Module 2

Formal Languages

Non-Regular Languages

A formal language is often defined by means of a formal grammar such as a regular grammar or context-free grammar.

- L is a regular language.
- There is a DFA D such that $\mathscr{L}(D) = L$.
- There is an NFA N such that $\mathscr{L}(N) = L$.
- There is a regular expression R such that $\mathscr{L}(R) = L$.

- Automata theory is a branch of the theory of computation. It deals with the study of abstract machines and their capacities for computation.
- An abstract machine is called the automata. It includes the design and analysis of automata, which are mathematical models that can perform computations on strings of symbols according to a set of rules.
- Theory of computation is the branch of computer science that studies the nature and ranges of computation. It includes analysis and design of algorithms computation systems, formal languages, automata theory, compatibility theory, and complexity theory.
- In this Automata Tutorial, you'll learn all the basic to advanced topics like Regular languages and finite automata, Context free Grammar and Context-free language, turning machines, etc.

- Regular languages correspond to problems that can be solved with finite memory.

- Nonregular languages correspond to problems that cannot be solved with finite memory bcos it cannot remember the symbols it had read.
- E.g. exact number pf a's and b's….

- The theorem which is used to identify, if a language is regular or not is based on pigeon hole principle- called as pumping lemma.

Non-regular languages

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

$$\{vv^R : \ v \in \{a,b\}^*\}$$

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

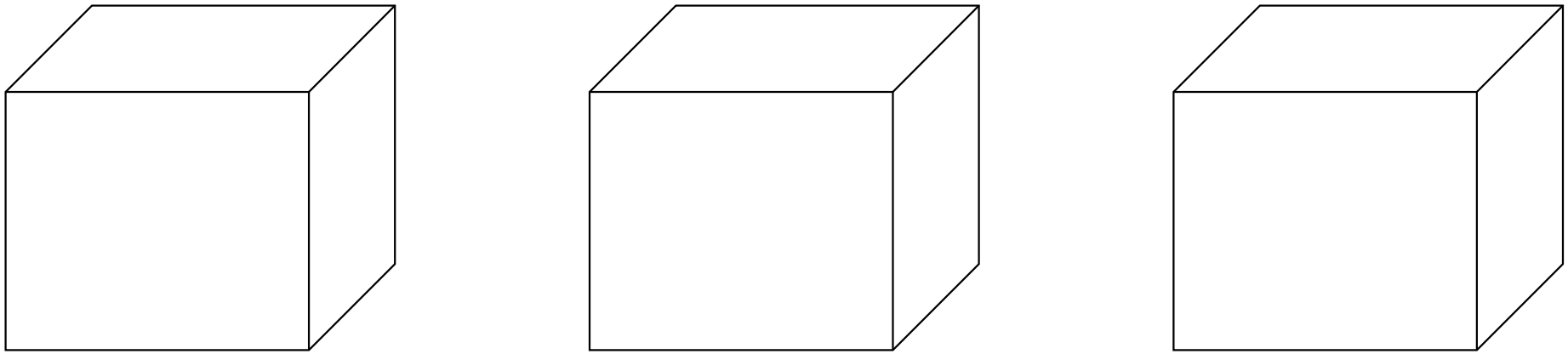
Problem: this is not easy to prove

Solution: the Pumping Lemma !!!

