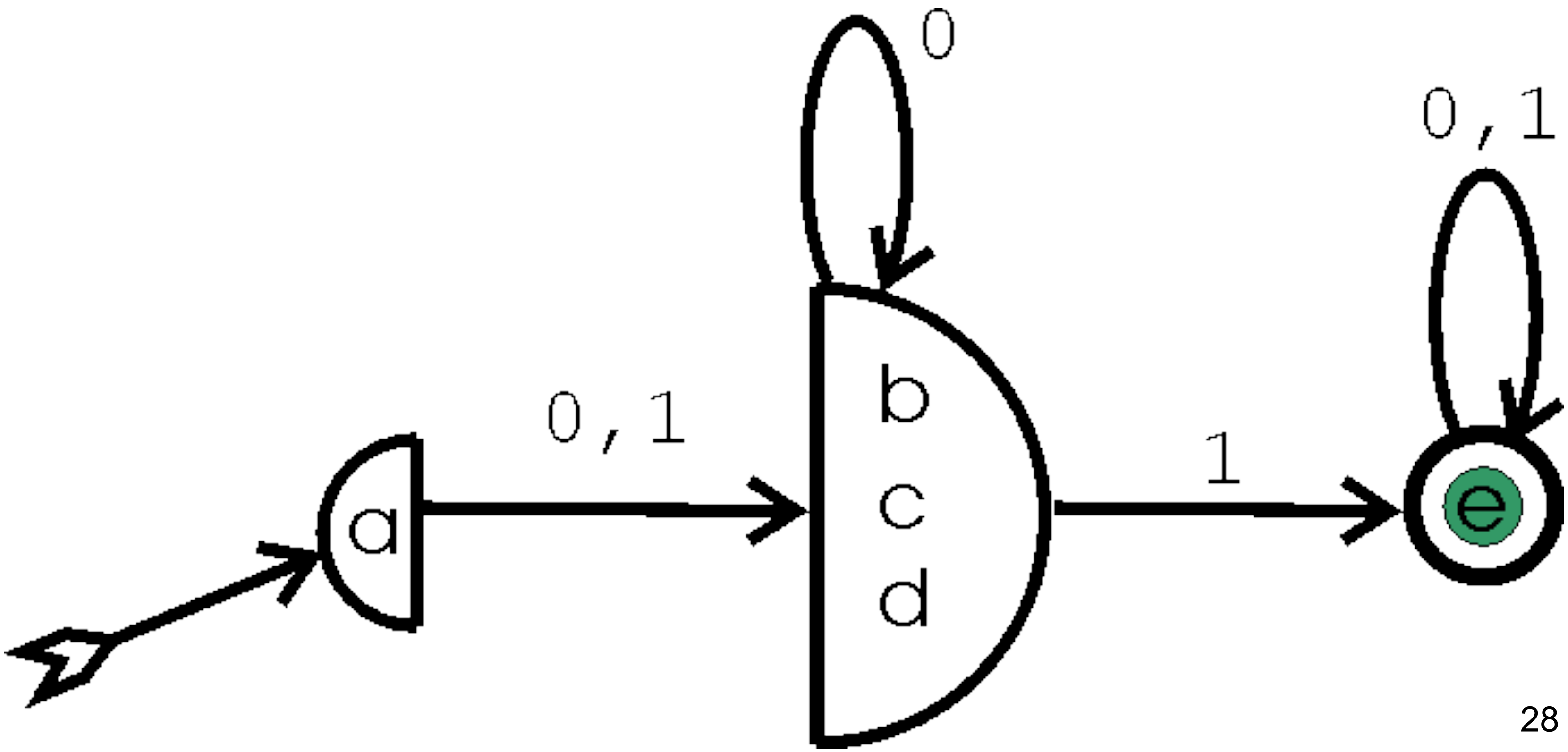
DFA Minimization

Example

End Result

States of the minimal automata are remaining.



