More Properties of Regular Languages

We have proven

Regular languages are closed under:

- Union
- Concatenation
- Star operation
- Reverse

Namely, for regular languages L_1 and L_2 :

Union $L_1 \cup L_2$

Concatenation $L_1 L_2$

Star operation L_1^*

Reverse L_1^R

} Regular Languages

We will prove

Regular languages are closed under:

Complement

Intersection

Namely, for regular languages L_1 and L_2:

Complement $\overline{L_1}$

Intersection $L_1 \cap L_2$

} Regular Languages

Complement

Theorem: For regular language L the complement $\overline{L}$ is regular

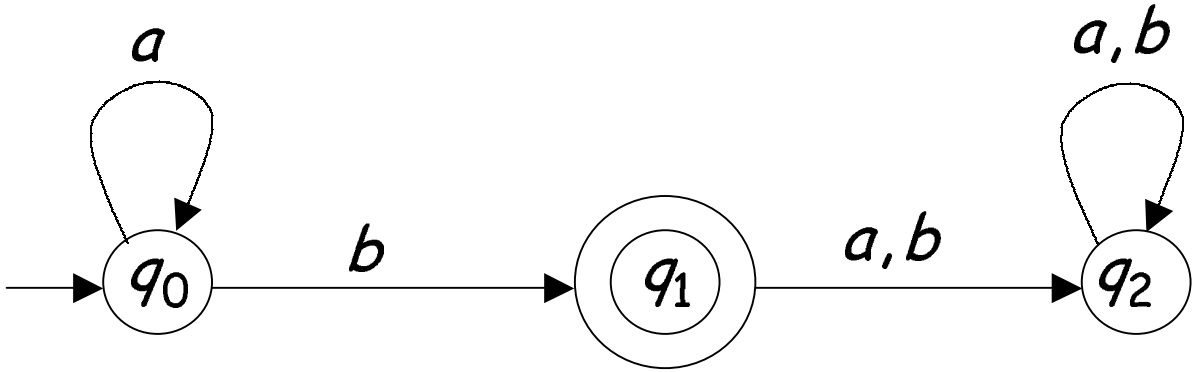
Proof: Take DFA that accepts L and make

- nonfinal states final
- final states nonfinal

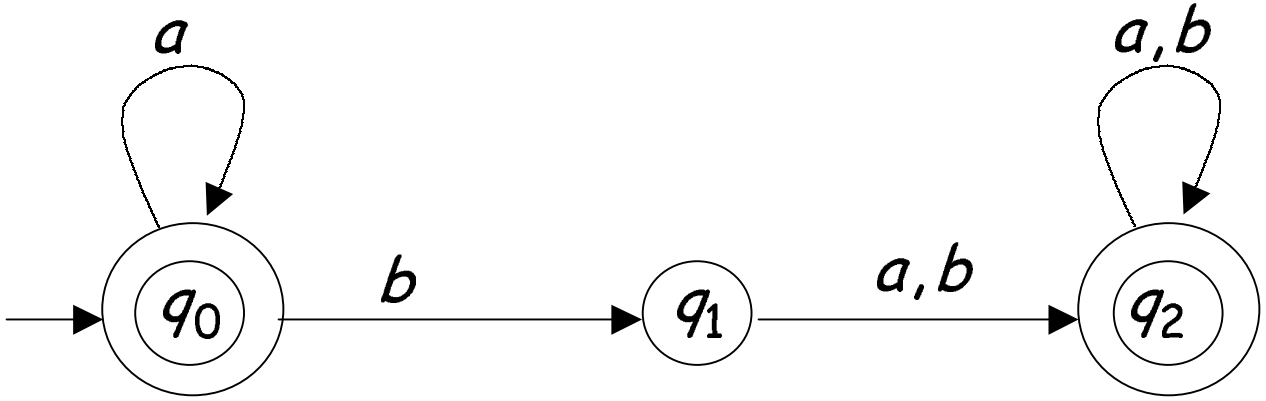
Resulting DFA accepts $\overline{L}$

Example:

$$L = L(a^*b)$$



$$\overline{L} = L(a^* + a^*b(a+b)(a+b)^*)$$



Intersection

Theorem: For regular languages L_1 and L_2 the intersection $L_1 \cap L_2$ is regular

Proof: Apply DeMorgan's Law:

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

L_1, L_2 regular

$\Rightarrow \overline{L_1}, \overline{L_2}$ regular

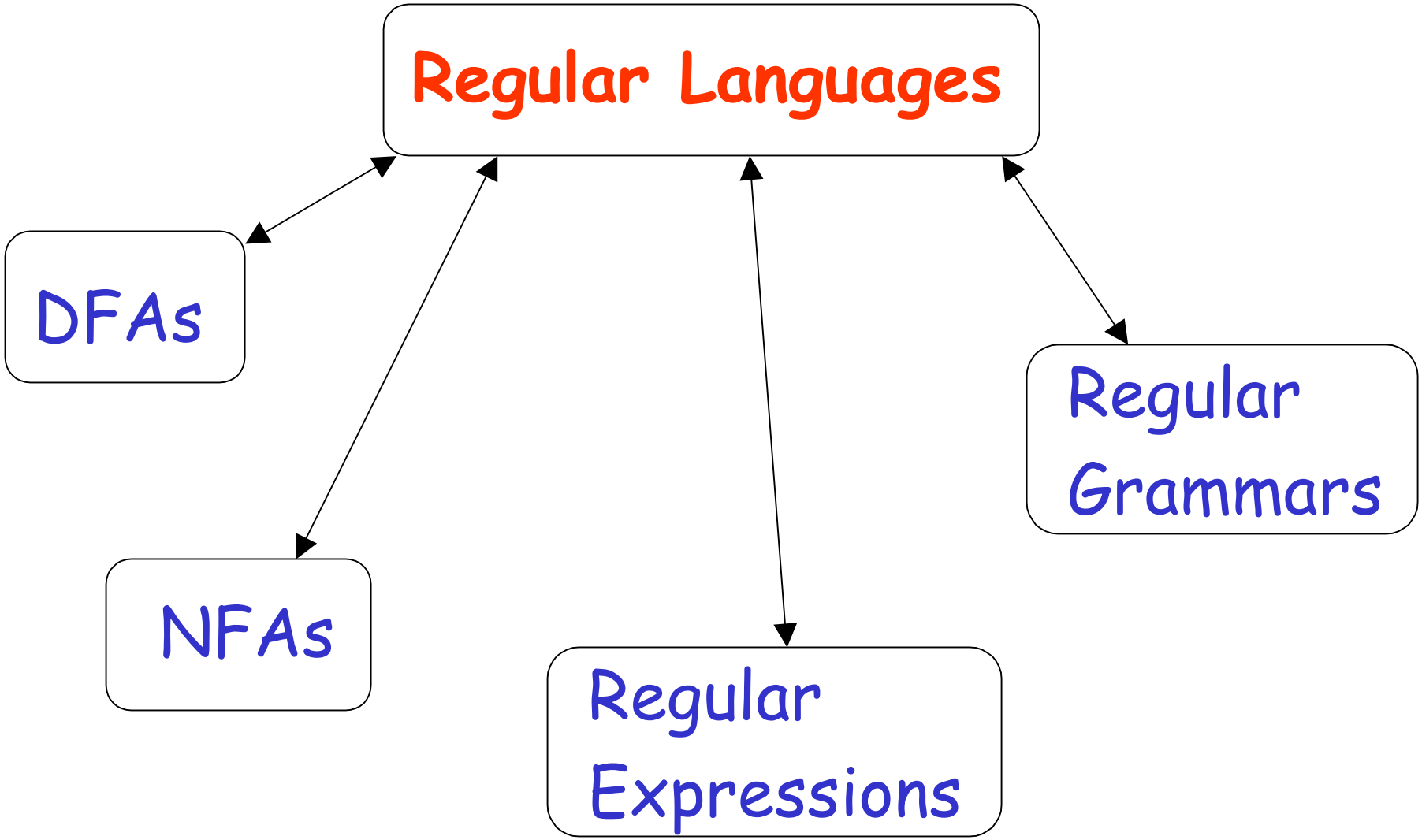
$\Rightarrow \overline{L_1} \cup \overline{L_2}$ regular

$\Rightarrow \overline{\overline{L_1} \cup \overline{L_2}}$ regular

$\Rightarrow L_1 \cap L_2$ regular

Standard Representations of Regular Languages

Standard Representations of Regular Languages



When we say: We are given a Regular Language L

We mean: Language L is in a standard representation

Elementary Questions

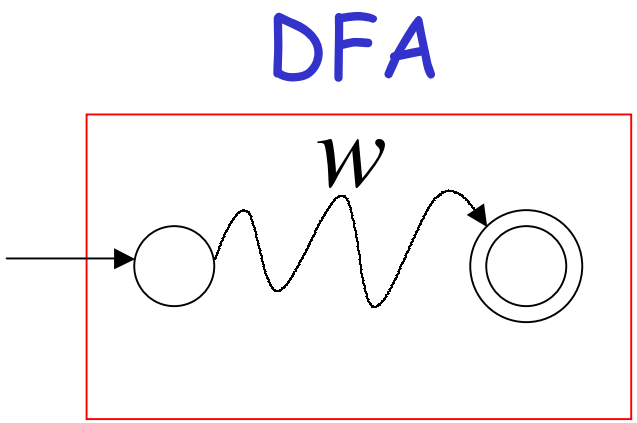
about

Regular Languages

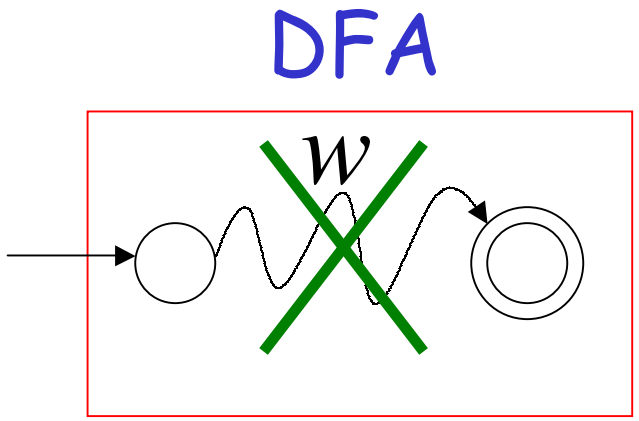
Membership Question

Question: Given regular language L
and string w
how can we check if $w \in L$?

