Damascus University

Non-regular languages & DFA minimization

Dr. Mohammad Ahmad

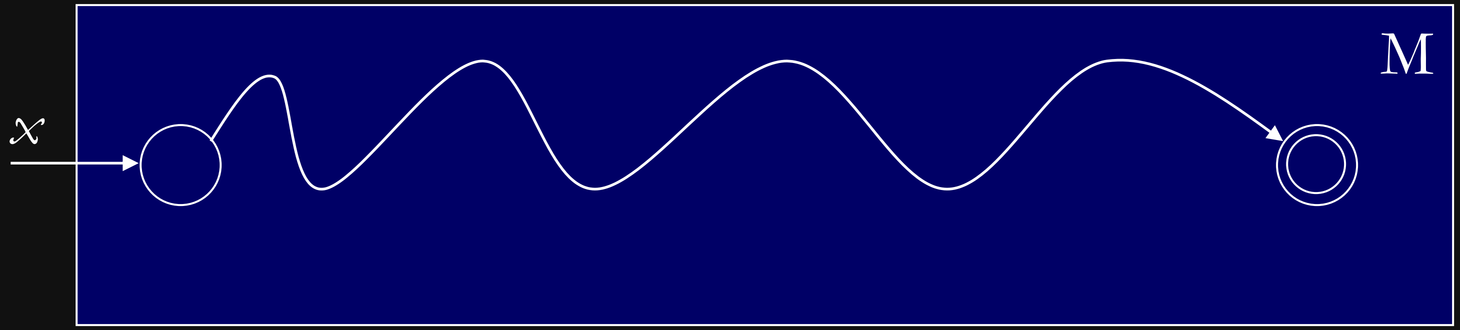
A non-regular language

- An example

$L = \{0^n1^n : n \geq 0\}$ is not regular.

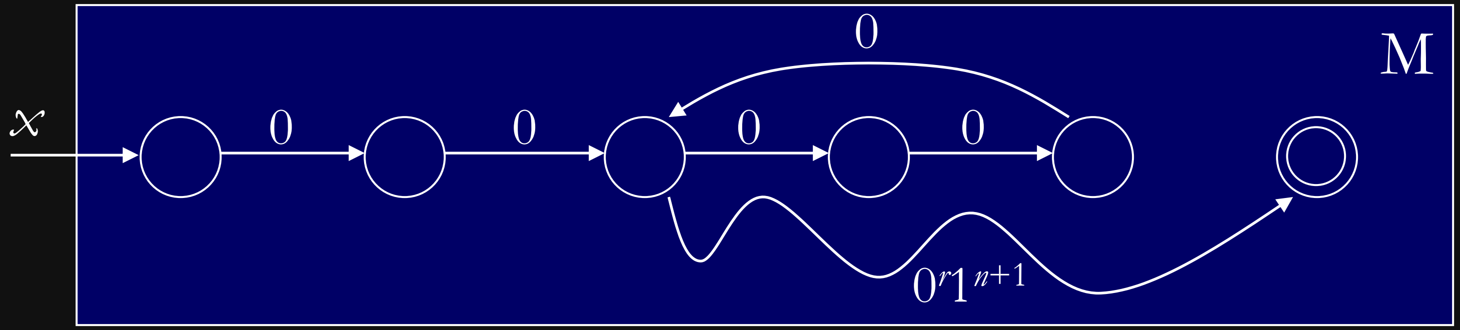
- We reason by contradiction:
  - Suppose we have managed to construct a DFA M for L
  - We argue something must be wrong with this DFA
  - In particular, M must accept some strings outside L

A non-regular language


imaginary DFA for L with n states

- What happens when we run M on input $x = 0^{n+1}1^{n+1}$?
  - M better accept, because $x \in L$

A non-regular language



- What happens when we run M on input $x = 0^{n+1}1^{n+1}$?
  - M better accept, because $x \in L$
  - But since M has n states, it must revisit at least one of its states while reading 0^{n+1}

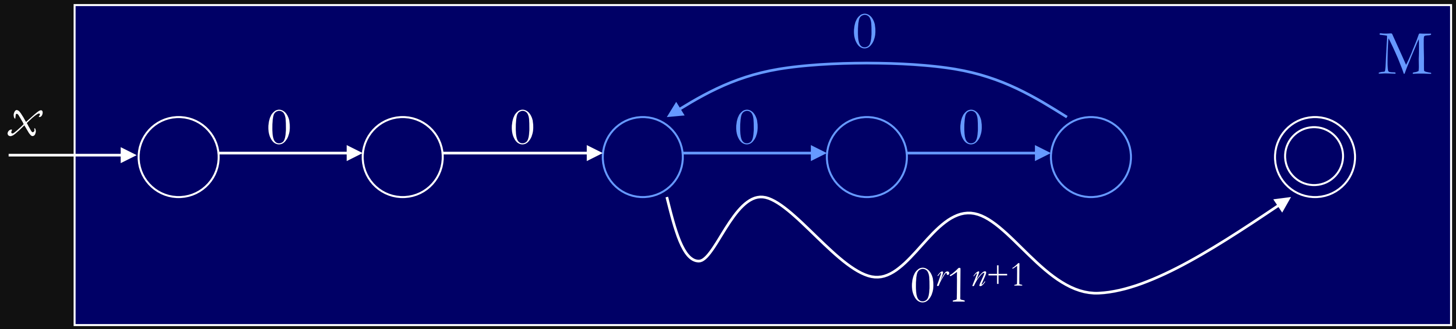
Pigeonhole principle

Suppose you are tossing $n + 1$ balls into n bins. Then two balls end up in the same bin.

- Here, balls are 0s, bins are states:

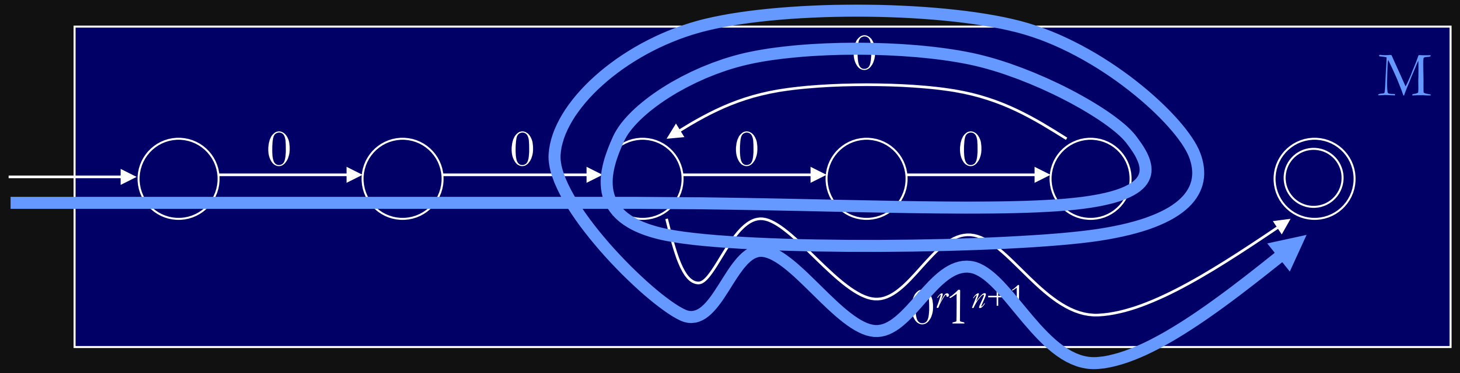
If you have a DFA with n states and it reads $n + 1$ consecutive 0s, then it must end up in the same state twice.

A non-regular language



- What happens when we run M on input $x = 0^{n+1}1^{n+1}$?
  - M better accept, because $x \in L_2$
  - But since M has n states, it must revisit at least one of its states while reading 0^{n+1}
  - But then the DFA must contain a loop with 0s

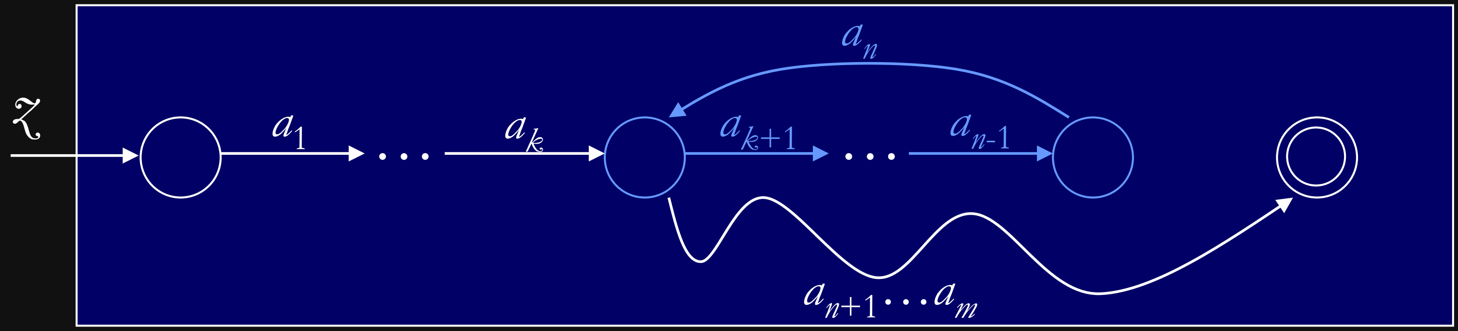
A non-regular language



- The DFA will then also accept strings that go around the loop multiple times
- But such strings have more 0s than 1s, so they are not in L_2!

