Module 2

Regular Expressions
&
Context Free Grammar

Regular Expressions

Regular expressions
describe regular languages

Example: $(a+b\cdot c)^*$

describes the language

$$\{a,bc\}^* = \{\lambda, a, bc, aa, abc, bca, ...\}$$

Recursive Definition

Primitive regular expressions: $\emptyset, \quad \lambda, \quad \alpha$

Given regular expressions r_1 and r_2

$$\left.\begin{array}{l} r_1 + r_2 \\ r_1 \cdot r_2 \\ r_1\,* \\ (r_1) \end{array}\right\}$$ Are regular expressions

Examples

A regular expression: $(a+b\cdot c)^{*}\cdot(c+\varnothing)$

Not a regular expression: $(a+b+)$

Languages of Regular Expressions

$L(r)$: language of regular expression r

Example

$$L((a+b\cdot c)^*)=\{\lambda, a, bc, aa, abc, bca, ...\}$$

Definition

For primitive regular expressions:

$$L(\varnothing) = \varnothing$$

$$L(\lambda) = \{\lambda\}$$

$$L(a) = \{a\}$$

Definition (continued)

For regular expressions r_1 and r_2

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$

$$L(r_1 \cdot r_2) = L(r_1)\, L(r_2)$$

$$L(r_1 *) = (L(r_1))*$$

$$L((r_1)) = L(r_1)$$

Basic Regular Expressions

Regular Expression	Regular language
φ	Φ or { }
ϵ	{ϵ}
a	{a}
a+b	{a,b}
a.b	{ab}
ab+cd	{ab, cd}
a*	{ϵ,a,aa,aaa,aaaa-----}
{a, b}*	{ϵ,a,b,ab,ba,aa,bb,ababa,----}
a+	{a,aa,aaa,aaaa,----}

Example

Regular expression: $(a+b)\cdot a*$

$$
\begin{aligned}
L((a+b)\cdot a*) &= L((a+b))\,L(a*) \\
&= L(a+b)\,L(a*) \\
&= (L(a)\cup L(b))\,(L(a))* \\
&= (\{a\}\cup\{b\})\,(\{a\})* \\
&= \{a,b\}\,\{\lambda,a,aa,aaa,\ldots\} \\
&= \{a,aa,aaa,\ldots,b,ba,baa,\ldots\}
\end{aligned}
$$

Example

Regular expression $r = (a + b)^*(a + bb)$

$$L(r) = \{a, bb, aa, abb, ba, bbb, ...\}$$

Example

Regular expression $r = (aa)^*(bb)^*b$

$$L(r) = \{a^{2n}b^{2m}b: \quad n, m \geq 0\}$$

Example

Regular expression $r = (0+1)^*\,00\,(0+1)^*$

$L(r)$ = { all strings with at least two consecutive 0 }

Examples

$a^*+ b = \{b, \lambda, a, aa, aaa, aaaa, aaaaa, \ldots \}$

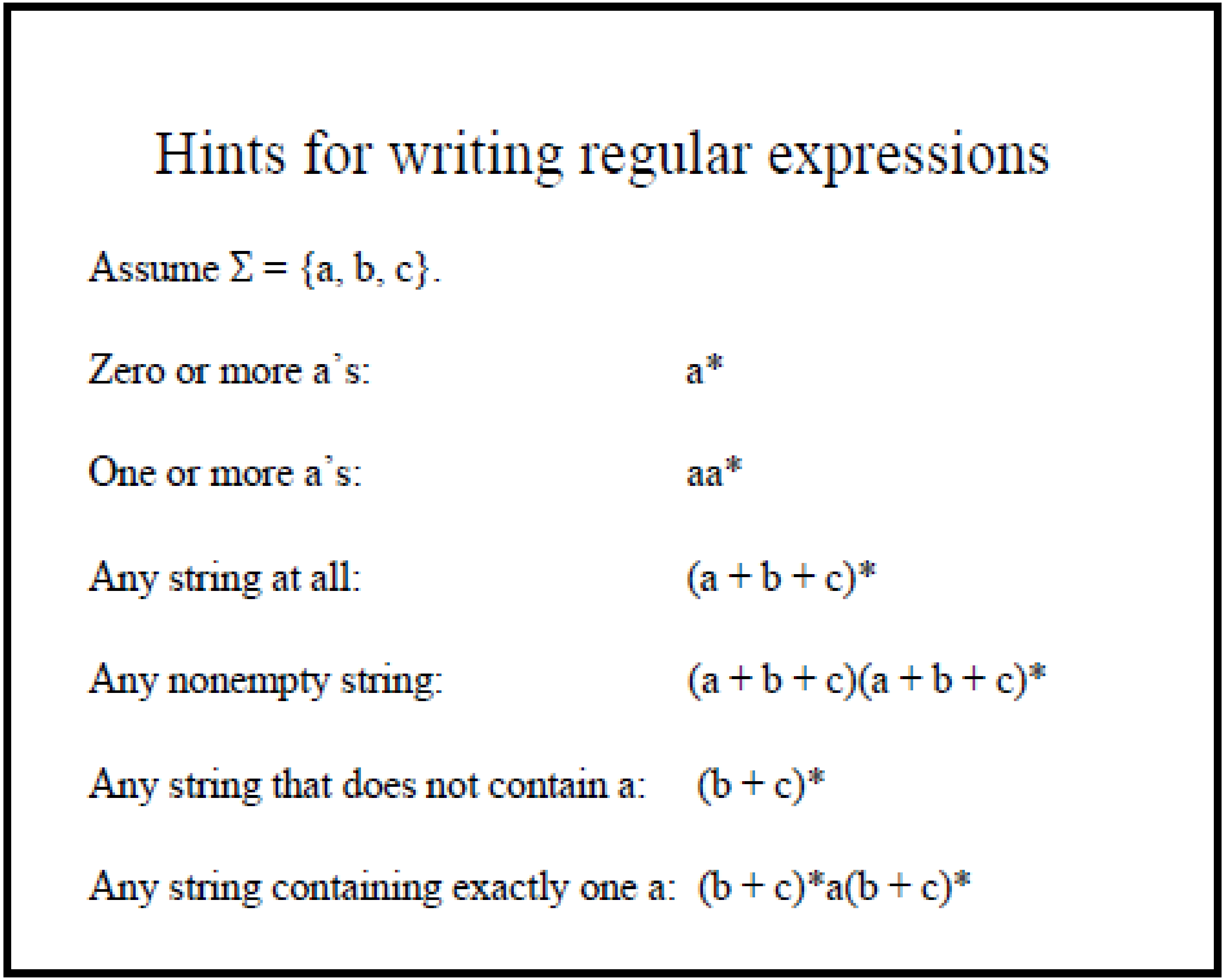
$a^*ba^* = \{w \in \Sigma^* \mid w \text{ has exactly one } b\}$

$(a + b)^*aa\ (a + b)^* = \{w \in \Sigma^* : w \text{ contains } aa\}$

$(a + b)^*aa\ (a + b)^* + (a + b)^*bb\ (a + b)^* =$
$\{w \in \Sigma^* : w \text{ contains } aa \text{ or } bb\}$

$(a + \lambda)b^* = \{ab^n : n \geq 0\} \cup \{b^n : n \geq 0\}$

As with arithmetic expressions, there is an order of precedence for operators -- unless you change it using parentheses. The order is: star closure first, then concatenation, then union.



Hints for writing regular expressions

Assume $\Sigma = \{a, b, c\}$.

Zero or more a's: a^*

One or more a's: aa^*

Any string at all: $(a + b + c)^*$

Any nonempty string: $(a + b + c)(a + b + c)^*$

Any string that does not contain a: $(b + c)^*$

Any string containing exactly one a: $(b + c)^*a(b + c)^*$

More practice

Give regular expressions for the following languages, where the alphabet is Σ = {a, b, c}.

--all strings ending in b

--all strings containing no more than two a's

-- all strings of even length

Find RE for below Languages

1. The set of all strings that begin with 110.

RE=

2. The set of all strings that contain 1011.

RE=

3. The set of all strings that contain exactly three 1's.

RE=

4. The set of all strings such that the number of 0's is odd.

RE=

Find RE for below Languages

5. All strings not ending in 01.

RE=

6. L={n_a(W) mod 3=0, where w ϵ $\{a,b\}^*$}

RE=

7. L={$a^{2n}b^{2m+1}$: n>=0, m>=0}

RE=

Find RE for below Languages

8. L={w: w has at least one pair of consecutive zeros}

RE=

9. L={w: w has no pair of consecutive zeros}

RE=

10. L={$a^n b^m$: n>=3, m<=4}

RE=

11. L={$a^n b^m$: n>=3,m is odd}

RE=

Find RE for below Languages

12. L={ $a^n b^m$: (n+m) is odd}

RE

13. Obtain a regular expression for strings of a's and b's whose lengths are multiples of three.

RE

14. Obtain a regular expression for strings of a's and b's where third symbol from the right is a and fourth symbol from the right is b.

