DFA Minimization, Pumping Lemma

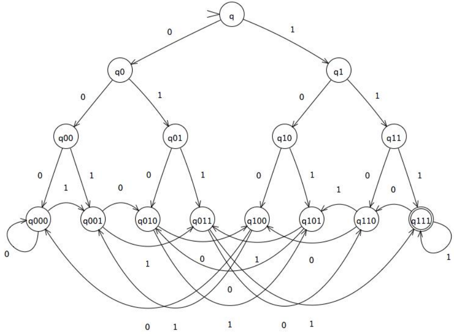
CSCI 3130 Formal Languages and Automata Theory

Siu On CHAN

Chinese University of Hong Kong

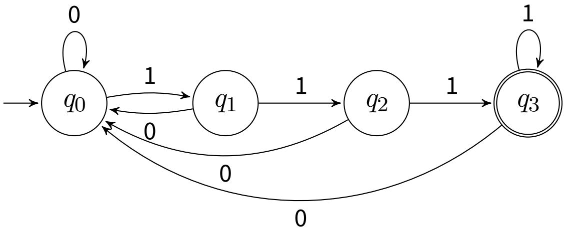
Fall 2015

$L =$ strings ending in 111



There is a simpler one…

$L =$ strings ending in 111



Can we do it in 3 states?

Even smaller DFA?

$L =$ strings ending in 111

Intuitively, needs to remember number of ones recently read

We will show

ε, 1, 11, 111 are pairwise distinguishable by L

In other words

$(\varepsilon, 1), (\varepsilon, 11), (\varepsilon, 111), (1, 11), (1, 111), (11, 111)$

are all distinguishable by L

Then use this result from last lecture:

If strings $x_1, \ldots, x_n$ are pairwise distinguishable by L, any DFA accepting L must have at least n states

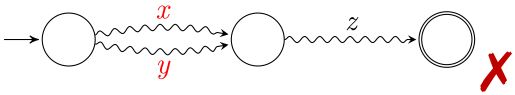
Recap: distinguishable strings

What do we mean by "1 and 11 are distinguishable"?

(x, y) are distinguishable by L if there is string z such that $xz \in L$ and $yz \notin L$ (or the other way round)

We saw from last lecture

If x and y are distinguishable by L, any DFA accepting L must reach different states upon reading x and y



Distinguishable strings

Why are 1 and 11 distinguishable by L?

$L =$ strings ending in 111

Distinguishable strings

Why are 1 and 11 distinguishable by L?

$L =$ strings ending in 111

Take $z = 1$

$11 \notin L \qquad 111 \in L$

More generally, why are 1^i and 1^j distinguishable by L?