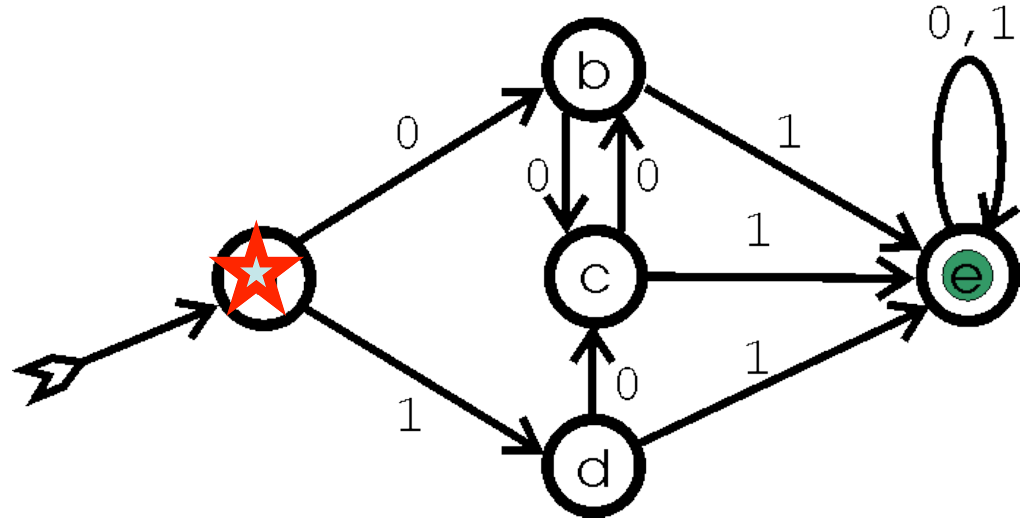


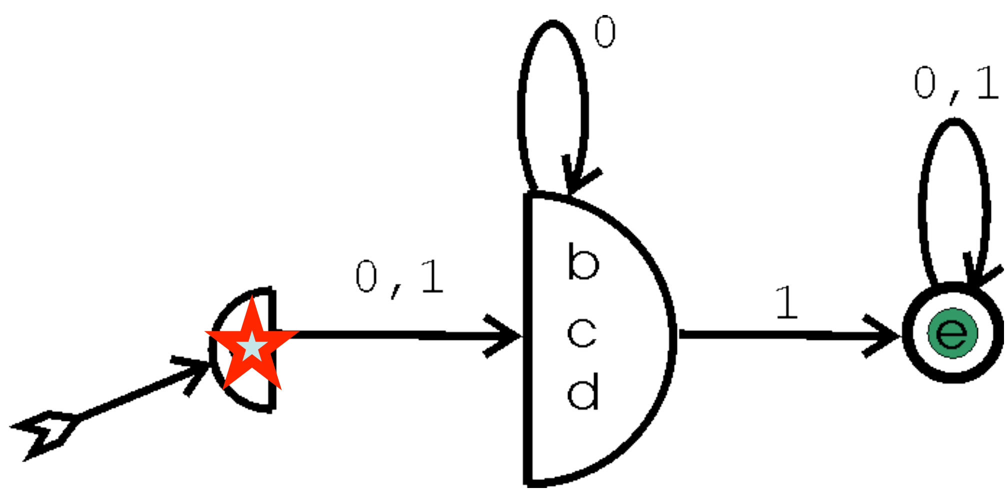
Example

Minimization Example. Compare

100100101

↑

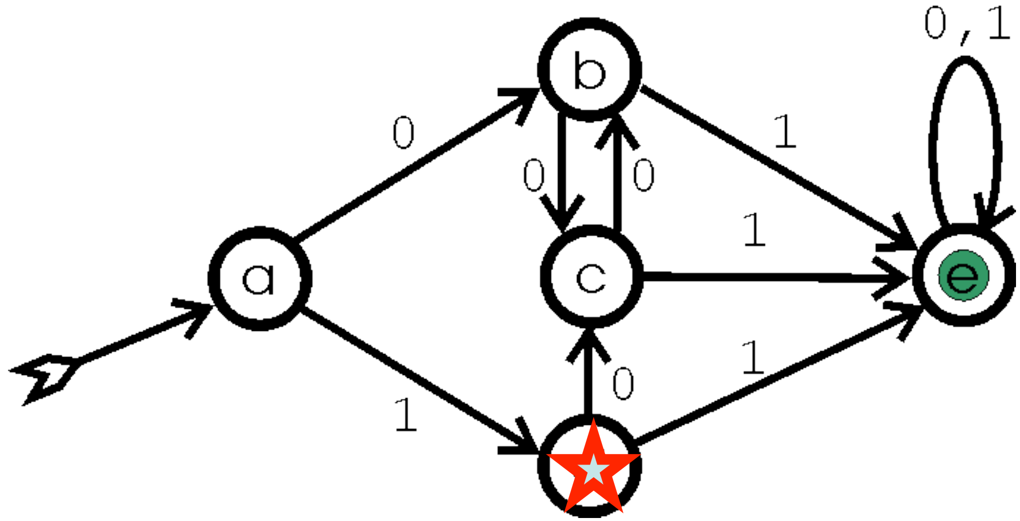


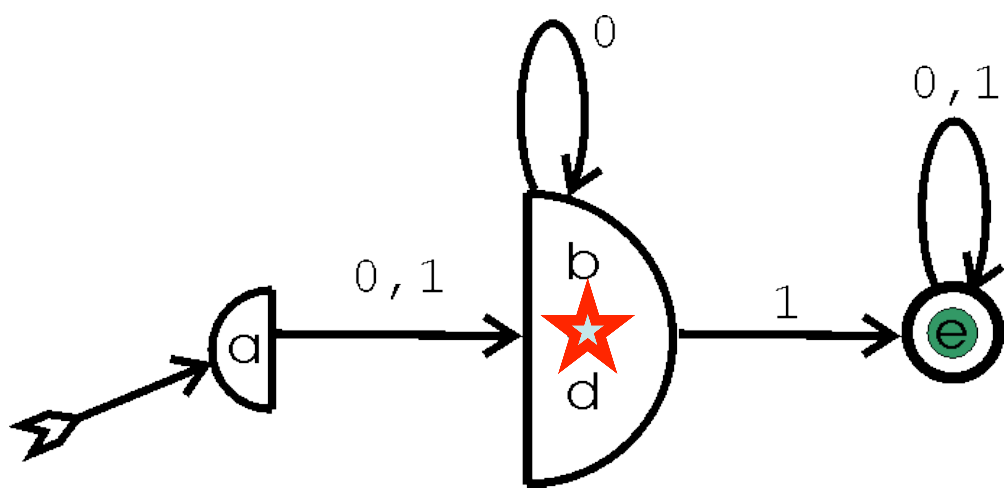


Example

Minimization Example. Compare

100100101

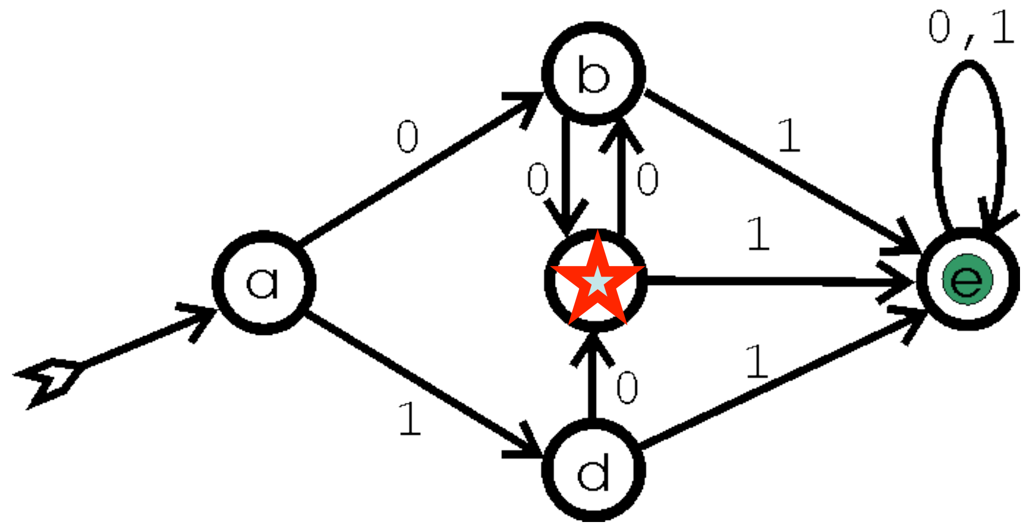


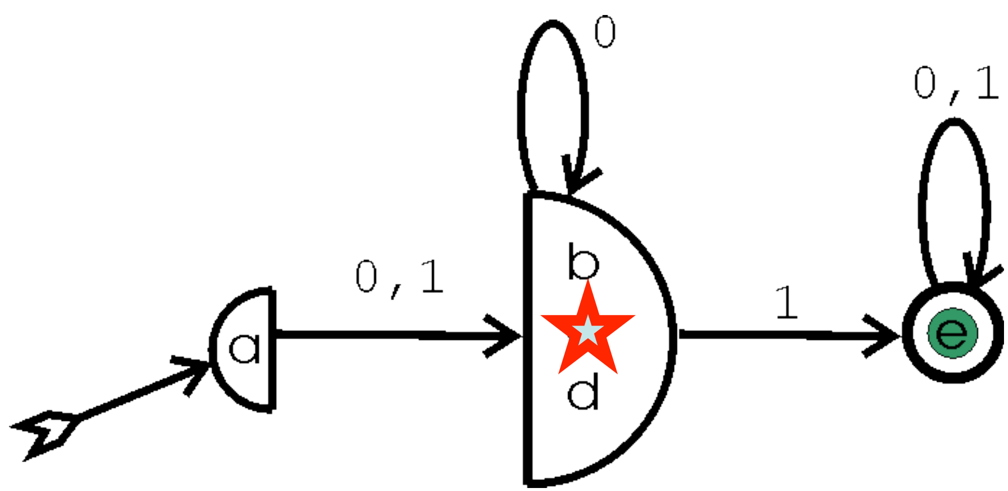


Example

Minimization Example. Compare

100100101



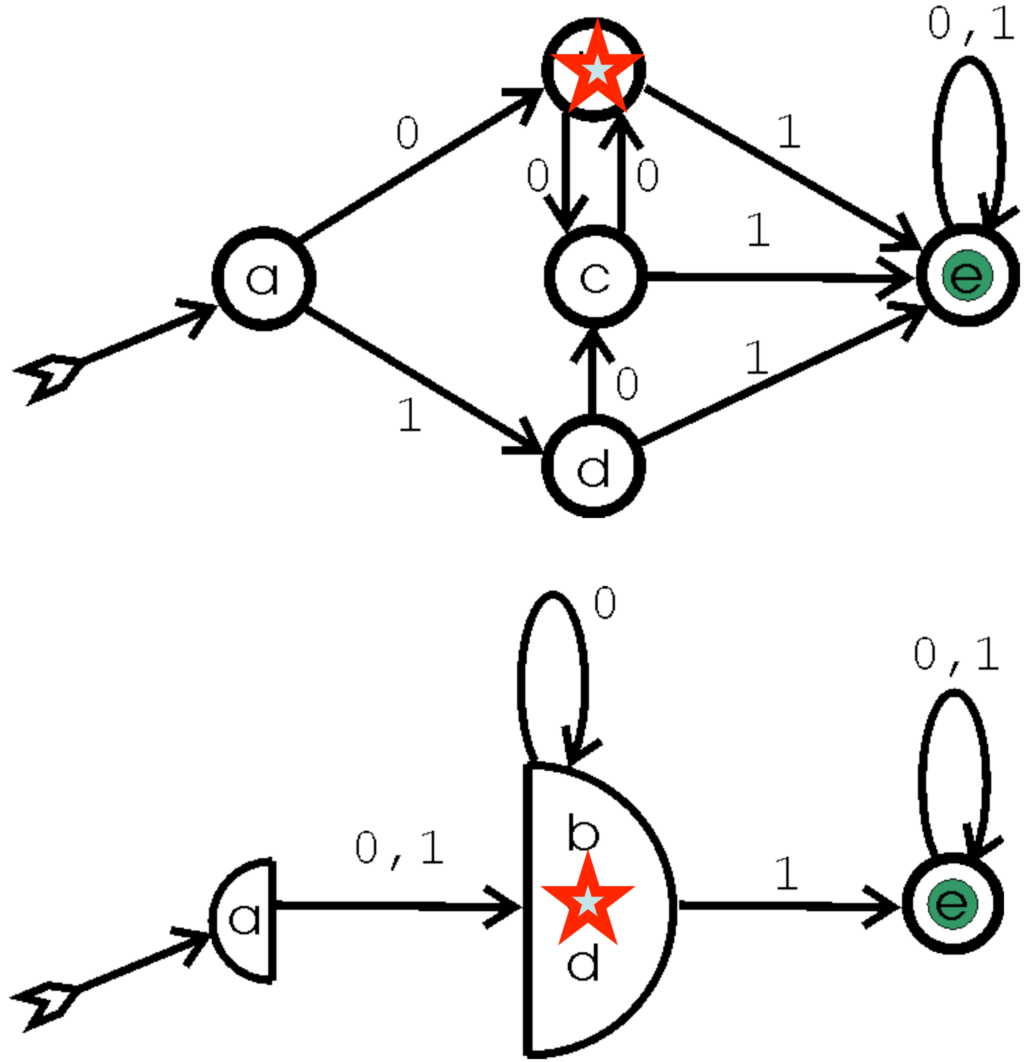