General method for showing non-regularity

- Every regular language L has a property:



- For every sufficiently long input z in L, there is a "middle part" in z that, even if repeated any number of times, keeps the input inside L

Pumping lemma for regular languages

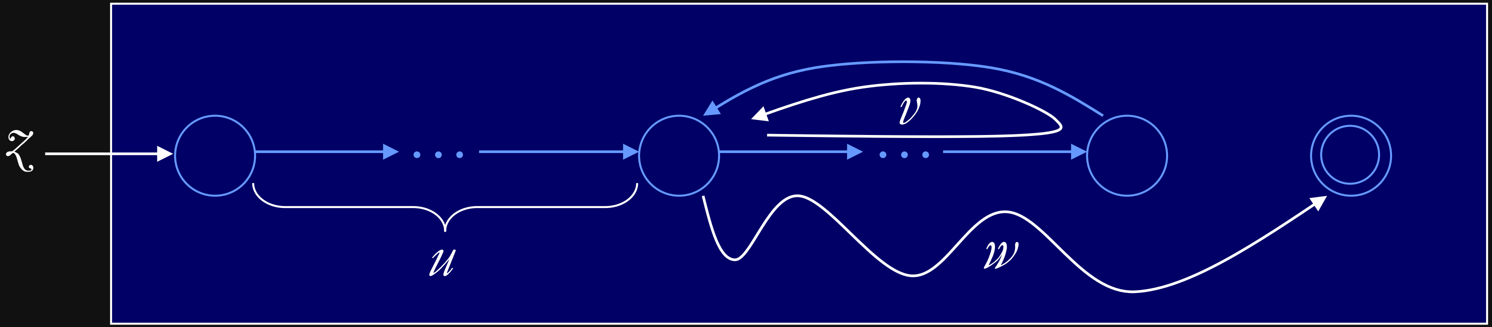
- Pumping lemma: For every regular language L

There exists a number n such that for every string z in L, we can write $z = u\, v\, w$ where

① $|uv| \leq n$

② $|v| \geq 1$

③ For every $i \geq 0$, the string $u\, v^i\, w$ is in L.



Arguing non-regularity

- If L is regular, then:

> There exists n such that for every z in L, we can write $z = u\ v\ w$ where ① $|uv| \leq n$, ② $|v| \geq 1$ and ③ For every $i \geq 0$, the string $u\ v^i\ w$ is in L.

- So to prove L is not regular, it is enough to show:

Proving non regularity

For every n there exists z in L, such that for every way of writing $z = u\;v\;w$ where ① $|uv| \leq n$ and ② $|v| \geq 1$, the string $u\;v^i\;w$ is not in L for some $i \geq 0$.

- This is a game between you and an imagined adversary

	adversary	you
1	choose n	choose $z \in L$
2	write $z = uvw$ $(\lvert uv \rvert \leq n, \lvert v \rvert \geq 1)$	choose i
		you win if $uv^iw \notin L$

Arguing non-regularity

- You need to give a strategy that, regardless of what the adversary does, always wins you the game

	adversary	you
1	choose n	choose $z \in L$
2	write $z = uvw$ $(\lvert uv \rvert \leq n, \lvert v \rvert \geq 1)$	choose i
		you win if $uv^iw \notin L$

Example

	adversary	you
1	choose n	choose $z \in L$
2	write $z = uvw$ $(\lvert uv \rvert \leq n, \lvert v \rvert \geq 1)$	choose i
		you win if $uv^iw \notin L$

$$\boxed{L = \{0^n1^n \colon n \geq 0\}}$$

	adversary	you
1	choose n	$z = 0^n1^n$
2	write $z = uvw$	$i = 2$
	$\underbrace{000000000}_{u}\underbrace{0000}_{v}\underbrace{00111111111111111}_{w}$	$uv^2w \ = 0^{n+k}1^n \notin L$
	$\underbrace{000000000}_{u}\underbrace{0000}_{v}\underbrace{0000}_{v}\underbrace{00111111111111111}_{w}$	

Example

$$L^{DUP} = \{0^n10^n1 \colon n \geq 0\}$$

	adversary	you
1	choose n	$z = 0^n10^n1$
2	write $z = uvw$	$i = 2$
		$uv^2w = 0^{n+k}10^n1 \notin L$

$$\underbrace{000000000}_{u}\underbrace{0000}_{v}\underbrace{0100000000000000001}_{w}$$

$$\underbrace{000000000}_{u}\underbrace{0000}_{v}\underbrace{0000}_{v}\underbrace{0100000000000000001}_{w}$$

Which of these are regular?

$L_1 = \{x \colon x$ has same number of 0s and 1s$\}$ $\Sigma = \{0, 1\}$

$L_2 = \{x \colon x = 0^n 1^m,\ n > m \geq 0\}$

$L_3 = \{x \colon x$ has same number of patterns 01 and 10$\}$

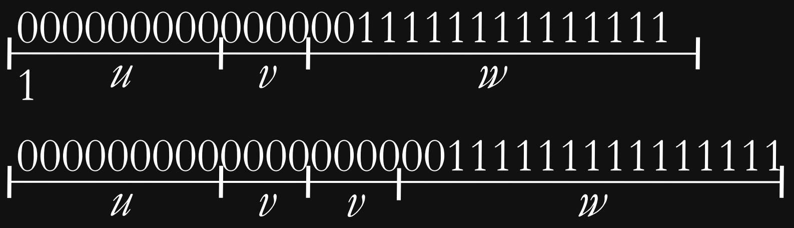
$L_4 = \{x \colon x$ has same number of patterns 01 and 10$\}$

$L_5 = \{x \colon x$ has different number of 0s and 1s$\}$

Example

$L_1 = \{x\text{: } x \text{ has same number of 0s and 1s}\}$

	adversary	you
1	choose n	$z = 0^n1^n$
2	write $z = uvw$	$i = 2$
		$uv^2w = 0^{n+k}1^n \notin L_3$



Example

$$L_2 = \{x\colon x = 0^m 1^n,\ m > n \geq 0\}$$

	adversary	you
1	choose n	$z = 0^{n+1}1^n$
2	write $z = uvw$	$i = 0$
		$uv^0w \quad = 0^{j+l}1^{n+1} \notin L_2$

$$\underbrace{000000000}_{u}\underbrace{0000}_{v}\underbrace{00111111111111111}_{w}$$

$$\underbrace{000000000}_{u}\underbrace{00111111111111111}_{w}$$

Example

$$\boxed{L_3 = \{x\text{: } x \text{ has same number of } 01\text{s and } 11\text{s}\}}$$

	adversary	you
1	choose n	$z = (01)^n(11)^n$
	$n = 1$	$z = 0111 \notin L_4$

has too many 11s

What we have in mind:

$n = 1$	$z = 011$
$n = 2$	$z = 010111$
$n = 3$	$z = 010101111$

$$\boxed{z = (01)^n 1^n}$$

has n 01s and n 11s

Example

$L_3 = \{x\colon x \text{ has same number of } 01\text{s and } 11\text{s}\}$

	adversary	you win!
1	choose n	$z = (01)^n 1^n$
2	write $z = uvw$	$i = 0$

$$\underbrace{01010}_{u}\underbrace{101}_{v}\underbrace{0101010111111111}_{w}$$

or $$\underbrace{010101}_{u}\underbrace{0}_{v}\underbrace{10101010111111111}_{w}$$

or $$\underbrace{01010}_{u}\underbrace{1}_{v}\underbrace{010101010111111111}_{w}$$

Taking out v will kill
at least one 01,
but it does not kill any 11s

so $uv^0w \notin L_3$

Example

$L_4 = \{x\text{: } x \text{ has same number of 01s and 01s}\}$

	adversary	you
1	choose n	$z = (01)^n(10)^n$

$n = 1$ $z = 0110$

$n = 2$ $z = 01011010$

→ → →

→ → →

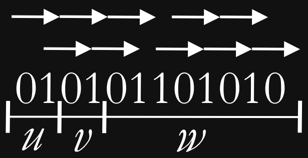
$n = 3$ $z = 010101101010$

Example

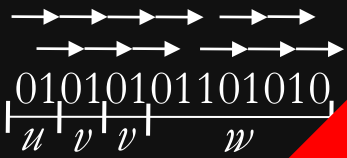
$L_4 = \{x\colon x$ has same number of 01s and 10s$\}$

is regular!

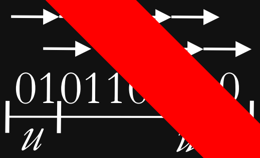
	adversary	you
1	choose n	$z = (01)^n(10)^n$
2	write $z = uvw$	$i = 2$



5 01 patterns
5 10 patterns

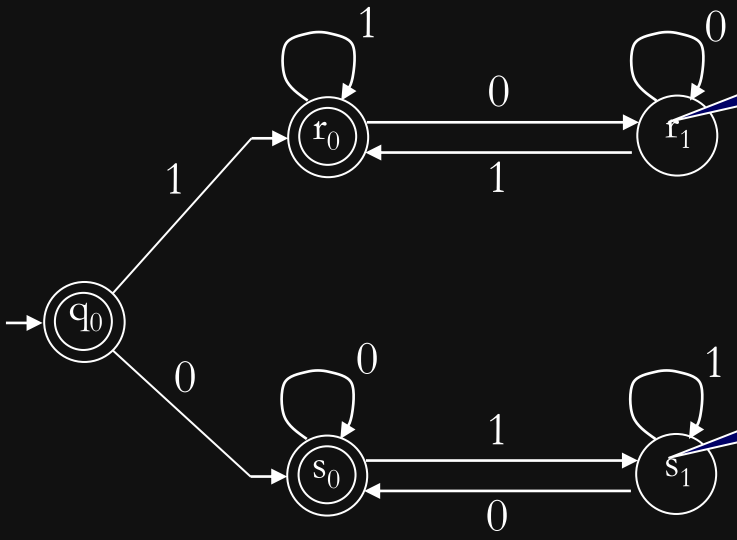


6 01 patterns
6 10 patterns



4 01 patterns
4 10 patterns

Example



one more 10

one more 01

$L_4 = \{x\text{: } x \text{ has same number of 01s and 10s}\}$

Example

$L_4 = \{x\colon x \text{ has different number of 0s than 1s}\}$

	adversary	you
1	choose n	$z = ?$

there is an easier way!