$(0 \leqslant i < j \leqslant 3)$

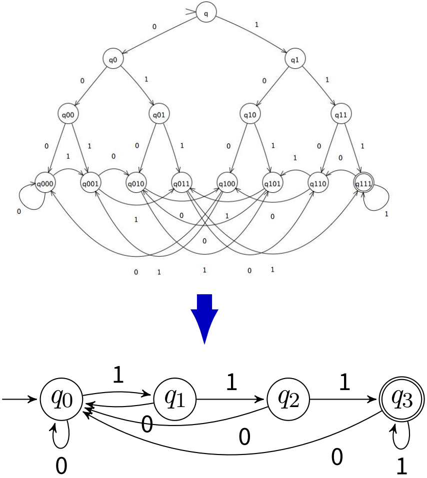
Take $z = 1^{3-j}$

$1^i 1^{3-j} \notin L \qquad 1^j 1^{3-j} \in L$

ε, 1, 11, 111 are pairwise distinguishable by L

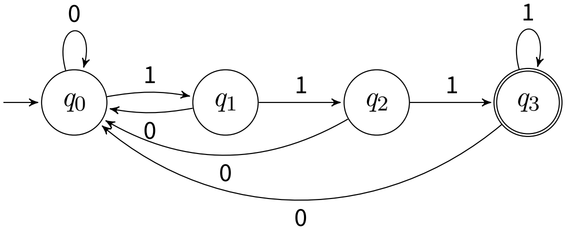
Thus our 4-state DFA is minimal

DFA minimization



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

Minimal DFA and distinguishability

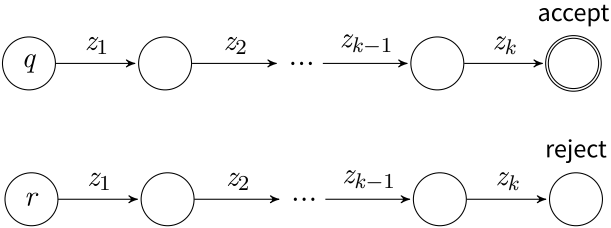


Distinguishable strings must be in different states
Indistinguishable strings may end up in the same state

DFA minimial $\Leftrightarrow$ Every pair of states is distinguishable

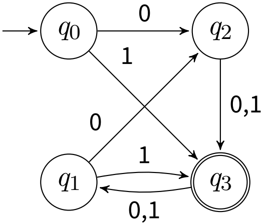
Distinguishable states

Two states q and r are distinguishable if



on the same continuation string $z = z_1 \ldots z_k$,
one accepts, but the other rejects

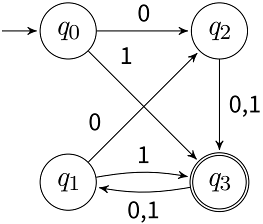
Examples of distinguishable states



Which of the following pairs are distinguishable? by which string?

(q_0, q_3)
(q_1, q_3)
(q_2, q_3)
(q_1, q_2)
(q_0, q_2)
(q_0, q_1)

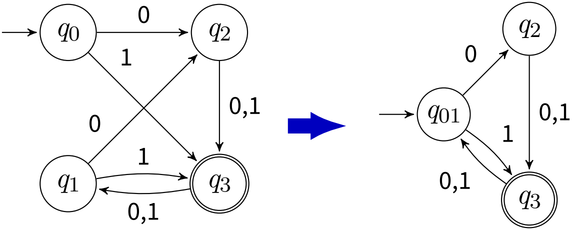
Examples of distinguishable states



Which of the following pairs are distinguishable? by which string?

(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

Examples of distinguishable states



Which of the following pairs are distinguishable? by which string?

(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

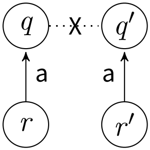
indistinguishable pairs can be merged

Finding (in)distinguishable states

Phase 1:

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

Phase 2:



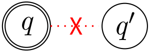
If (q, q') are marked
Mark (r, r') as distinguishable (X)

Phase 3:

Unmarked pairs are indistinguishable
Merge them into groups

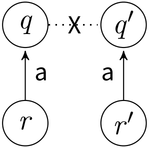
Finding (in)distinguishable states

Phase 1:



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

Phase 2:



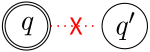
If (q, q') are marked
Mark (r, r') as distinguishable (X)

Phase 3:

Unmarked pairs are indistinguishable
Merge them into groups

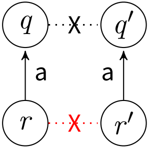
Finding (in)distinguishable states

Phase 1:



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

Phase 2:

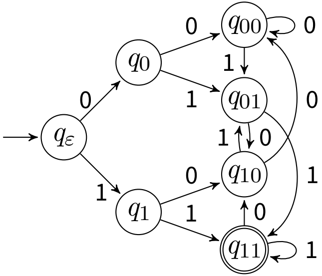


If (q, q') are marked
Mark (r, r') as distinguishable (X)

Phase 3:

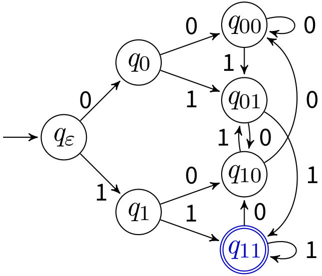
Unmarked pairs are indistinguishable
Merge them into groups

DFA minimization: example



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

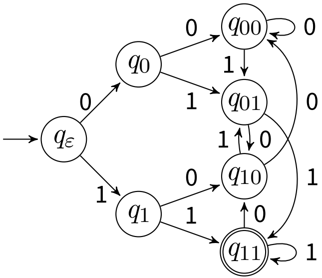
DFA minimization: example



	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

(Phase 1) q_{11} is distinguishable from all other states

DFA minimization: example

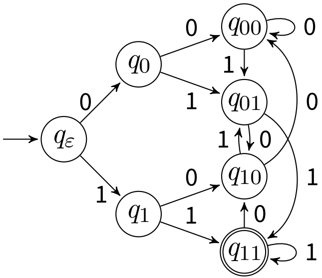


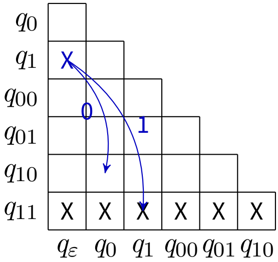
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1						
q_{00}						
q_{01}						
q_{10}						
q_{11}	X	X	X	X	X	X

(Phase 2) Looking at $(r, r') = (q_\varepsilon, q_0)$

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

DFA minimization: example

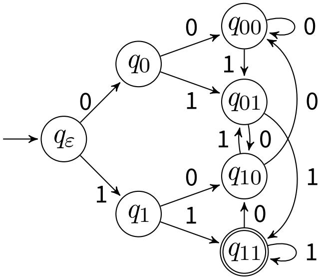




(Phase 2) Looking at $(r, r') = (q_\varepsilon, q_1)$

(q_1, q_{11}) is distinguishable

DFA minimization: example

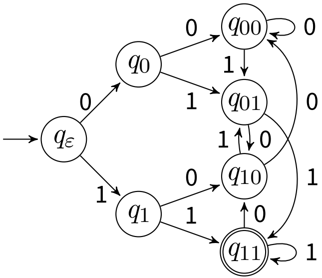


q_0						
q_1	X	X				
q_{00}			X			
q_{01}	X	X		X		
q_{10}			X		X	
q_{11}	X	X	X	X	X	X
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}

(Phase 2) After going through the whole table once

Now we make another pass

DFA minimization: example

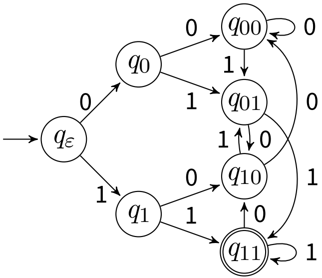


	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1	X	X				
q_{00}			X			
q_{01}	X	X		X		
q_{10}			X		X	
q_{11}	X	X	X	X	X	X

(Phase 2) Looking at $(r, r') = (q_\varepsilon, q_0)$

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

DFA minimization: example

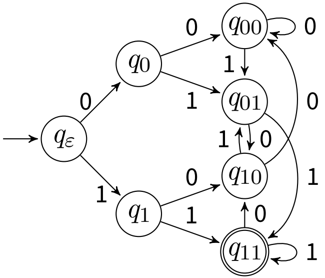


	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1	X	X				
q_{00}			X			
q_{01}	X	X		X		
q_{10}			X		X	
q_{11}	X	X	X	X	X	X

(Phase 2) Looking at $(r, r') = (q_\varepsilon, q_{00})$

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

DFA minimization: example

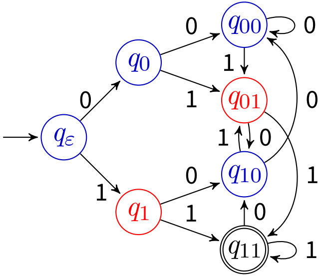


	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0						
q_1	X	X				
q_{00}			X			
q_{01}	X	X		X		
q_{10}			X		X	
q_{11}	X	X	X	X	X	X