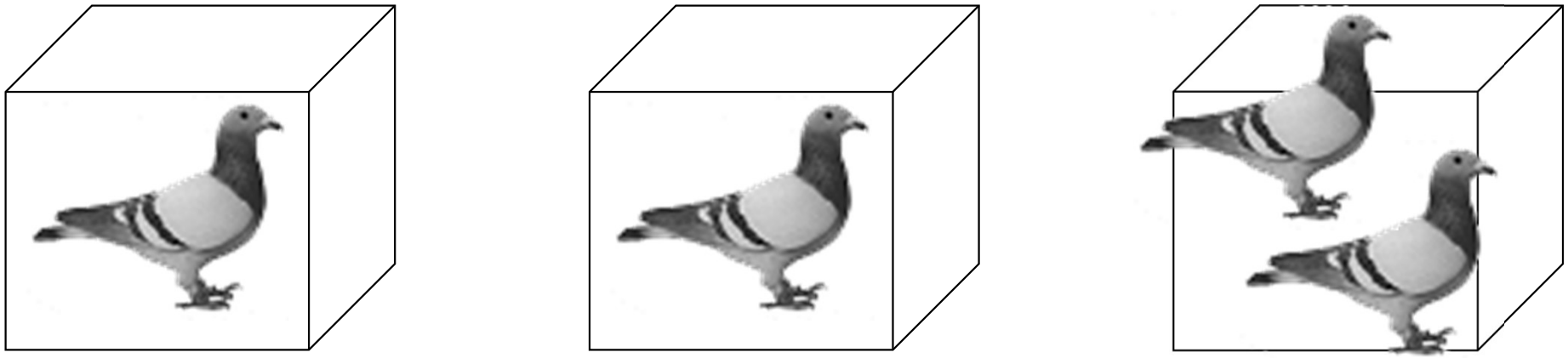
The Pigeonhole Principle

4 pigeons

3 pigeonholes

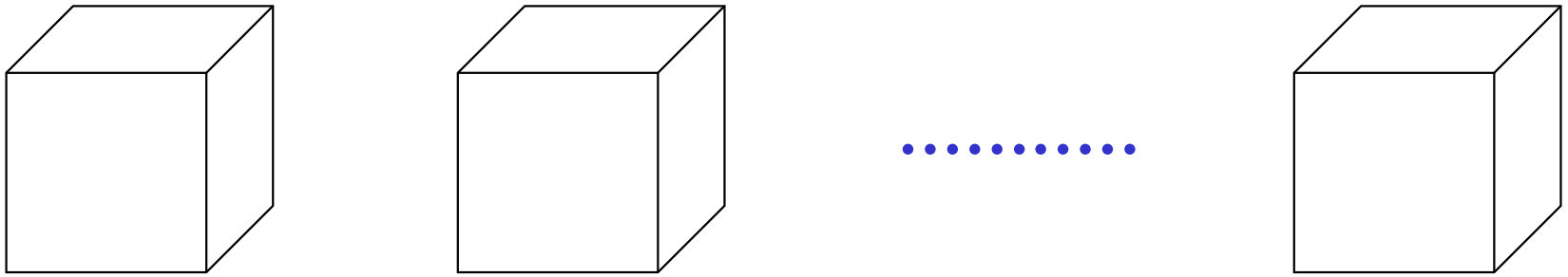


A pigeonhole must
contain at least two pigeons



n pigeons

m pigeonholes $n > m$



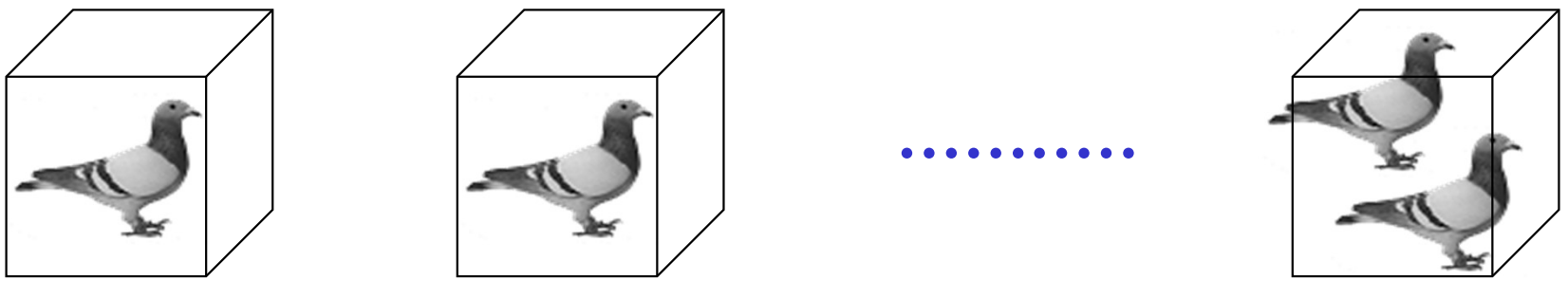
The Pigeonhole Principle

n pigeons

m pigeonholes

$n > m$

There is a pigeonhole with at least 2 pigeons

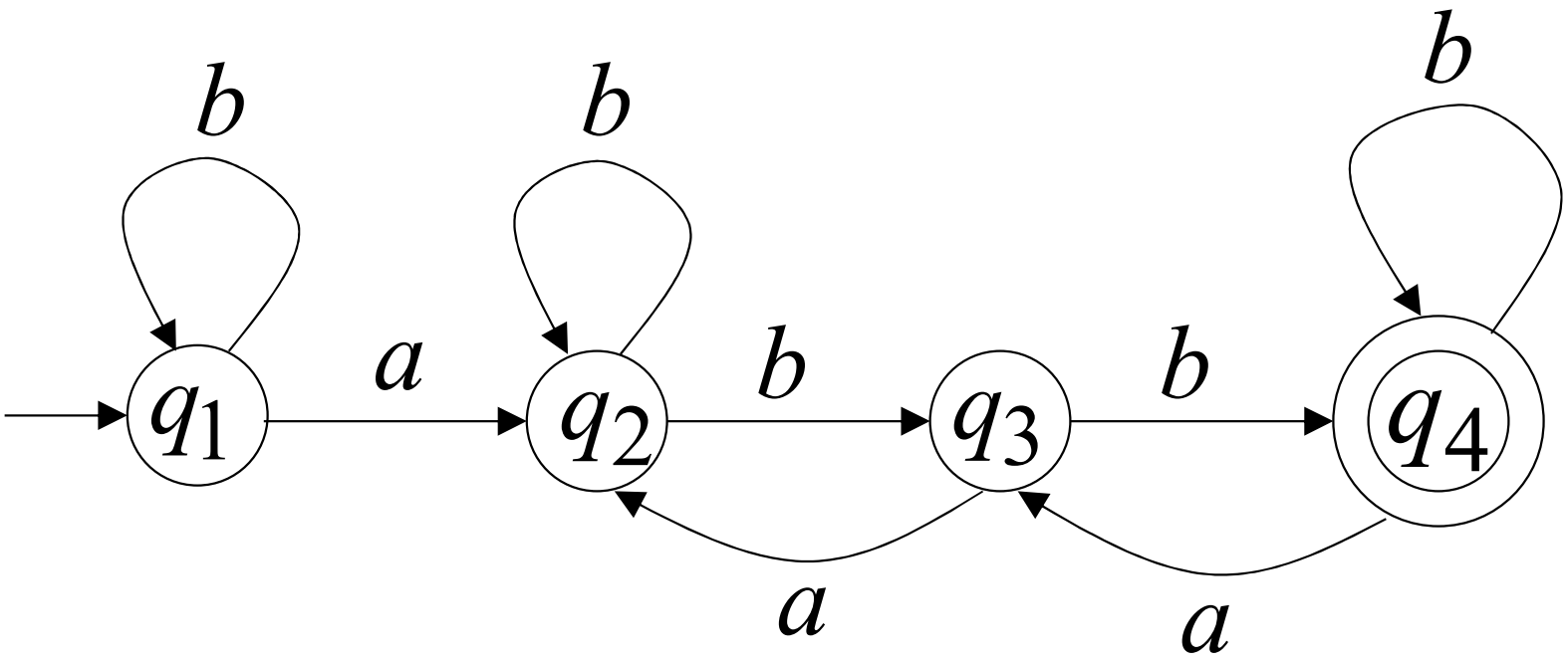


The Pigeonhole Principle

and

DFAs

DFA with 4 states

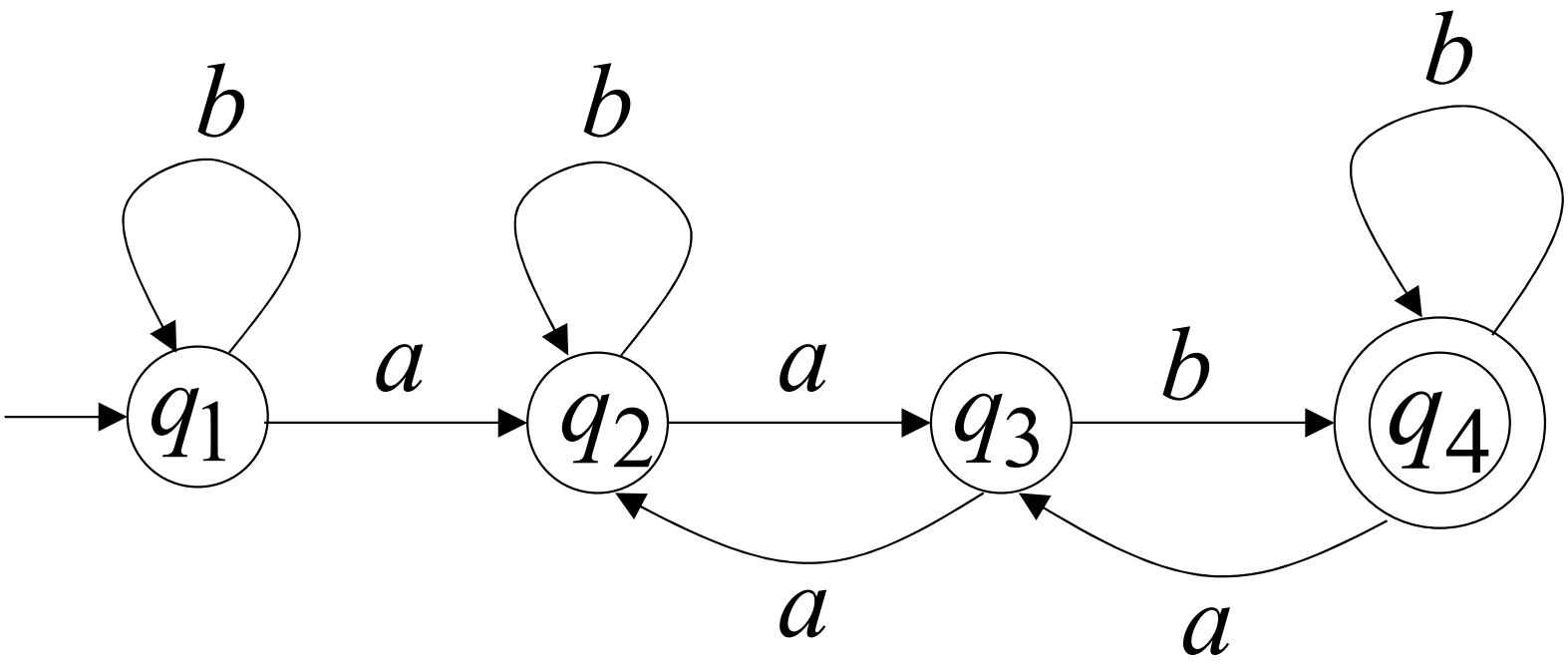


In walks of strings: a

aa

aab

no state is repeated



In walks of strings:

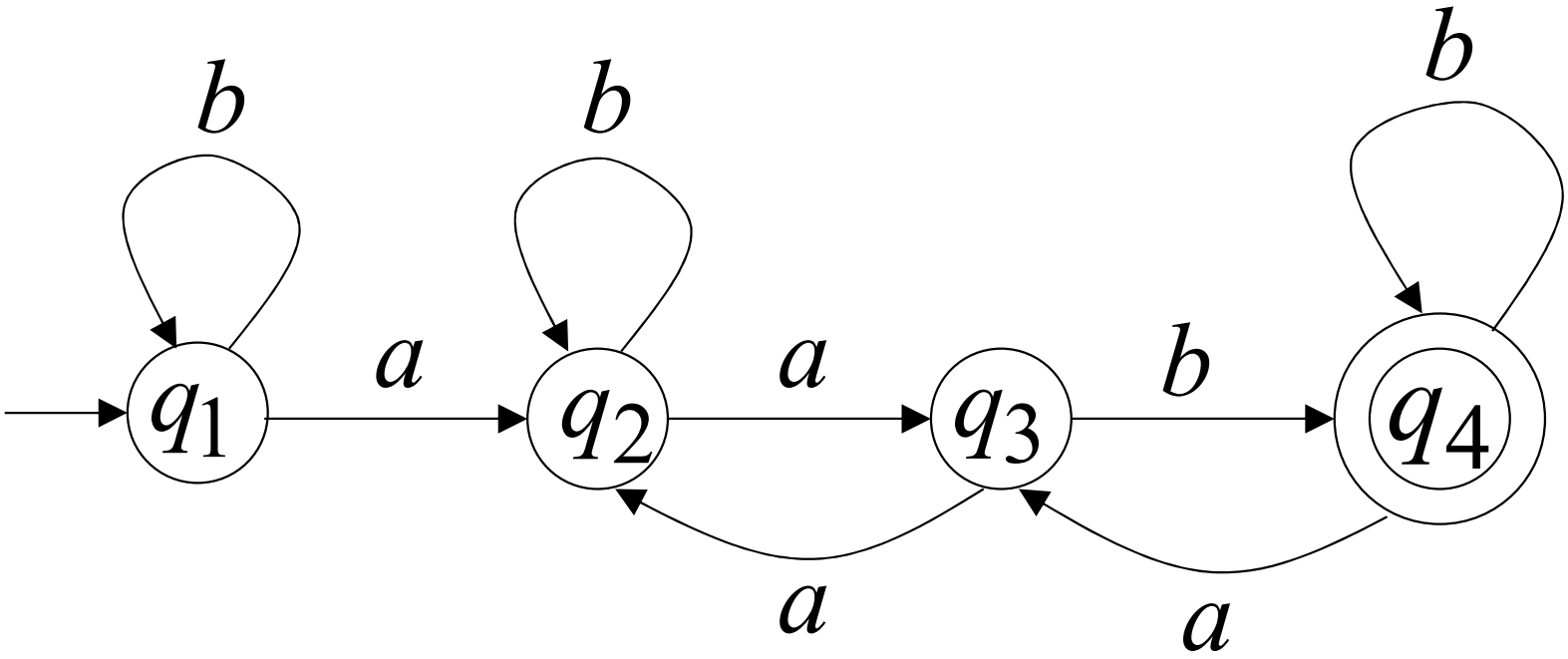
$aabb$

$bbaab$

$abbabb$

$abbbabbabb...$

a state is repeated

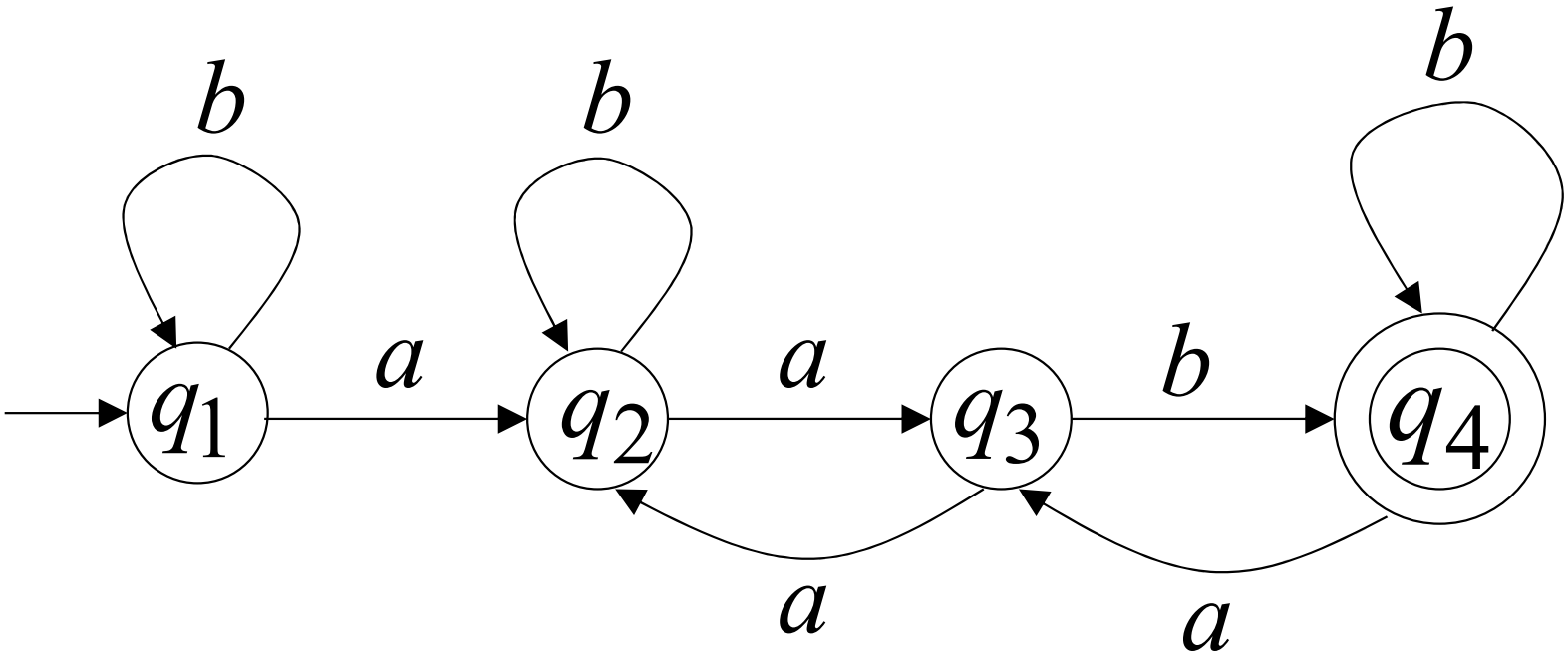


For the given DFA….

If string w has length $|w| \geq 4$:

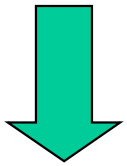
Then the transitions of string w are more than the states of the DFA

Thus, a state must be repeated

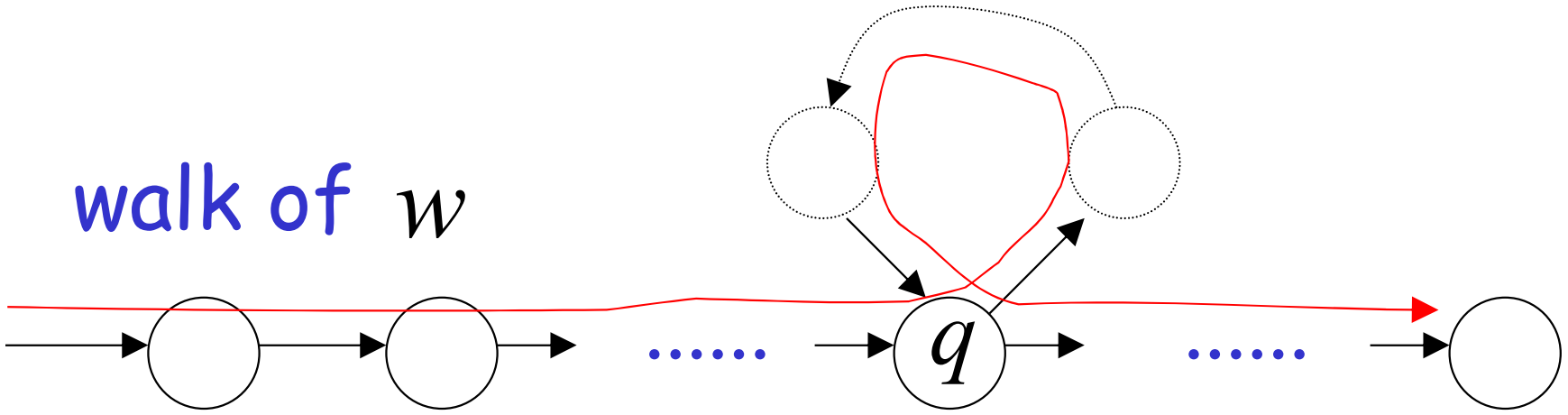


In general, for any DFA:

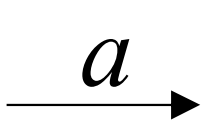
String w has length $\geq$ number of states



A state q must be repeated in the walk of w

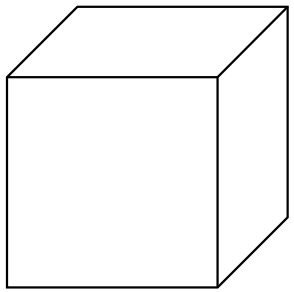


In other words for a string w:

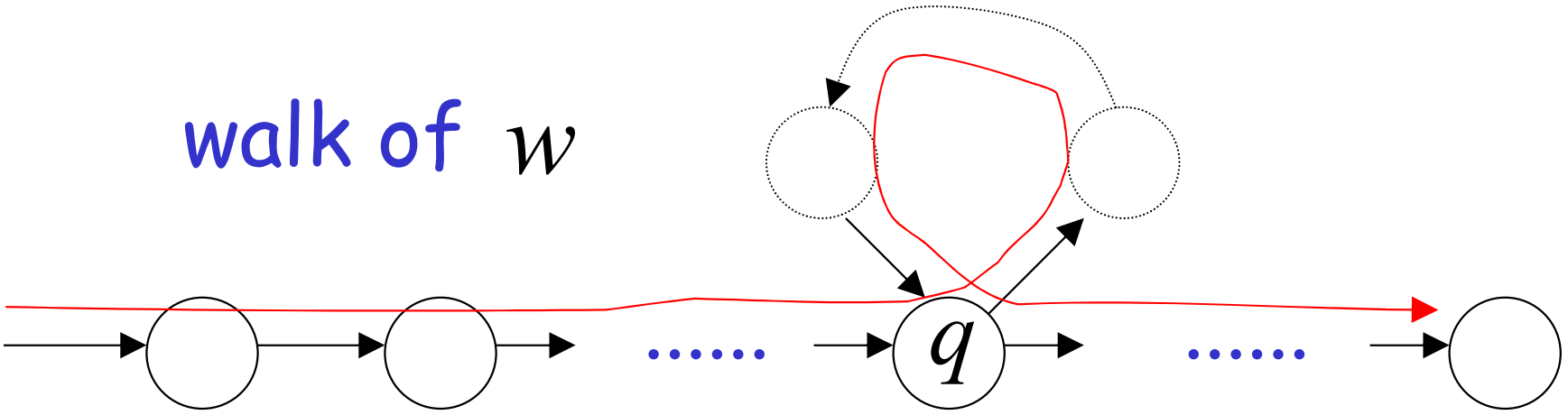


transitions are pigeons

states are pigeonholes



walk of w



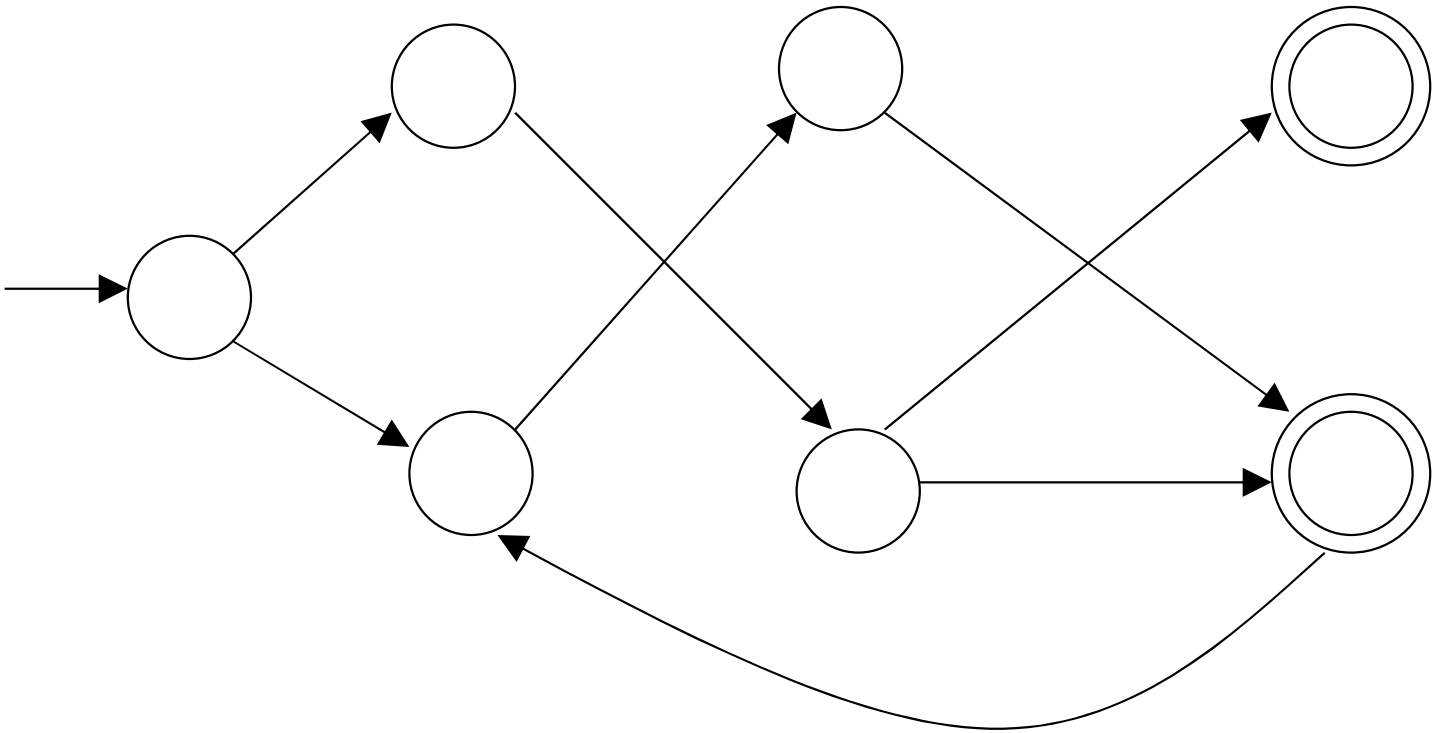
Repeated state

The Pumping Lemma

Take an infinite regular language L

Eg. a*, aba*, a* b a* etc…

There exists a DFA that accepts L



m states

Take string w with $w \in L$

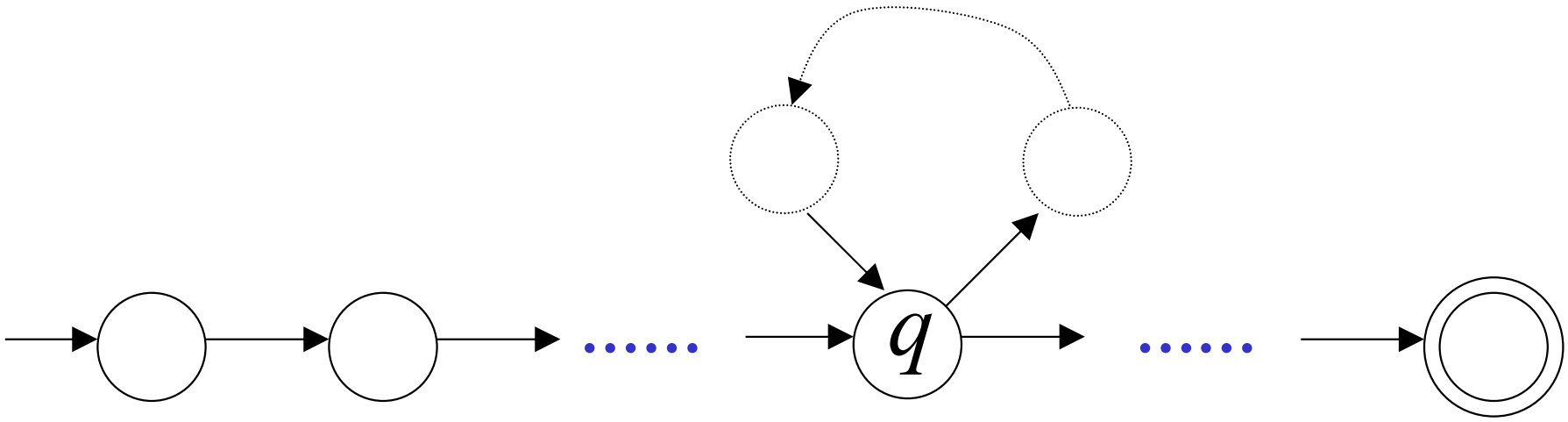
There is a walk with label w:

walk w

If string w has length $|w| \geq m$ no. of states

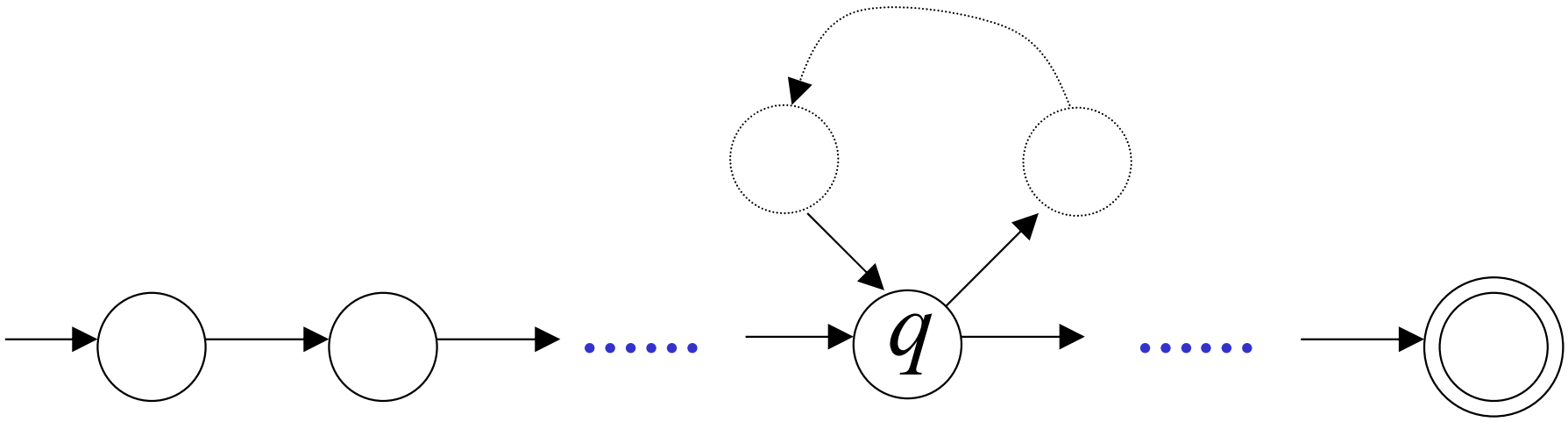
then, from the pigeonhole principle:

a state is repeated in the walk w

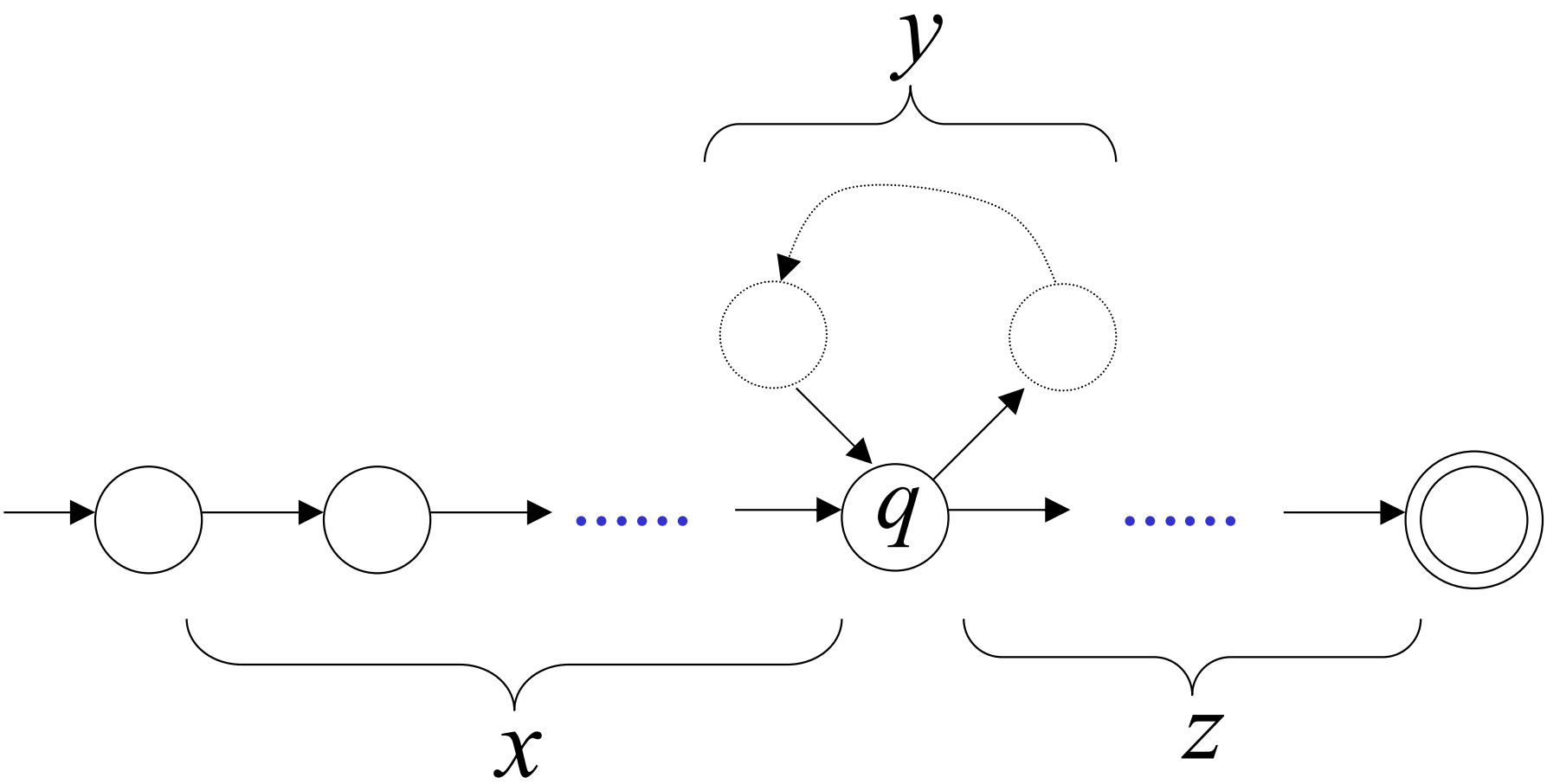


walk w

Let q be the first state repeated in the walk of w

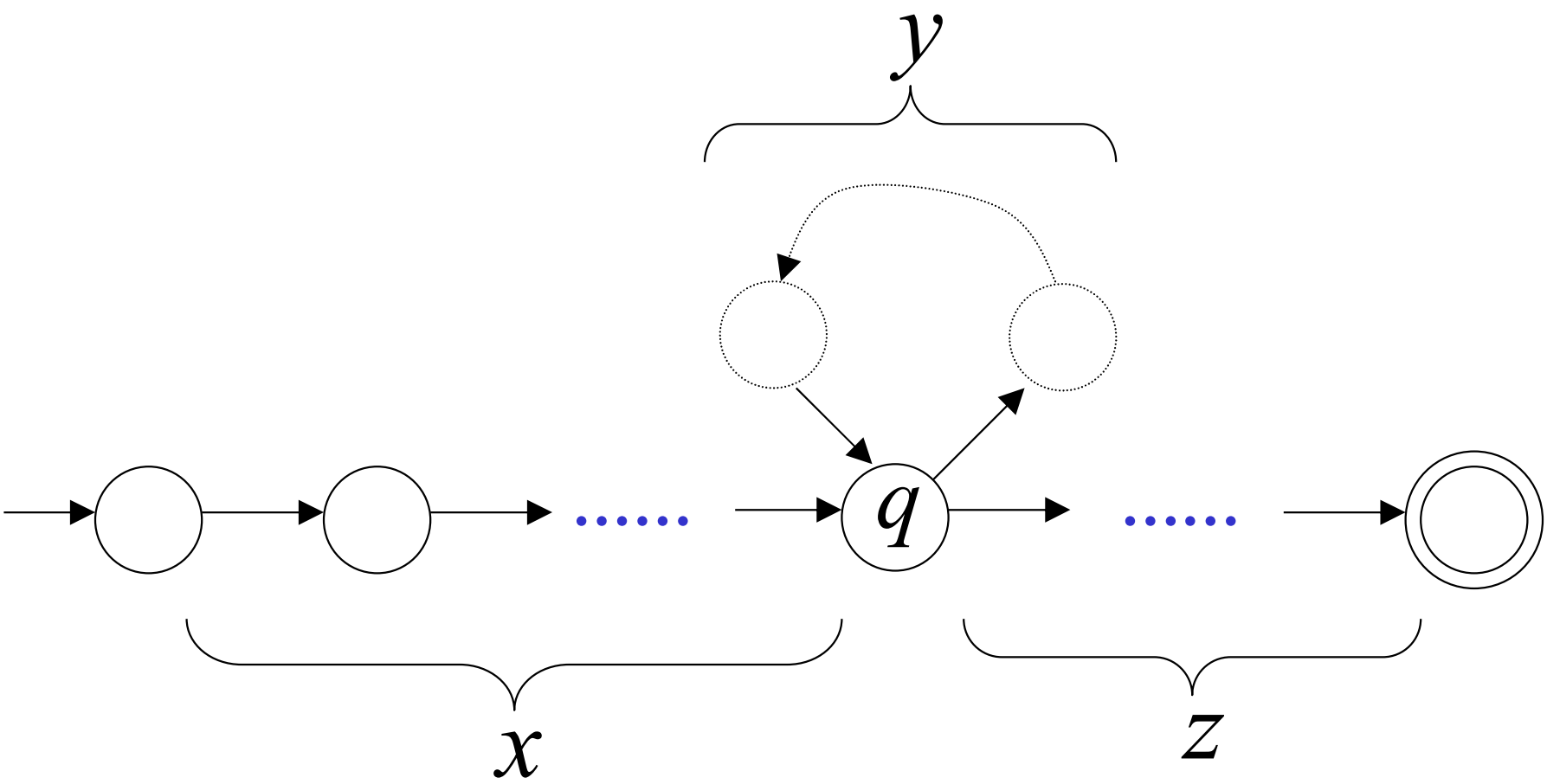


Write $w = x\ y\ z$