RE

TOC Online Class

Non-regular languages
&
Pumping Lemma Theorem

Non-regular languages
(FA is not possible)

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

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

Regular languages
(FA is possible)

$$a^*b \qquad b^*c+a$$

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

$$etc...$$

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

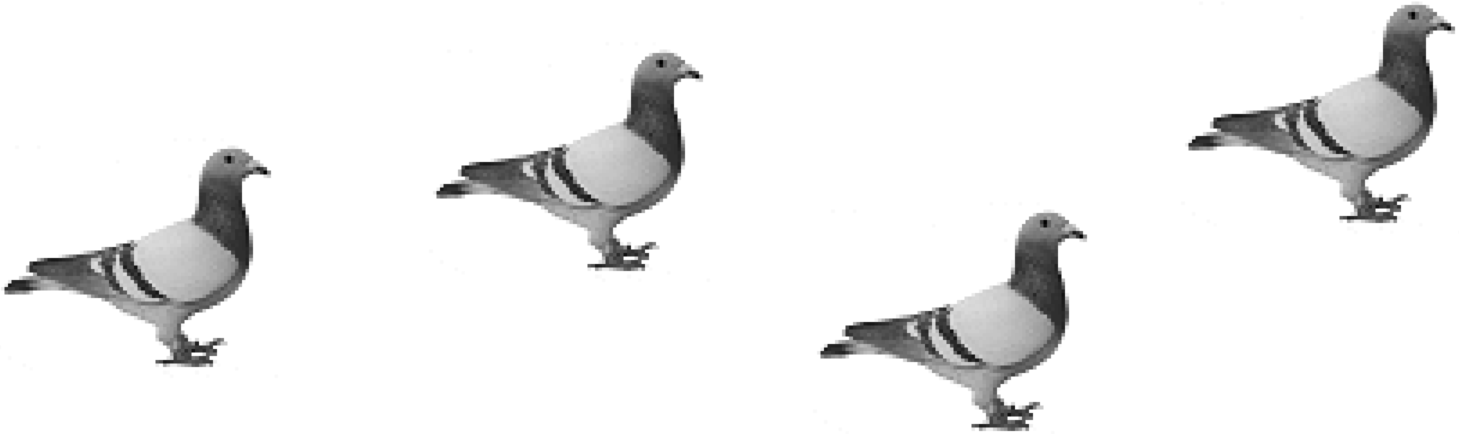
Problem: To Prove that there is no DFA that accepts L

Solution: The Pumping Lemma Theorem

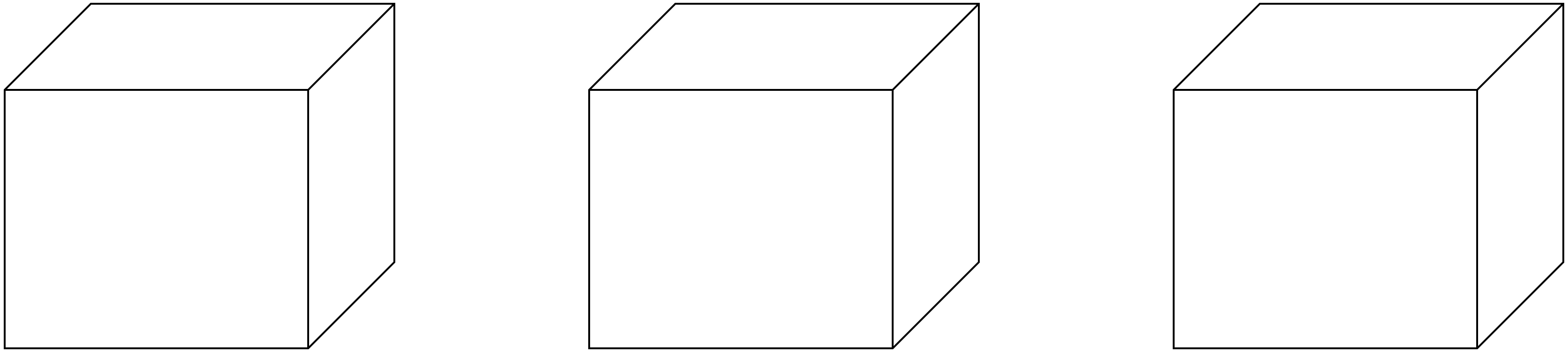
The Pigeonhole Principle

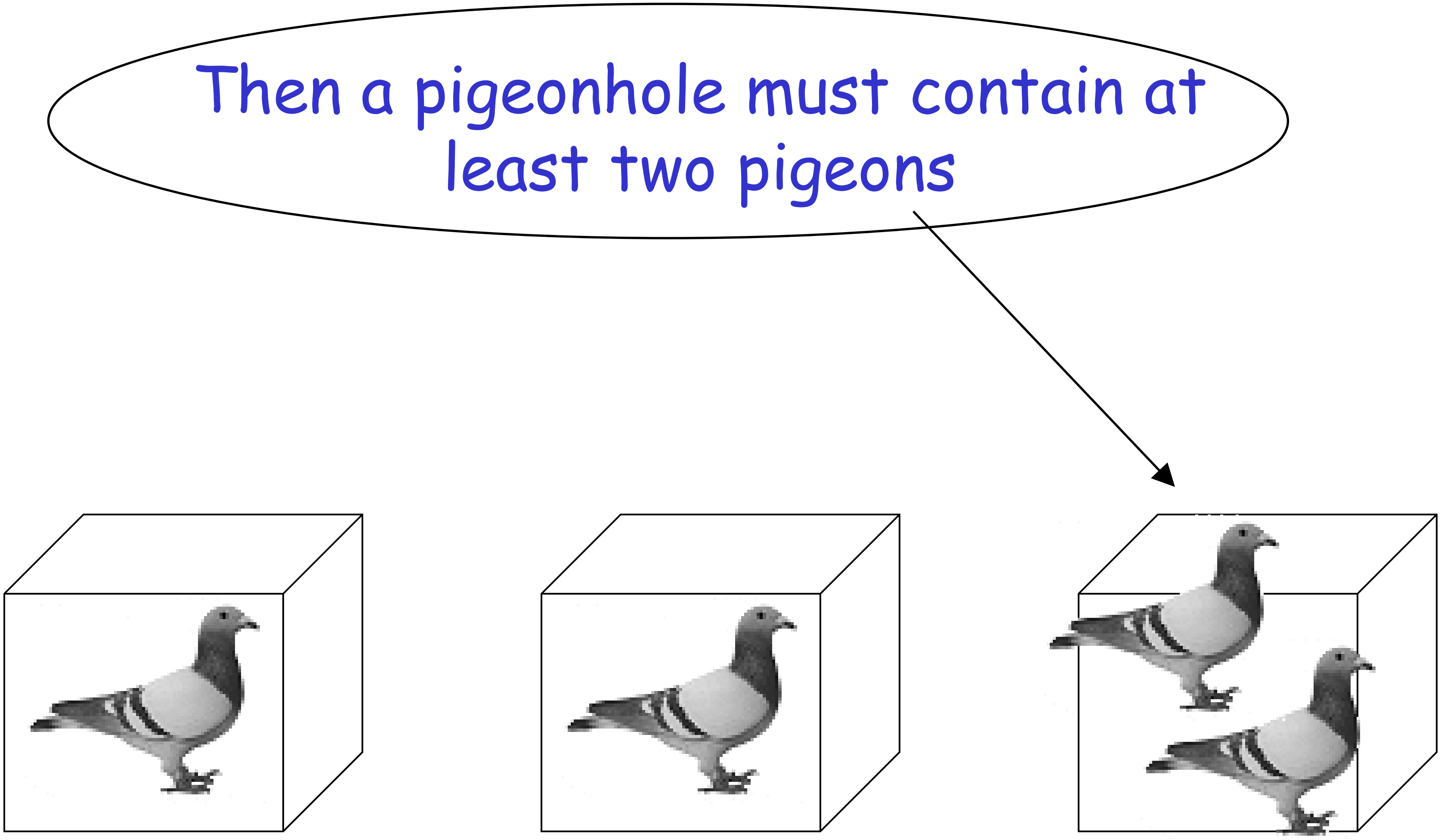
(Basic principle required to prove Pumping Lemma Theorem)

If 4 pigeons



& 3 pigeonholes





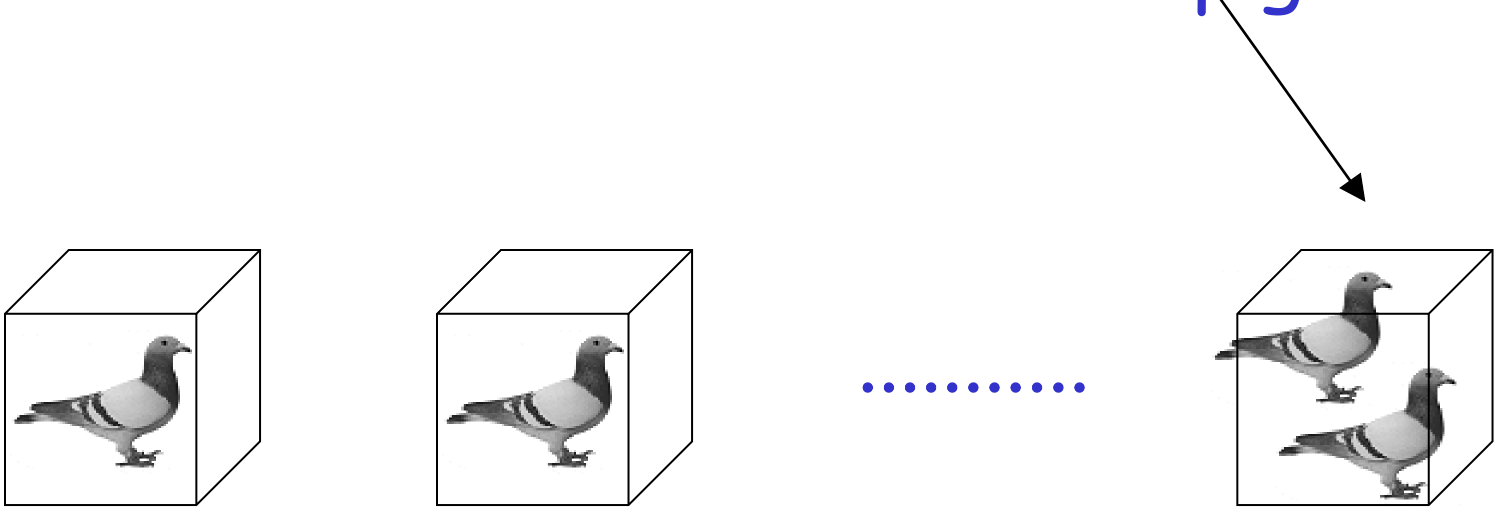
Then a pigeonhole must contain at least two pigeons