DFA Minimization

Example

Minimization Example. Compare

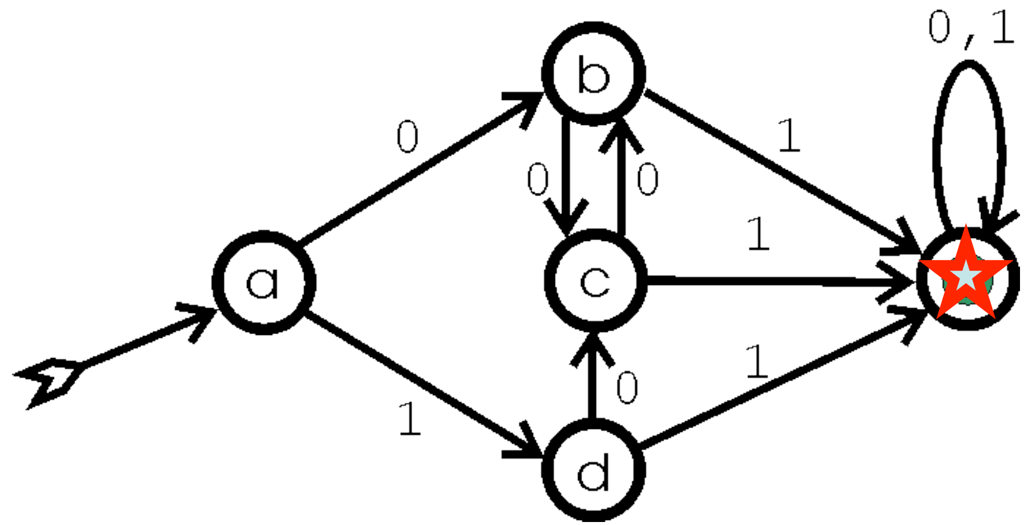
100100101

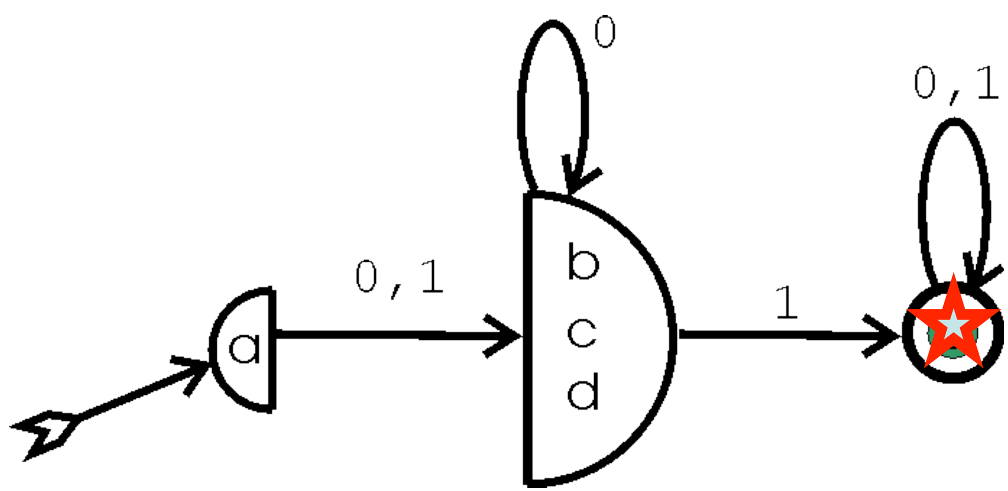


Example

Minimization Example. Compare

100100101

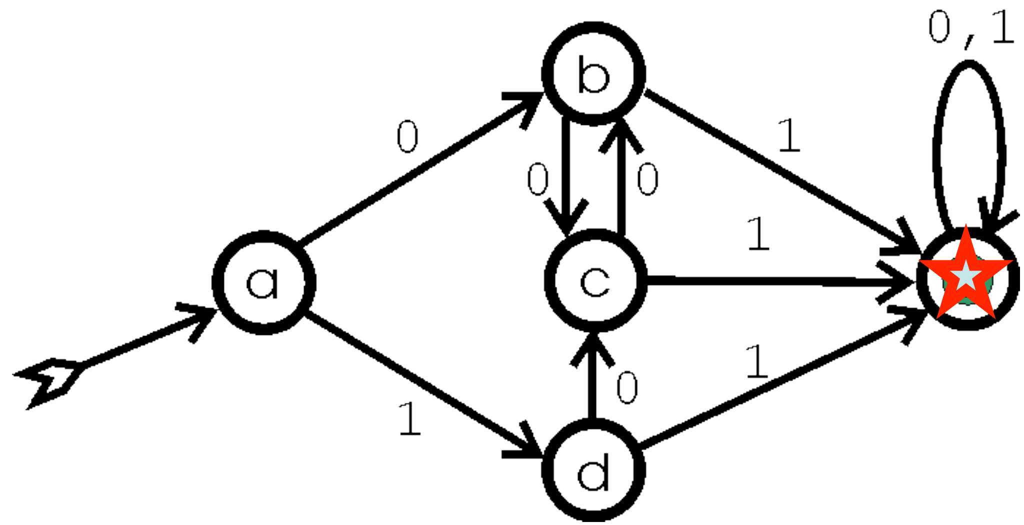


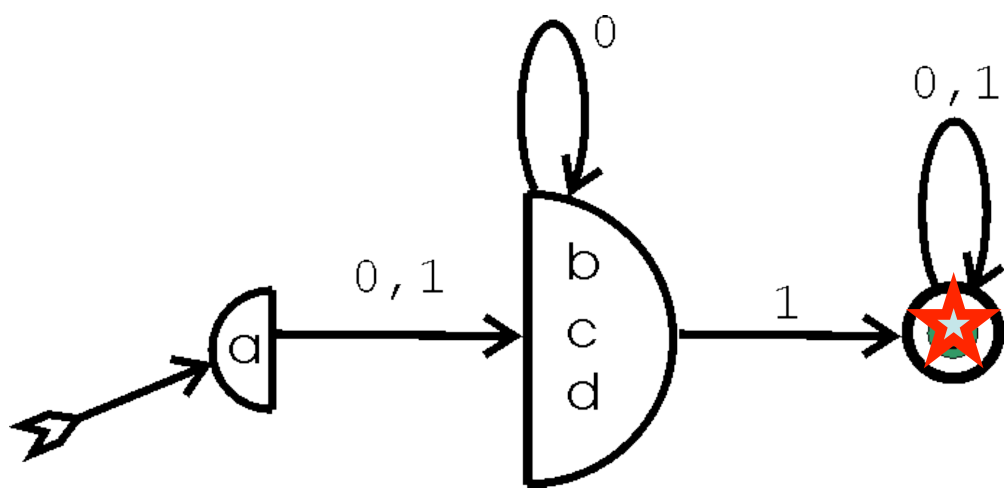


Example

Minimization Example. Compare

100100101

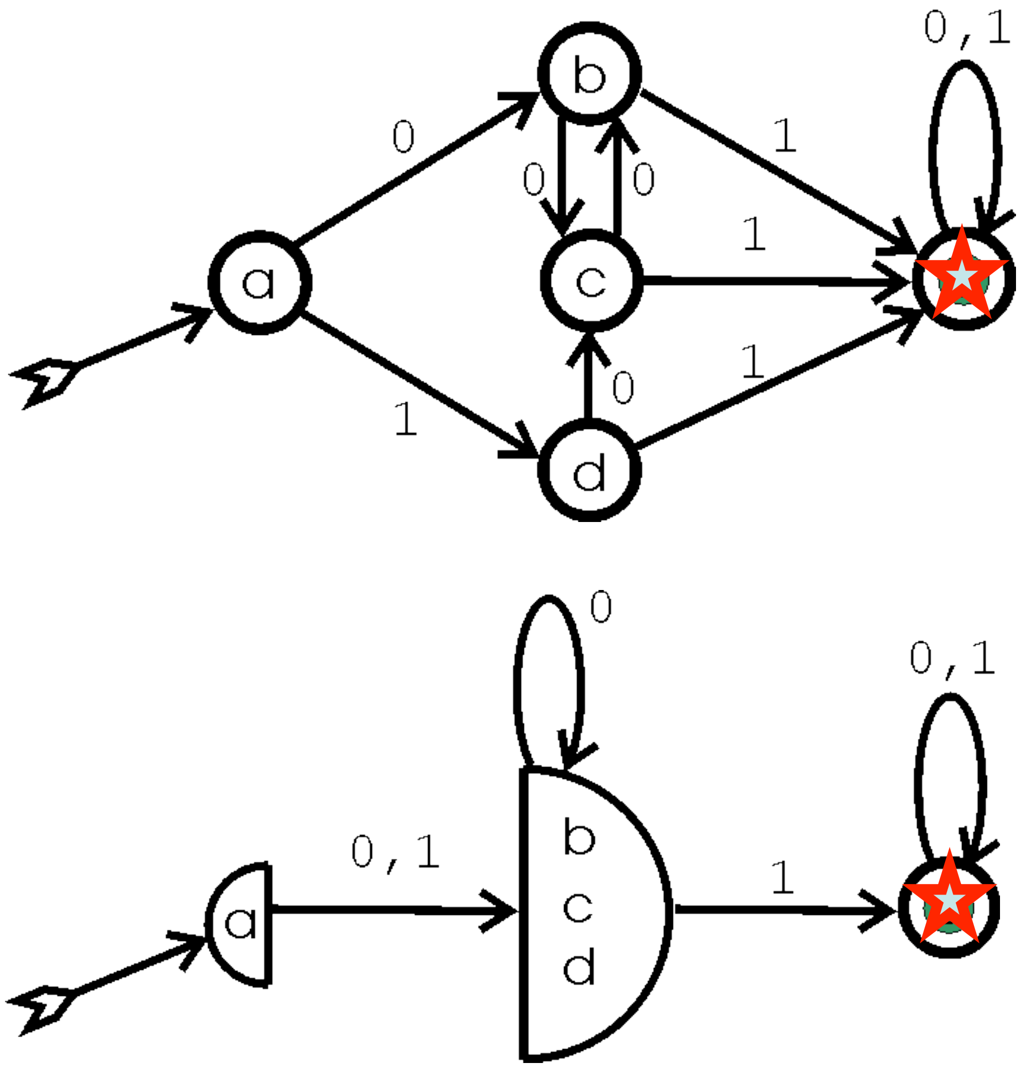




Example

Minimization Example. Compare