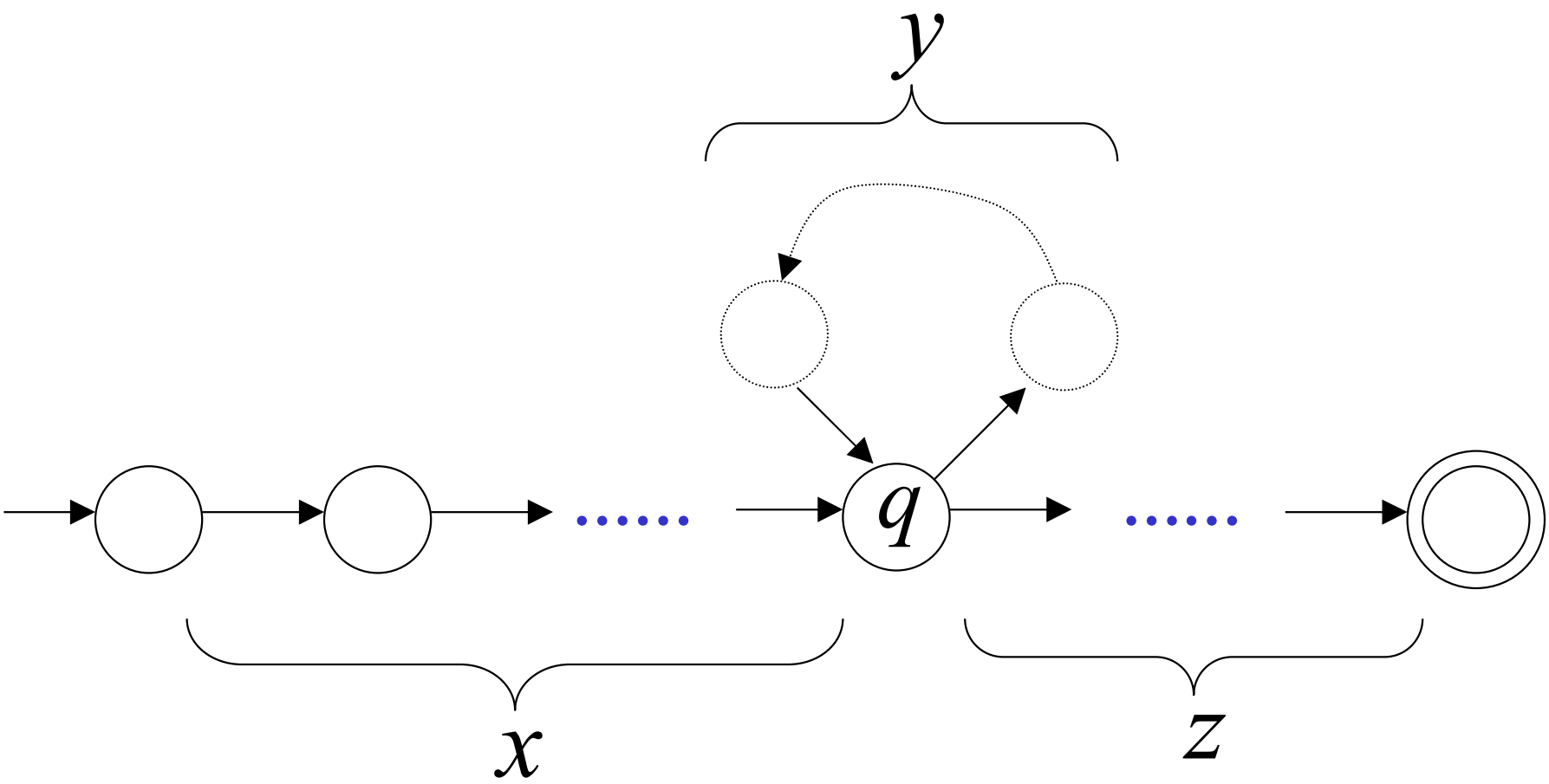


Observations: length $|xy| \leq$ m

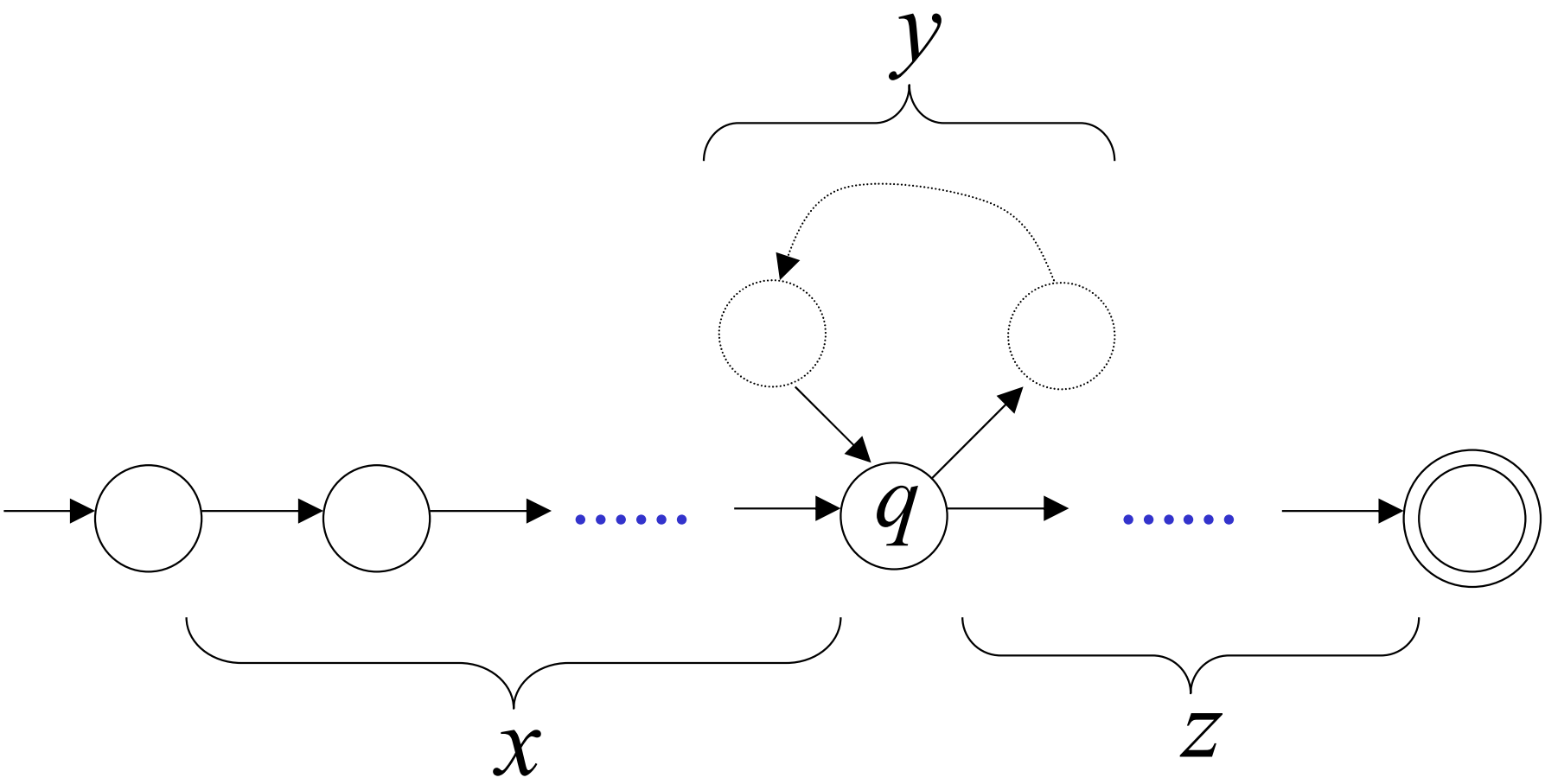
length $|y| \geq 1$



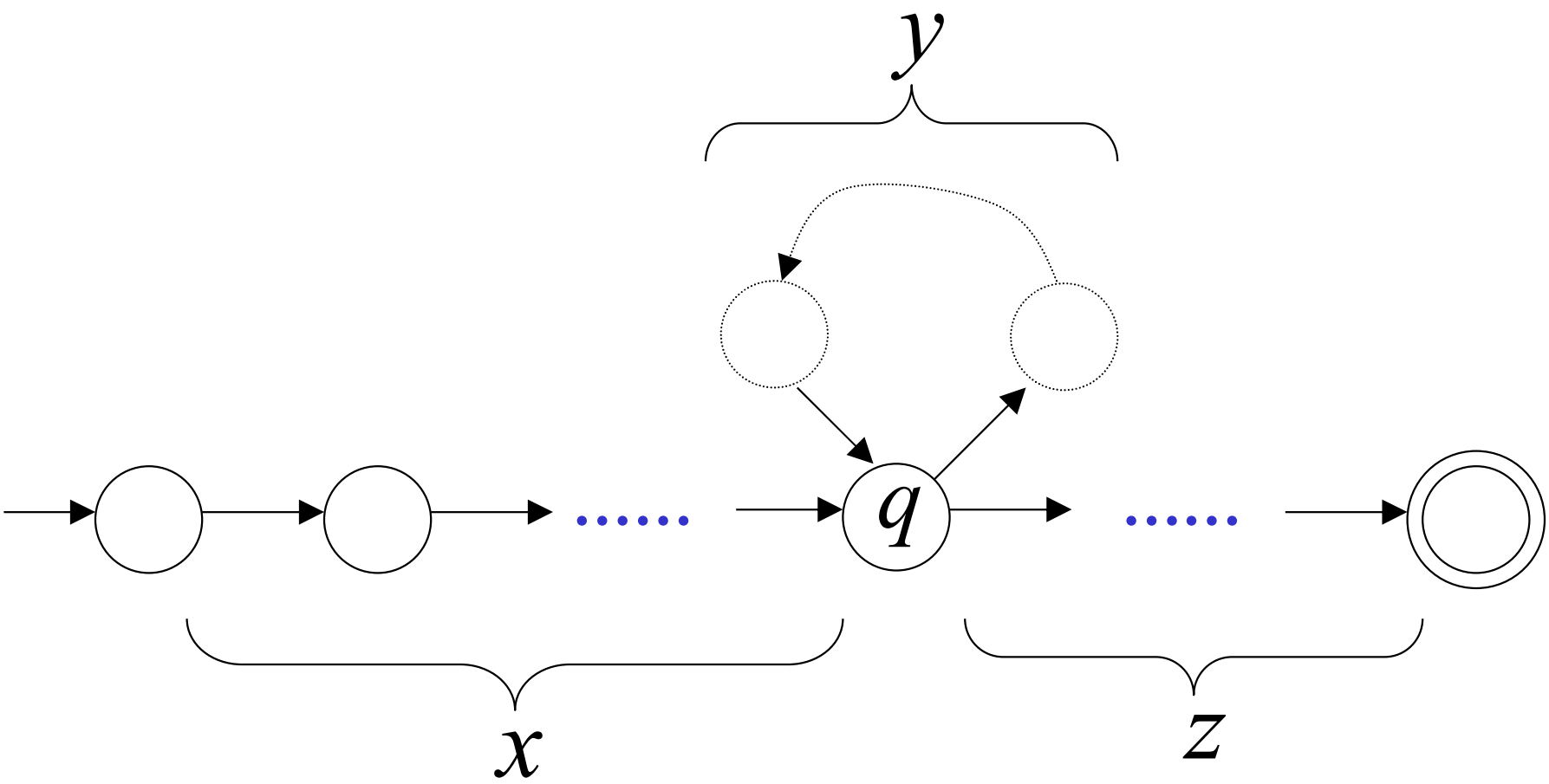