$$L_1 = \{x\colon x \text{ has same number of 0s and 1s}\} = \overline{L_4}$$

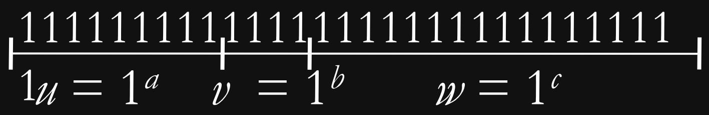
If L_4 is regular, then $L_1 = \overline{L_4}$ is also regular

But L_1 is not regular, so L_4 cannot be regular

Extra example

$L_5 = \{1^p\text{: } p \text{ is prime}\}$

	adversary	you
1	choose n	$z = 1^p\text{: } p > n$ is prime
2	write $z = uvw = 1^a 1^b 1^c$	$i = a + c$

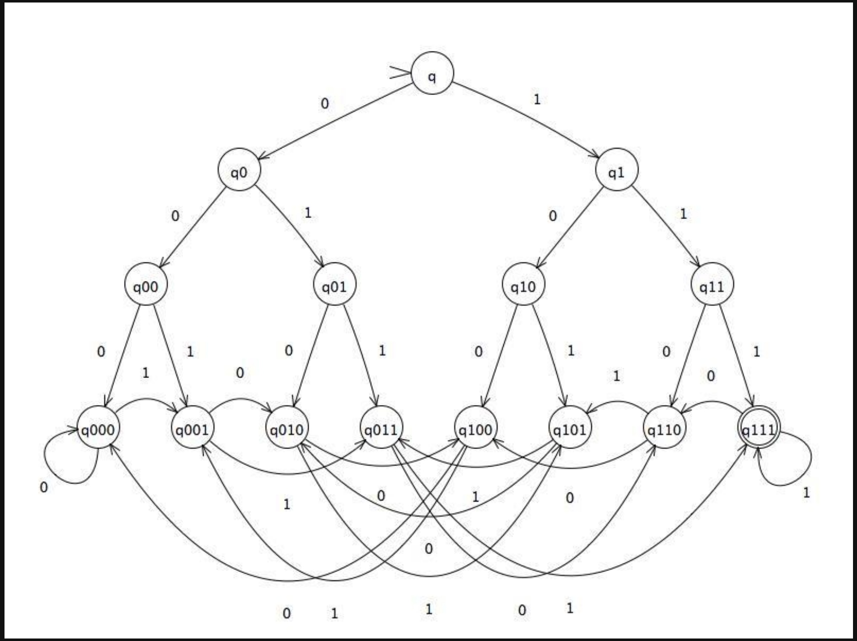


$$\begin{aligned} uv^iw &= 1^a 1^{ib} 1^c \\ &= 1^{(a+c)+ib} \\ &= 1^{(a+c)+(a+c)b} \\ &= 1^{(a+c)(b+1)} \\ &= 1^{\text{composite}} \notin L_5 \end{aligned}$$

DFA minimization

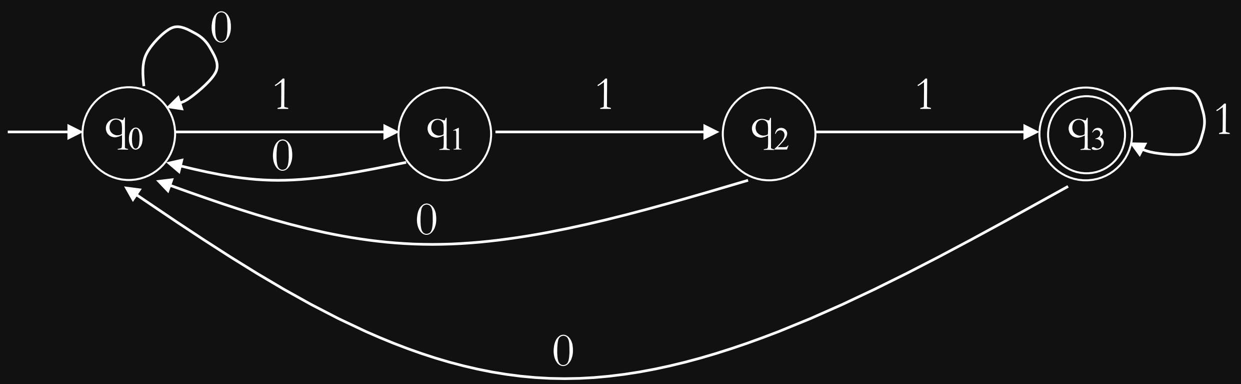
Example

- Construct a DFA over alphabet $\{0, 1\}$ that accepts those strings that end in 111



Isn’t there a smaller one?

Smaller DFA

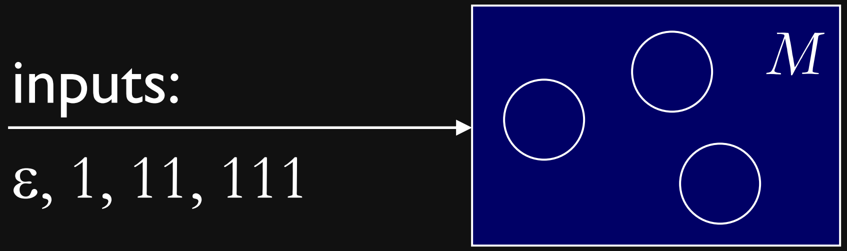


Can we do it with 3 states?

Even smaller DFA?

- Suppose we had a 3 state DFA M for L

  … let’s imagine what happens when:



inputs:

ε, 1, 11, 111

- By the pigeonhole principle, on two of these inputs M ends in the same state

Pigeonhole principle

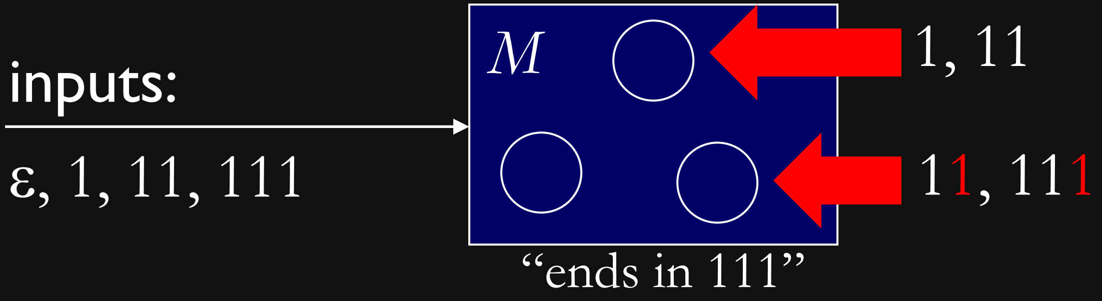
Suppose you are tossing $n + 1$ balls into n bins, and Then two balls end up in the same bin.

- Here, balls are inputs, bins are states:

If you have a DFA with n states and you run it on $n + 1$ inputs, then two of them end up in same state.

A smaller DFA

- Suppose M ends up in the same state after reading inputs $x = 1$ and $y = 11$

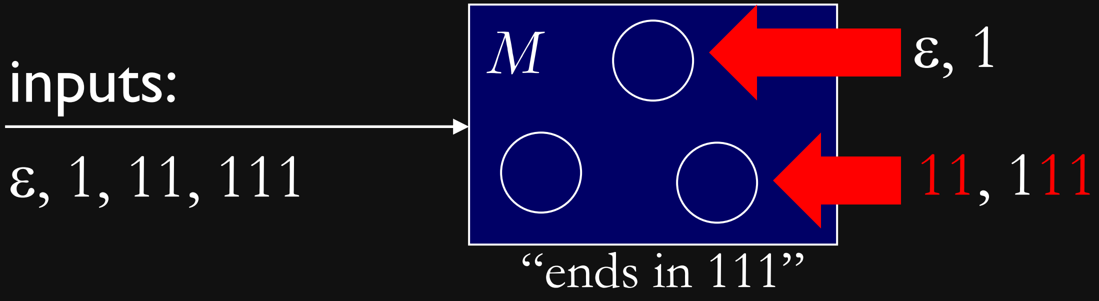


- Then after reading one more 1
  - The state of $x1 = 11$ should be rejecting
  - The state of $y1 = 111$ should be accepting

… but they are both the same state!

A smaller DFA

- Suppose M ends up in the same state after reading inputs $x = \varepsilon$ and $y = 1$



- Then after reading 11
  - The state of $x1 = 11$ should be rejecting
  - The state of $y1 = 111$ should be accepting

… but they are both the same state!

No smaller DFA!

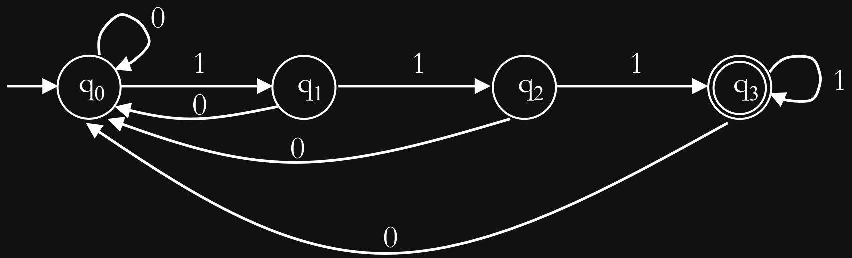
- After looking at all possible pairs for $x, y, x \neq y$

  (ε, 1) (ε, 11) (ε, 111) (1, 11) (1, 111) (11, 111)