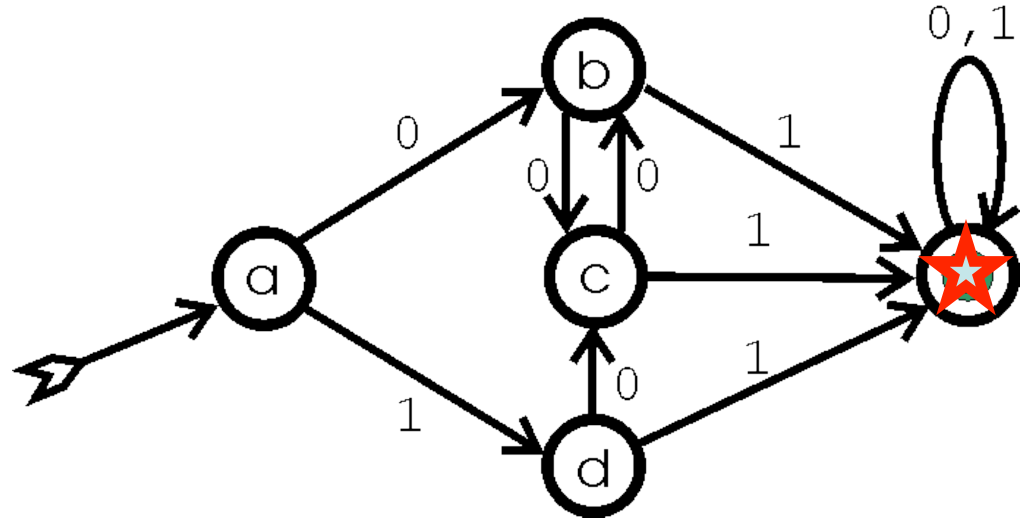
100100101

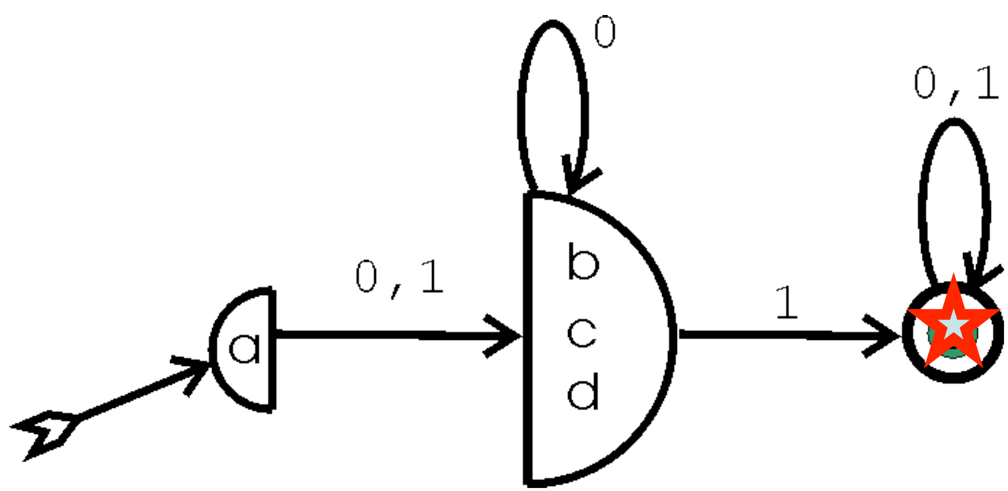


Example

Minimization Example. Compare

100100101



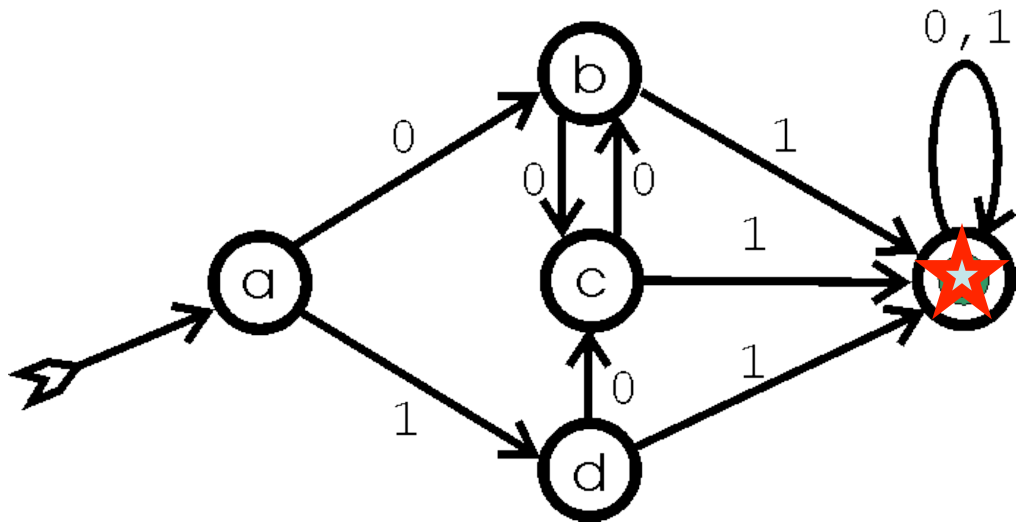


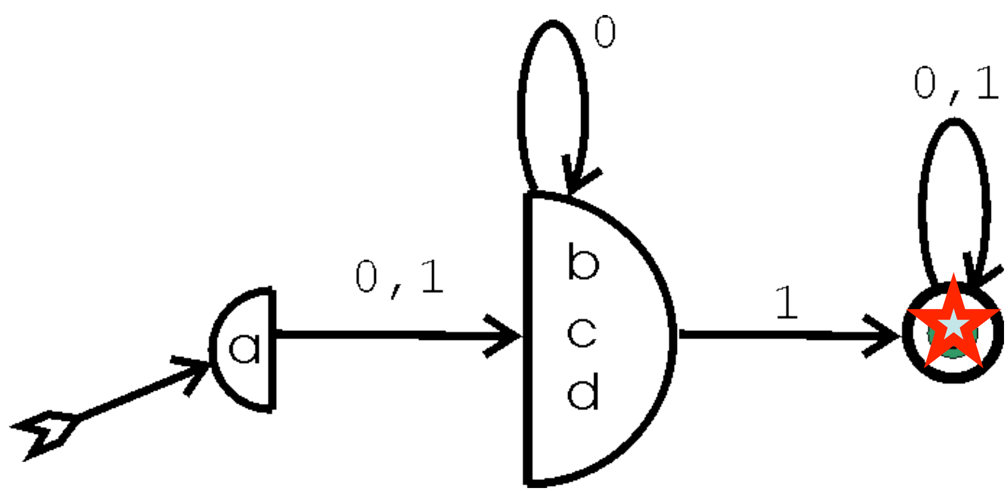
DFA Minimization

Example

Minimization Example. Compare

100100101



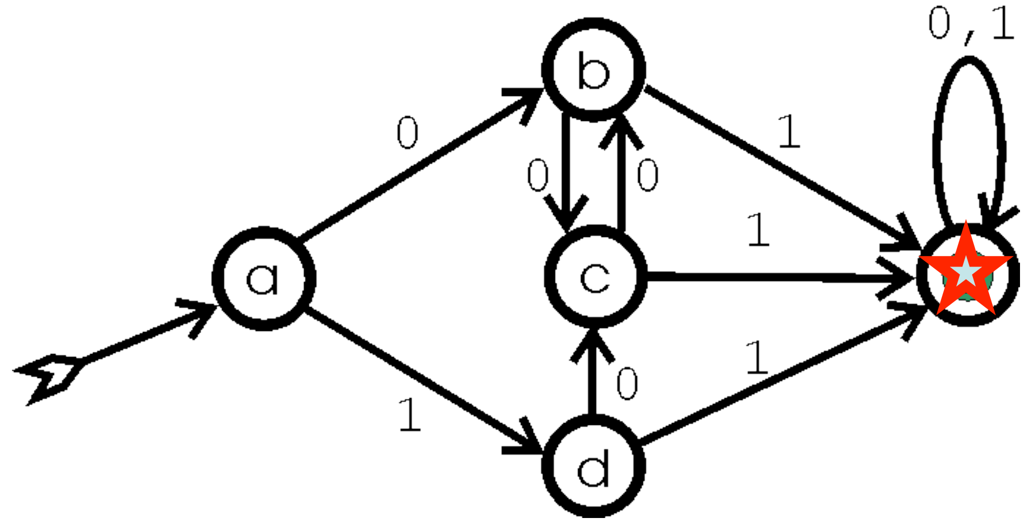


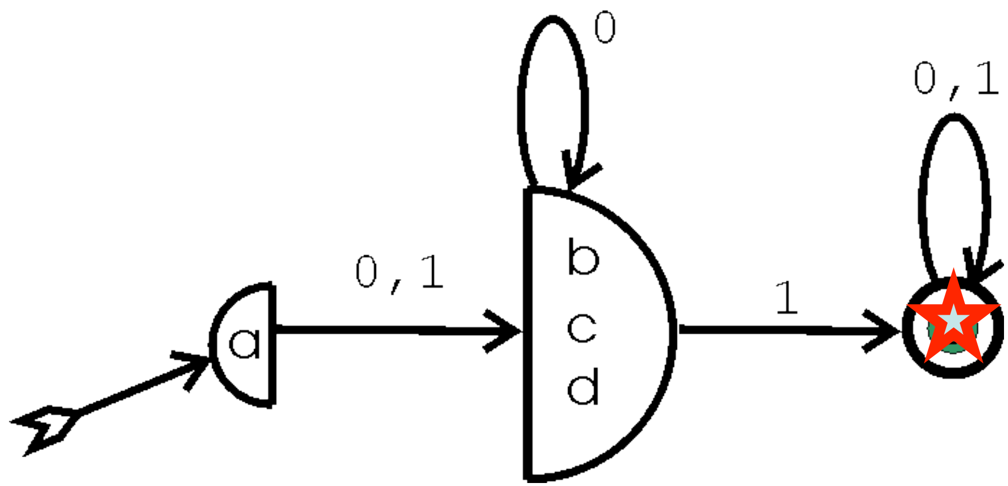
Example

Minimization Example. Compare

100100101

ACCEPTED.