Answer: Take the DFA that accepts L
and check if w is accepted



$w \in L$



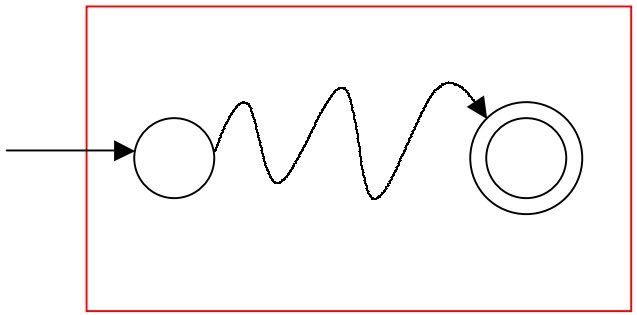
$w \notin L$

Question: Given regular language L
how can we check
if L is empty: $(L = \emptyset)$?

Answer: Take the DFA that accepts L

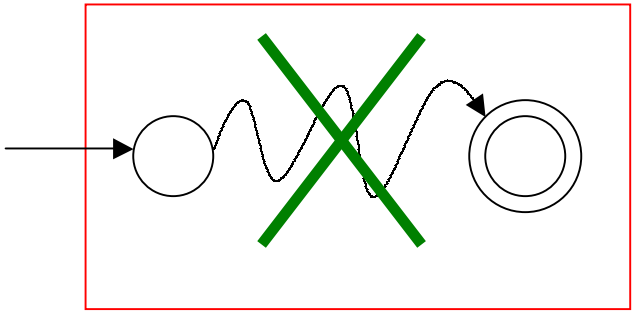
Check if there is a path from
the initial state to a final state

DFA



$L \neq \emptyset$

DFA



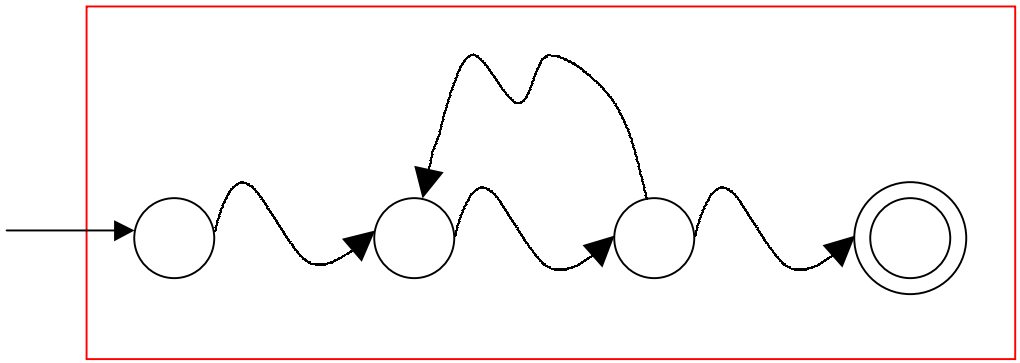
$L = \emptyset$

Question: Given regular language L
how can we check
if L is finite?

Answer: Take the DFA that accepts L

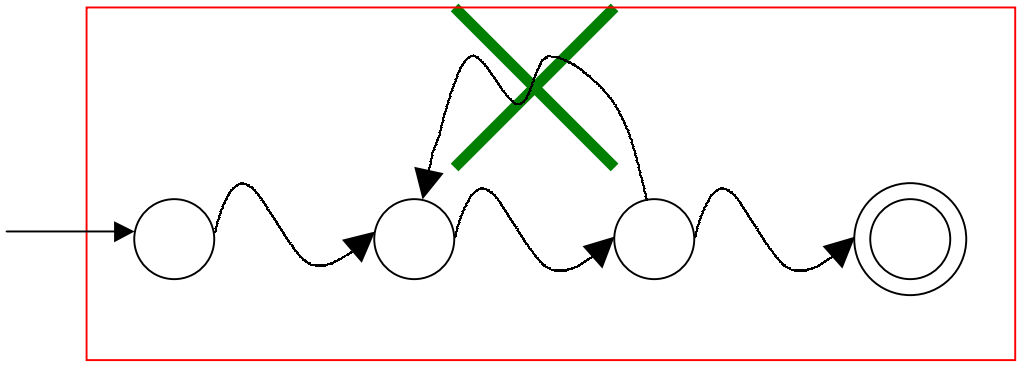
Check if there is a walk with cycle
from the initial state to a final state

DFA



L is infinite

DFA

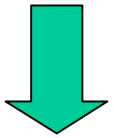


L is finite

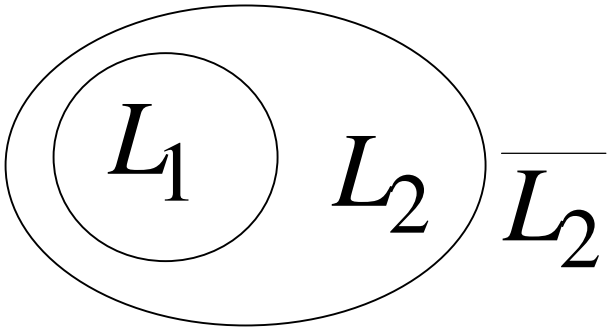
Question: Given regular languages L_1 and L_2 how can we check if $L_1 = L_2$?

Answer: Find if $(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) = \emptyset$

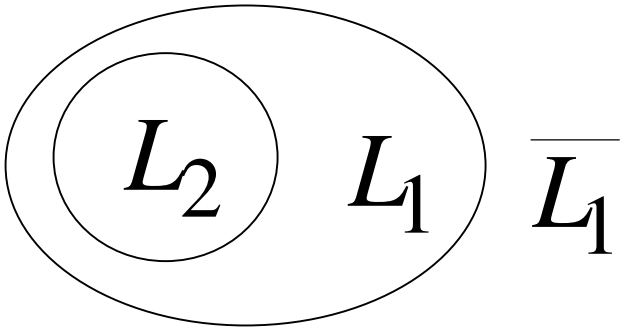
$$(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) = \emptyset$$



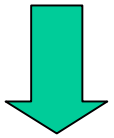
$L_1 \cap \overline{L_2} = \emptyset$ and $\overline{L_1} \cap L_2 = \emptyset$



$L_1 \subseteq L_2$

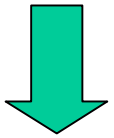


$L_2 \subseteq L_1$

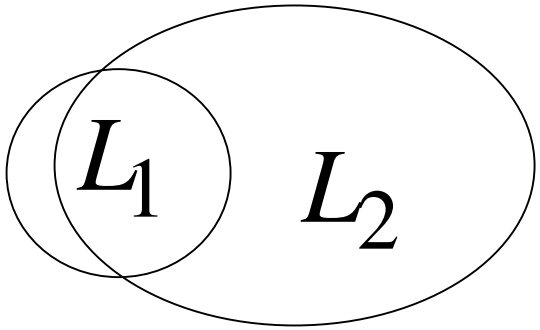


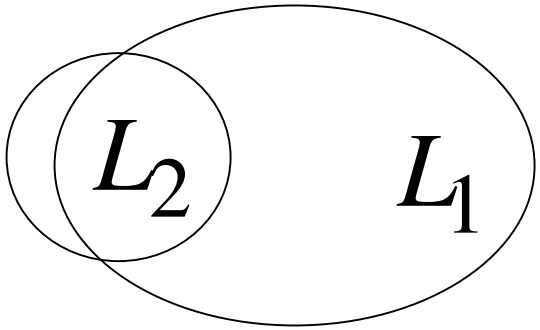
$$L_1 = L_2$$

$$(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2) \neq \emptyset$$

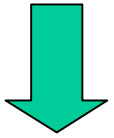


$L_1 \cap \overline{L_2} \neq \emptyset$ or $\overline{L_1} \cap L_2 \neq \emptyset$





$L_1 \not\subset L_2$ $L_2 \not\subset L_1$



$L_1 \neq L_2$

Non-regular languages

Non-regular languages

$$\{a^n b^n : \ n \geq 0\}$$

$$\{ww^R : \ w \in \{a,b\}^*\}$$

Regular languages

a^*b

$b^*c + a$

$b + c(a+b)^*$

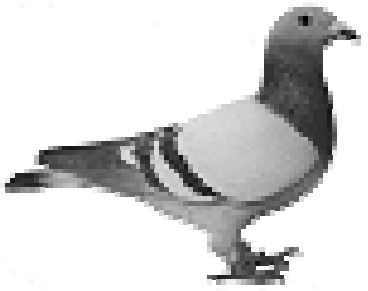
$etc...$

How can we prove that a language L is not regular?

Prove that there is no DFA that accepts L

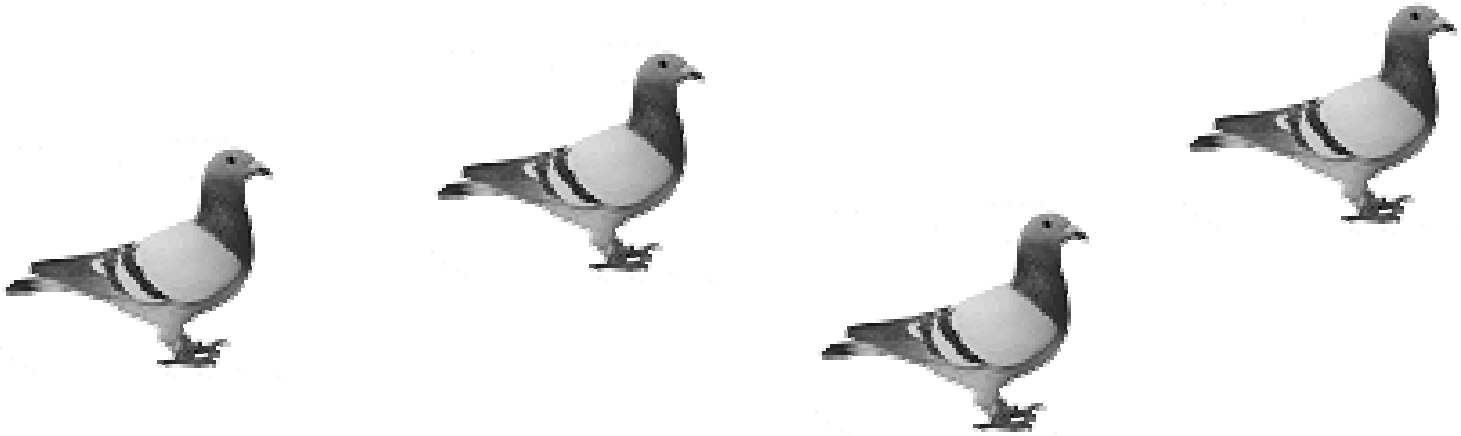
Problem: this is not easy to prove

Solution: the Pumping Lemma !!!