The Pigeonhole Principle

If n pigeons

& m pigeonholes such that $n > m$

Then there is a pigeonhole with at least 2 pigeons



The Pumping Lemma

The Pumping Lemma Theorem

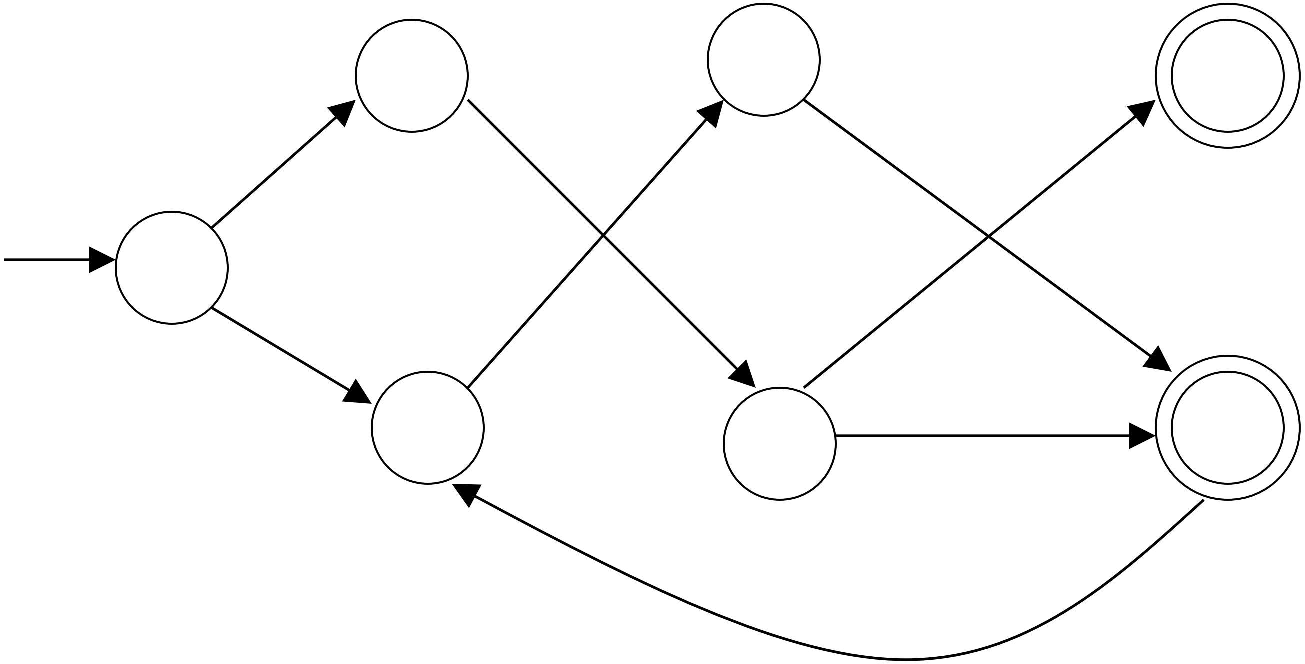
- For any regular language L, there exists an integer m for any string $w \in L$ with length $|w| \geq m$, and $x, y, z \in \sum^*$ such that $w = x\ y\ z$ then

  1) $|x\ y| \leq m$
  2) $|y| \geq 1$
  3) $w_i = x\ y^i\ z \in L$, For all $i = 0, 1, 2, \ldots$

Proof:

Take an infinite regular language L

Then there exists a DFA that accepts L
With m no of states



Take string w with

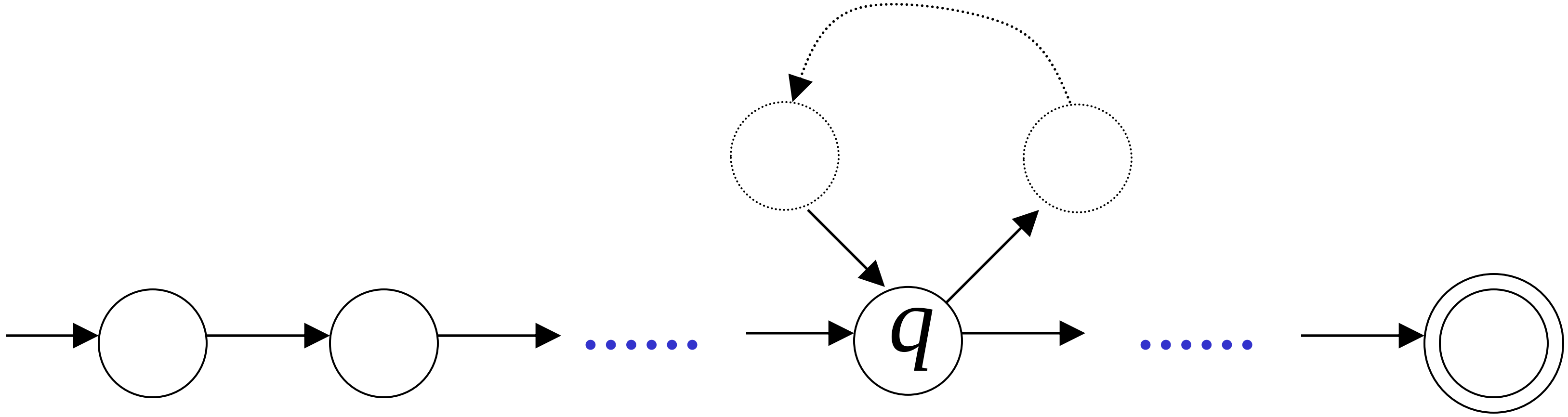
There is a walk with label w :

walk w

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

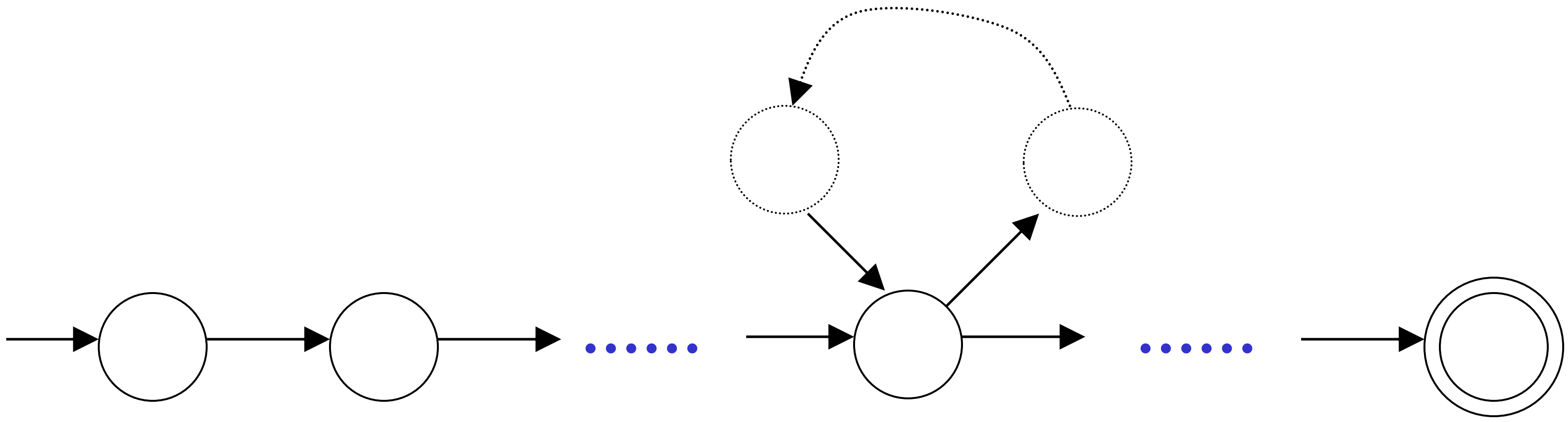
then, from the pigeonhole principle:

a state is repeated in the walk w



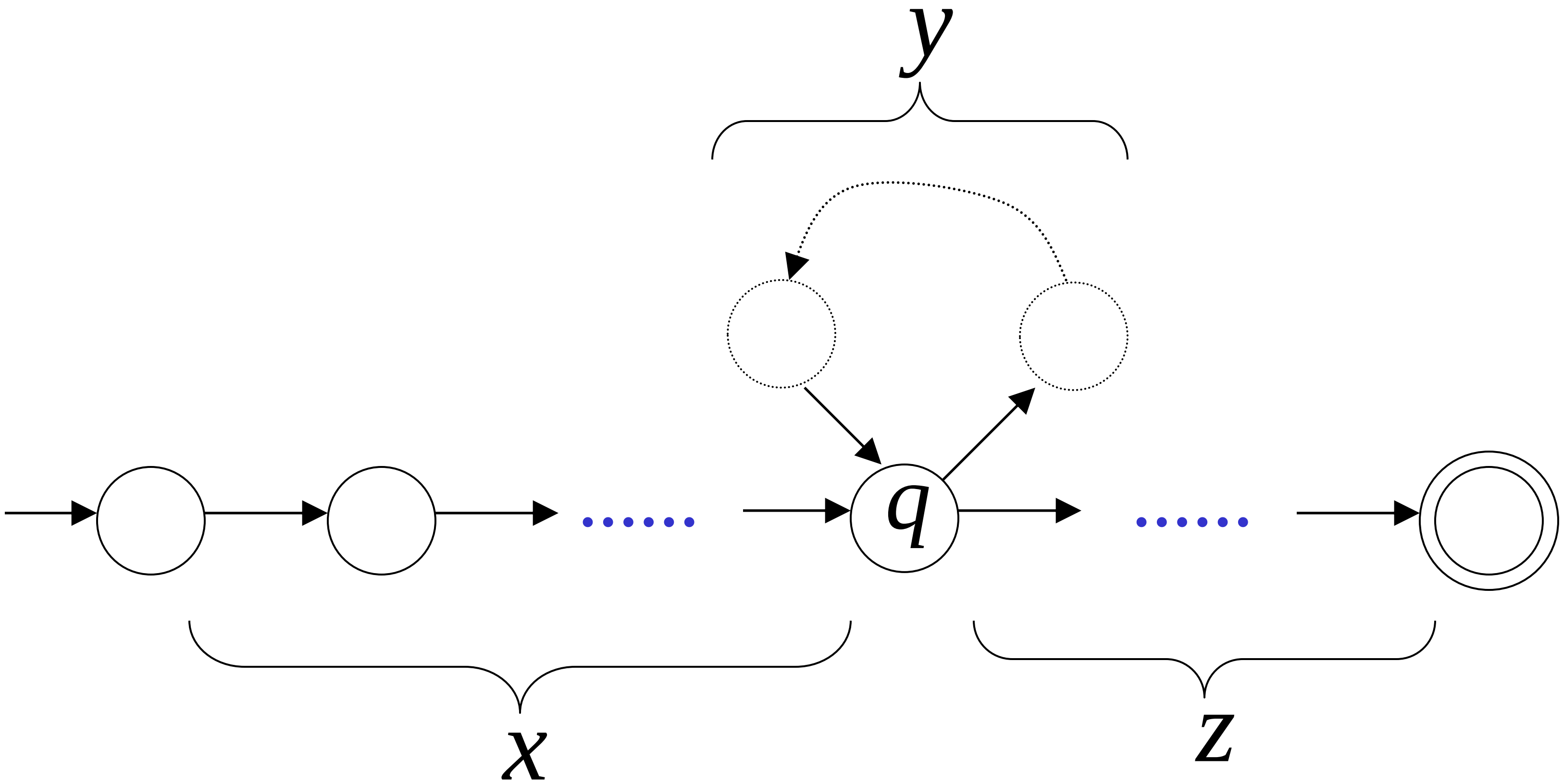
walk w

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



walk w

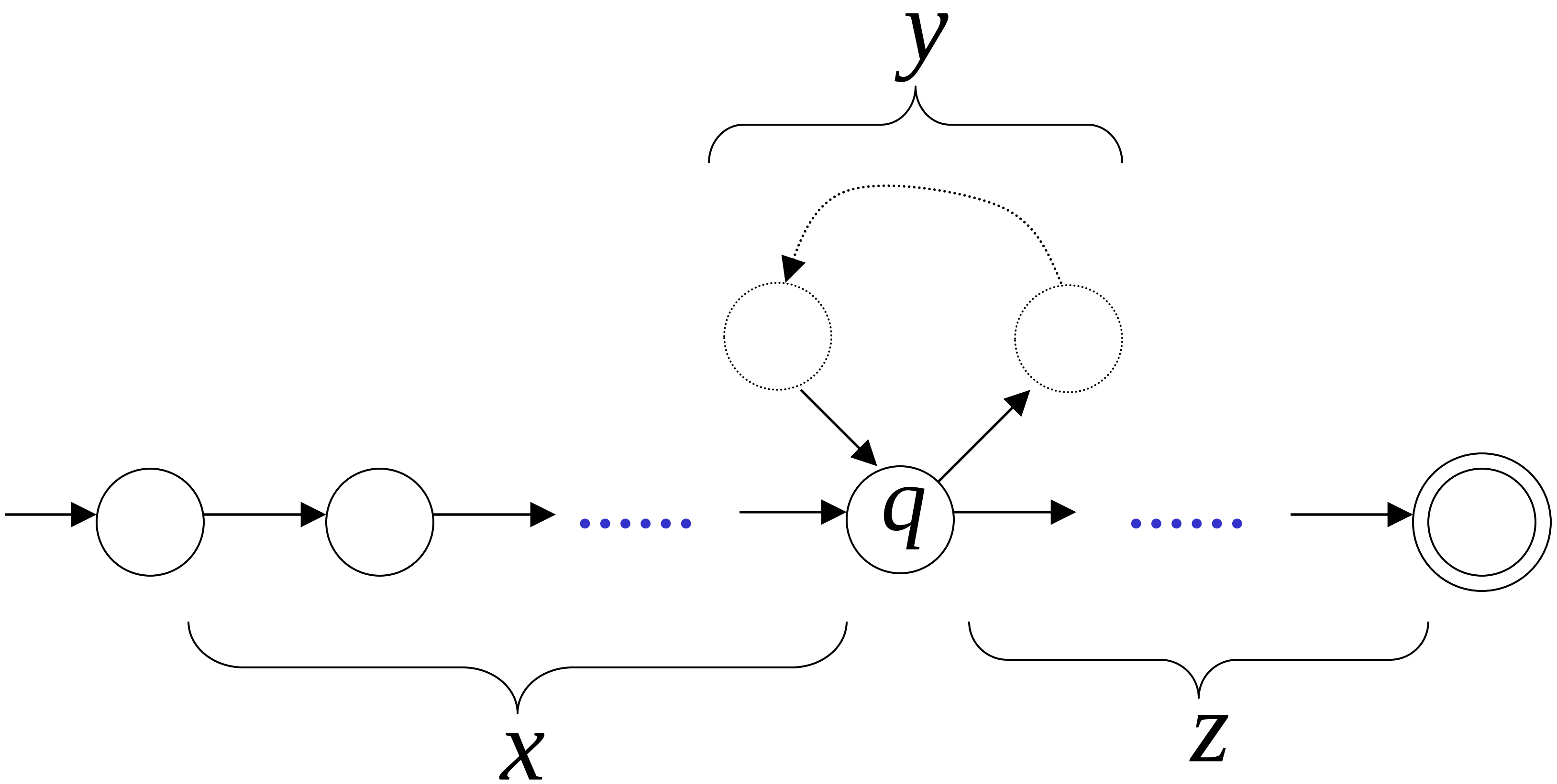
Write $w = x\ y\ z$



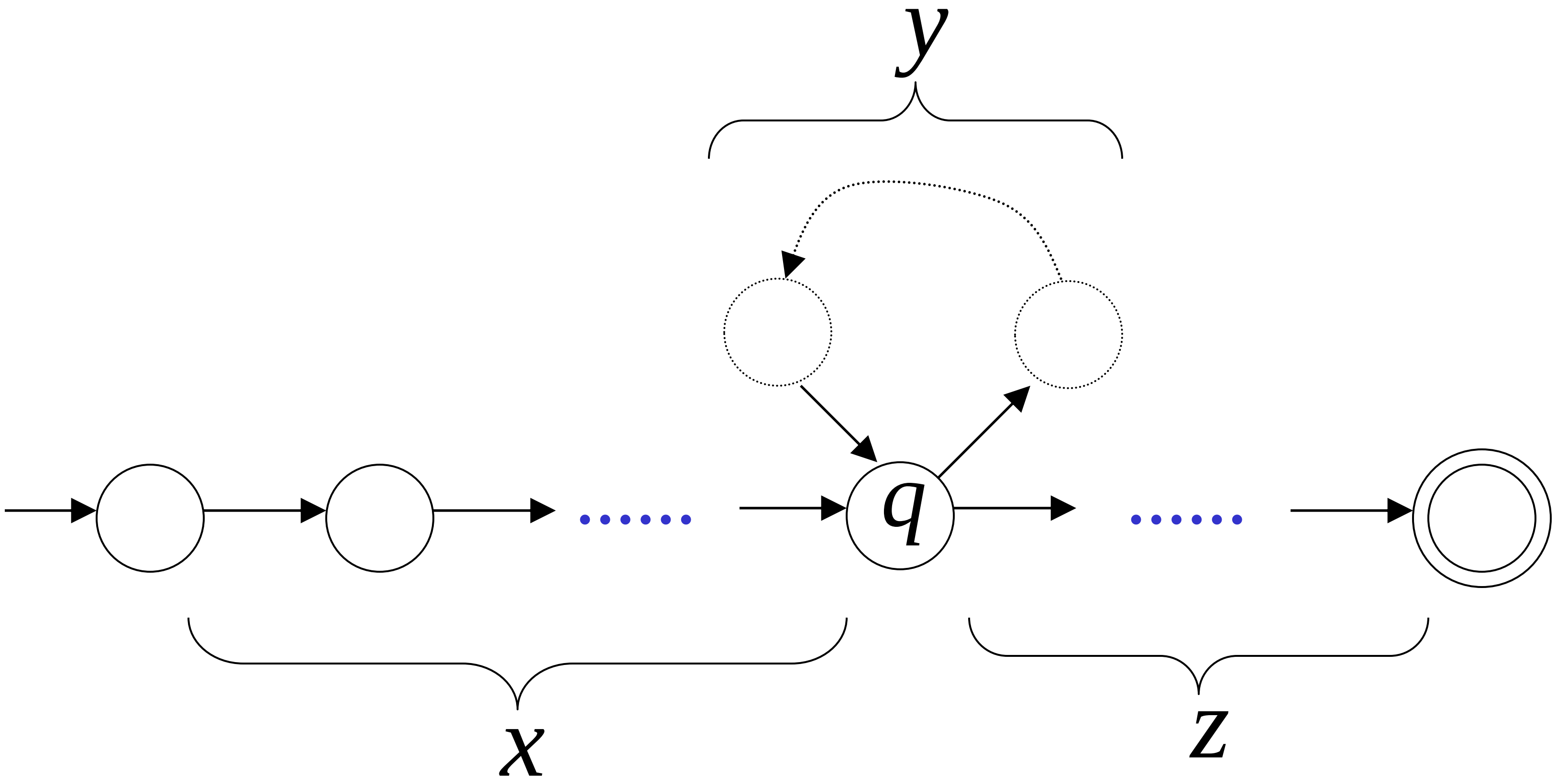
Observations:

length $|x\,y| \leq m$ number of states of DFA

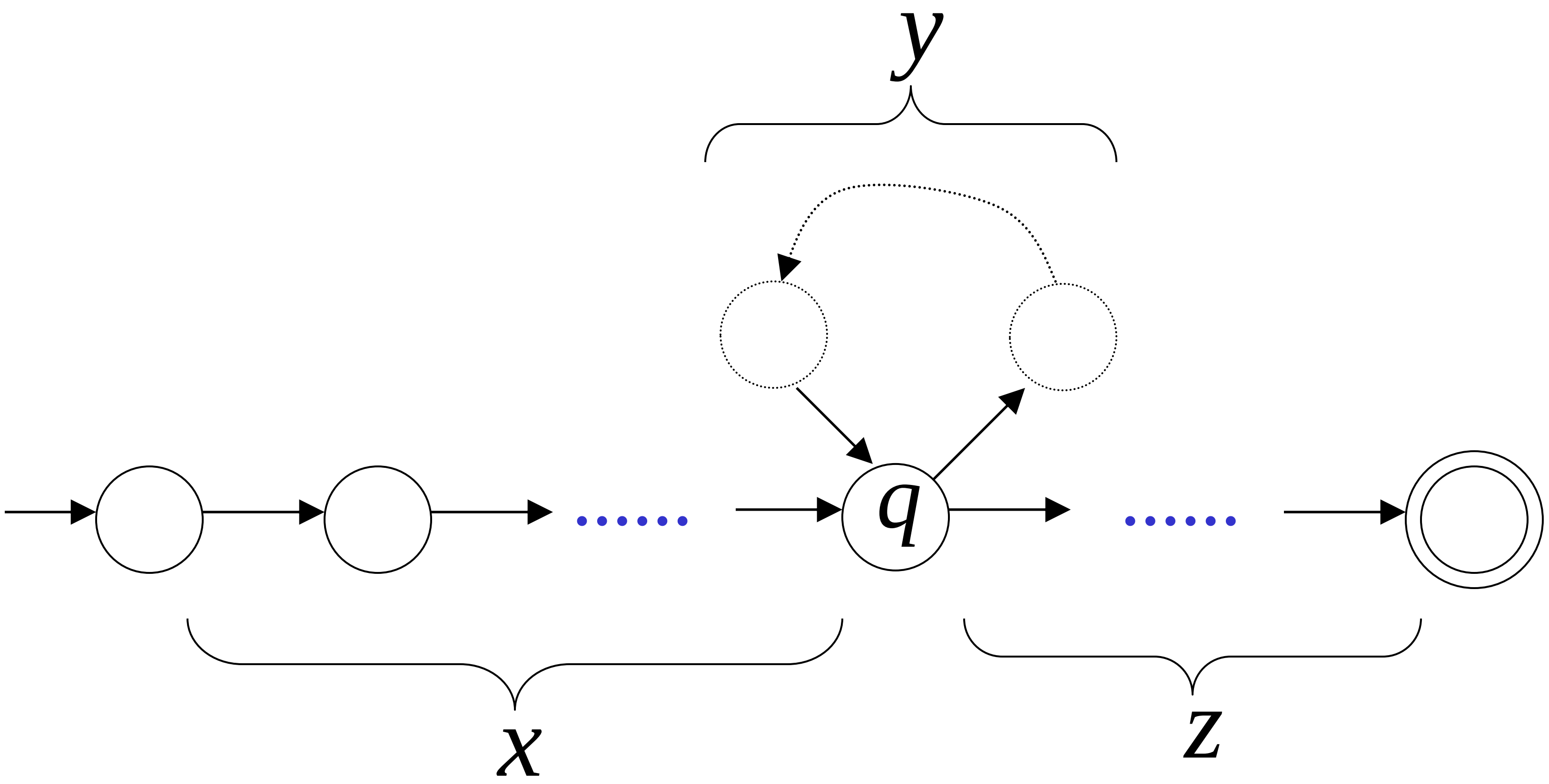
length $|y| \geq 1$



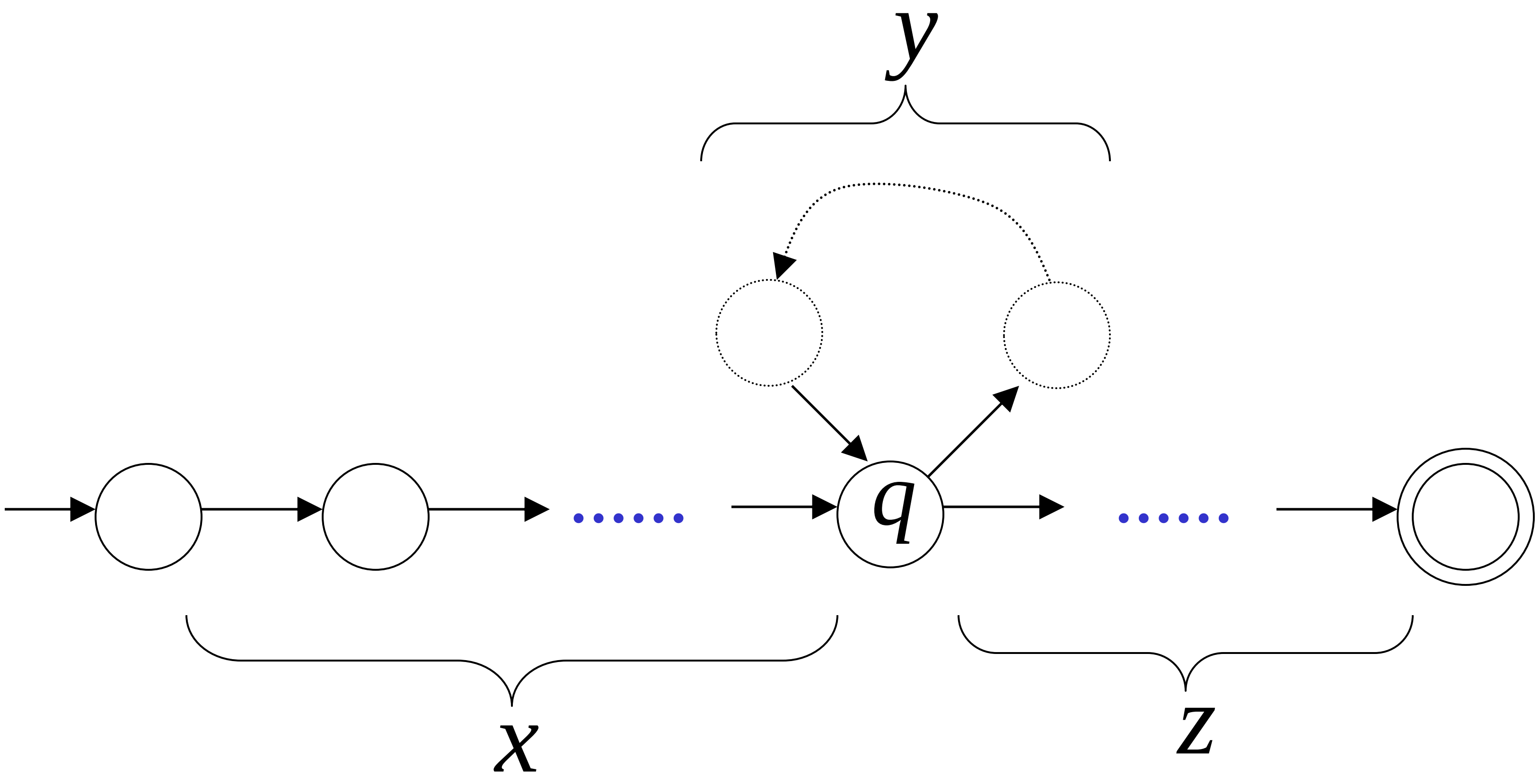
Observation: The string $x\,z$ is accepted



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



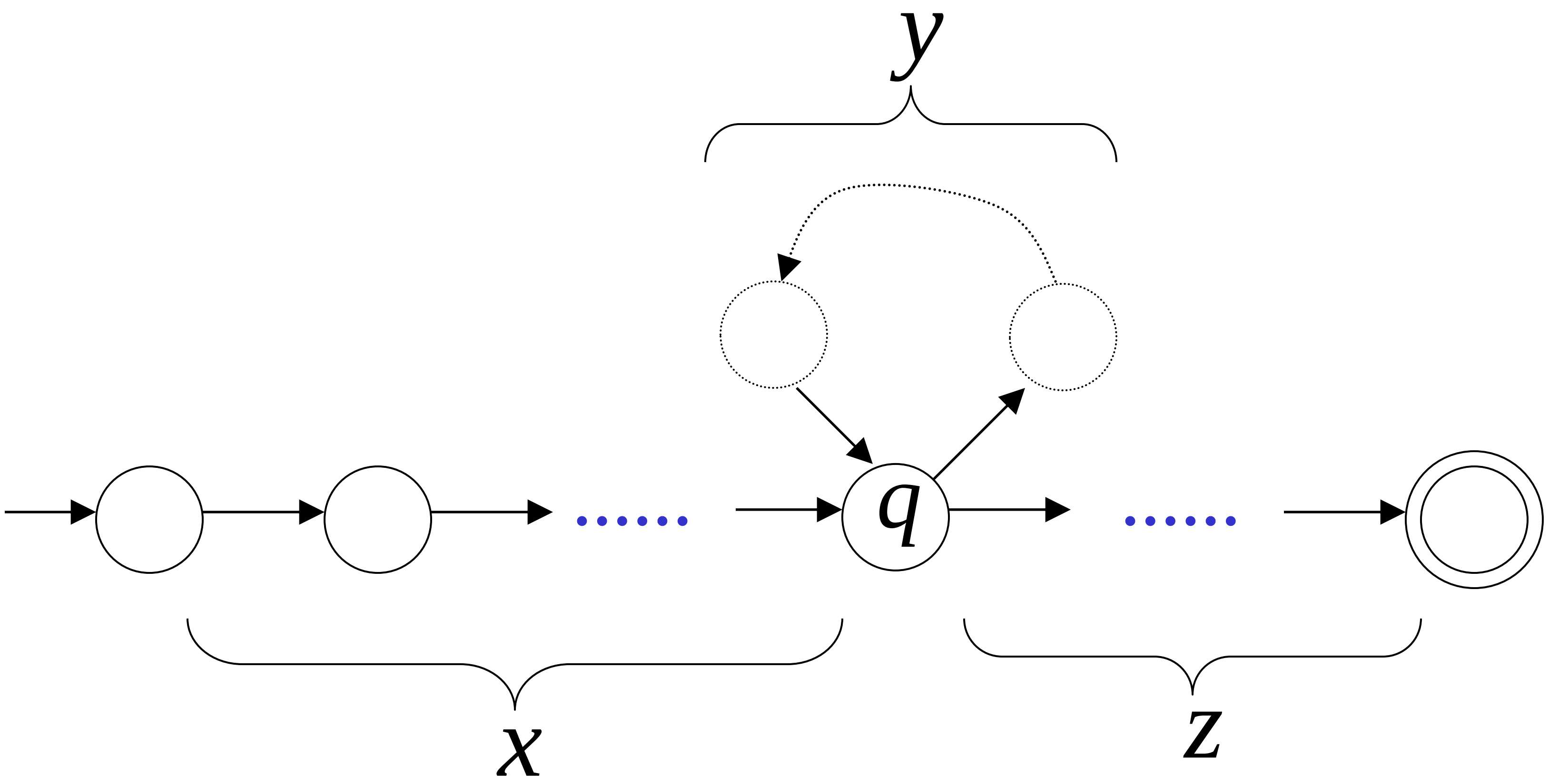
Observation: The string $x\ y\ y\ y\ z$ is accepted