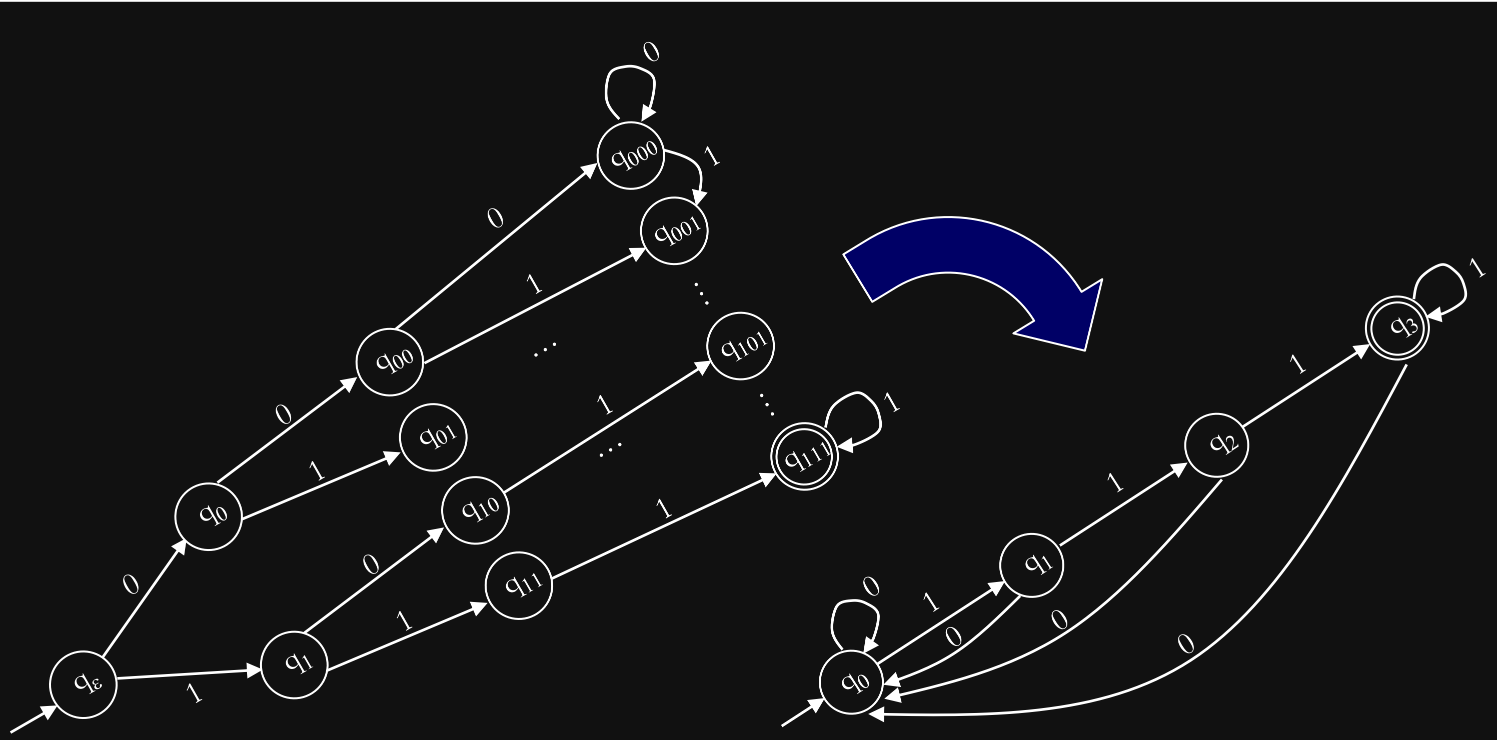
  we conclude that

  There is no DFA with 3 states for L

- So, this DFA is minimal



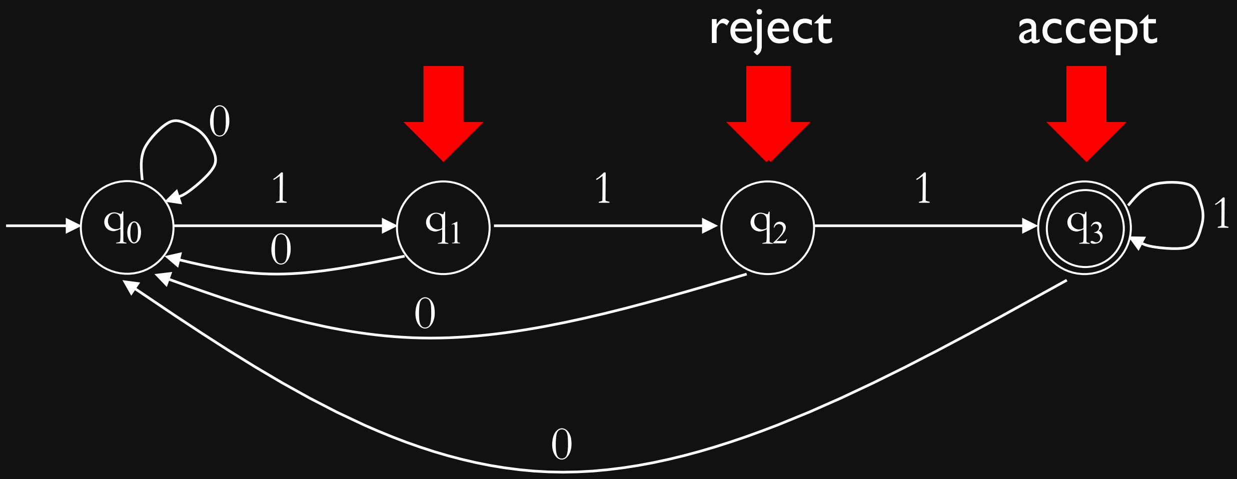
DFA minimization



We will show how to turn any DFA for L into the minimal DFA for L

Minimal DFAs and distinguishable states

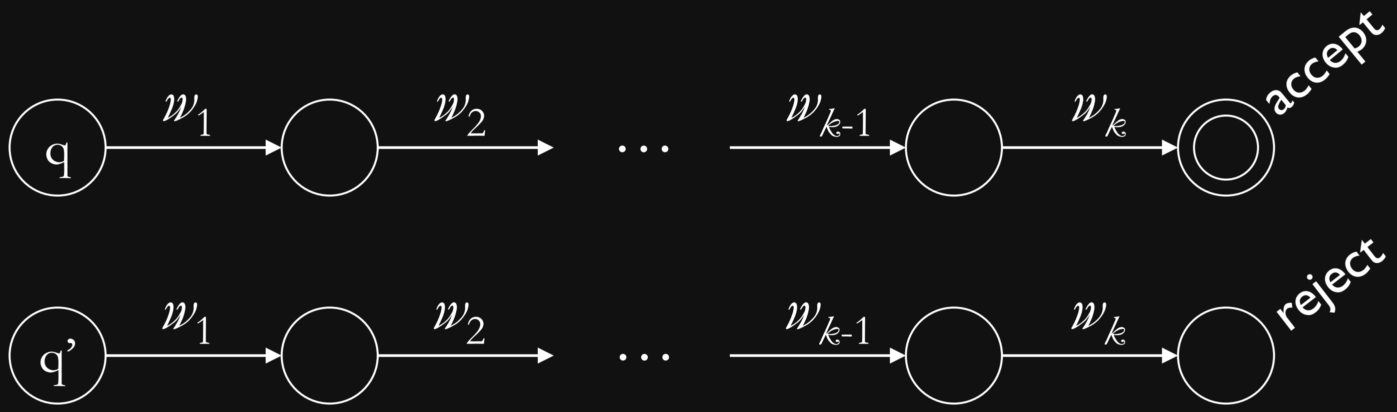
- First, we have to understand minimal DFAs:



minimal DFA ⟺ every pair of states is distinguishable

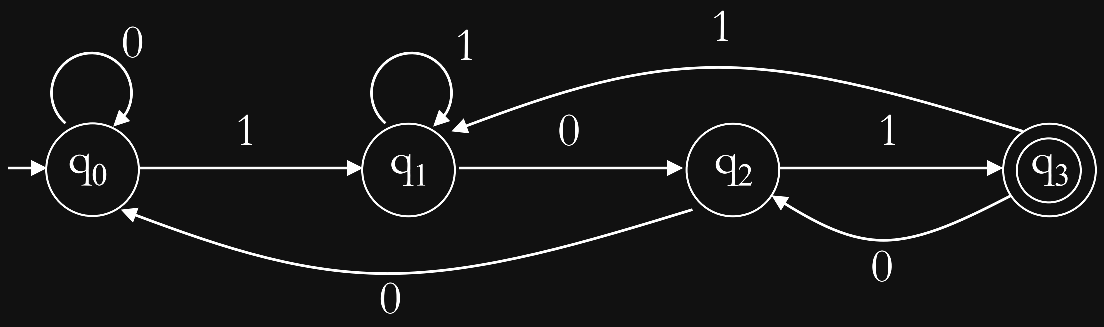
Distinguishable states

- Two states q and q' are distinguishable if



on the same continuation string $w_1w_2\dots w_k$, one accepts, but the other rejects

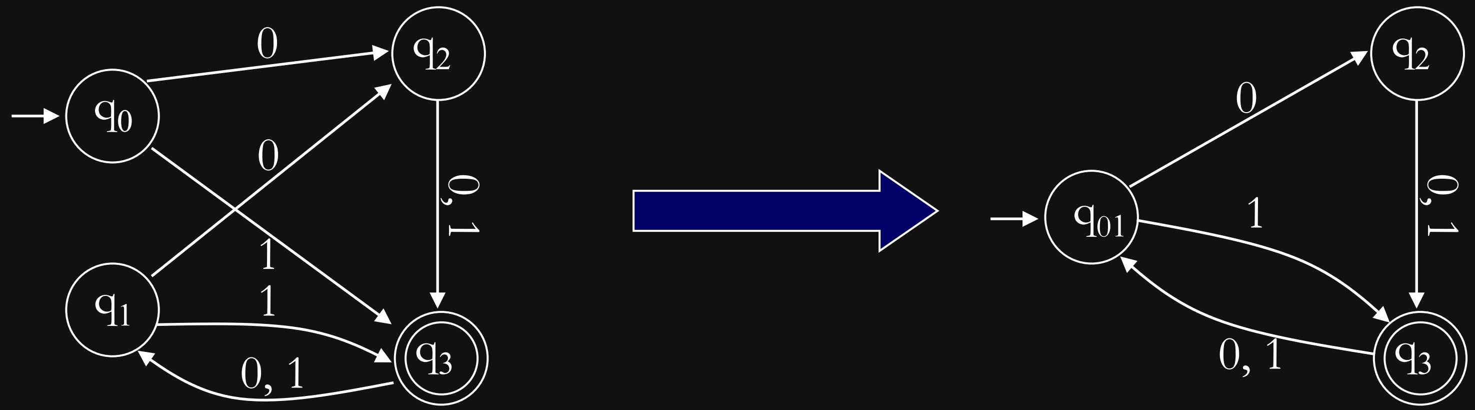
Examples of distinguishable states



(q_0, q_1)	distinguishable by 01
(q_0, q_2)	distinguishable by 1
(q_0, q_3)	distinguishable by ε
(q_1, q_2)	distinguishable by 1
(q_1, q_3)	distinguishable by ε
(q_2, q_3)	distinguishable by ε