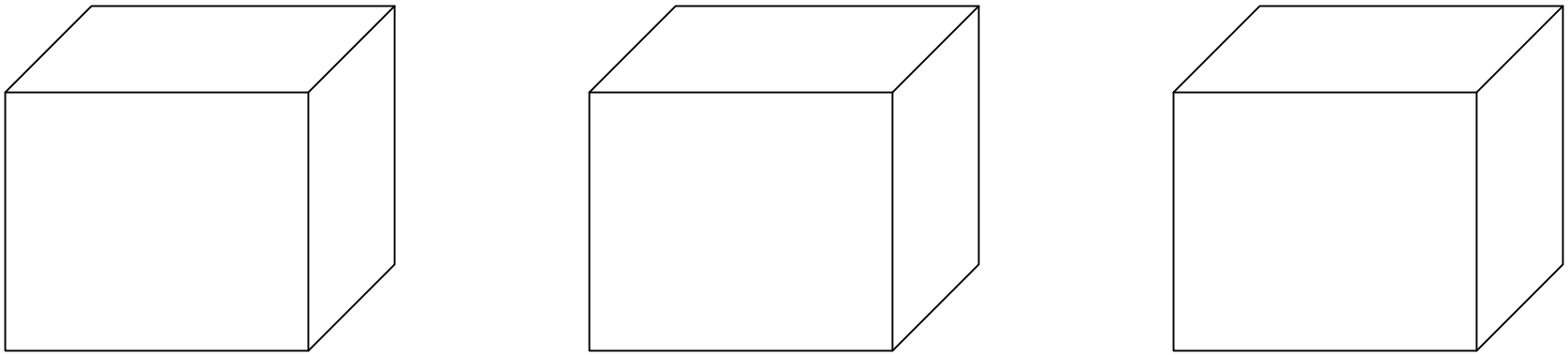


The Pigeonhole Principle

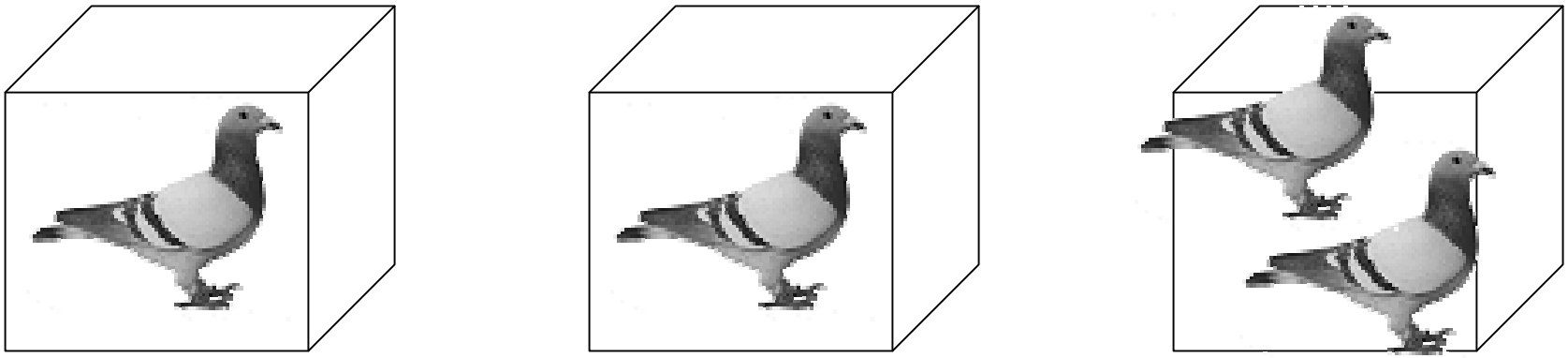
4 pigeons



3 pigeonholes

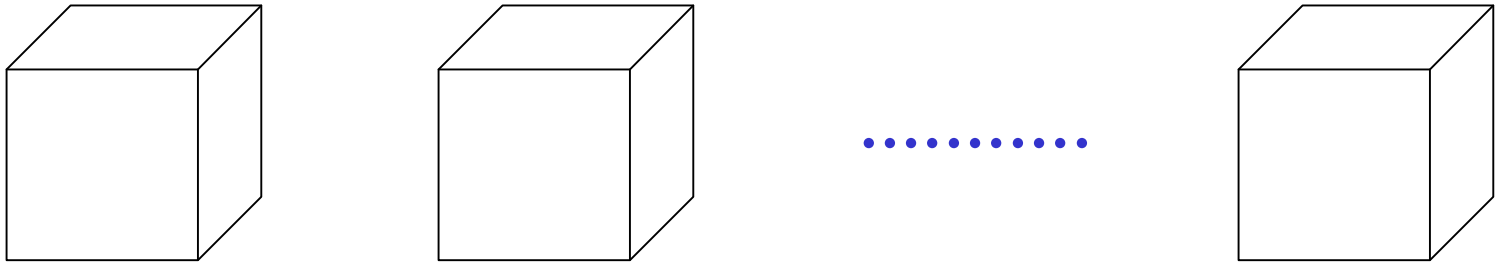


A pigeonhole must
contain at least two pigeons



n pigeons

m pigeonholes $n > m$



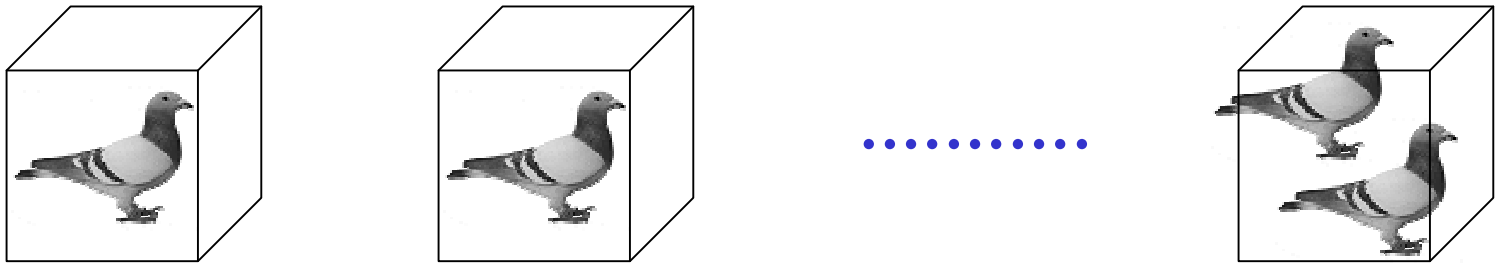
The Pigeonhole Principle

n pigeons

m pigeonholes

$n > m$

There is a pigeonhole with at least 2 pigeons

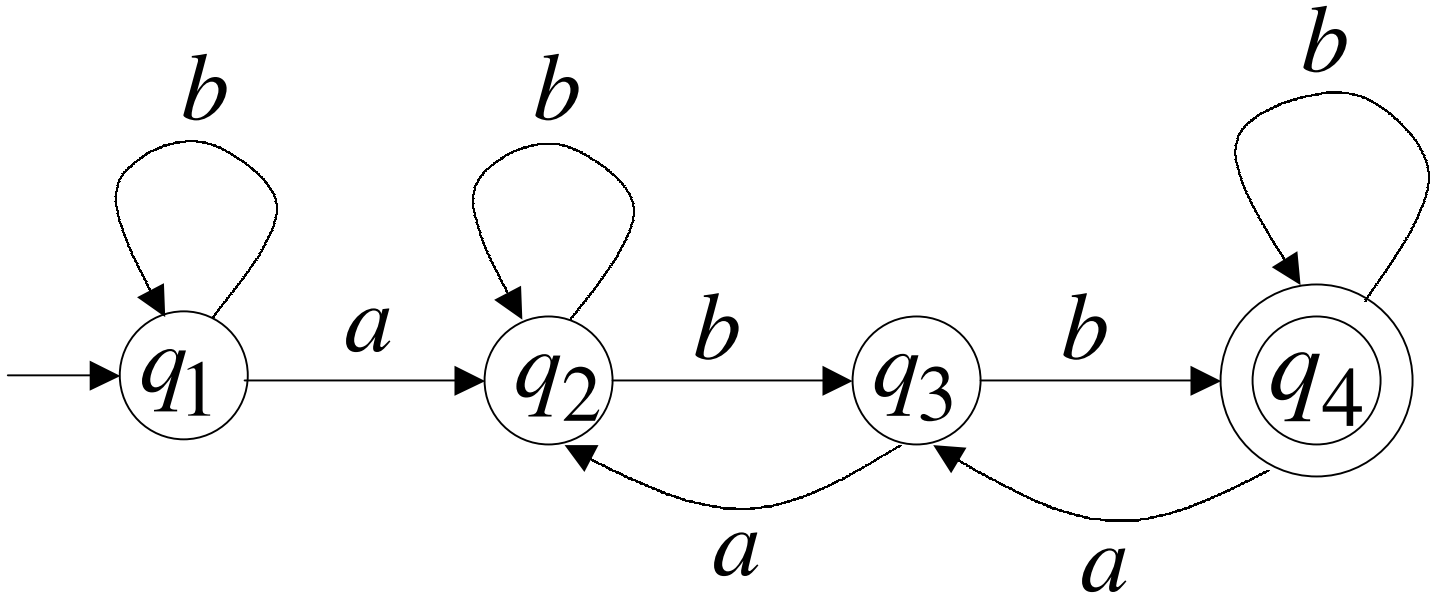