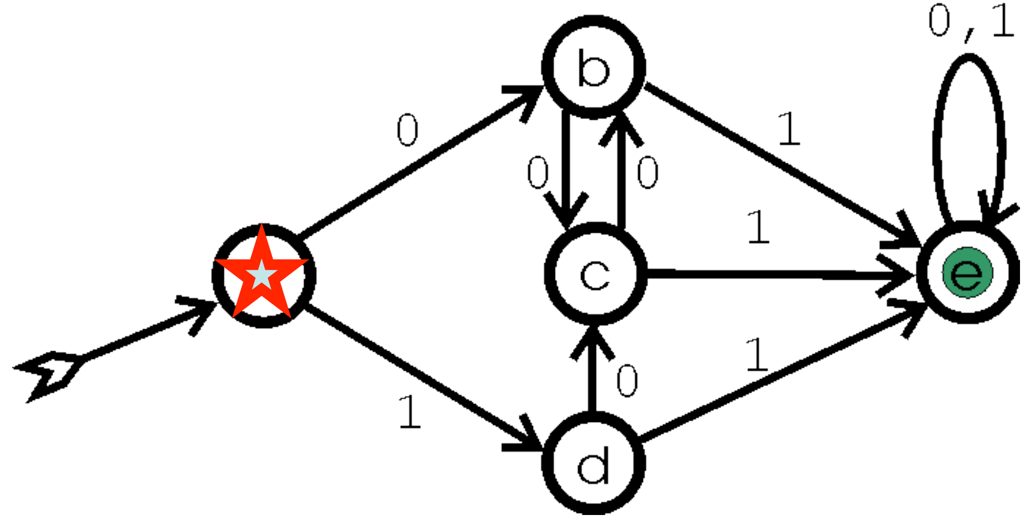


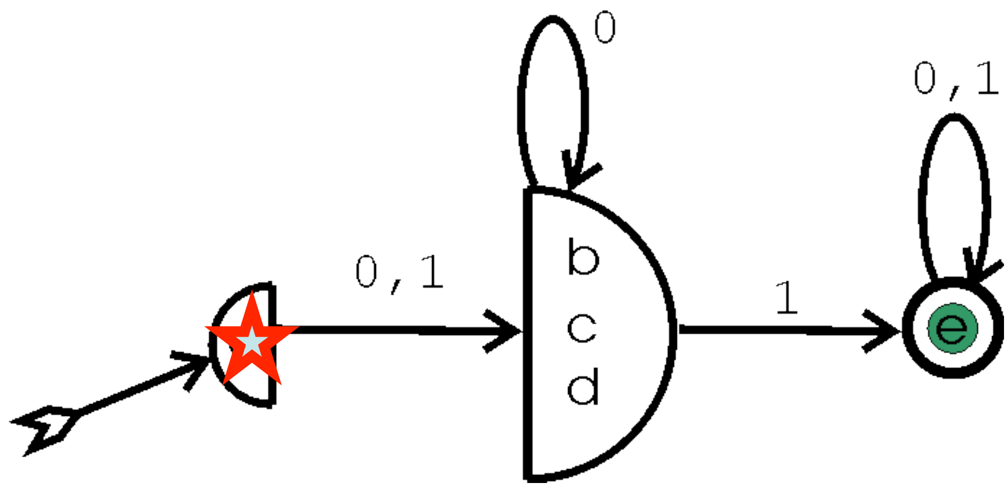
DFA Minimization

Example

Minimization Example. Compare

10000

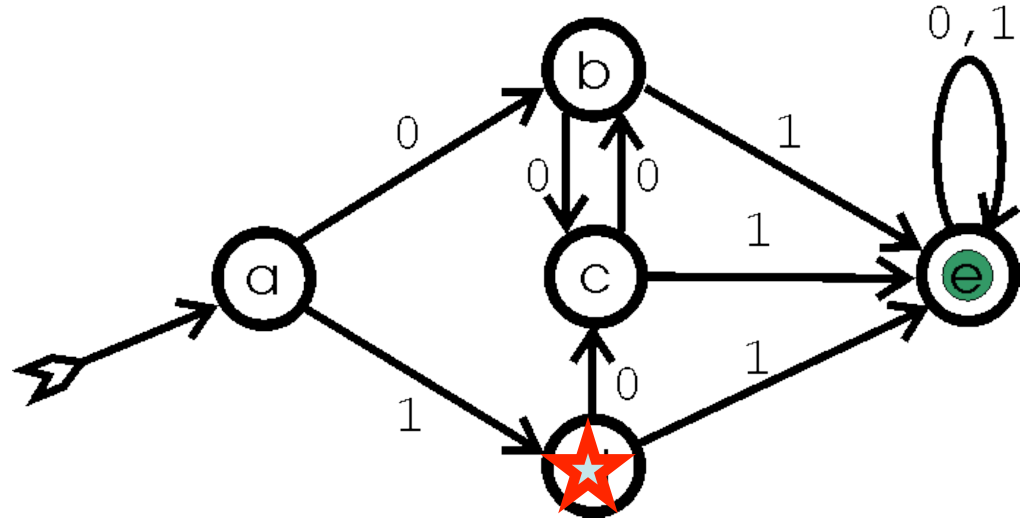


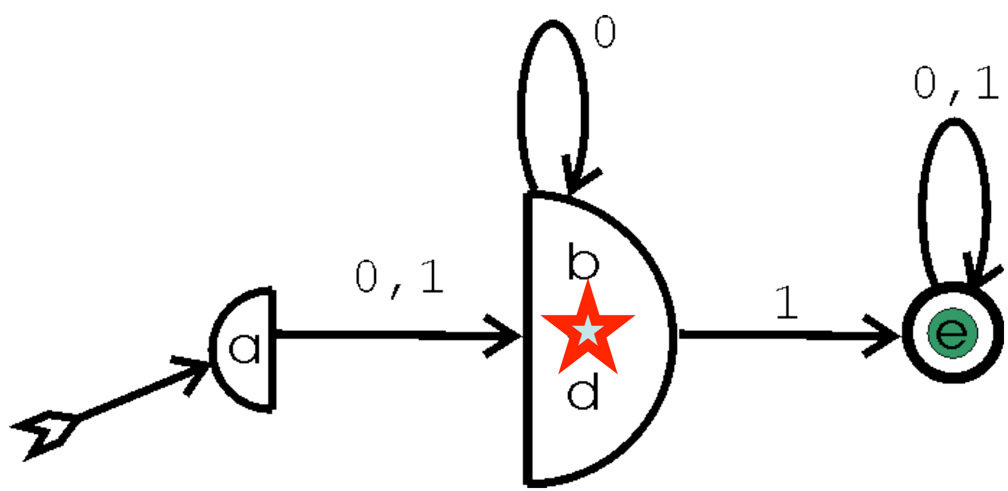


Example

Minimization Example. Compare

10000

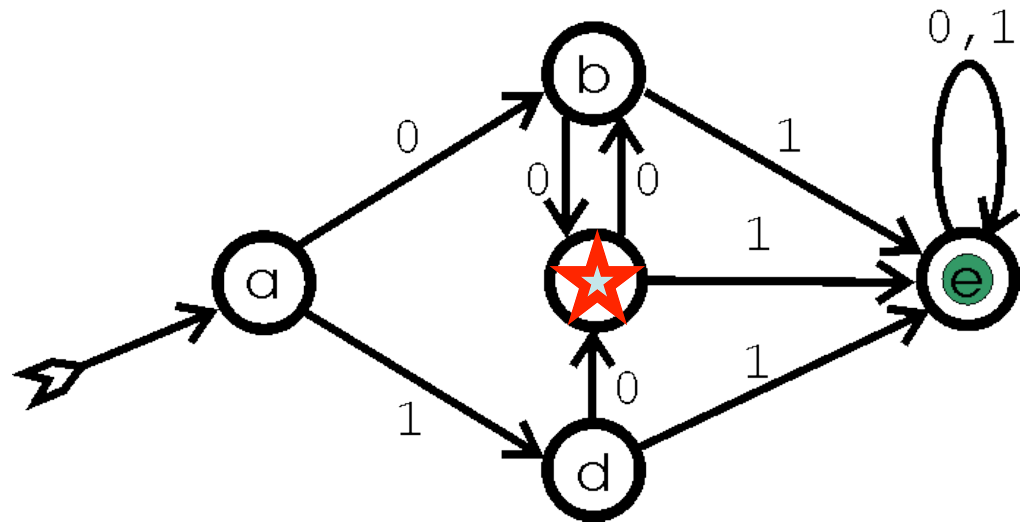


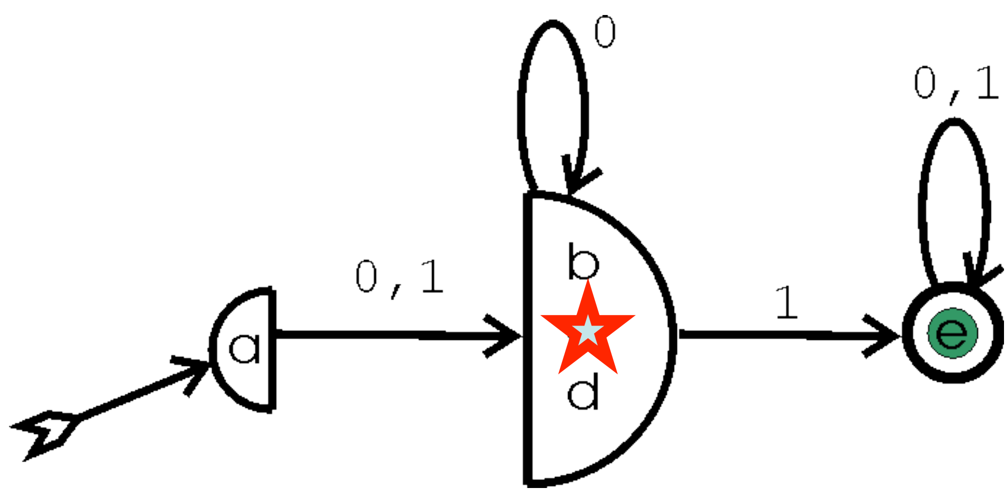


Example

Minimization Example. Compare

10000

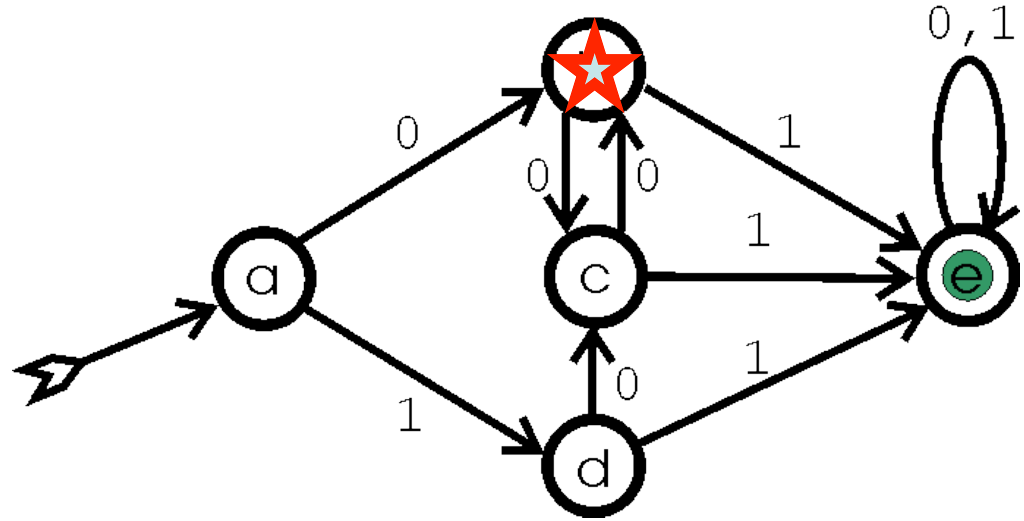


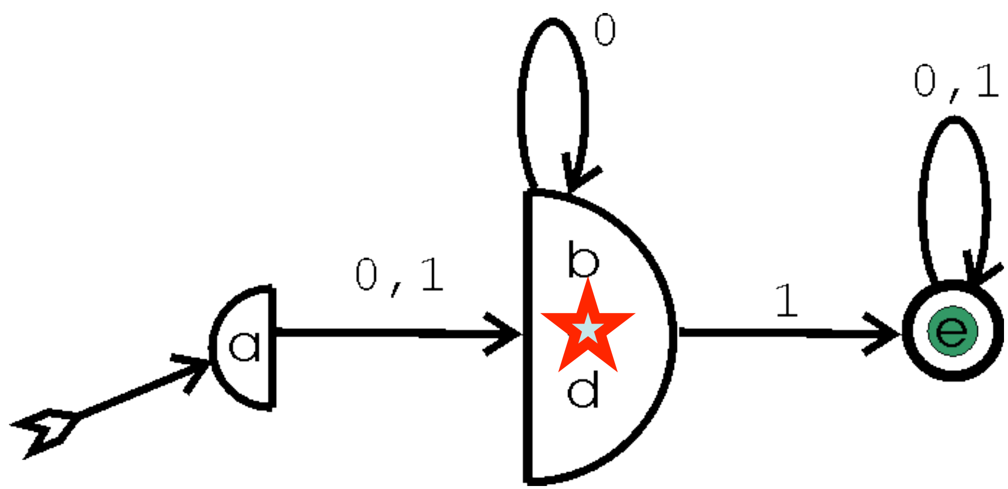


Example

Minimization Example. Compare

10000

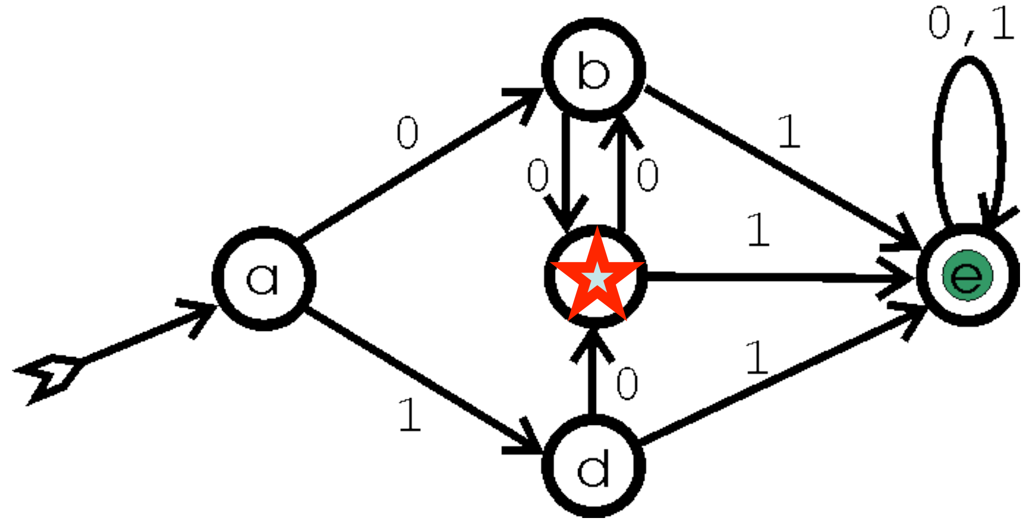


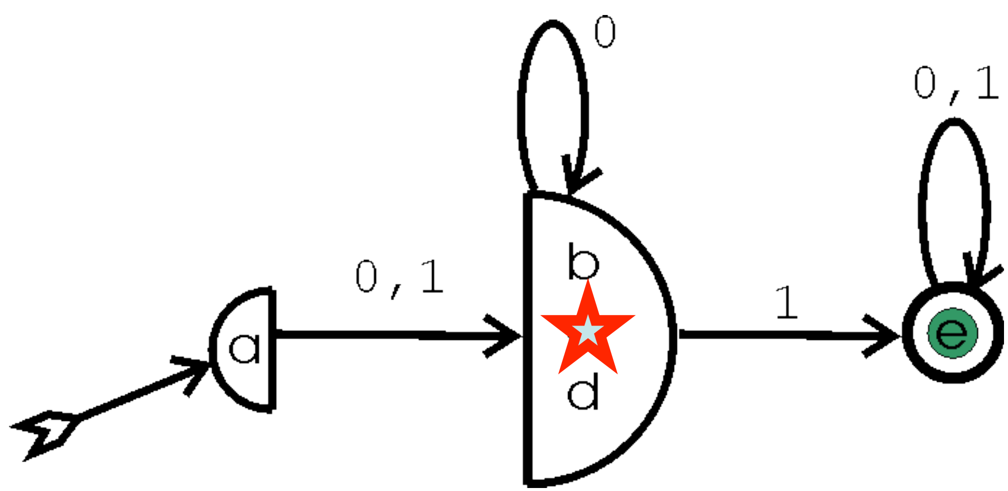


Example

Minimization Example. Compare

10000



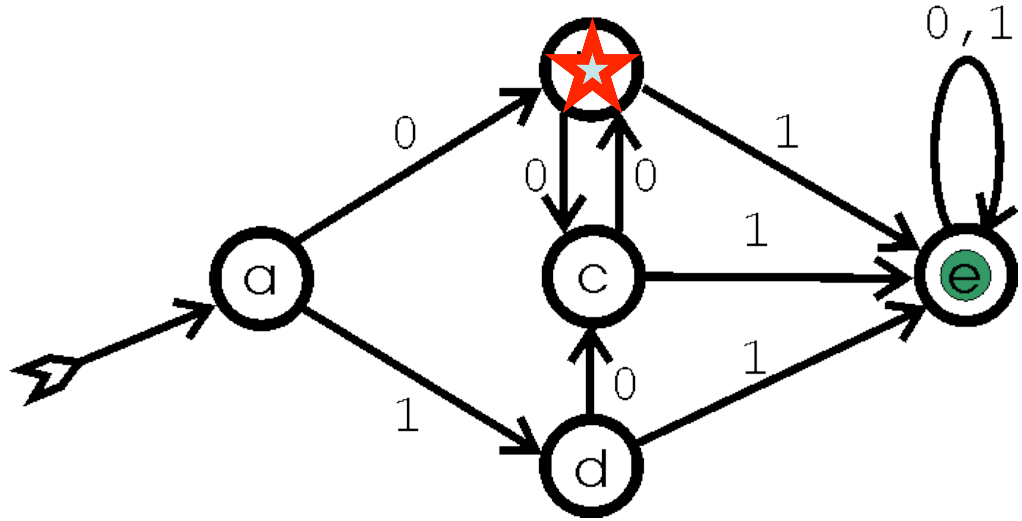


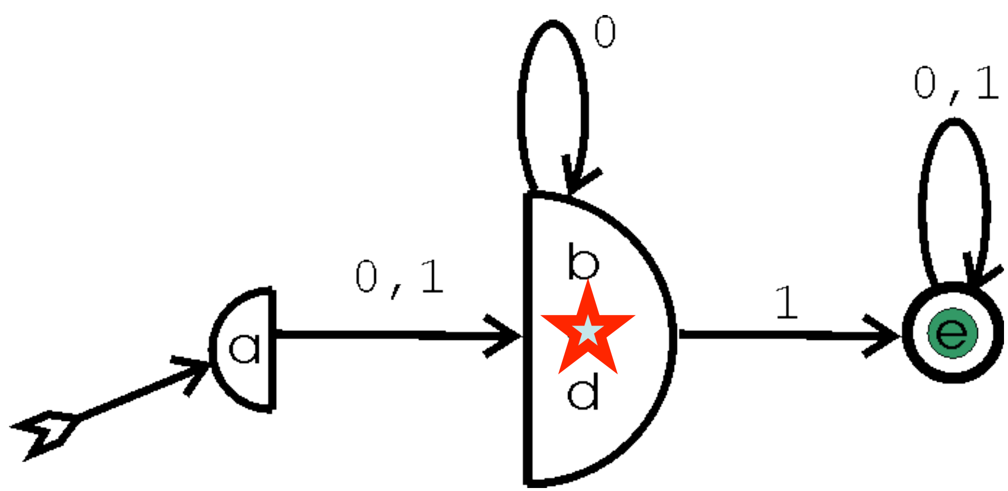
Example

Minimization Example. Compare

10000

REJECT.





Minimal Automaton

Previous algorithm guaranteed to produce an irreducible FA. Why should that FA be the smallest possible FA for its accepted language?

Minimal Automaton

Theorem (Myhill-Nerode): The minimization algorithm produces the smallest possible automaton for its accepted language.

Proof Show that any irreducible automaton is the smallest for its accepted language *L*:

We say that two strings $u,v \in \Sigma^*$ are ***indistinguishable*** if for all suffixes *x*, *ux* is in *L* exactly when *vx* is.

Notice that if *u* and *v* *are* distinguishable, the path from their paths from the start state must have different endpoints.

DFA Minimization

Proof (cont.)