In General:

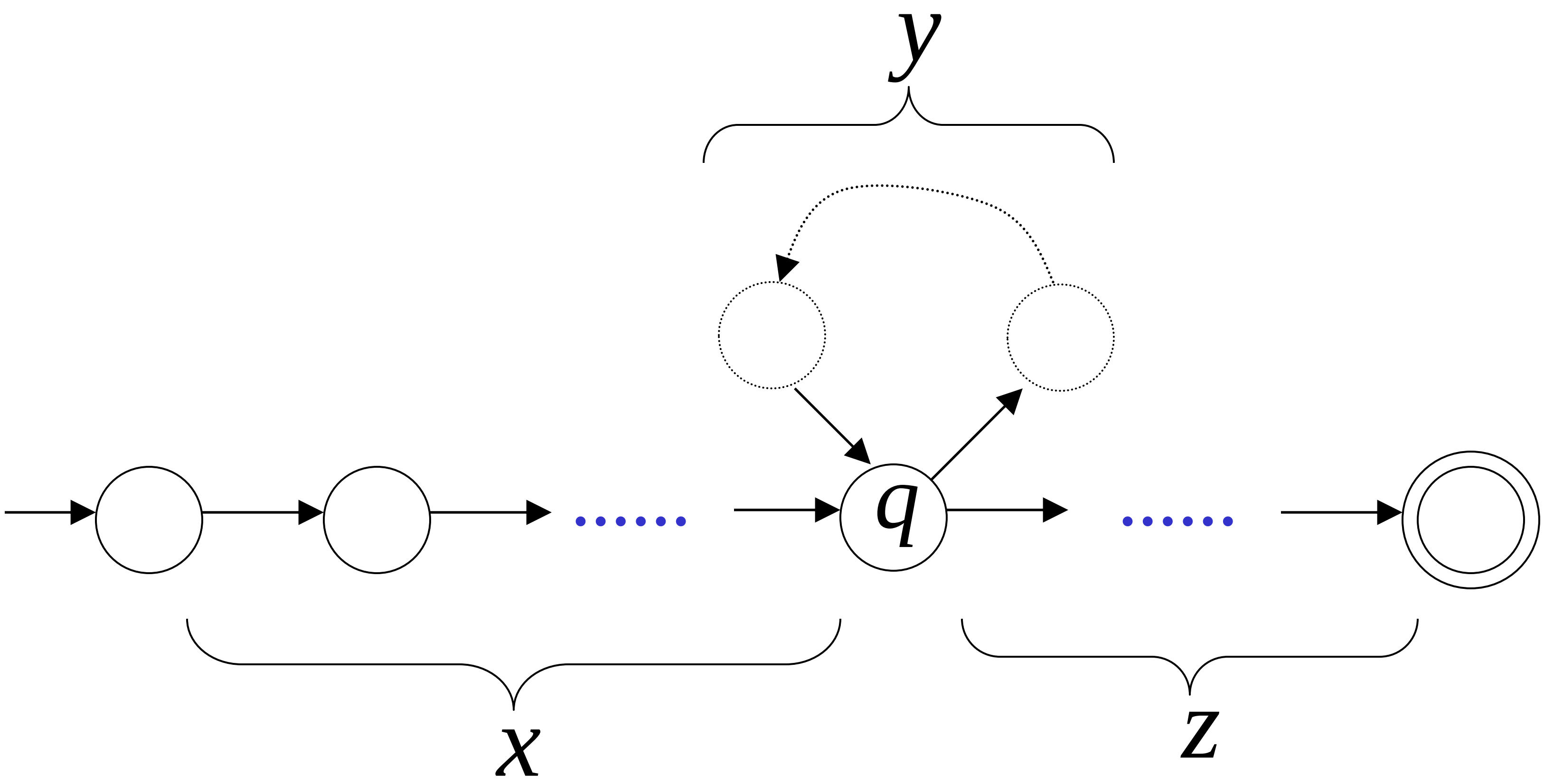
The string $x\,y^i\,z$

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



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

Language accepted by the DFA



Applications

of

the Pumping Lemma

Problem 1

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

is not regular using Pumping Lemma theorem

Proof: Assume that L is a regular language

Since L is an regular language, we can apply the Pumping Lemma

Let m be the integer in the Pumping Lemma

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

consider L is RL

For DFA, let m = no of sts

consider $w \in L$ & $|w| \geq m$

$w = a^m b^m \quad |w| = 2m$

$w = xyz = \underbrace{a \cdots a}_{x}\underbrace{a \cdots a}_{y}\underbrace{a \cdots a\, b \cdots b}_{z}$

$|xy| \leq m, \; |y| \geq 1$

$\boxed{y = a^k}, \; k = 1, 2, 3, \cdots$

$xy^i z \in L, \; i = 0, 1, 2, 3, \cdots$

$i = 2$

xy^2z

$= x \cdot y \cdot y \cdot z$

$= a \cdots a a \cdots a$

$\underbrace{a \cdots a}_{y}\, \underbrace{a \cdots a\, b \cdots b}_{z}$

$= a^{m+k} \cdot b^m$

$\notin L$

Pick a string w such that: (1) $w \in L$ and

(2) length $|w| \geq m$

Therefor, $w = a^m b^m$

Using Pumping Lemma theorem

we can write that $a^m b^m = x\,y\,z$

Also, $|x\,y| \leq m, \quad |y| \geq 1$

$$w = xyz = a^m b^m \;=\; \rlap{$\overbrace{}^{m}$}\underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\underbrace{a\ldots a\overbrace{b\ldots b}^{m}}_{z}$$

Thus, $y = a^k, \quad k \geq 1$

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

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

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

Take i=2, thus $w_2 = x\,y^2\,z \in L$

$$xy^2z = \overbrace{a\ldots aa\ldots aa\ldots aa\ldots a}^{m+k}\overbrace{b\ldots b}^{m} \in L$$

$$xy^2z = \underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\underbrace{a\ldots a}_{y}\underbrace{a\ldots ab\ldots b}_{z} \in L$$

Thus, $a^{m+k}b^m \in L$

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

$$\Downarrow$$

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

Therefor, Proof by Contradiction,

our assumption that is, L is a regular language is not true

Conclusion: L is not a regular language

More Applications

of

the Pumping Lemma

Problem 2

Theorem: The language

$$L = \{vv^R : v \in \Sigma^*\} \qquad \Sigma = \{a, b\}$$

is not regular

Proof: Use the Pumping Lemma

$$L = \{vv^R : v \in \Sigma^*\}$$

consider L is RL

m = no of Sts

$w \in L$ & $|w| \geq m$

$w = a^m b^m b^m a^m$ $\quad |w| = 4m$

$$w = xyz = \overbrace{\underbrace{a\cdots a}_{x}\,\underbrace{a\cdots a}_{y}\,a\cdots a}^{m}\,\underbrace{\overbrace{b\cdots b}^{m}\,\overbrace{b\cdots b}^{m}\,\overbrace{a\cdots a}^{m}}_{z}$$

✓ $|xy| < m$

✓ $|y| \geq 1$

$$L = \{vv^R : v \in \Sigma^*\}$$

Assume for contradiction
that L is a regular language

Since L is infinite
we can apply the Pumping Lemma

$$L = \{vv^R : v \in \Sigma^*\}$$

Let m be the integer in the Pumping Lemma

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

length $|w| \geq m$

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

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

From the Pumping Lemma

it must be that length $|\,x\ y\,| \leq m, \quad |\,y\,| \geq 1$

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

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

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

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

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

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

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

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

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

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

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

BUT: $L = \{vv^R : v \in \Sigma^*\}$

$$\Downarrow$$

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

CONTRADICTION!!!