The Pigeonhole Principle

and

DFAs

DFA with 4 states

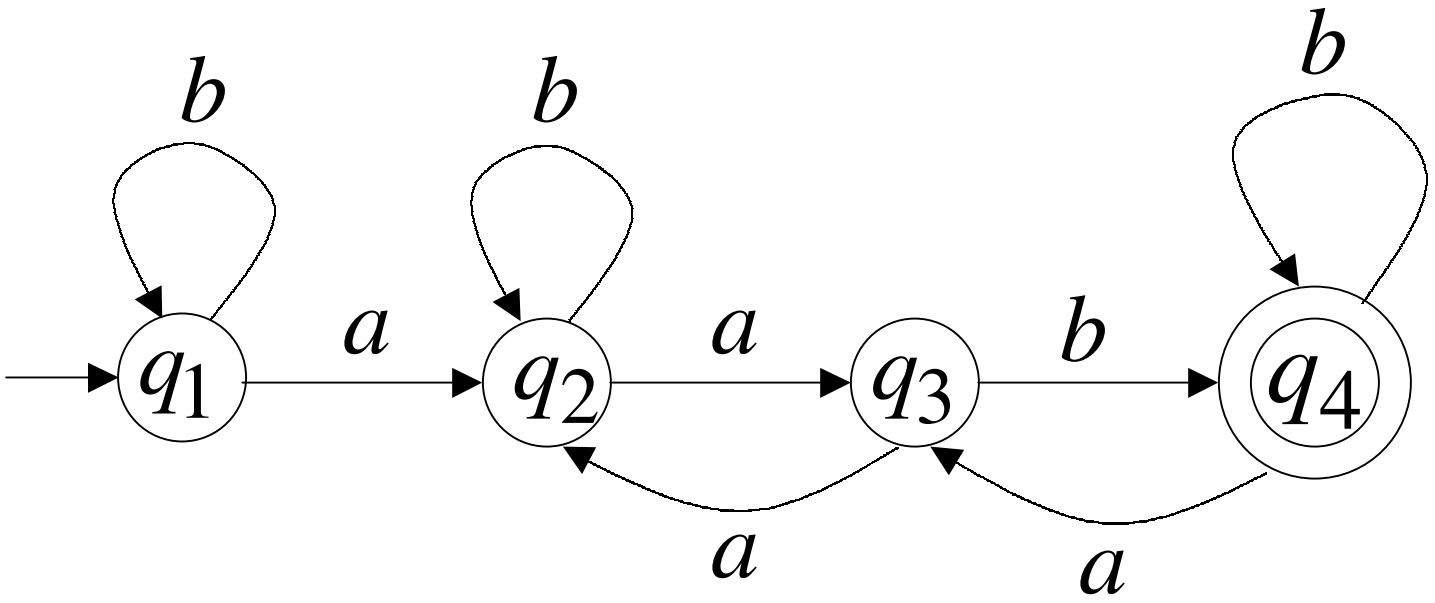


In walks of strings: a

aa

aab

no state is repeated



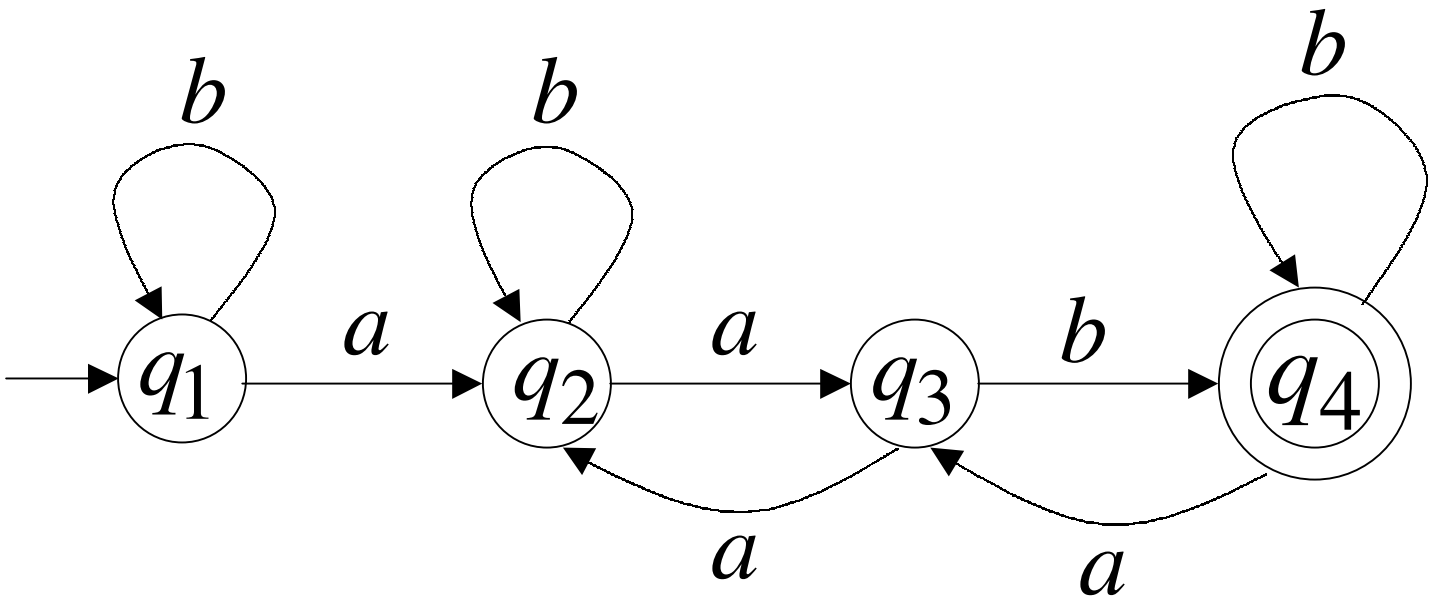
In walks of strings: $aabb$

$bbaa$

$abbabb$

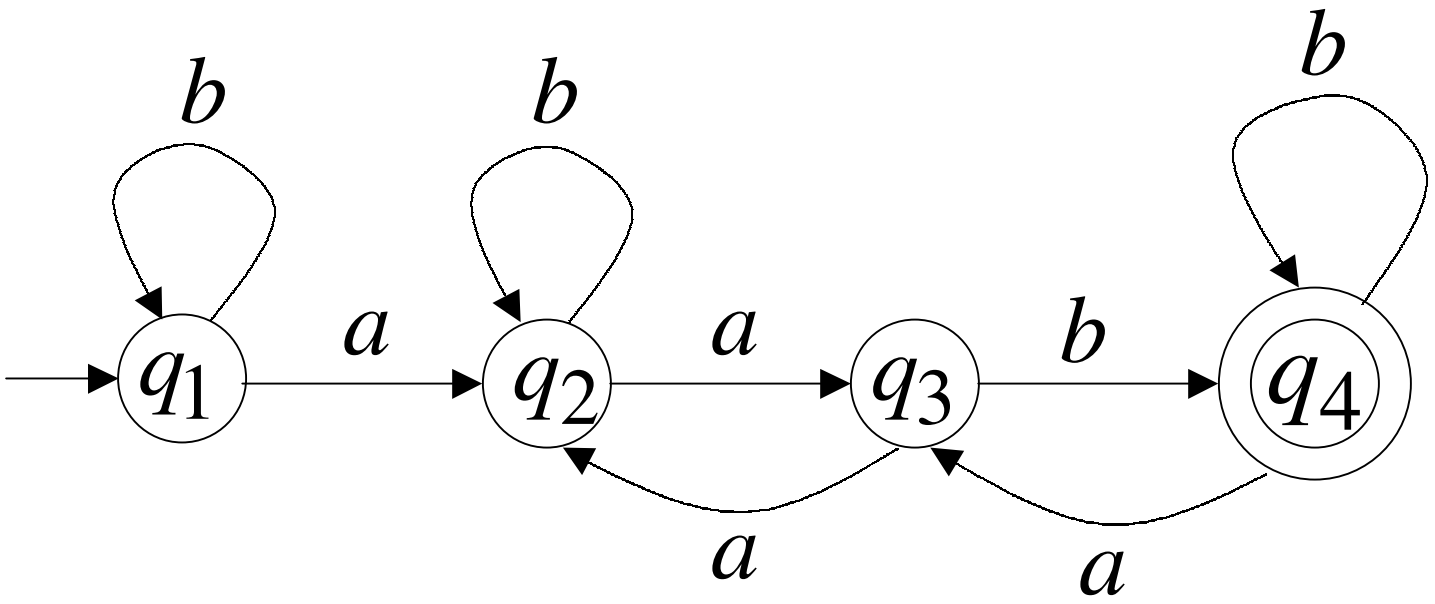
$abbbabbabb...$

a state is repeated



If the walk of string w has length $|w| \geq 4$

then a state is repeated

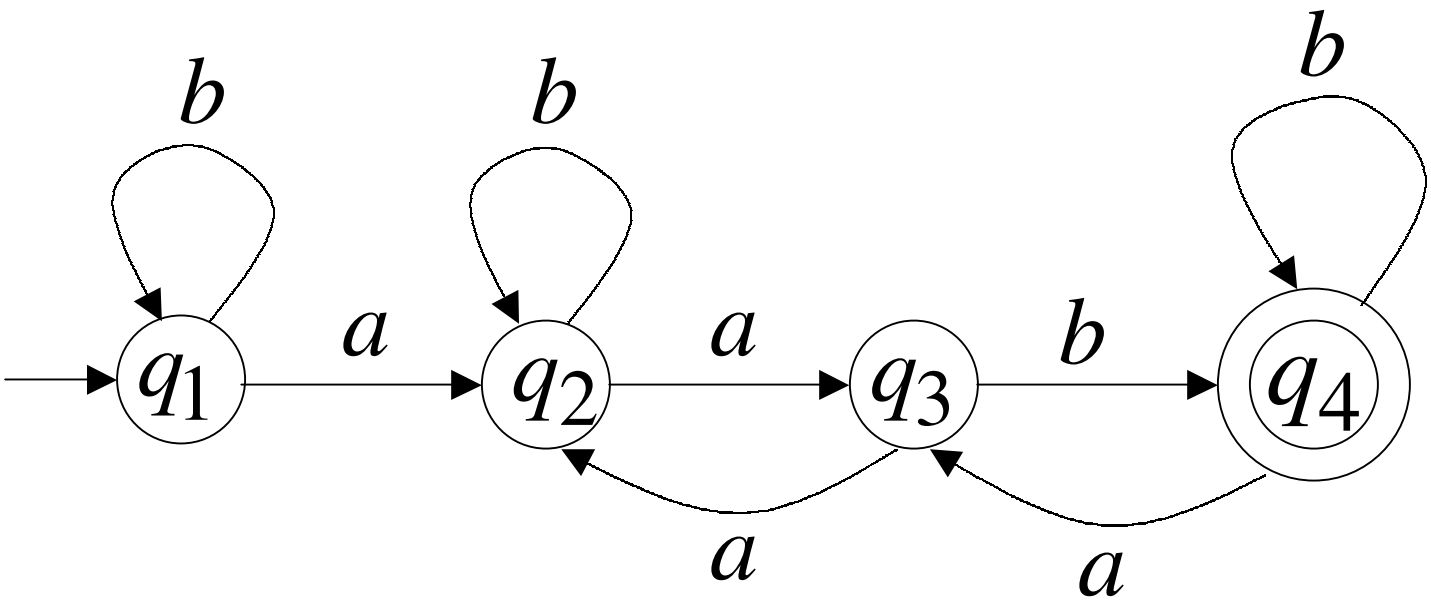


Pigeonhole principle for any DFA:

If in a walk of a string w

transitions $\geq$ states of DFA

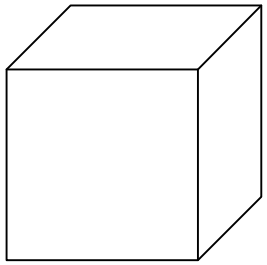
then a state is repeated

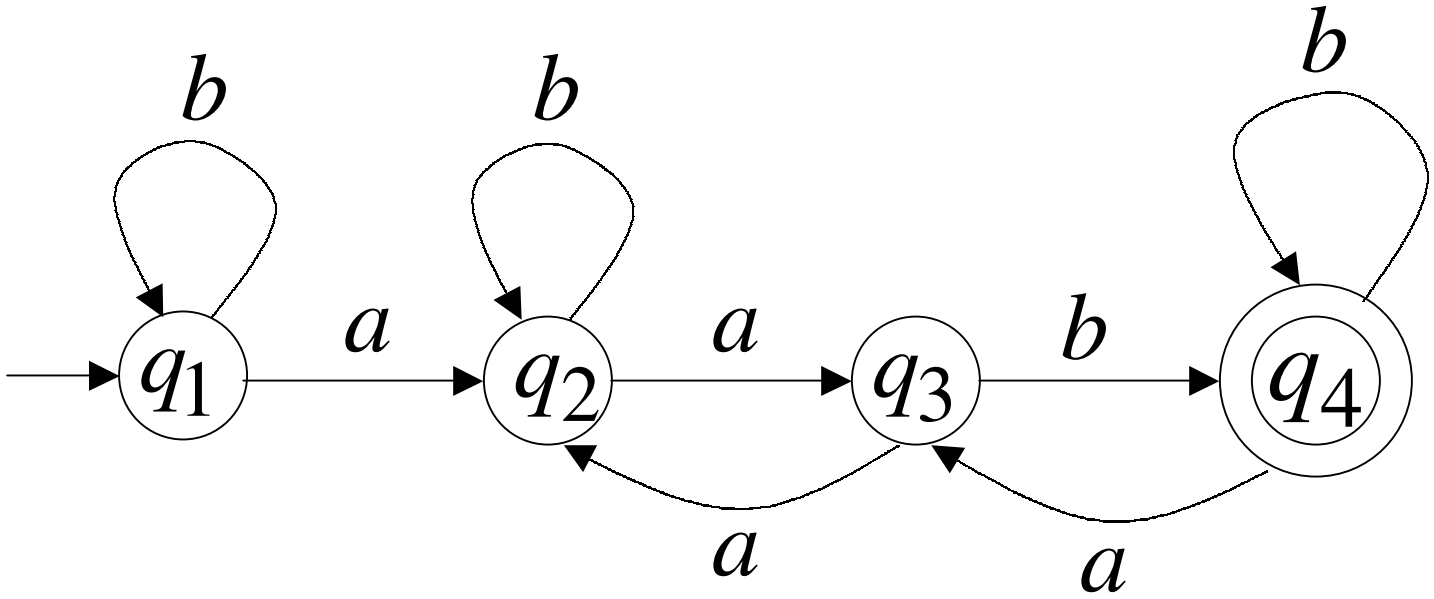


In other words for a string w:

transitions are pigeons

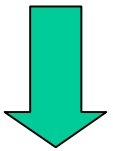
states are pigeonholes



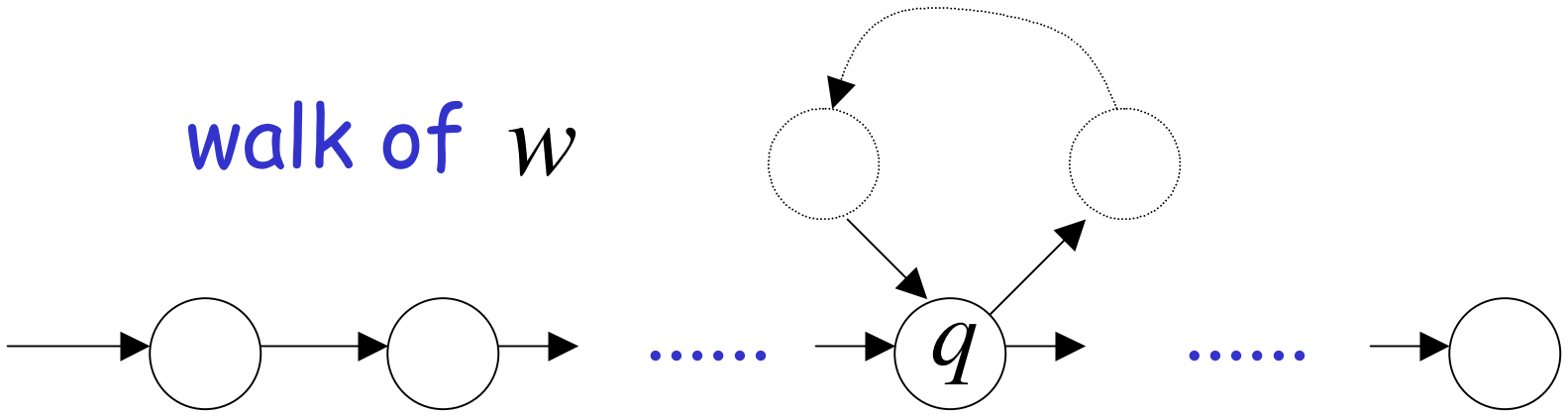


In general:

A string w has length $\geq$ number of states



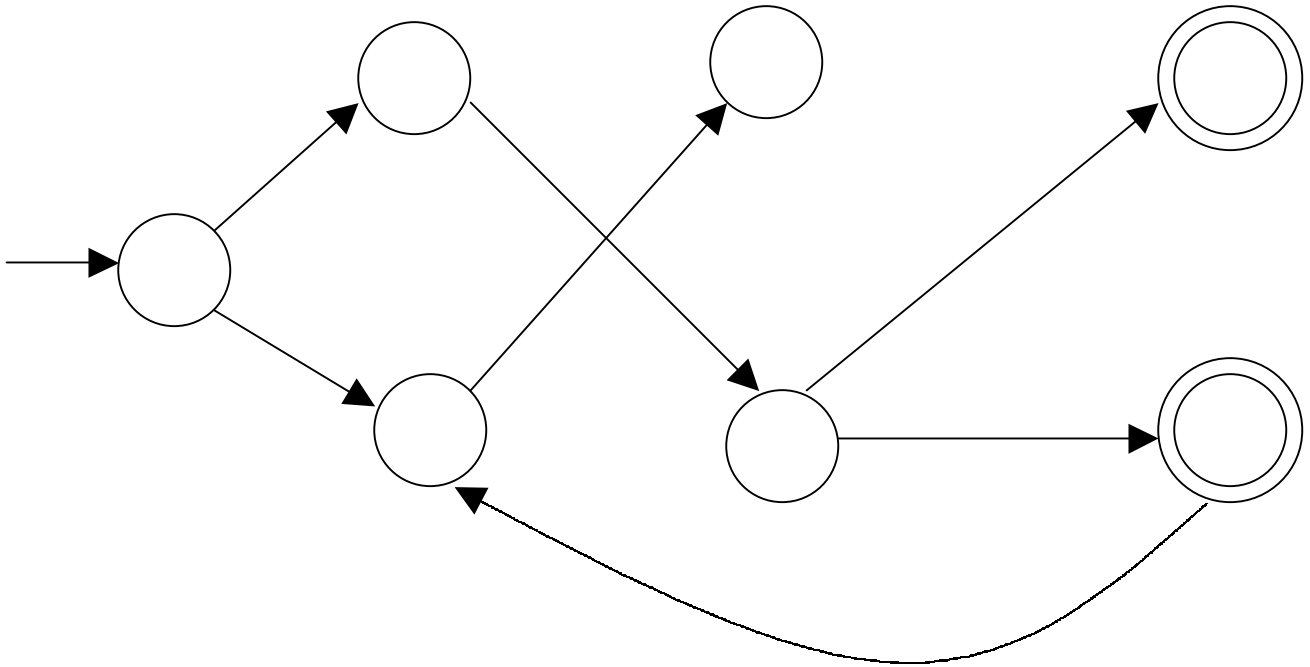
A state q must be repeated in the walk w