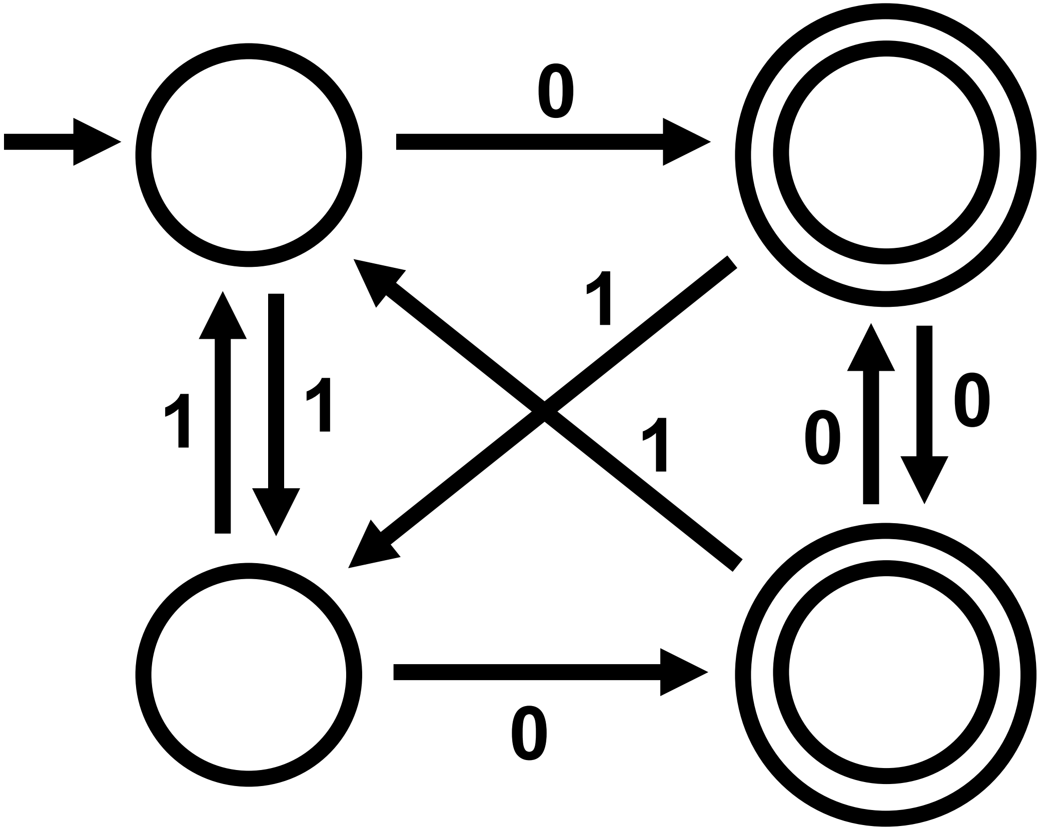
Consequently, the number of states in any DFA for *L* must be as great as the number of mutually distinguishable strings for *L*.

But an irreducible DFA has the property that every state gives rise to another mutually distinguishable string!

Therefore, any other DFA must have at least as many states as the irreducible DFA

Quiz 1

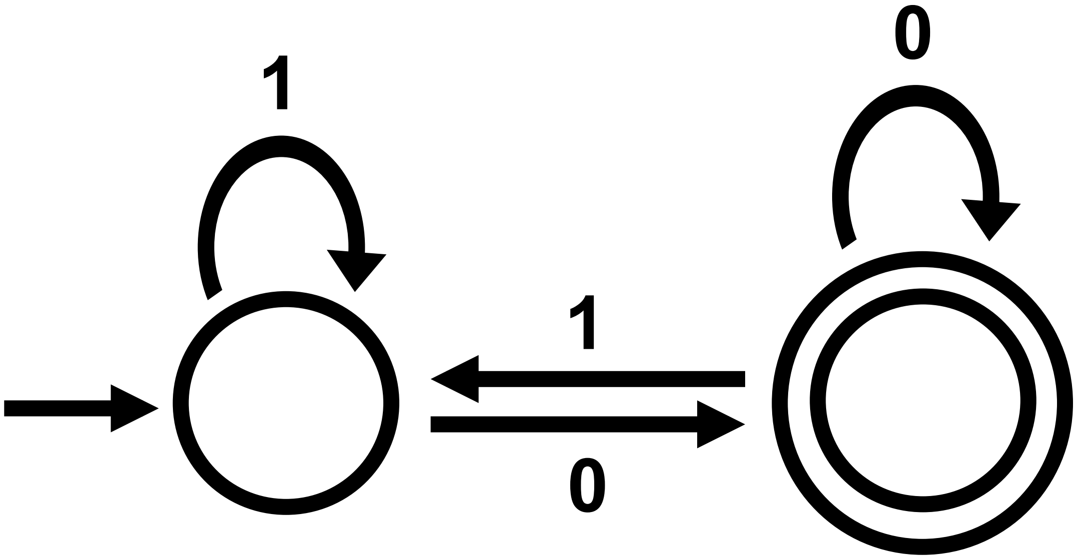
IS THIS DFA **MINIMAL?**



NO

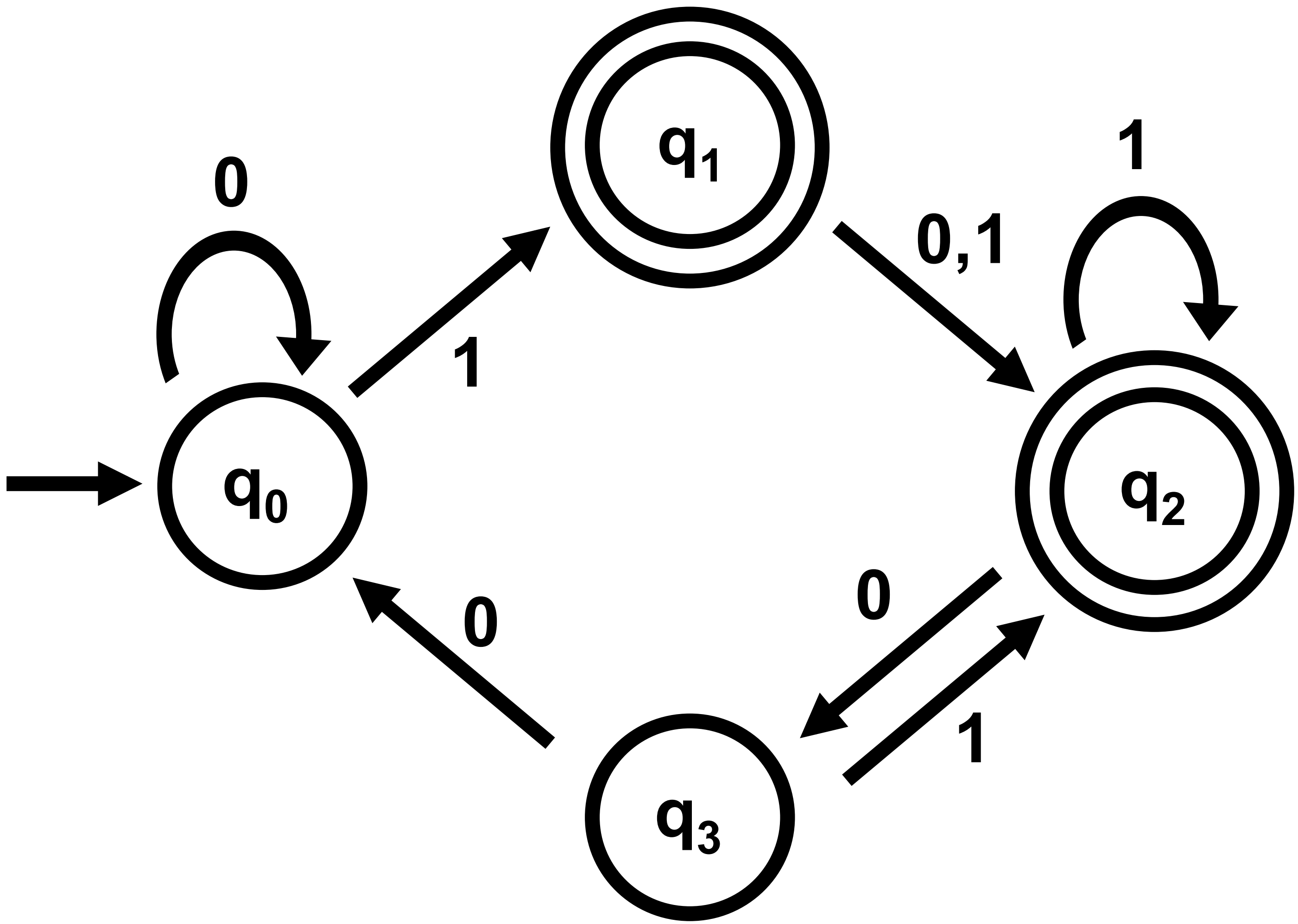
Quiz 1

ITS MINIMAL DFA IS



Quiz 2

IS THIS DFA **MINIMAL?**



DFA Minimization

Definition

For a DFA M = $(Q, \Sigma, \delta, q_0, F)$, let $p, q, r \in Q$

Definition:

$p \sim q$ iff p is indistinguishable (equivalent) from q

$p \not\sim q$ iff p is distinguishable from q

Proposition: ~ is an equivalence relation

$p \sim p$ (reflexive)

$p \sim q \Rightarrow q \sim p$ (symmetric)

$p \sim q$ and $q \sim r \Rightarrow p \sim r$ (transitive)

Definition

For a DFA $M = (Q, \Sigma, \delta, q_0, F)$, let $p, q, r \in Q$

Definition:

$p \sim q$ iff p is indistinguishable from q

$p \nsim q$ iff p is distinguishable from q

Proposition: ~ is an equivalence relation

$p \sim p$ (reflexive)

$p \sim q \Rightarrow q \sim p$ (symmetric)

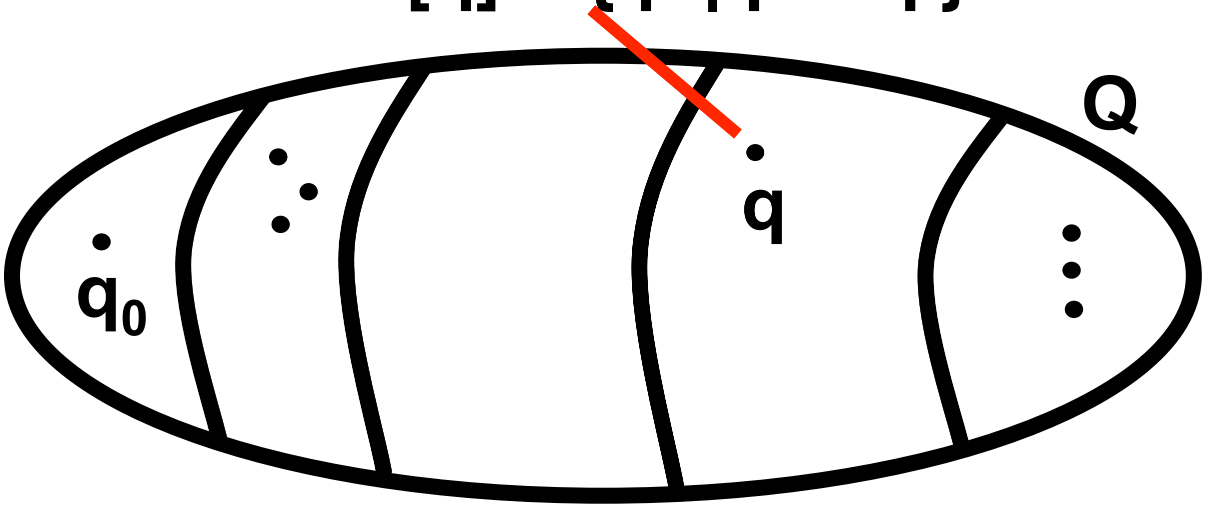
$p \sim q$ and $q \sim r \Rightarrow p \sim r$ (transitive)