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

Conclusion: L is not a regular language

Problem 3

Theorem: The language

$$L = \{a^n b^l c^{n+l} : \ n, l \geq 0\}$$

is not regular

Proof: Use the Pumping Lemma

$$L = \{a^n b^l c^{n+l} : \; n,l \geq 0\}$$

Assume for contradiction
that L is a regular language

Since L is infinite
we can apply the Pumping Lemma

$$L = \{a^n b^l c^{n+l} : \ n, l \geq 0\}$$

Let m be the integer in the Pumping Lemma

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

length $|w| \geq m$

We pick $w = a^m b^m c^{2m}$

Write $a^m b^m c^{2m} = x\ y\ z$

From the Pumping Lemma

it must be that length $|x\ y| \leq m, \quad |y| \geq 1$

$$xyz = \mathrlap{\overbrace{}^{m}}\underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\underbrace{a\ldots a\overbrace{b\ldots b}^{m}\overbrace{c\ldots cc\ldots c}^{2m}}_{z}$$

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

$$x\,y\,z = a^m b^m c^{2m} \qquad y = a^k, \quad k \geq 1$$

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

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

Thus: $x\,y^0\,z \;=\; xz \;\in\; L$

$$x\;y\;z = a^m b^m c^{2m} \qquad\qquad y = a^k, \quad k \ge 1$$

From the Pumping Lemma: $xz \in L$

$$xz = \mathrlap{\overbrace{}^{m-k}\overbrace{}^{m}\overbrace{}^{2m}}\underbrace{a\ldots a}_{x}\underbrace{a\ldots ab\ldots bc\ldots cc\ldots c}_{z} \in L$$

Thus: $a^{m-k} b^m c^{2m} \in L$

$$a^{m-k}b^{m}c^{2m} \in L \qquad k \geq 1$$

BUT: $L = \{a^{n}b^{l}c^{n+l} : \; n, l \geq 0\}$

$$\Downarrow$$

$$a^{m-k}b^{m}c^{2m} \notin L$$

CONTRADICTION!!!

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

Conclusion: L is not a regular language

Problem 4

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

is not regular

$$n! = 1 \cdot 2 \cdots (n-1) \cdot n$$

Proof: Use the Pumping Lemma

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

Assume for contradiction
that L is a regular language

Since L is infinite
we can apply the Pumping Lemma

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

Let m be the integer in the Pumping Lemma

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

length $|w| \geq m$

We pick $w = a^{m!}$

Write $a^{m!} = x\ y\ z$

From the Pumping Lemma
it must be that length $|x\ y| \leq m, \quad |y| \geq 1$

$$xyz = a^{m!} = \overbrace{\underbrace{a\ldots a}_{x}\underbrace{a\ldots a}_{y}\rlap{\underbrace{}_{z}}a\ldots a}^{m}\overbrace{a\ldots aa\ldots a}^{m!-m}$$

Thus: $y = a^{k}, \quad 1 \leq k \leq m$

$$x\ y\ z = a^{m!} \qquad\qquad y = a^k,\quad 1 \le k \le m$$

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

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

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

$$x\,y\,z = a^{m!} \qquad y = a^{k},\ \ 1 \le k \le m$$

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

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

Thus: $a^{m!+k} \in L$

$$a^{m!+k} \in L \qquad 1 \le k \le m$$

Since: $L = \{a^{n!} : \ n \ge 0\}$

$$\Downarrow$$

There must exist p such that:

$$m! + k = p!$$

However: $m!+k \leq m!+m$ for $m>1$

$$< m!+m!$$

$$< m!m + m!$$

$$< m!(m+1)$$

$$< (m+1)!$$

$$\Downarrow$$

$$m!+k < (m+1)!$$

$$\Downarrow$$

$m!+k \neq p!$ for any p

$$a^{m!+k} \in L \qquad 1 \leq k \leq m$$

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

$$\Downarrow$$

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

CONTRADICTION!!!

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

Conclusion: L is not a regular language

Context-Free Grammars (CFG)

What is a grammar?

- ☐ A grammar consists of one or more variables that represents a language.
- ☐ A grammar is defined with 4 tuples as:

G = (V, T, P, S)

Where V-> finite set of variables(non-terminal symbols) represented with capital letters

T->finite set of terminals(input letters)

P->Set of productions rules that represents recursive definition of language

S->Start symbol (first production rule always starts with S)

Definition of Context-Context Free Grammar

A Context Free Grammar is defined as:

$G = (V, T, P, S)$

Where V-> finite set of variables(non-terminal symbols) represented with capital letters (e.g. A,S,B,…etc)

T->finite set of terminals(input letters ex. a,b,c,x,---etc)

S->Start symbol (first production rule always starts with S)

P->Set of productions rules having the form:

$A \rightarrow x$

Where $A \in V$ and

$x \in (V \cup T)^*$

Example

1) The grammar $G=(\{S\}, \{a, b\}, S, P)$

Where $P = \{S \rightarrow aSa,\ S \rightarrow bSb,\ S \rightarrow \epsilon\}$ is a CFG.

In this CFG,

V={S}=set of variables

T={a, b}=set of terminals

S=Start Symbol

P=Set of production rules

Production rules

Each production rule consists of three parts as:

1. A variable taken from V set. This variable is often called the *head* of the production

2. The production symbol →

3. A string of empty symbol or more terminals and variables. This string, called the *body* of the production, represents one way to form strings in the language of the variable of the head.

Ex: S→aSa, S→bSb, S→ϵ

or can also be written as: S→aSa|bSb|ϵ

Context Free Language

1) The grammar G=({S}, {a, b}, S, P) Where P ={S → aSa, S → bSb, S→ϵ} is a CFG. Find the language represented by given grammar.

Solution:

S → aSa

S→aaSaa (since S → aSa)

S→aabSbaa (since S → bSb)

S→aabbaa (since S→ϵ)

S → bSb

S→baSab (since S → aSa)

S→babSbab (since S → bSb)

S→babbSbbab (since S → bSb)

S→babbbbab (since S→ϵ)

$L(G)=\{ww^R$: w ϵ {a,b}*} is a Context Free Language

Context Free Language continue..

2) The grammar G=({S}, {a, b}, S, P)

Where P ={S → aaSbb|ab|ϵ} is a CFG. Find the language represented by given grammar.

Solution:

S → aaSbb

S→aaaaSbbbb (since S → aaSbb)

S→aaaaabbbbb (since S → ab)

S → aaSbb

S→aaaaSbbbb (since S → aaSbb)

S→aaaabbbb (since S→ϵ)

L(G)={a^n b^n: n >=0} is a Context Free Language

Context Free Grammar Examples

1. L={$a^n b^n$, n is even} find CFG

n= {0, 2, 4, 6, 8, …}

L= {ϵ, aabb, aaaabbbb, aaaaaabbbbbb,….}

G=(V,T,S,P)

T={a,b}   S= start state

P= {   S→ ϵ

S → aaSbb

}

V={S}

CFG= ({S},{a,b},S,{S→ ϵ,S → aaSbb})

Context Free Grammar Examples

2. L={$a^n b^n$, n is odd} find CFG

n= 1,3,5,7….

L={ab, aaabbb,aaaaabbbbb,aaaaaaabbbbbbb….}

G=(V,T,S,P)

T={a,b} S=start state

P={ S→ ab

S→ aaSbb

}

V={S}

CFG= ({S},{a,b},S,{S→ab,S→aaSbb})

Context Free Grammar Examples

3. L={$a^n b^n$, n is a multiple of three} find CFG

n= 3,6,9,12,..

L={aaabbb,aaaaaabbbbbb,aaaaaaaaabbbbbbbbb,..}

G=(V,T,S,P)

T={a,b} S=start state

P={ S→ aaabbb

S→ aaaSbbb }

V={S}

CFG=({S},{a,b},S,{S→aaabbb,S→aaaSbbb})

Context Free Grammar Examples

4.L={$a^n b^n$, n is not a multiple of three} find CFG

n=1,2,4,5,7,8,11,---

L={ab, aabb,aaaabbbb,aaaaabbbbb,---}

G=(V,T,S,P)

T={a,b} S=start state

P={ S→ab
S→aabb
S→aaaSbbb
}

V={S}

CFG=({S},{a,b},S,{S→ab,S→aabb,S→aaaSbbb})

Context Free Grammar Examples

5. $L=\{a^n b^m,\ n=m+3\}$

m=0,1,2,3-- n=3,4,5,6,...

L={aaa,aaaab,aaaaabb,aaaaaabbb,---}

G=(V,T,P,S)

T={a,b} S=start symbol

P= { S→aaa

S→aSb }

V={S}

CFG=({S},{a,b},S,{S→aaa,S→aSb})

Context Free Grammar Examples

6. $L=\{a^n b^m, n=m-1\}$

$m=1,2,3,4,--$ $n=0,1,2,3,4—$

L={b,abb,aabbb,aaabbbb,--}

G=(V,T,P,S)

T={a,b} S=start symbol

P={ S→b

S→aSb }

V={S}

CFG=({S},{a,b},S,{S→b,S→aSb})

Context Free Grammar Examples

7. $L=\{a^n b^m, n=2m\}$

m=0,1,2,3,-- n=0,2,4,6,--

L={ϵ,aab,aaaabb,aaaaaabbb,--}

G=(V,T,P,S)

T={a,b} S=start symbol

P={ S→ ϵ

S→aaSb }

V={S}

CFG=({S},{a,b},S,{S→ ϵ,S→aaSb})

Context Free Grammar Examples

8. L={w,w€{a,b}* : n_a(w)= n_b(w)}

Count of a = count of b

L={ϵ,ab,ba,abab,baba,aabb,bbaa,---}

G=(V,T,P,S)

T={a,b}   S=Start symbol

P={S→SS

S→aSb|bSa

S→ϵ }

V={S}

CFG=({S},{a,b},S,{S→SS|aSb|bSa|ϵ})

Context Free Grammar Examples continue..

2) Given: L(G)={a^n b^m: n = m+2 } Find Context Free Grammar

Solution: Let S is Start State

T={a,b}

P: { S → aSb, S→aa}

V={S}

Therefor, CFG is

G=({S}, {a,b}, S, {S → aa|aSb})

Context Free Grammar Examples continue..