DFA is minimal

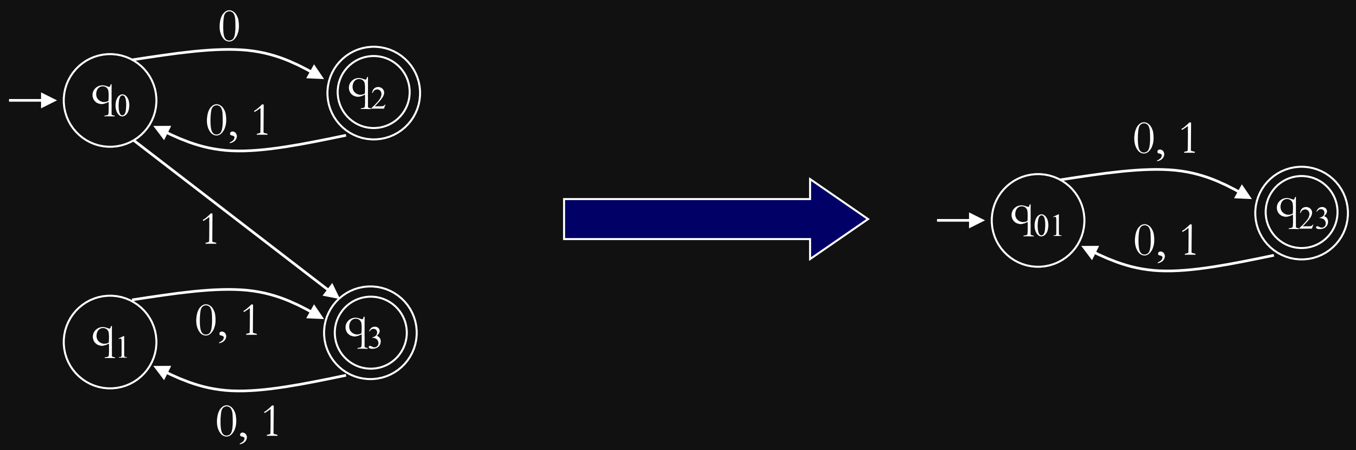
Examples of distinguishable states



(q_0, q_3)	distinguishable by ε
(q_1, q_3)	distinguishable by ε
(q_2, q_3)	distinguishable by ε
(q_1, q_2)	distinguishable by 0
(q_0, q_2)	distinguishable by 0
(q_0, q_1)	indistinguishable

indistinguishable pairs can be merged

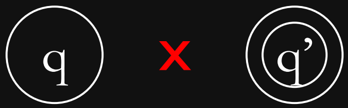
Examples of distinguishable states



(q_0, q_2)	distinguishable by ε
(q_1, q_2)	distinguishable by ε
(q_0, q_3)	distinguishable by ε
(q_1, q_3)	distinguishable by ε
(q_0, q_1)	indistinguishable
(q_2, q_3)	indistinguishable

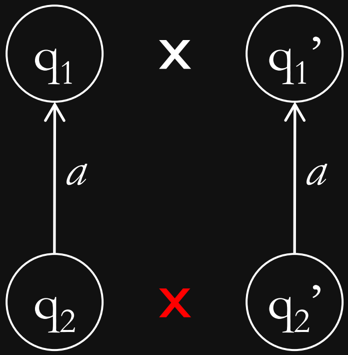
Finding (in)distinguishable states

Rule 1:



If q is accepting and q' is rejecting
Mark (q, q') as distinguishable (x)

Rule 2:

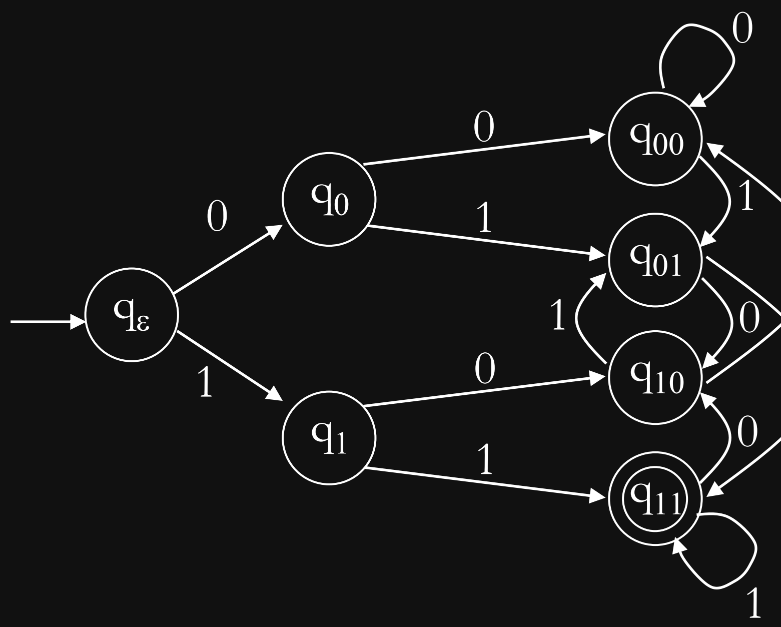


If (q_1, q_1') are marked,
Mark (q_2, q_2') as distinguishable (x)

Rule 3:

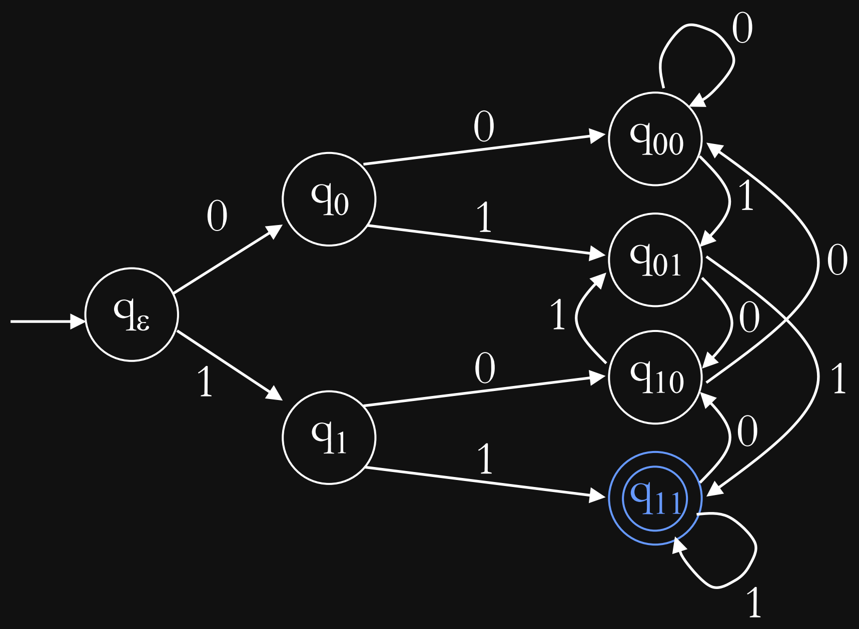
Unmarked pairs are indistinguishable
Merge them into groups

Example of DFA minimization



q_0						
q_1						
q_{00}						
q_{01}						
q_{10}						
q_{11}						
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}

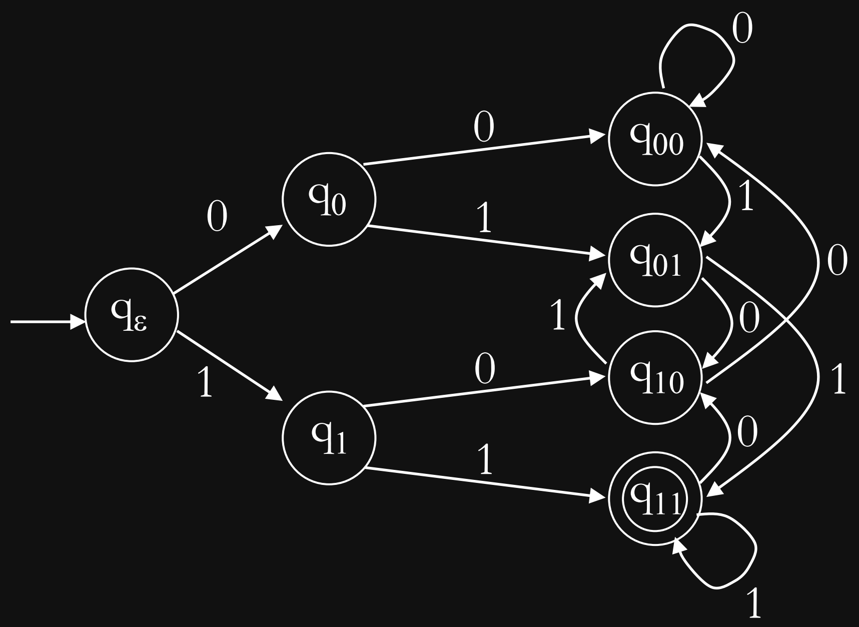
Example of DFA minimization

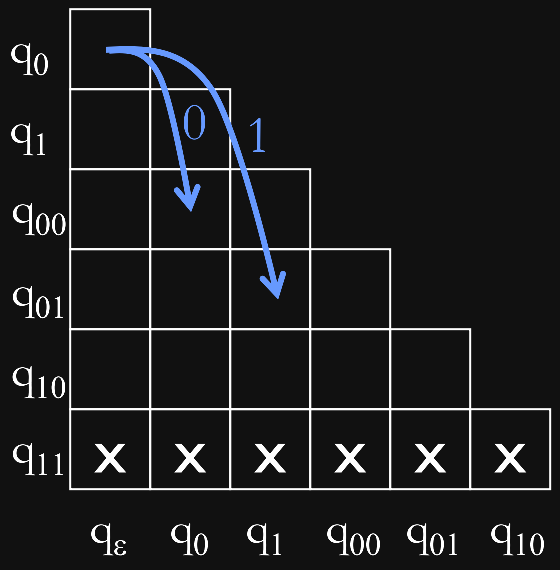


	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1						
q_{00}						
q_{01}						
q_{10}						
q_{11}	x	x	x	x	x	x