Definition

Because ~ is an equivalence relation

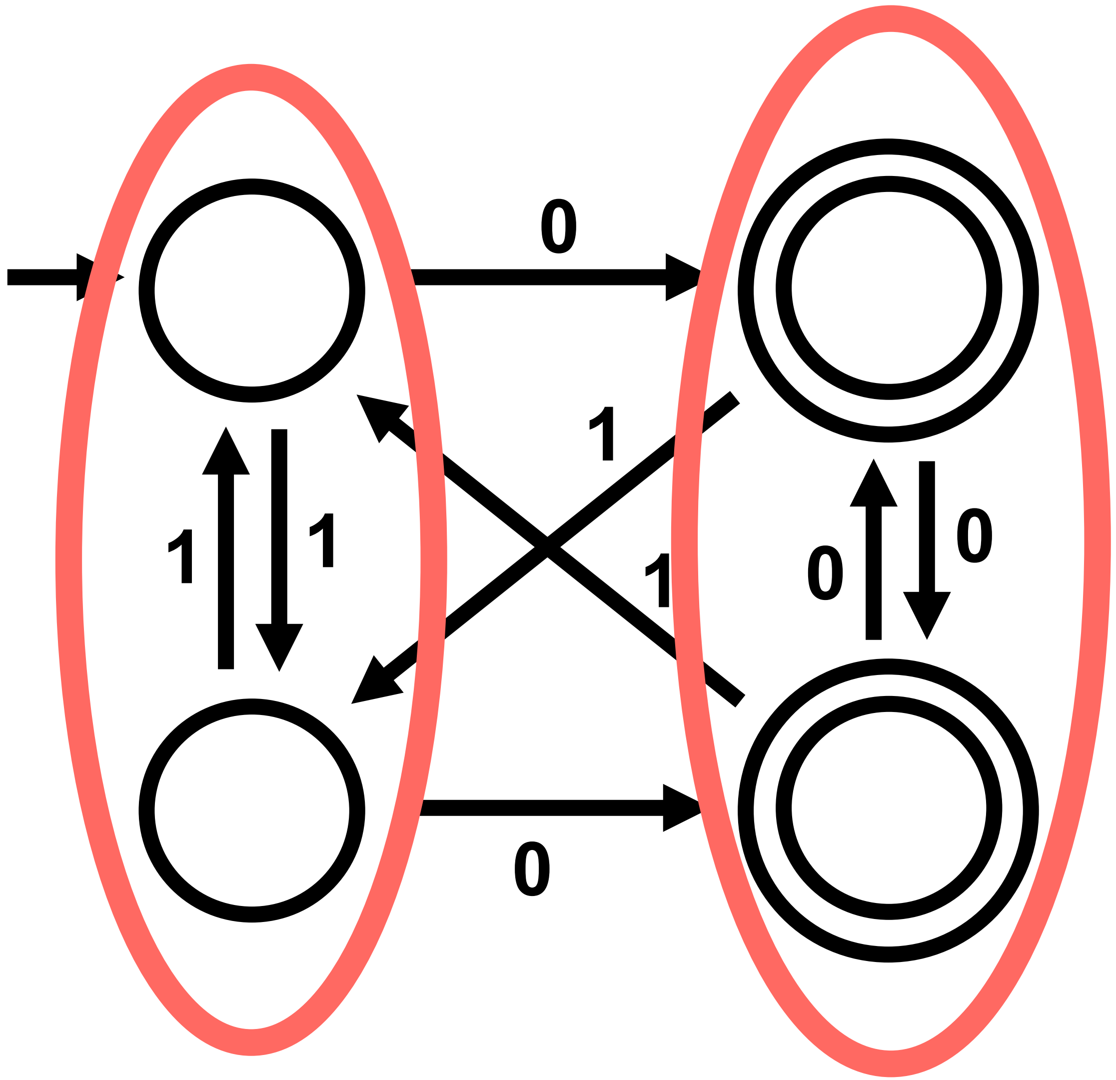
so ~ partitions the set of states of M into disjoint equivalence classes

$$[q] = \{ p \mid p \sim q \}$$



DFA Minimization

Example



Algorithm MINIMIZE

Input: DFA M

Output: DFA M_{MIN} such that:

$M \equiv M_{MIN}$

M_{MIN} has no inaccessible states

M_{MIN} is irreducible

||

states of M_{MIN} are pairwise distinguishable

Theorem: M_{MIN} is the unique minimum

Idea: States of M_{MIN} will be blocks of equivalent states of M

Algorithm

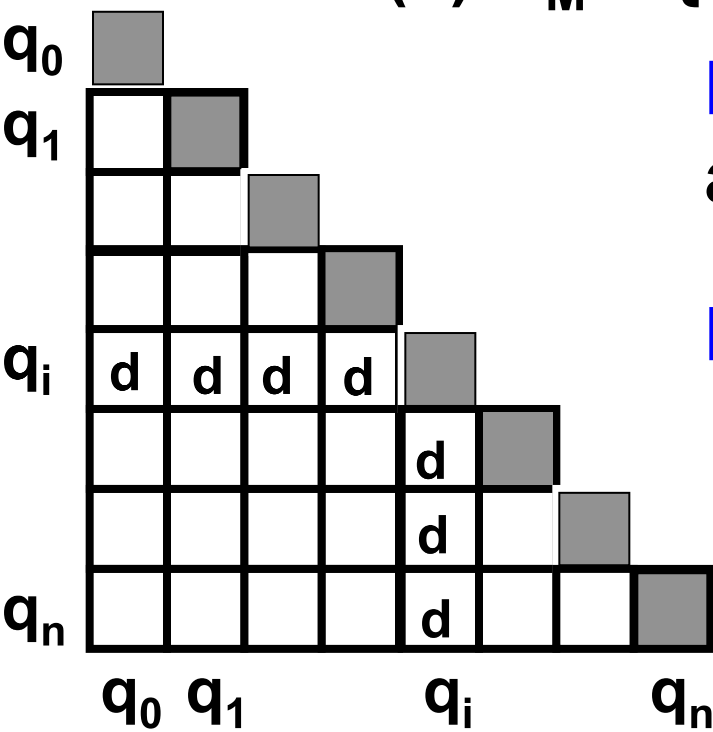
DFA Minimization

TABLE-FILLING ALGORITHM

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

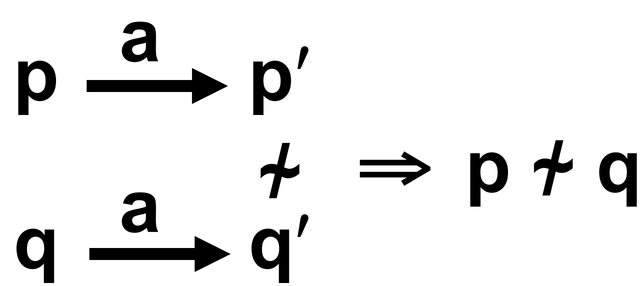
Output: (1) $D_M = \{ (p,q) \mid p,q \in Q \text{ and } p \not\sim q \}$

(2) $E_M = \{ [q] \mid q \in Q \}$



Base Case: p accepts and q rejects $\Rightarrow p \not\sim q$

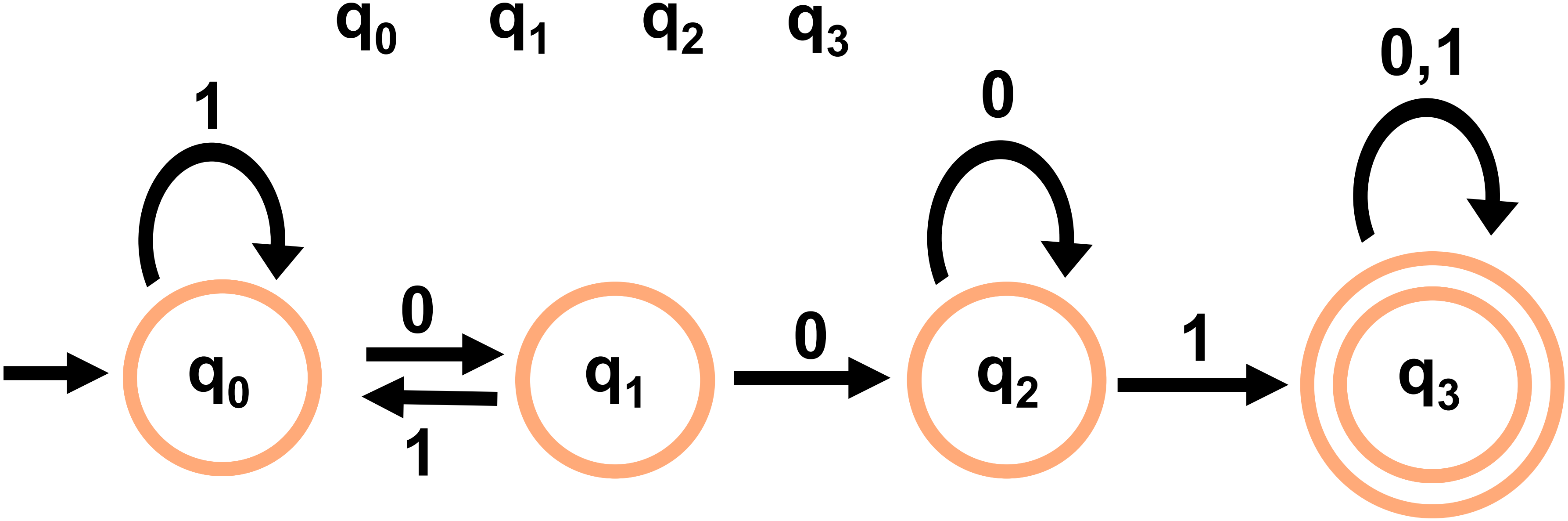
Recursion:



DFA Minimization

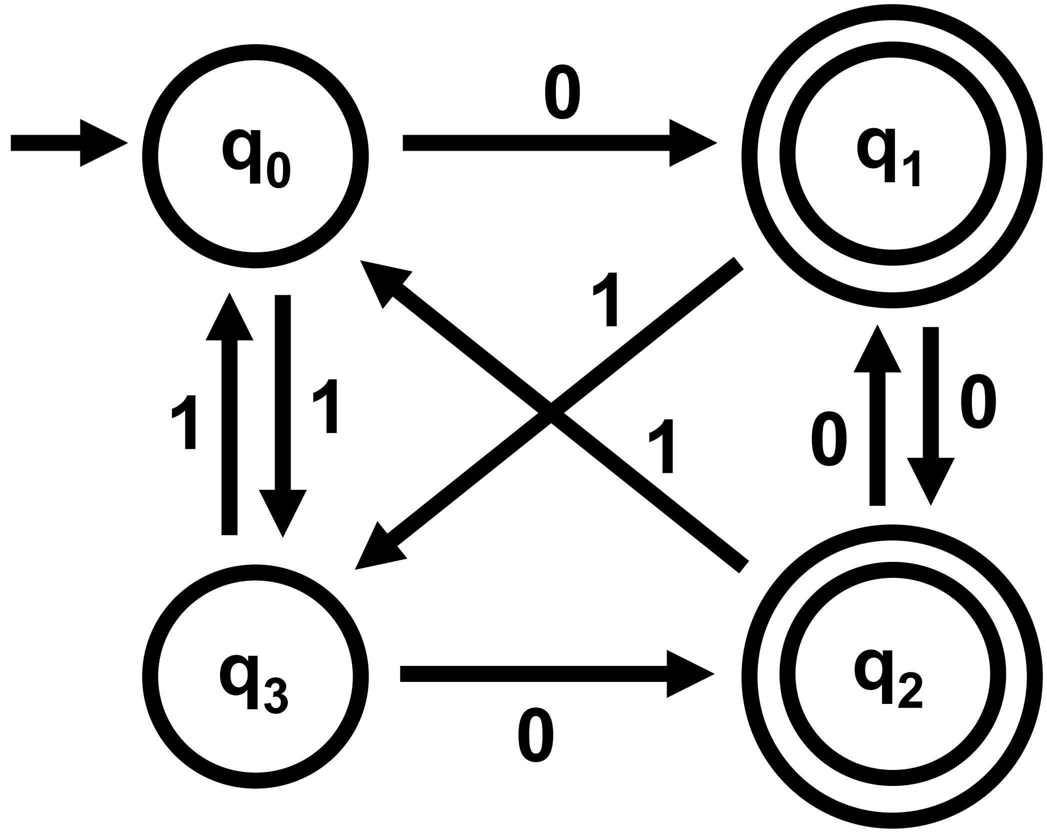
Example

	q_0	q_1	q_2	q_3
q_0				
q_1	d			
q_2	d	d		
q_3	d	d	d	



DFA Minimization

Example



	q_0	q_1	q_2	q_3
q_0				
q_1	d			
q_2	d			
q_3		d	d	

Algorithm

DFA Minimization

Algorithm MINIMIZE

Input: DFA M

Output: DFA M_{MIN}

(1) Remove all inaccessible states from M

(2) Apply Table-Filling algorithm to get $E_M = \{ [q] \mid q \text{ is an accessible state of } M \}$