The Pumping Lemma

Take an infinite regular language L

DFA that accepts L



m states

Take string w with $w \in L$

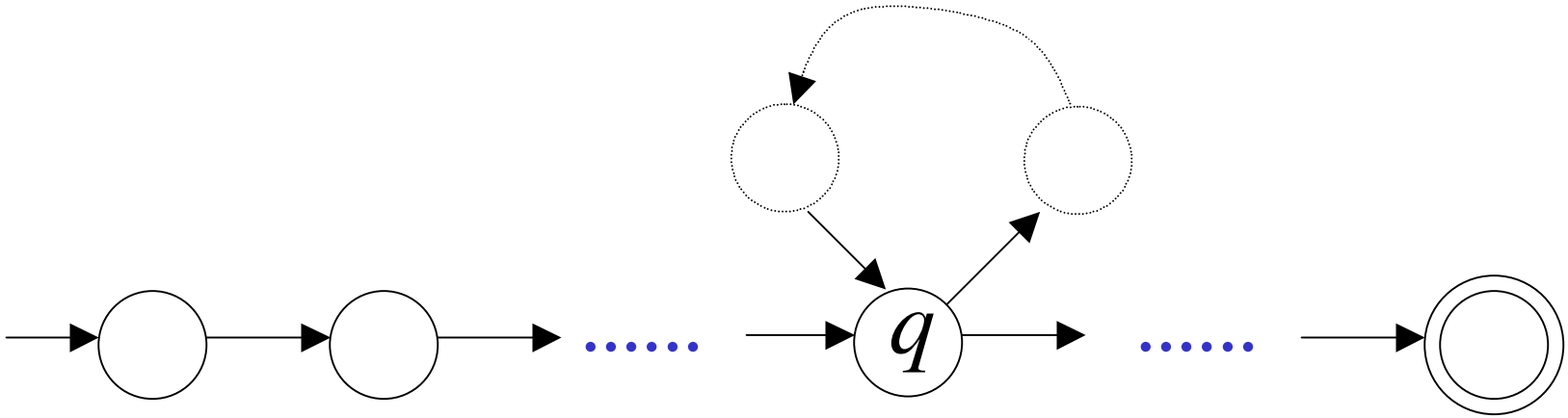
There is a walk with label w:

walk w

If string w has length $|w| \geq m$ number of states

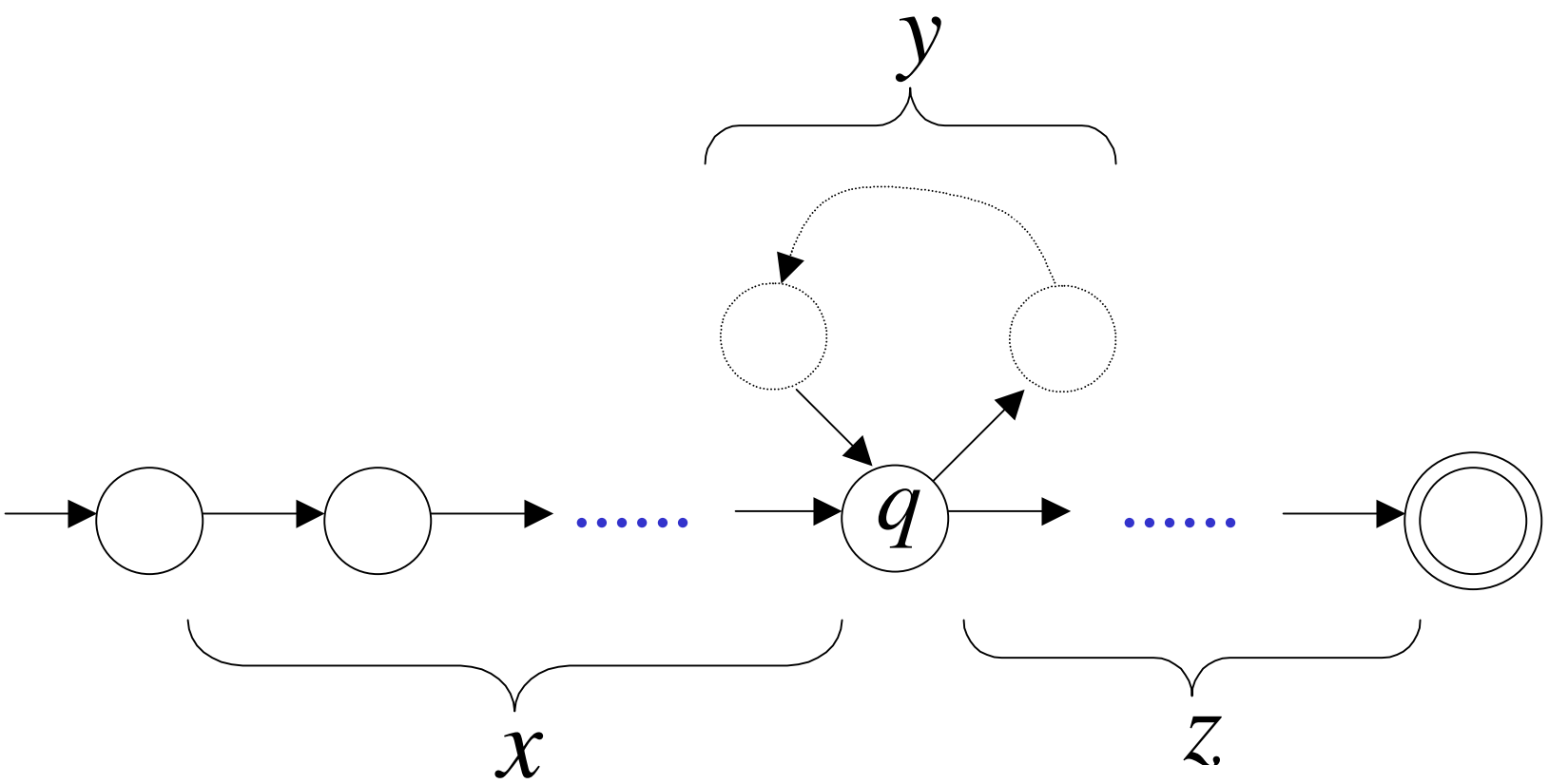
then, from the pigeonhole principle:

a state q is repeated in the walk w



walk w

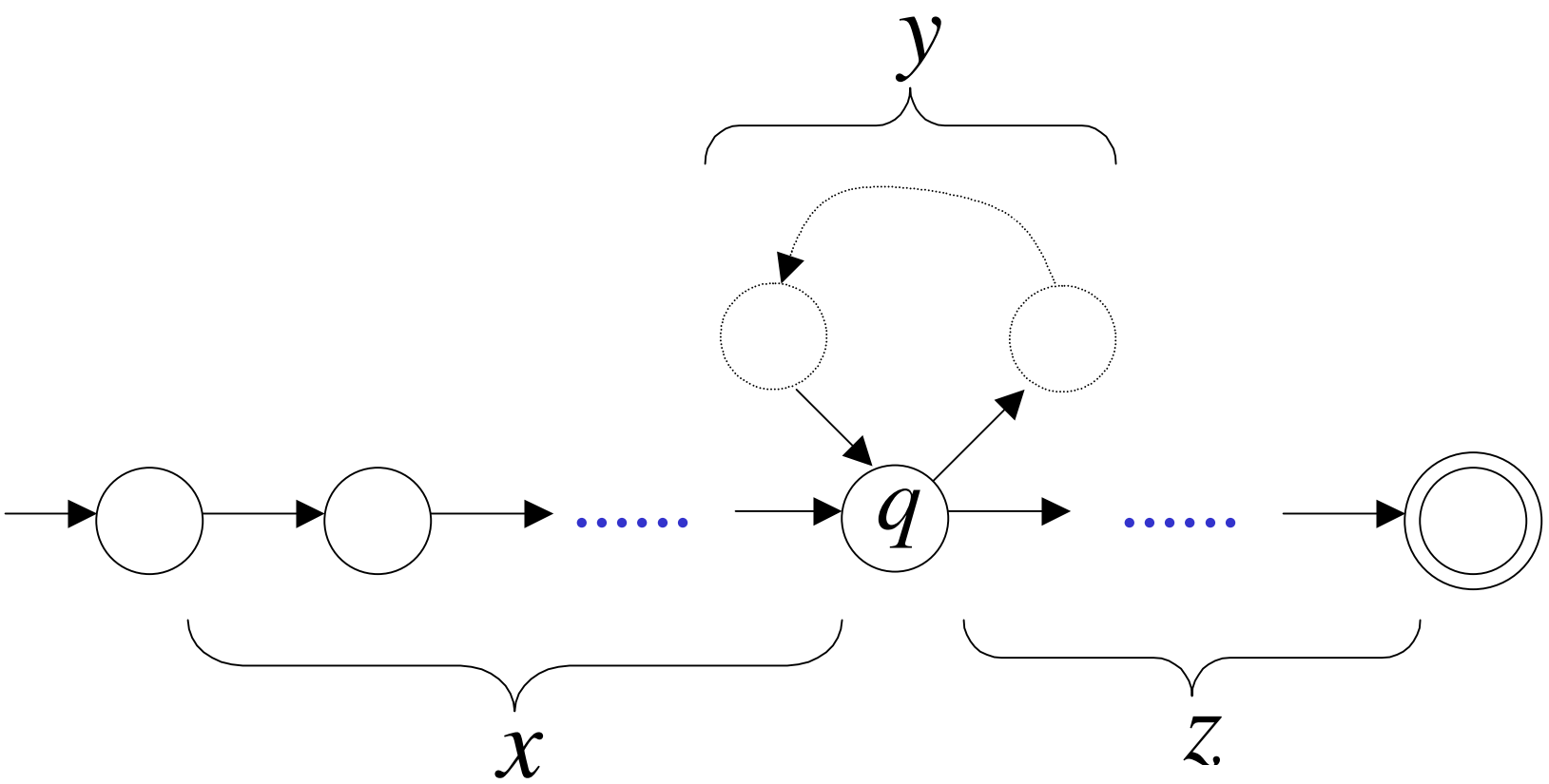
Write $w = x\ y\ z$



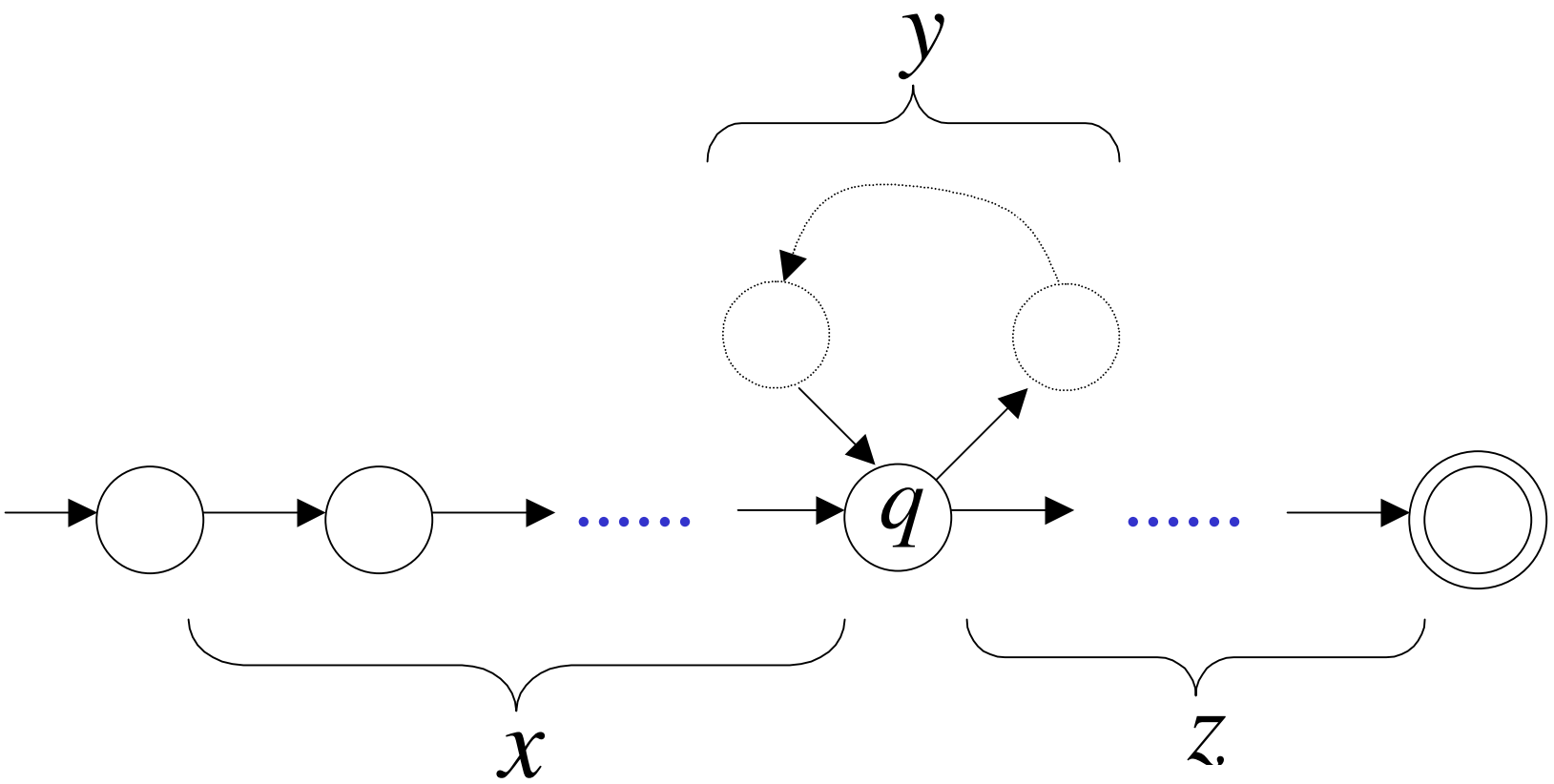
Observations:

length $|xy| \leq m$ number of states

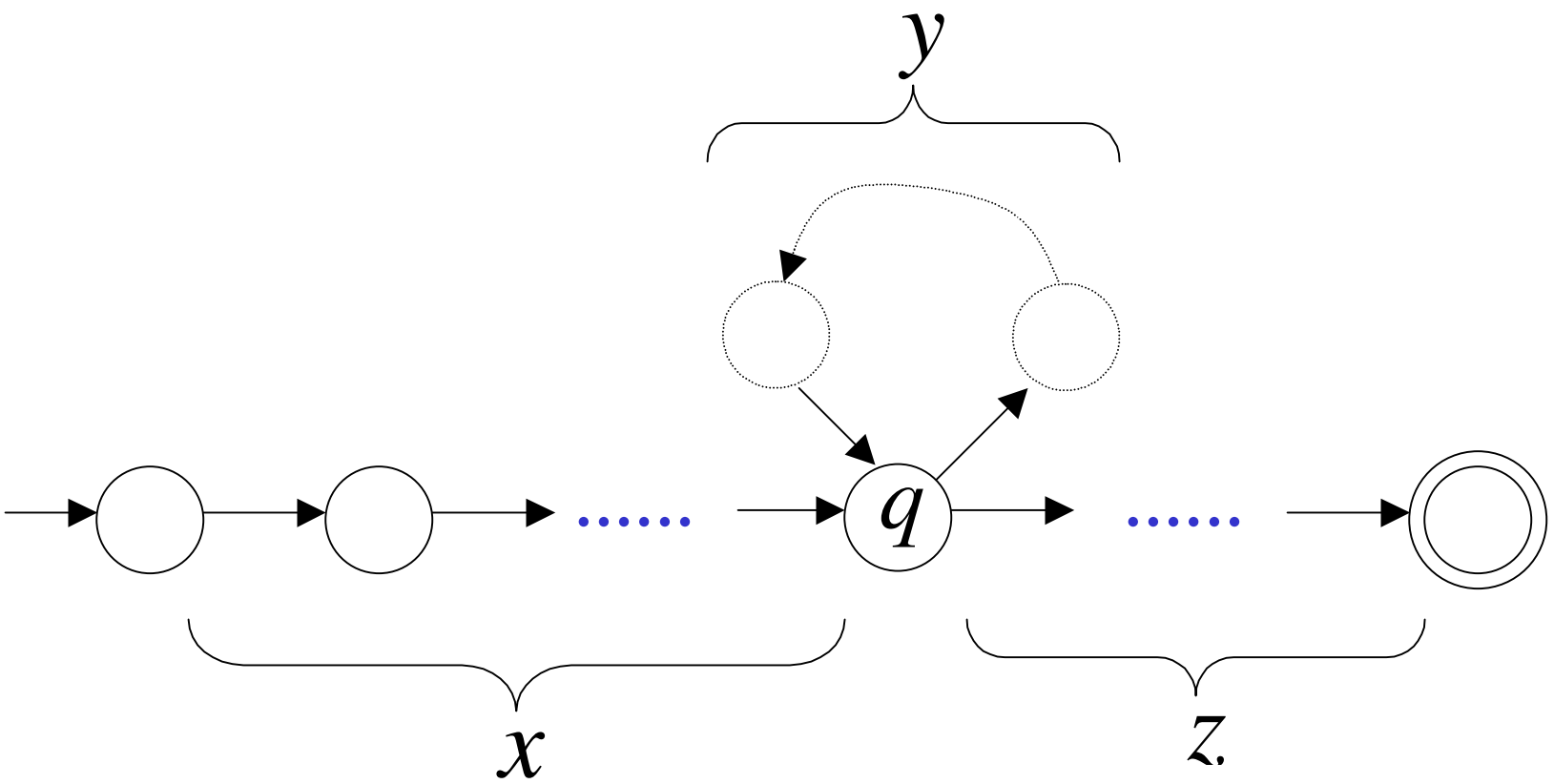
length $|y| \geq 1$



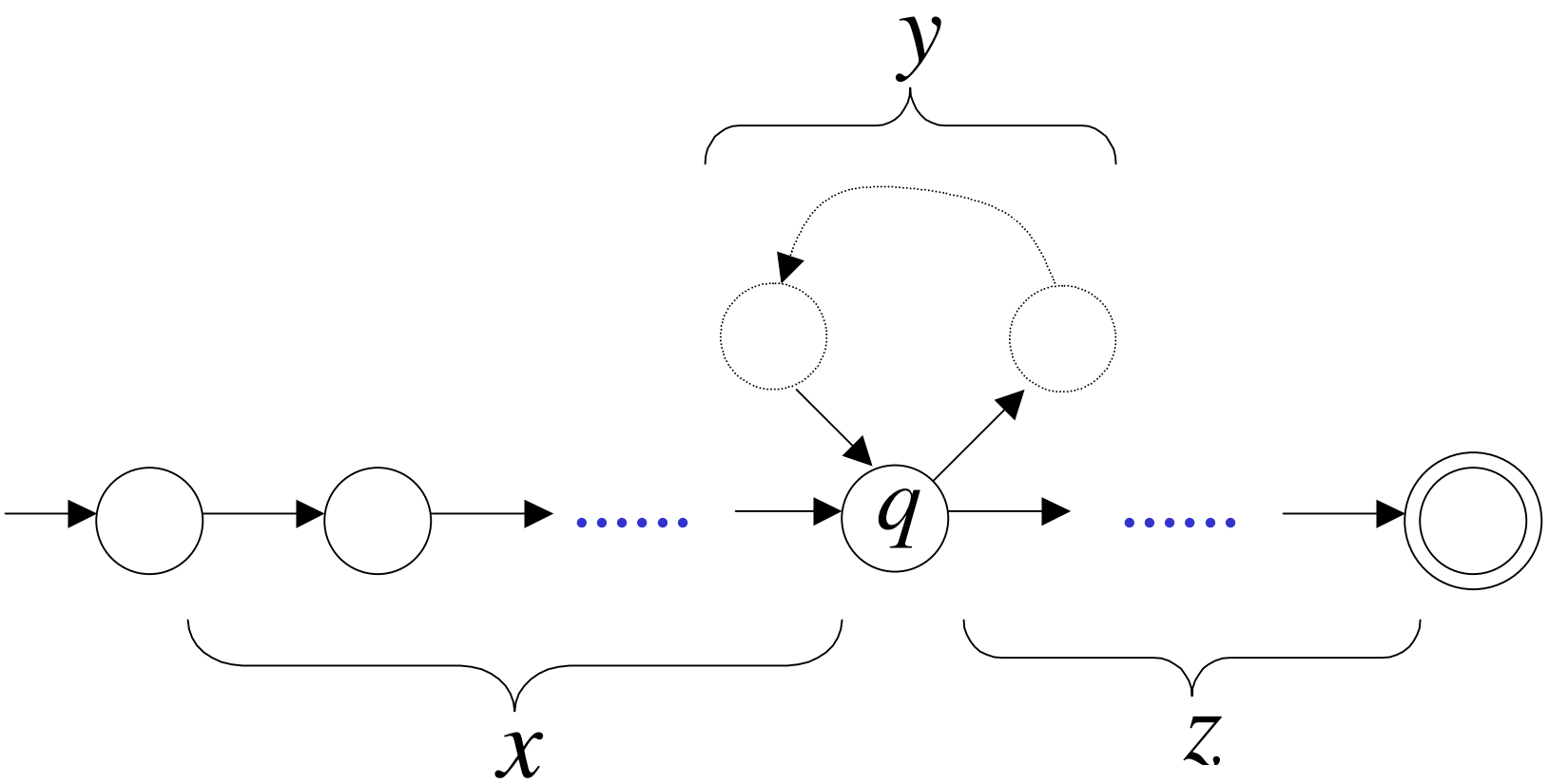
Observation: The string $x\,z$ is accepted