① q_{11} is distinguishable from all other states

Example of DFA minimization



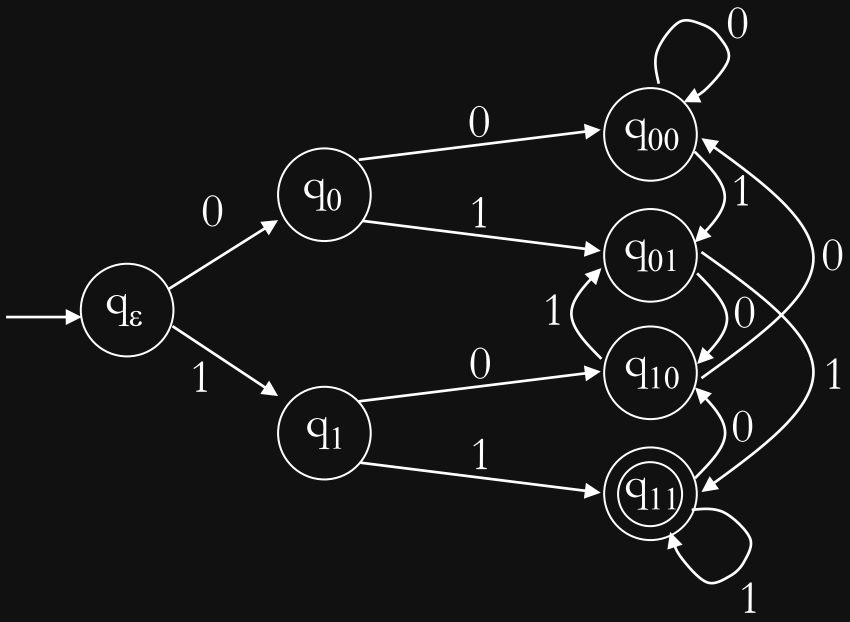


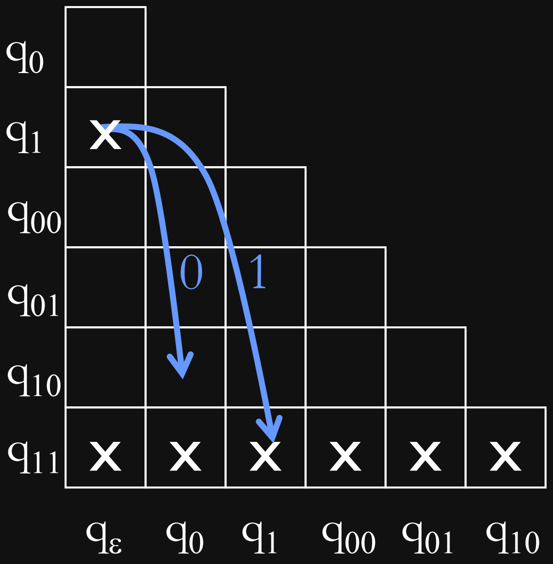
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1						
q_{00}						
q_{01}						
q_{10}						
q_{11}	x	x	x	x	x	x

② Look at pair q_ε, q_0

Neither (q_0, q_{00}) nor (q_1, q_{01}) are distinguishable

Example of DFA minimization



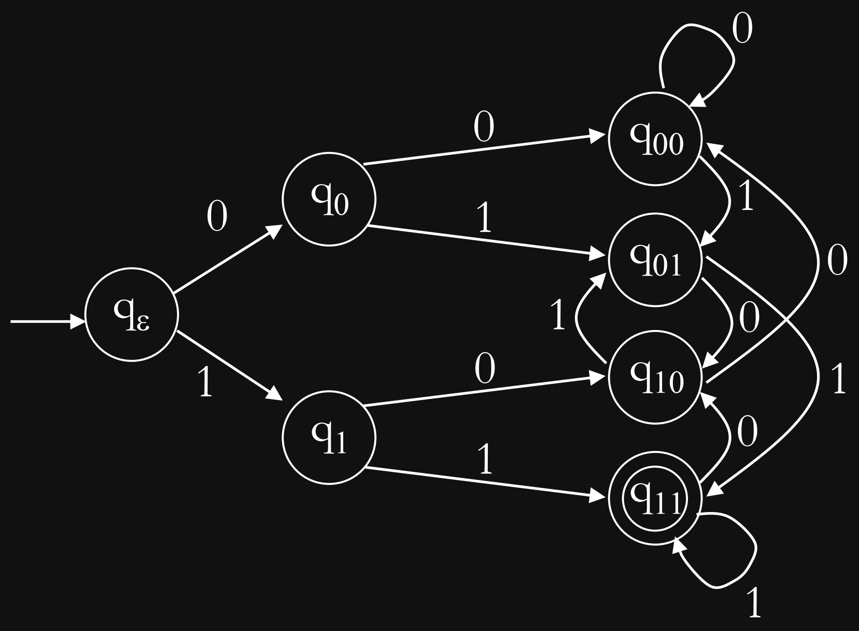


	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1	x					
q_{00}						
q_{01}						
q_{10}						
q_{11}	x	x	x	x	x	x

② Look at pair q_ε, q_1

(q_1, q_{11}) is distinguishable

Example of DFA minimization

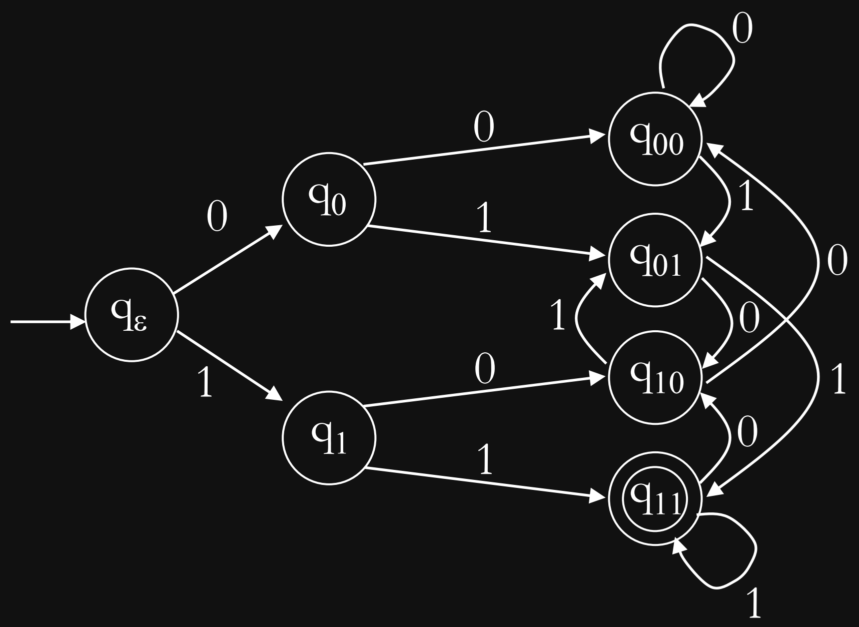


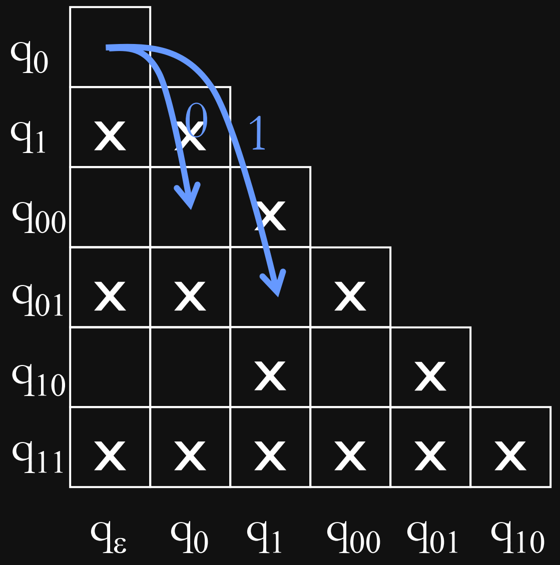
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1	x	x				
q_{00}			x			
q_{01}	x	x		x		
q_{10}			x		x	
q_{11}	x	x	x	x	x	x