$$M_{MIN} = (Q_{MIN}, \Sigma, \delta_{MIN}, q_{0\,MIN}, F_{MIN})$$

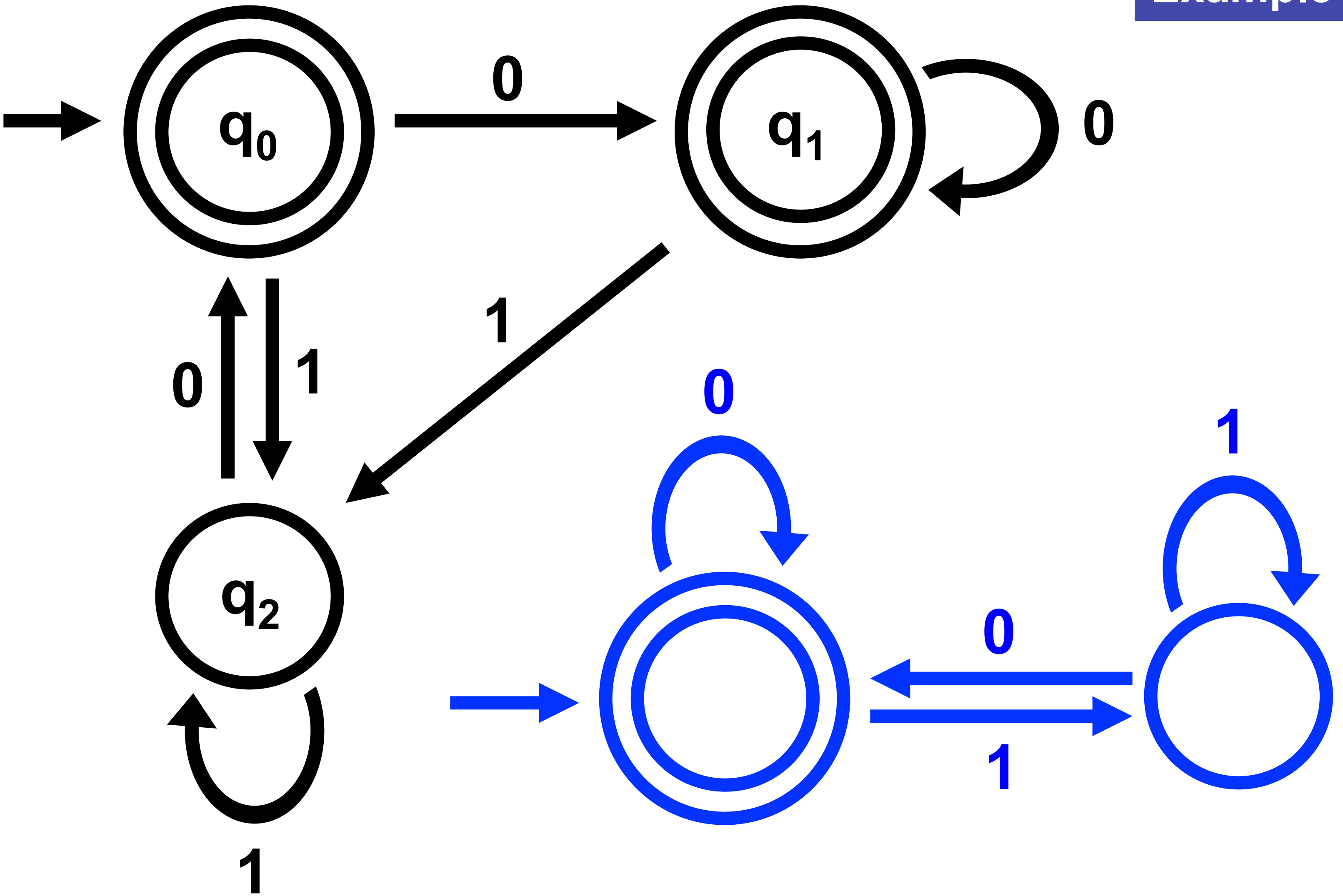
$$Q_{MIN} = E_M, \quad q_{0\,MIN} = [q_0], \quad F_{MIN} = \{ [q] \mid q \in F \}$$

$$\delta_{MIN}([q],\sigma) = [\delta(q,\sigma)]$$

MINIMIZE

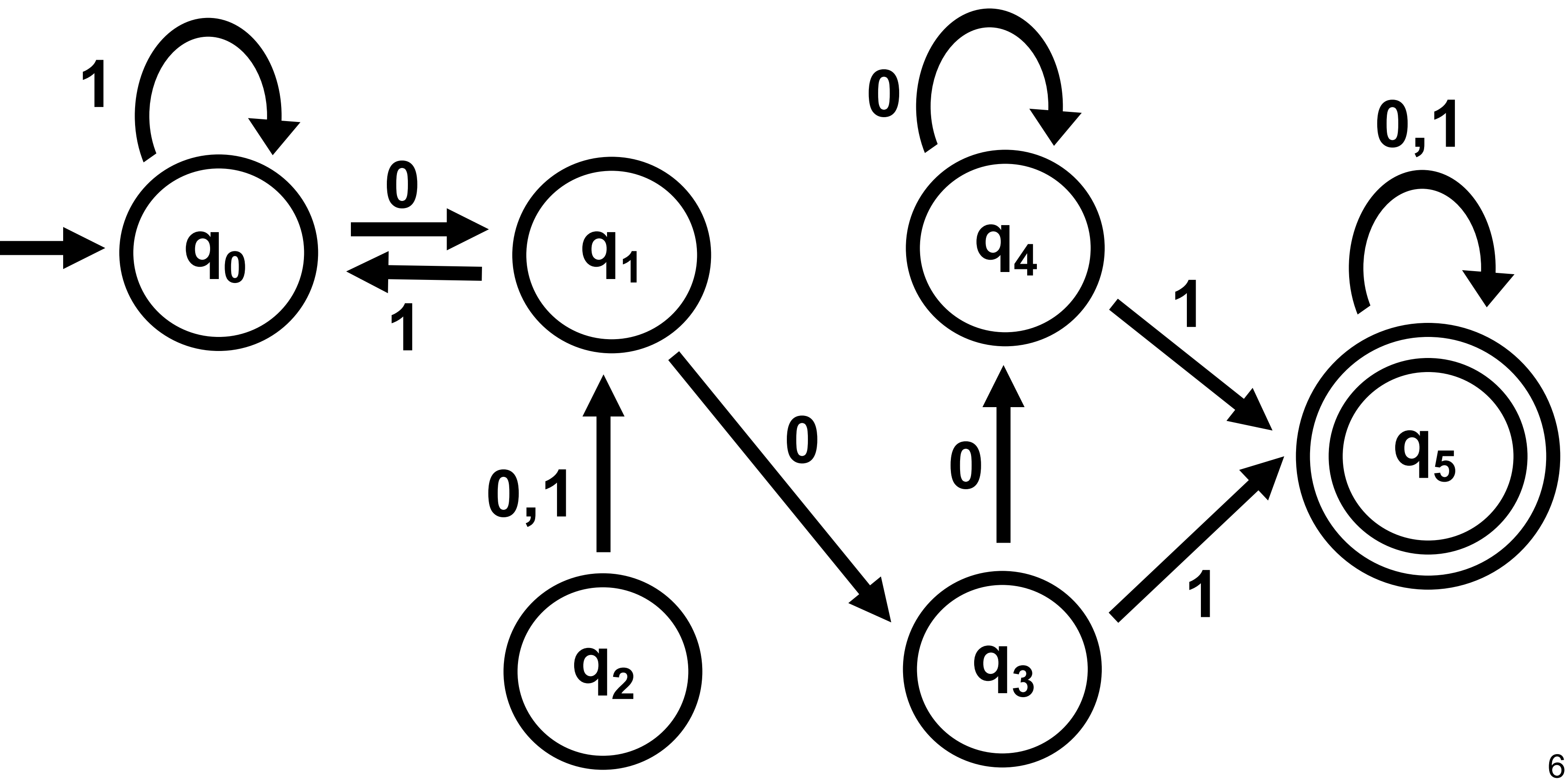
DFA Minimization

Example

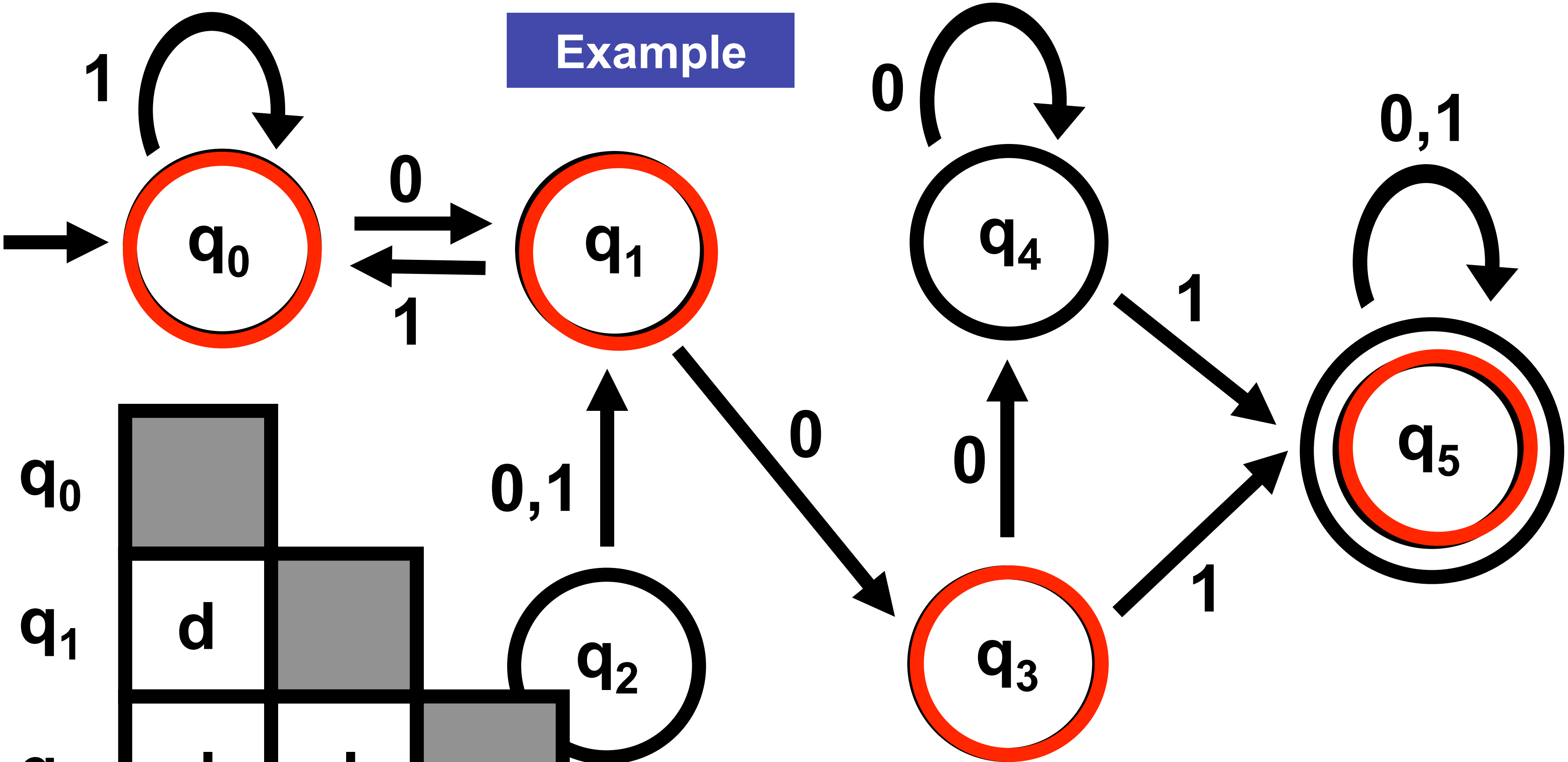


Exercise

Minimize the following DFA?



Example



	q_0	q_1	q_3	q_4	q_5
q_0					
q_1	d				
q_3	d	d			
q_4	d	d			
q_5	d	d	d	d	