Observation: The string $x\ y\ y\ z$ is accepted



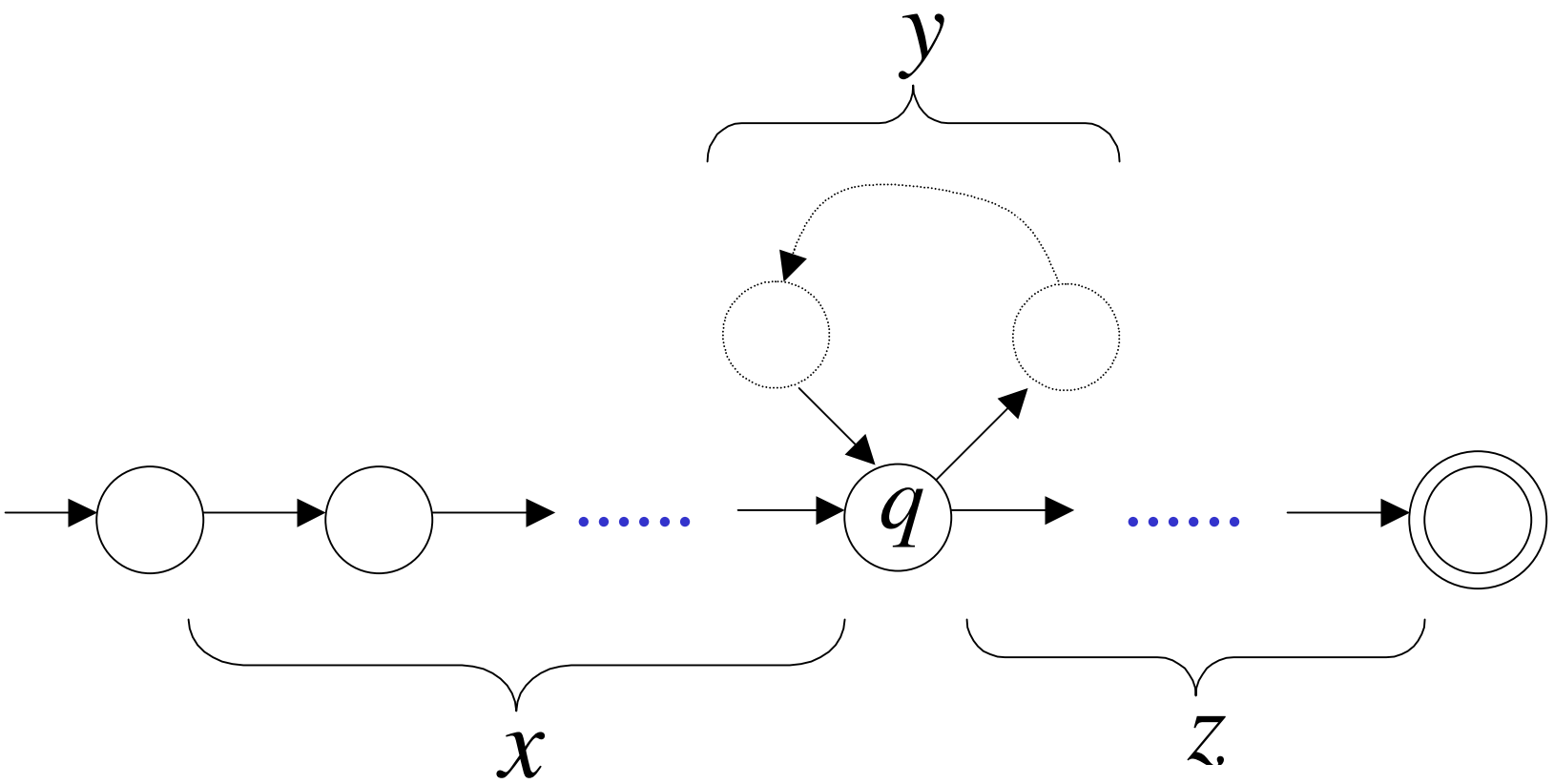
Observation: The string $x\ y\ y\ y\ z$ is accepted



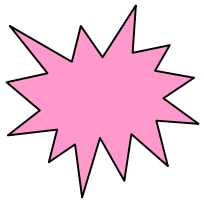
In General:

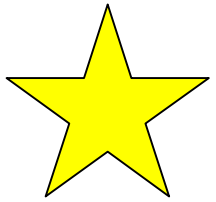
The string $x\, y^i\, z$

is accepted $i = 0, 1, 2, \ldots$

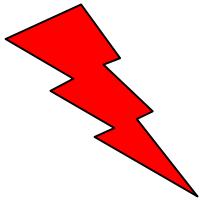


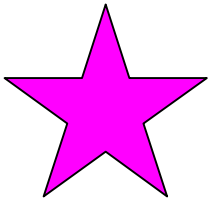
In other words, we described:





The Pumping Lemma !!!





The Pumping Lemma:

- Given a infinite regular language L
- there exists an integer m
- for any string $w \in L$ with length $|w| \geq m$
- we can write $w = x\ y\ z$
- with $|x\ y| \leq m$ and $|y| \geq 1$
- such that: $x\ y^i\ z \in L \qquad i = 0, 1, 2, \ldots$

Applications

of

the Pumping Lemma

Theorem: The language $L = \{a^n b^n : n \geq 0\}$

is not regular

Proof: Use the Pumping Lemma

$$L = \{a^n b^n : n \geq 0\}$$

Assume for contradiction
that L is a regular language

Since L is infinite
we can apply the Pumping Lemma

$$L = \{a^n b^n : n \geq 0\}$$

Let m be the integer in the Pumping Lemma

Pick a string w such that: $w \in L$

length $|w| \geq m$

Example: pick $w = a^m b^m$

Write: $a^m b^m = x\ y\ z$

From the Pumping Lemma

it must be that: length $|x\ y| \le m, \quad |y| \ge 1$

Therefore: $$a^m b^m = \overbrace{\underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\ldots a}^{m}\overbrace{\underbrace{b\ldots b}_{z}}^{m}$$

$$y = a^k, \quad k \ge 1$$

We have: $x\ y\ z = a^m b^m \qquad y = a^k, \quad k \geq 1$

From the Pumping Lemma: $x\ y^i\ z \ \in \ L$

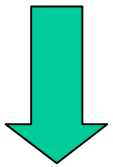
$i = 0, 1, 2, \ldots$

Thus: $x\ y^2\ z \ \in \ L$

$$x\ y^2\ z \ = \ x\ y\ y\ z \ = \ a^{m+k} b^m \ \in \ L$$

Therefore: $a^{m+k}b^m \in L$

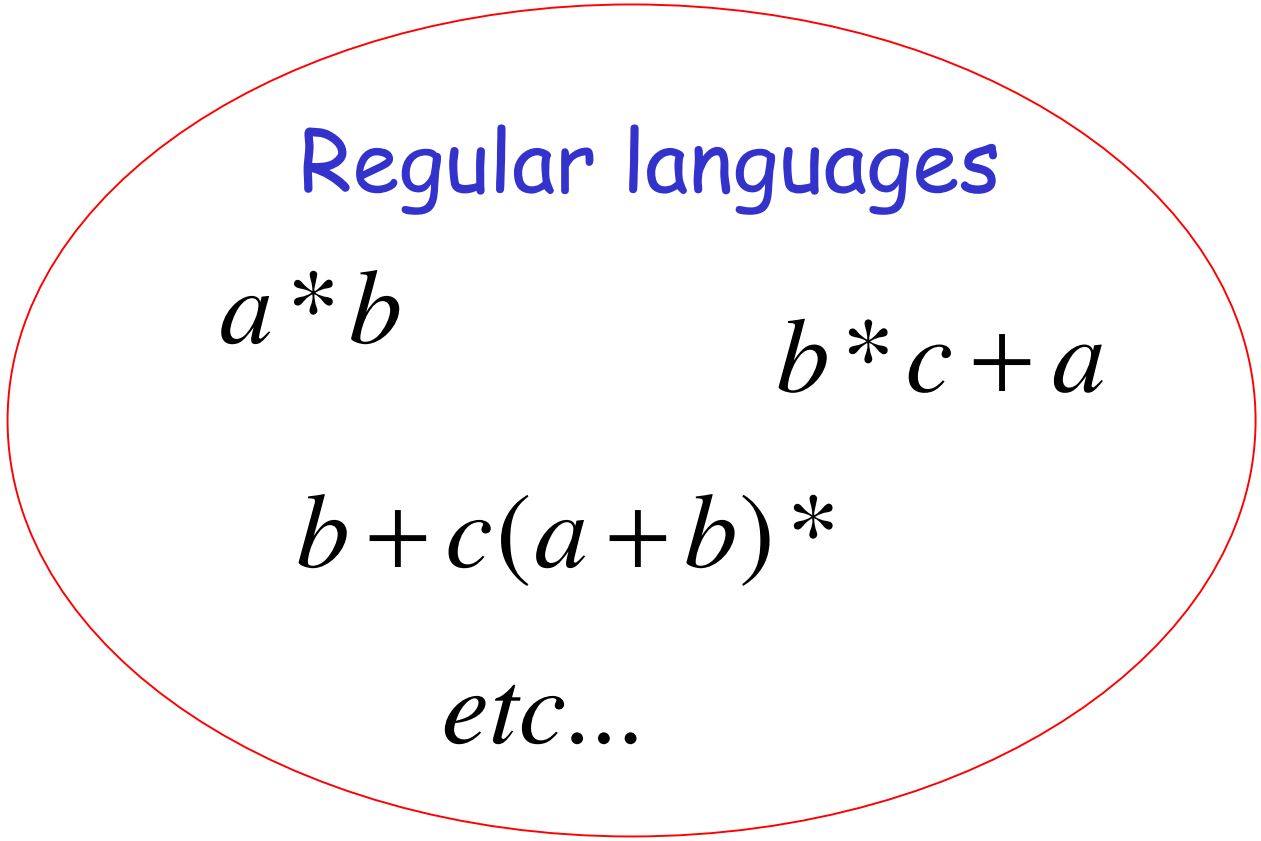
BUT: $L = \{a^n b^n : n \geq 0\}$



$a^{m+k}b^m \notin L$

CONTRADICTION!!!

Therefore: Our assumption that L is a regular language is not true

Conclusion: L is not a regular language

Non-regular languages $\{a^n b^n : \ n \geq 0\}$