② After going thru the whole table once

Now we make another pass

Example of DFA minimization



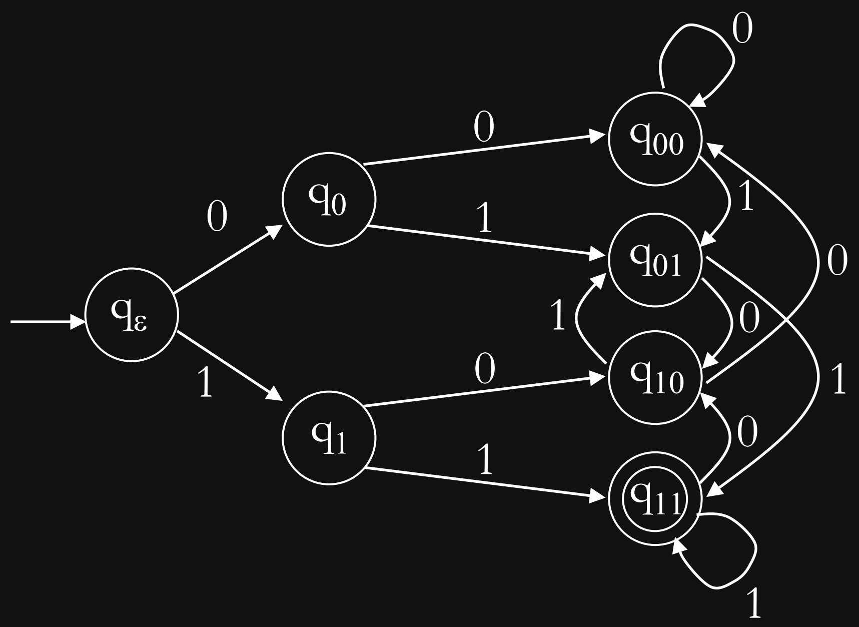


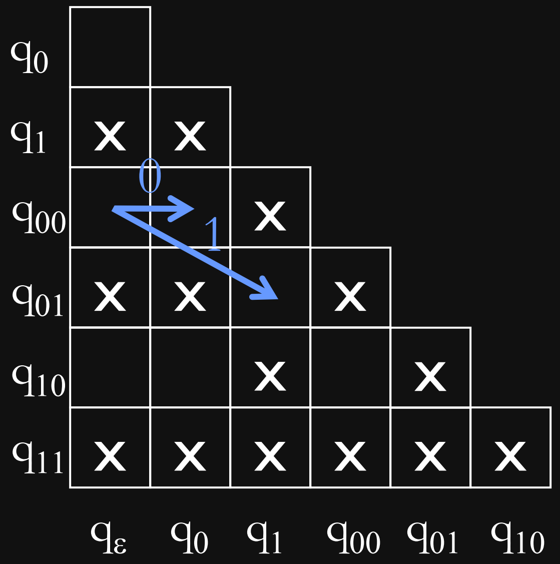
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1	x	x				
q_{00}			x			
q_{01}	x	x		x		
q_{10}			x		x	
q_{11}	x	x	x	x	x	x

② Look at pair q_ε, q_0

Neither (q_1, q_{00}) nor (q_1, q_{01}) are distinguishable

Example of DFA minimization



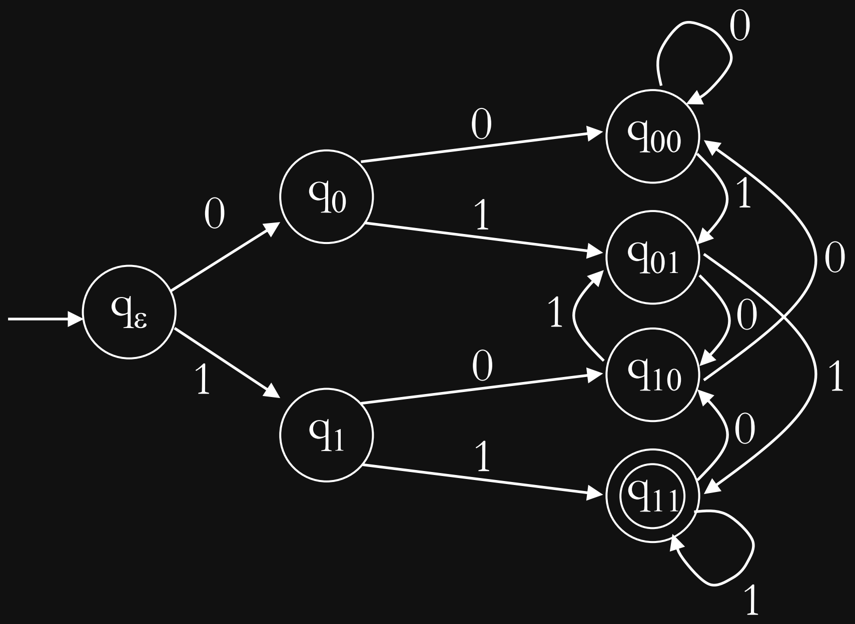


q_0						
q_1	x	x				
q_{00}			x			
q_{01}	x	x		x		
q_{10}			x		x	
q_{11}	x	x	x	x	x	x
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}

② Look at pair q_ε, q_{00}

Neither (q_0, q_{00}) nor (q_1, q_{01}) are distinguishable

Example of DFA minimization

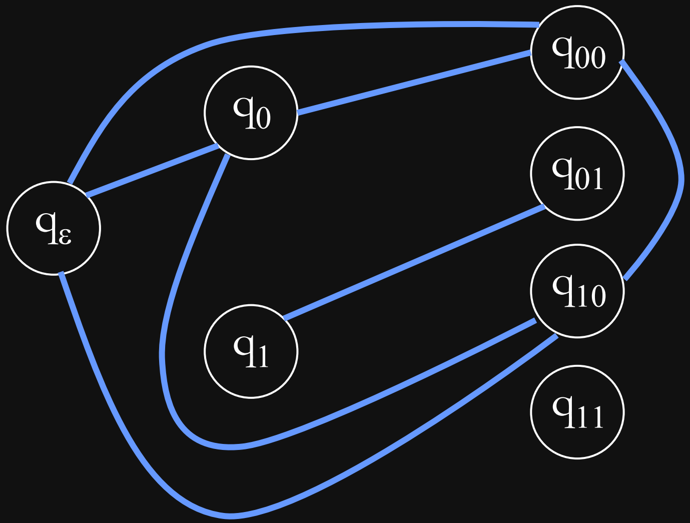


q_0						
q_1	x	x				
q_{00}			x			
q_{01}	x	x		x		
q_{10}			x		x	
q_{11}	x	x	x	x	x	x
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}

② In the second pass, nothing changes

So we are ready to apply Rule 3

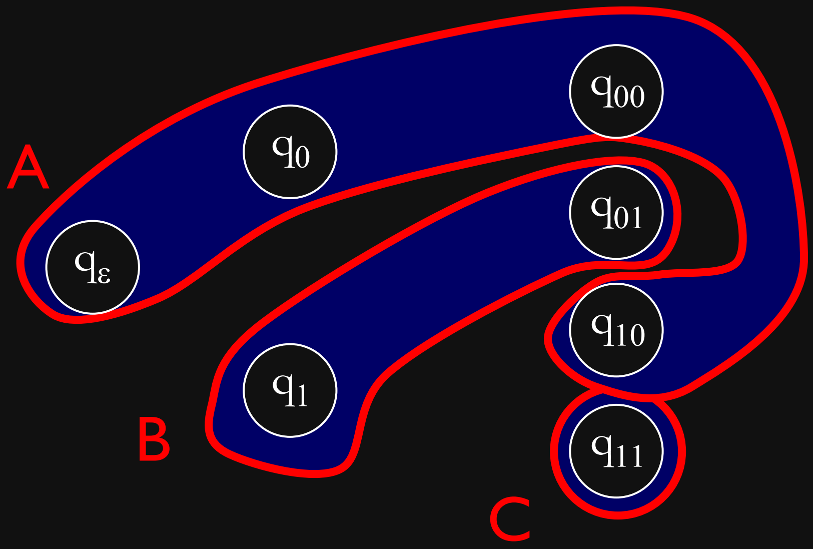
Example of DFA minimization



q_{0}						
q_{1}	x	x				
q_{00}			x			
q_{01}	x	x		x		
q_{10}			x		x	
q_{11}	x	x	x	x	x	x
	q_{ε}	q_{0}	q_{1}	q_{00}	q_{01}	q_{10}

③ Merge unmarked pairs into groups

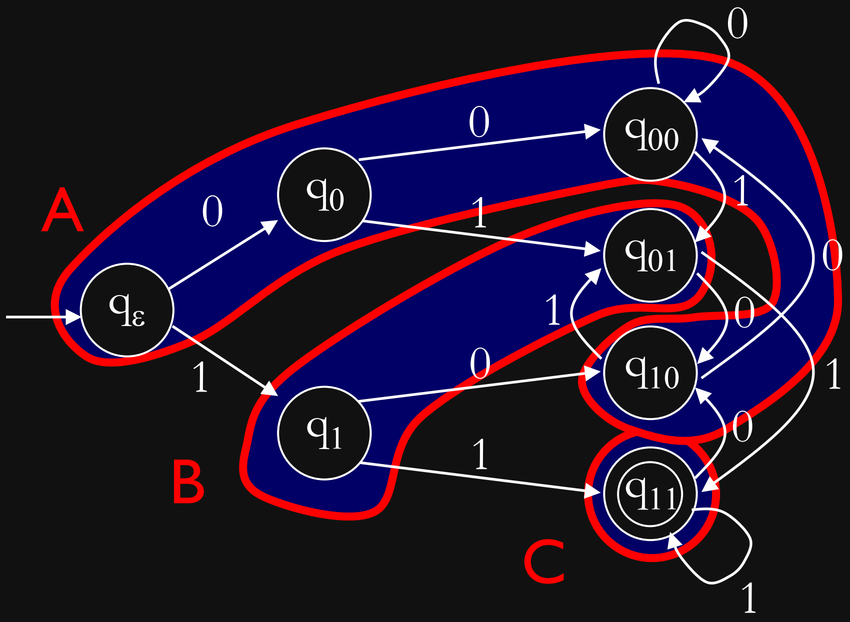
Example of DFA minimization



	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0	A					
q_1	x	x				
q_{00}	A	A	x			
q_{01}	x	x	B	x		
q_{10}	A	A	x	A	x	
q_{11}	x	x	x	x	x	x