(Phase 2) Nothing changes in the second pass

Ready to go to Phase 3

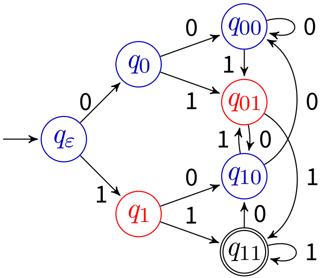
DFA minimization: example



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
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}

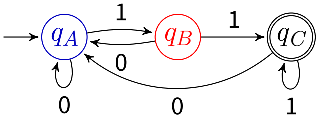
(Phase 3) Merge states into groups (also called equivalence classes)

DFA minimization: example



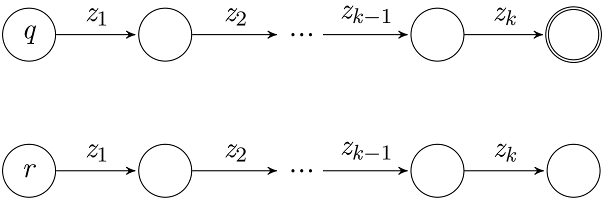
	q_ε	q_0	q_1	q_{00}	q_{01}	q_{10}
q_0	A					
q_1	X	X				
q_{00}	A	A	X			
q_{01}	X	X	B	X		
q_{10}	A	A	X	A	X	
q_{11}	X	X	X	X	X	X

Minimized DFA:



Why it works

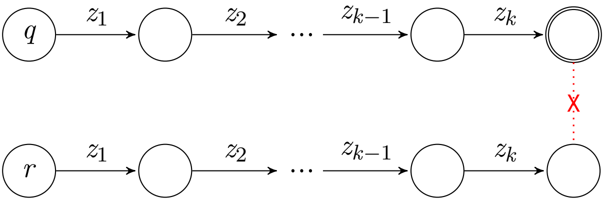
Why have we found all distinguishable pairs?



Because we work backwards

Why it works

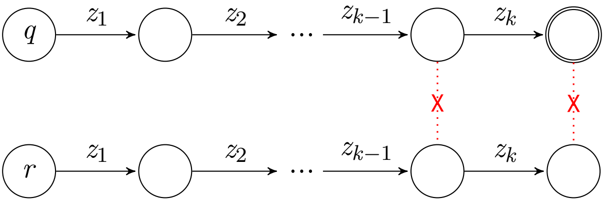
Why have we found all distinguishable pairs?



Because we work backwards

Why it works

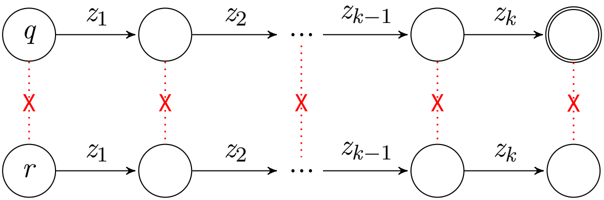
Why have we found all distinguishable pairs?



Because we work backwards

Why it works

Why have we found all distinguishable pairs?



Because we work backwards

Pumping Lemma

Pumping lemma

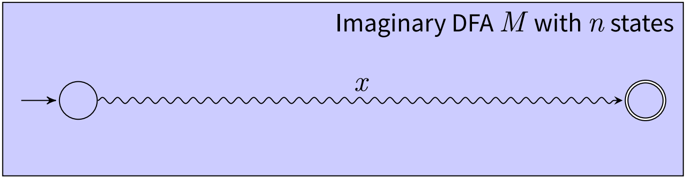
Another way to show some language is irregular

Example

$L = \{0^n 1^n \mid n \geqslant 0\}$ is irregular

We reason by contradiction:
Suppose we have a DFA M for L
Something must be wrong with this DFA
M must accept some strings outside L