Observation: The string $x\,z$ is accepted



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



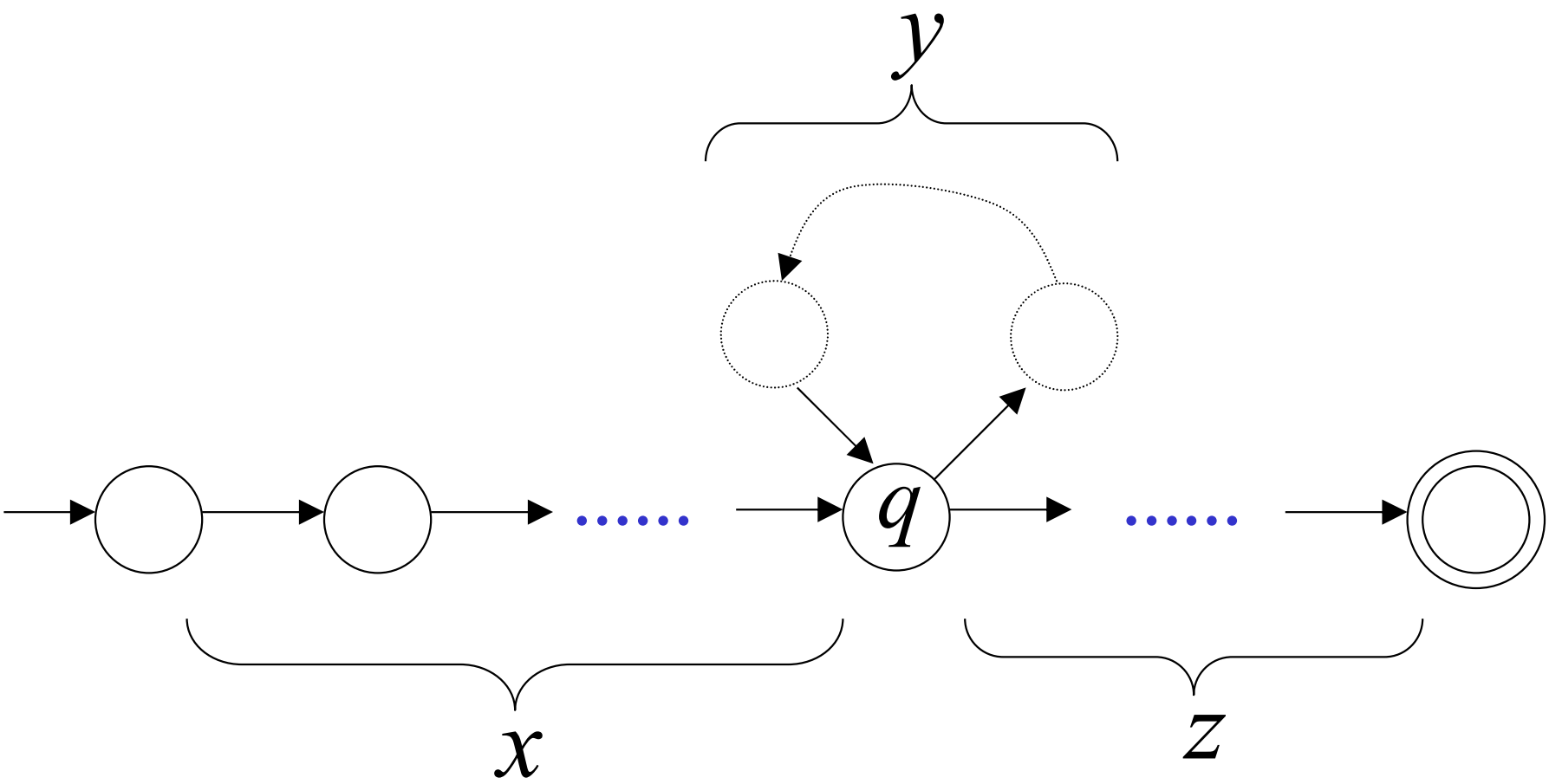
Observation: The string $x\ y\ y\ y\ z$ is accepted



In General:

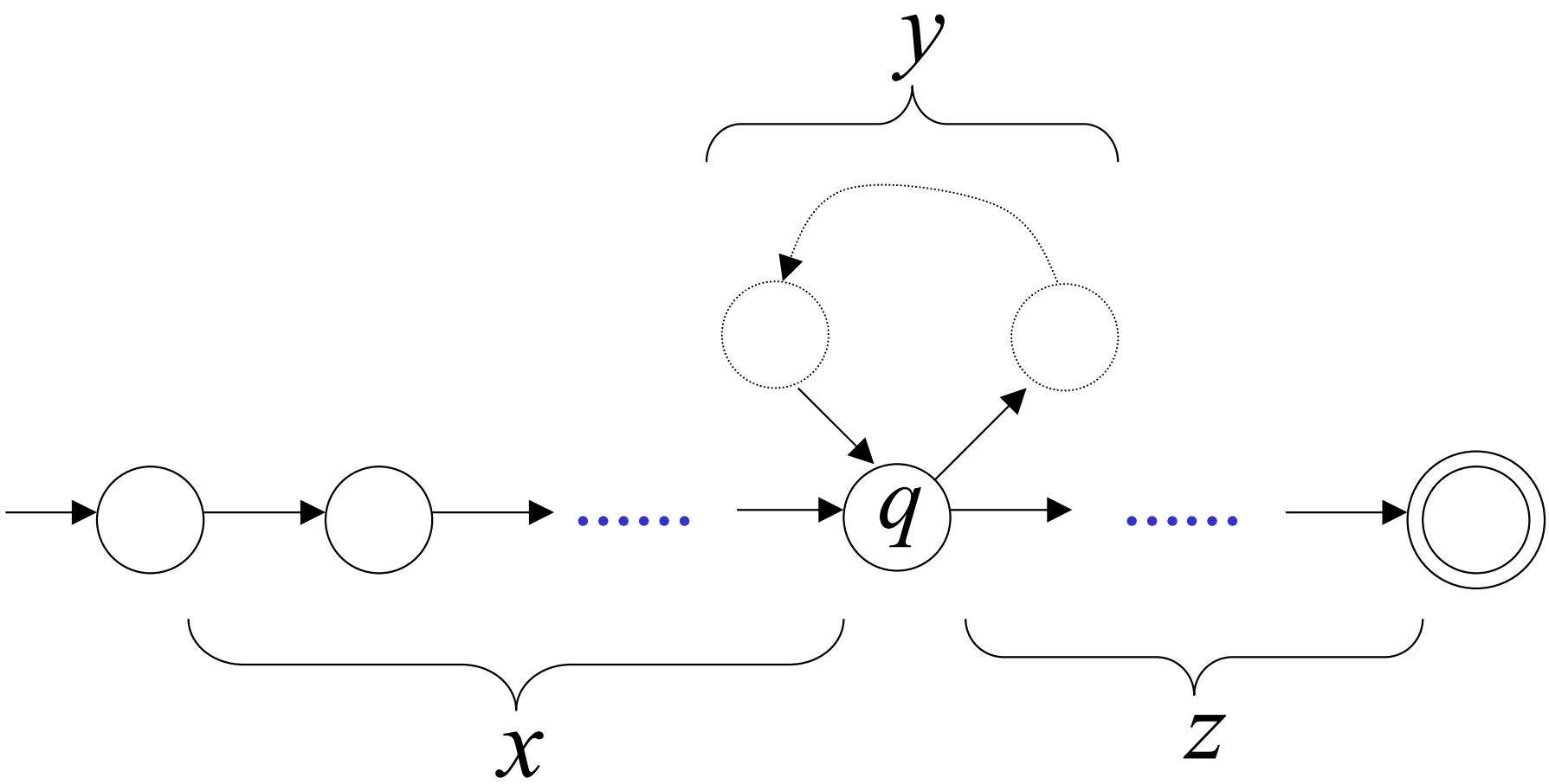
The string $x\, y^i\, z$

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

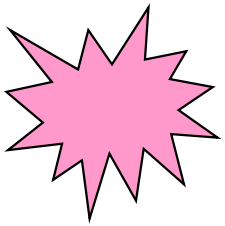


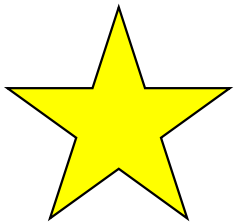
In General: $x\, y^i\, z \in L \qquad i = 0, 1, 2, \ldots$

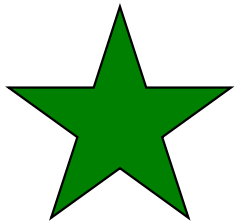
Language accepted by the DFA



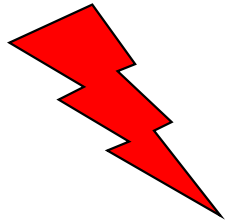
In other words, we described:

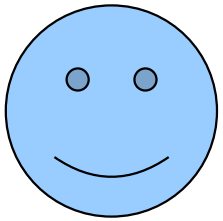


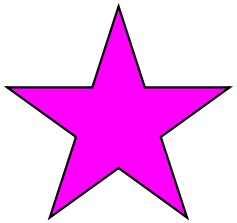




The Pumping Lemma !!!







The Pumping Lemma:

- Given an 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

Proof

A Pumping Lemma

The following result, known as the **pumping lemma** for regular languages, uses the pigeonhole principle in another form. The proof is based on the observation that in a transition graph with n vertices, any walk of length n or longer must repeat some vertex, that is, contain a cycle.

Proof: If L is regular, there exists a dfa that recognizes it. Let such a dfa have states labeled $q_0, q_1, q_2, ..., q_n$. Now take a string w in L such that $|w| \geq m = n+1$. Since L is assumed to be infinite, this can always be done. Consider the set of states the automaton goes through as it processes w, say

$$q_0, q_i, q_j, ..., q_f.$$

Since this sequence has exactly $|w| + 1$ entries, at least one state must be repeated, and such a repetition must start no later than the nth move. Thus, the sequence must look like

$$q_0, q_i, q_j, ..., q_r, ..., q_r, ..., q_f,$$

indicating there must be substrings x, y, z of w such that

$$\begin{aligned} \delta^*(q_0, x) &= q_r, \\ \delta^*(q_r, y) &= q_r, \\ \delta^*(q_r, z) &= q_f, \end{aligned}$$

with $|xy| \leq n + 1 = m$ and $|y| \geq 1$. From this it immediately follows that

$$\delta^*(q_0, xz) = q_f,$$

as well as

$$\begin{aligned} \delta^*(q_0, xy^2z) &= q_f, \\ \delta^*(q_0, xy^3z) &= q_f, \end{aligned}$$

and so on, completing the proof of the theorem. ■

Using the Pigeonhole Principle

The term "pigeonhole principle" is used by mathematicians to refer to the following simple observation. If we put n objects into m boxes (pigeonholes), and if $n > m$, then at least one box must have more than one item in it. This is such an obvious fact that it is surprising how many deep results can be obtained from it.

EXAMPLE 4.6

Is the language $L = \{a^n b^n : n \geq 0\}$ regular? The answer is no, as we show using a proof by contradiction.

Suppose L is regular. Then some dfa $M = (Q, \{a, b\}, \delta, q_0, F)$ exists for it. Now look at $\delta^*(q_0, a^i)$ for $i = 1, 2, 3, \ldots$. Since there are an unlimited number of i's, but only a finite number of states in M, the pigeonhole principle tells us that there must be some state, say q, such that

$$\delta^*(q_0, a^n) = q$$

and

$$\delta^*(q_0, a^m) = q,$$

with $n \neq m$. But since M accepts $a^n b^n$ we must have

$$\delta^*(q, b^n) = q_f \in F.$$

From this we can conclude that

$$\begin{aligned} \delta^*(q_0, a^m b^n) &= \delta^*(\delta^*(q_0, a^m), b^n) \\ &= \delta^*(q, b^n) \\ &= q_f. \end{aligned}$$

This contradicts the original assumption that M accepts $a^m b^n$ only if $n = m$, and leads us to conclude that L cannot be regular.

Inference:

Finite automaton has limited memory. To accept all $a^n b^n$, automaton should differentiate between all prefixes.

Since there are only finite number of internal states, there are some n and m , where then distinctions cannot be made.

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 $|w| \geq m$

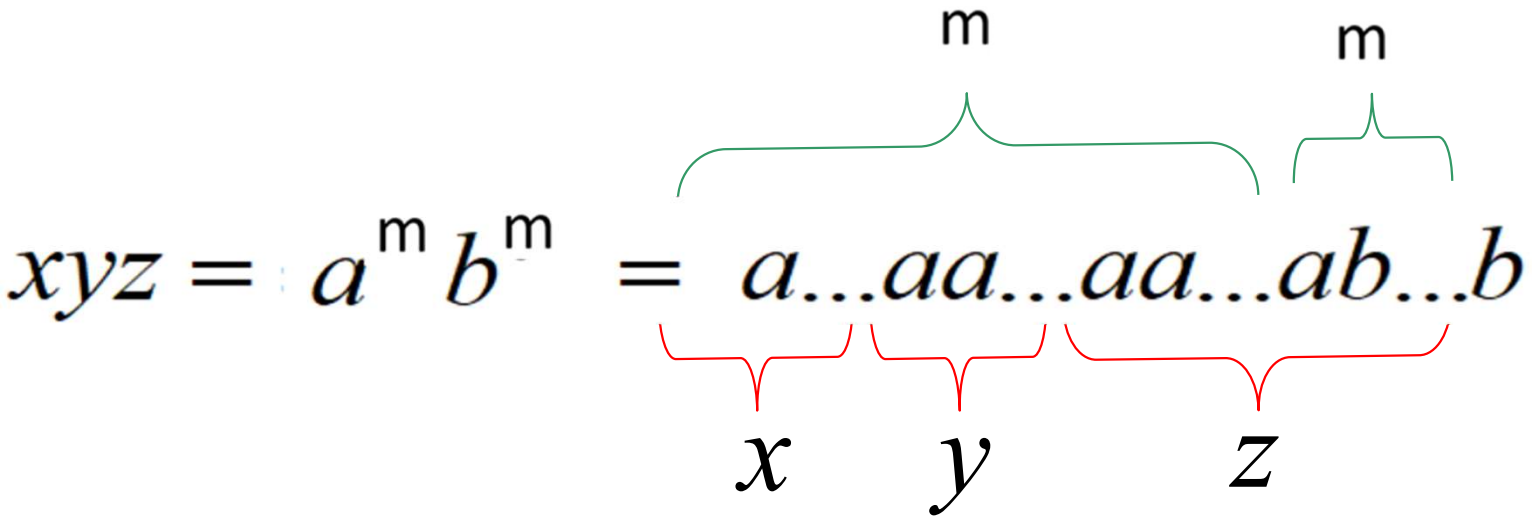
Let m be the pumping length..

We pick $w = a^m b^m$

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

From the Pumping Lemma

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



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

$$x\,y\,z = a^m\,b^m \qquad y = a^k,\ \ k \geq 1$$

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

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

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

$$x\,y\,z = a^{m}\,b^{m} \qquad y = a^{k},\ \ k \geq 1$$

From the Pumping Lemma: $x\,y^{2}\,z \in L$

$$xy^{2}z = \overbrace{\underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\underbrace{a\ldots a}_{y}\underbrace{a\ldots a}_{z}}^{m+k}\overbrace{b\ldots b}^{m} \in L$$

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

$$a^{m+k}b^{m} \notin L \qquad k \geq 1$$

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