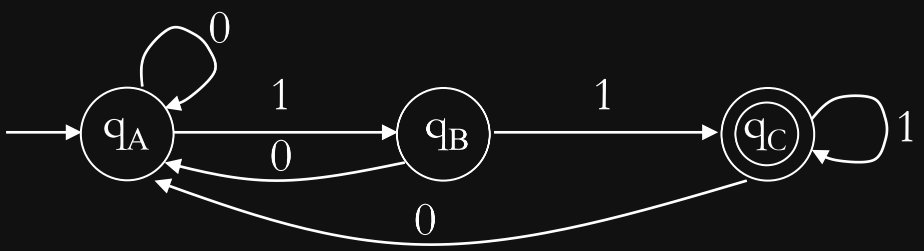
③ Merge unmarked pairs into groups

Example of DFA minimization

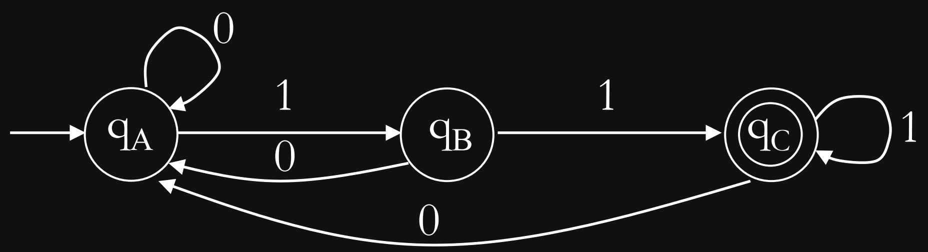


	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0	A					
q_1	x	x				
q_{00}	A	A	x			
q_{01}	x	x	B	x		
q_{10}	A	A	x	A	x	
q_{11}	x	x	x	x	x	x

minimized DFA:



Example of DFA minimization



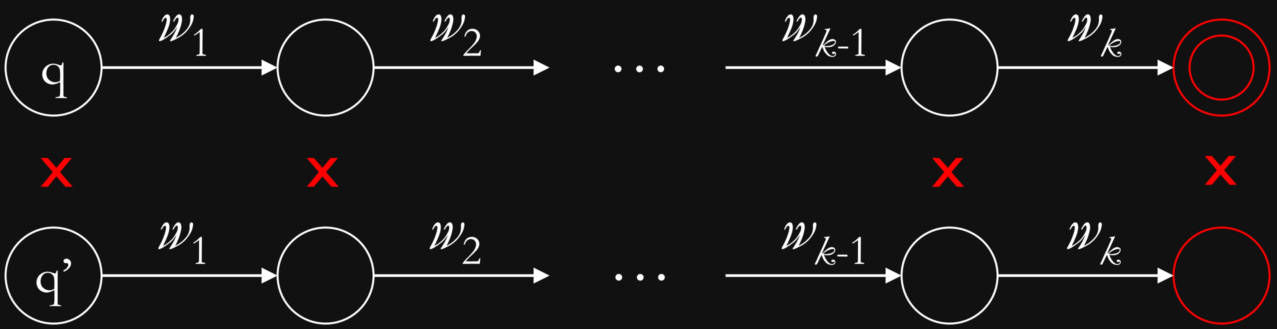
How do we know this DFA is minimal?

Answer: All pairs are distinguishable

q_B	1	
q_C	ε	ε
	q_A	q_B

Why it works

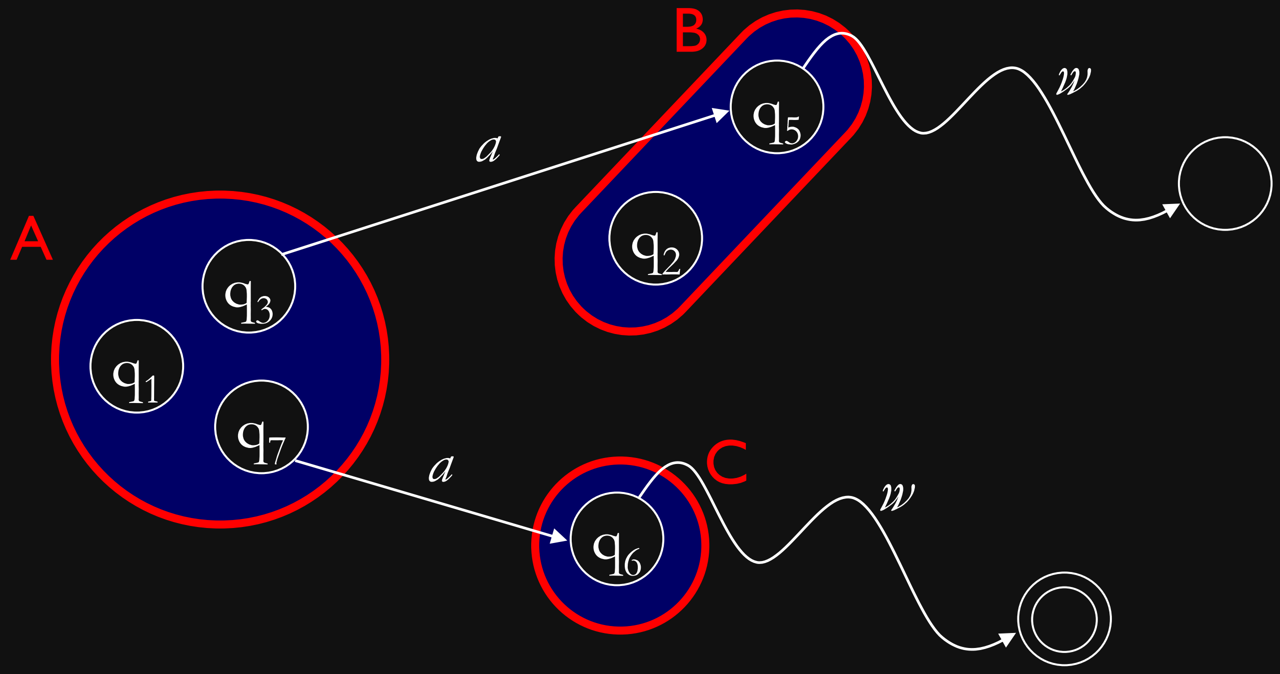
- Why do we end up finding all distinguishable pairs?



Because we work backwards

Why it works

- Why are there no inconsistencies when we merge?



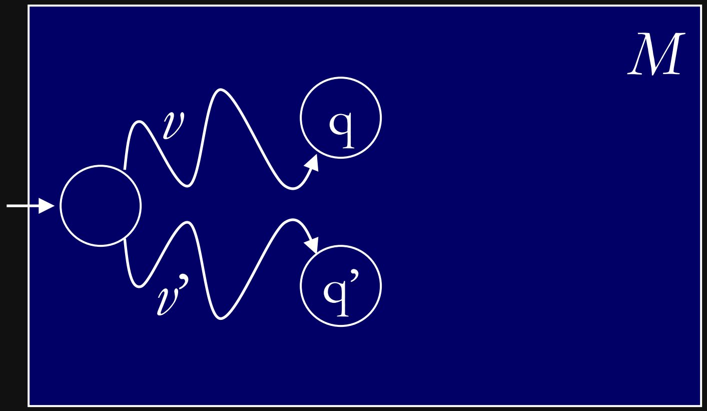
Because we only merge indistinguishable states

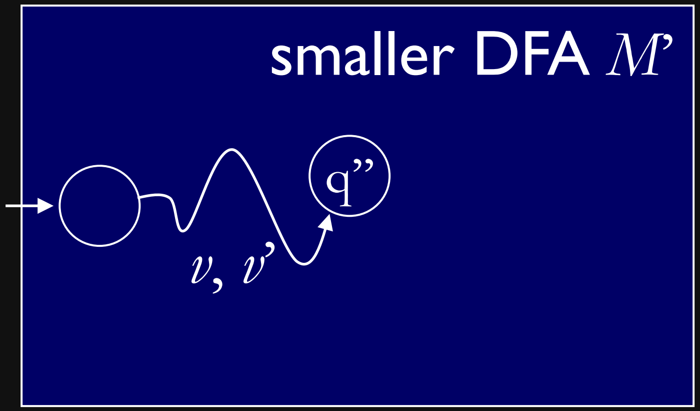
Why it works

- Why is there no smaller DFA?

  Suppose there is

  By the pigeonhole principle this must happen:

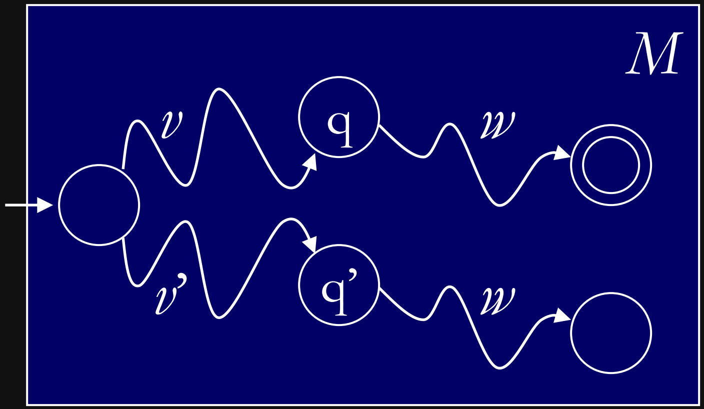




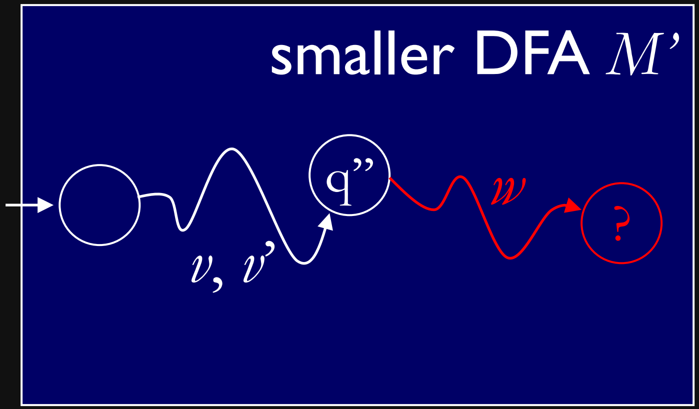
Why it works

- Why is there no smaller DFA?

But then



Every pair of states
is distinguishable



q'' cannot exist!