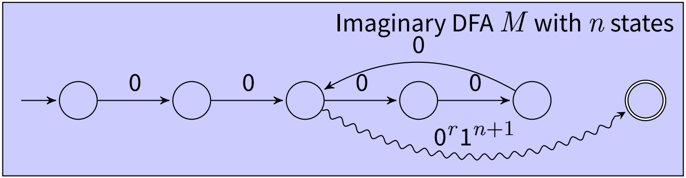
Towards a contradiction



What happens when M gets input $x = 0^{n+1}1^{n+1}$?

M accepts x because $x \in L$

Towards a contradiction

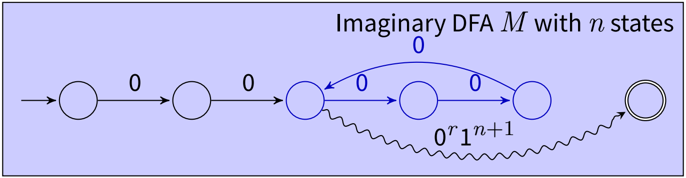


What happens when M gets input $x = 0^{n+1}1^{n+1}$?

M accepts x because $x \in L$

Since M has n states, it must revisit one of its states while reading 0^{n+1}

Towards a contradiction



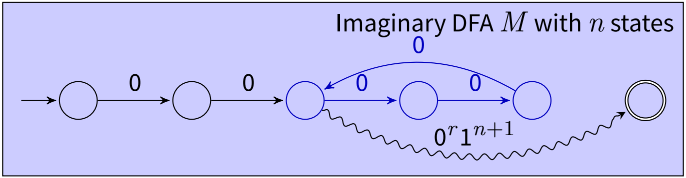
What happens when M gets input $x = 0^{n+1}1^{n+1}$?

M accepts x because $x \in L$

Since M has n states, it must revisit one of its states while reading 0^{n+1}

The DFA must contain a loop with 0s

Towards a contradiction



What happens when M gets input $x = 0^{n+1}1^{n+1}$?

M accepts x because $x \in L$

Since M has n states, it must revisit one of its states while reading 0^{n+1}

The DFA must contain a loop with 0s

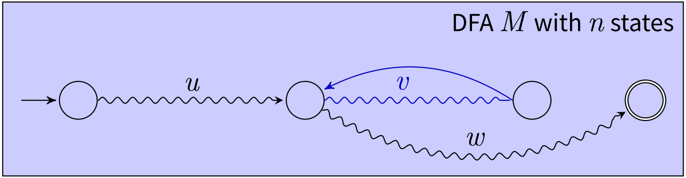
The DFA will also accept strings that go around the loop multiple times

But such strings have more 0s than 1s and cannot be in L

Pumping lemma for regular languages

For every regular language L, there exists a number n such that
for every string $s \in L$ longer than n symbols,
we can write $s = uvw$ where

1. $|uv| \leqslant n$
2. $|v| \geqslant 1$
3. For every $i \geqslant 0$, the string uv^iw is in L



Proving languages are irregular

For every regular language L, there exists a number n such that
for every string $s \in L$ longer than n symbols,
we can write $s = uvw$ where

1. $|uv| \leqslant n$
2. $|v| \geqslant 1$
3. For every $i \geqslant 0$, the string uv^iw is in L

To show that a language L is irregular
we need to find arbitrarily long s
so that no matter how the lemma splits s into u, v, w
we can find $uv^iw \notin L$

Example

$$L_2 = \{\mathtt{0}^m\mathtt{1}^n \mid m > n \geqslant 0\}$$

1. For any n (number of states of an imaginary DFA)
2. There is a string $s = \mathtt{0}^{n+1}\mathtt{1}^n$
3. Pumping lemma splits s into uvw
4. Choose $i = 0$ so that $uv^iw \notin L_2$

Example: 00000011111