3) Given: L(G)={a^n b^n: n is odd} Find Context Free Grammar

Solution: Let S is Start State

T={a,b}

P: { S → aaSbb, S→ab}

V={S}

Therefor, CFG is

G=({S}, {a,b}, S, {S → aaSbb, S→ab})

Derivation Tree/Parse Tree

Generation of Derivation Tree

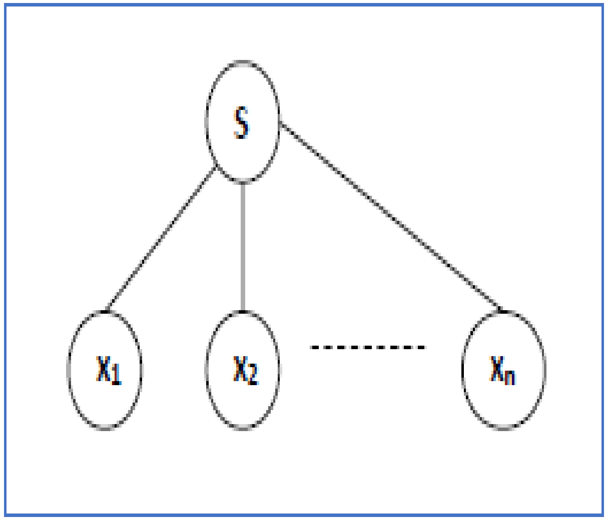
A derivation tree or parse tree is an ordered rooted tree that graphically represents the semantic information for a string derived from a context-free grammar.

Representation Technique

1. Root vertex: Must be labeled by the start symbol.
2. Vertex: Labeled by a non-terminal symbol.
3. Leaves: Labeled by a terminal symbol or ε.

Example

If S → x1x2 …… xn is a production rule in a CFG, then the parse tree / derivation tree will be as follows:



Derivation or Yield of a Tree

The derivation or the yield of a parse tree is the final string obtained by concatenating the labels of the leaves of the tree from left to right, ignoring the Nulls(Empty strings). However, if all the leaves are Null, derivation is Null.

Example

1) Let a CFG {V,T,P,S} be V = {S}, T = {a, b}, Starting symbol = S, P = S → SS | aSb | ε Find the derivation tree from the above CFG for the String "abaabb"

Solution

S → SS

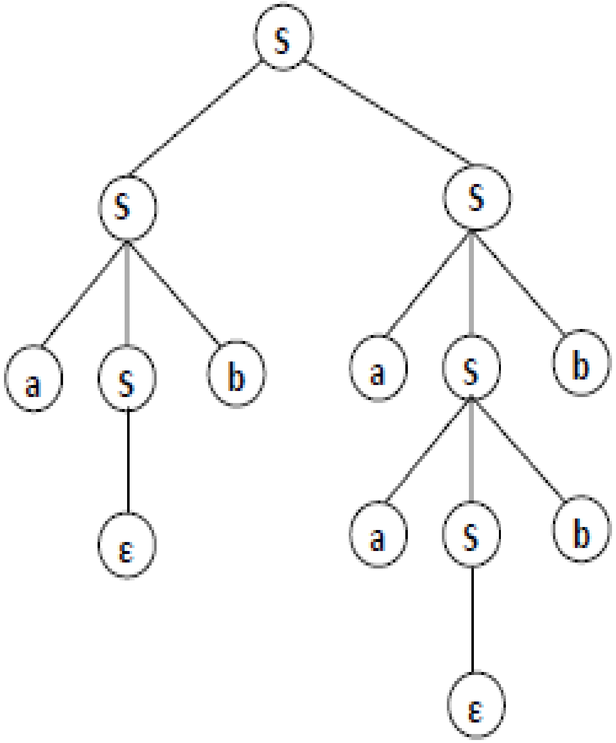
S → aSbS (replace S →aSb)

S →abS (replace S →ε)

S → abaSb (replace S →aSb)

S → abaaSbb (replace S →aSb)

S→ abaabb (replace S →ε)



Sentential Form and Partial Derivation Tree

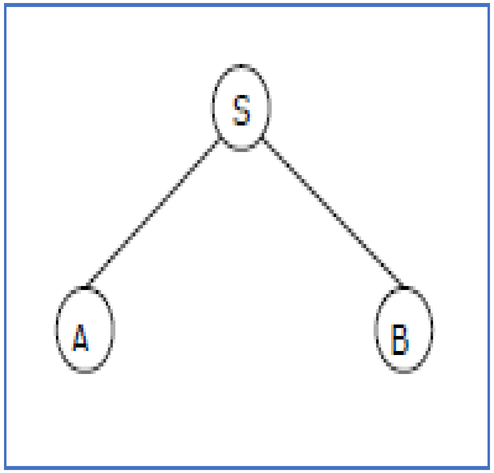
A partial derivation tree is a sub-tree of a derivation tree/parse tree such that either all of its children are in the sub-tree or none of them are in

Example

If in any CFG the productions are:

$S \rightarrow AB, A \rightarrow aaA \mid \varepsilon, B \rightarrow Bb \mid \varepsilon$

The partial derivation tree is shown as



If a partial derivation tree contains the root S, it is called a sentential form. The above sub-tree is also in sentential form.

Types of Derivation Tree

1. **Leftmost derivation**

- A leftmost derivation is obtained by applying production to the leftmost variable in each step.

2. **Rightmost derivation**

- A rightmost derivation is obtained by applying production to the rightmost variable in each step.

LMD Example

1) Let any set of production rules in a CFG be

X → X+X | X*X |X| a

over an alphabet {a,+,*}.

Find the leftmost derivation for the string

"a+a*a".

Solution

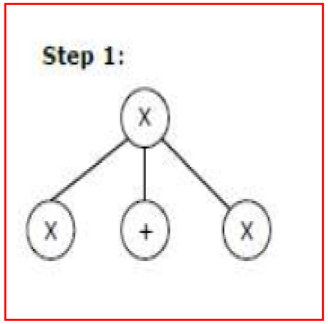
X → X+X

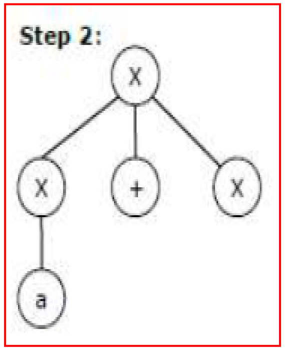
X→ a+X (replace X→a)

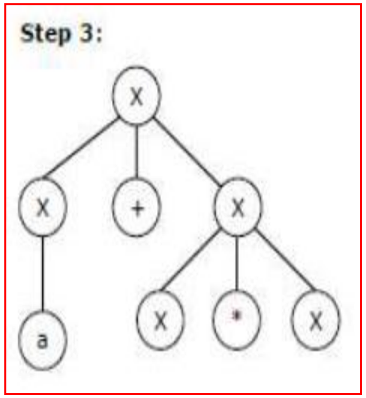
X→ a+ X*X (replace →X*X)

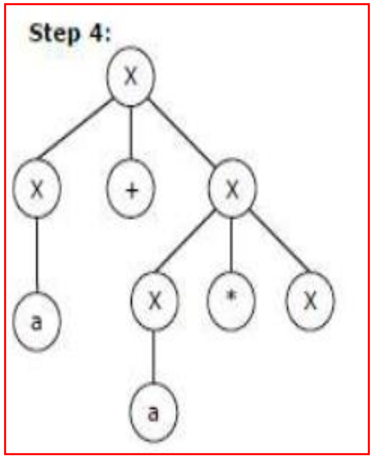
X→a+a*X (replace X→a)

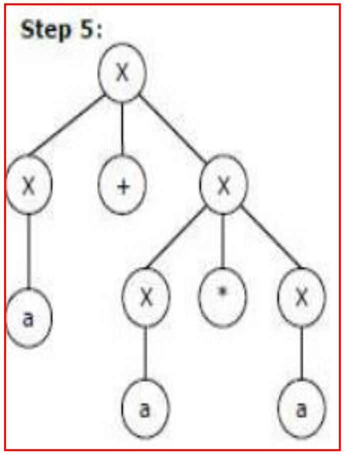
X→ a+a*a (replace X→a)


Step 1:


Step 2:


Step 3:


Step 4:


Step 5:

RMD Example

2) Let any set of production rules in a CFG be

X → X+X | X*X |X| a

over an alphabet {a}.

Find the Rightmost derivation for the string

"a+a*a“.

Solution

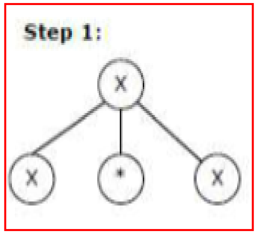
X → X*X

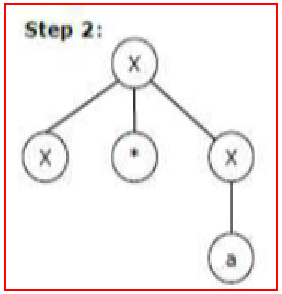
X→ X*a (replace X→a)

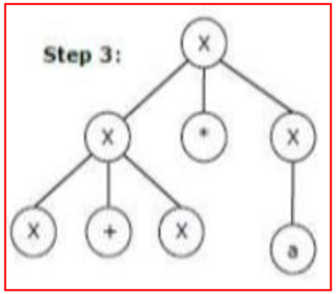
X → X+X*a (replace X→X+X)

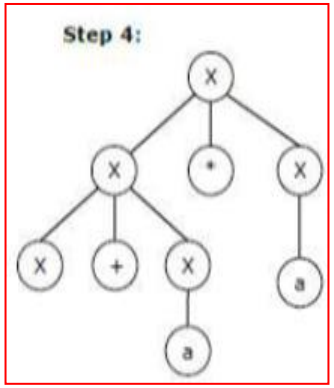
X →X+a*a (replace X→a)

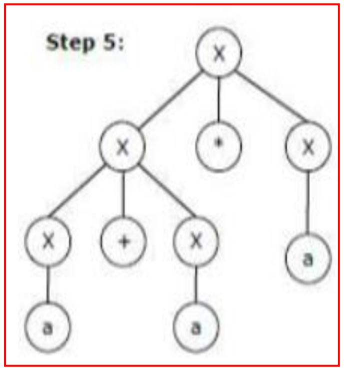
X→ a+a*a (replace X→a)


Step 1:

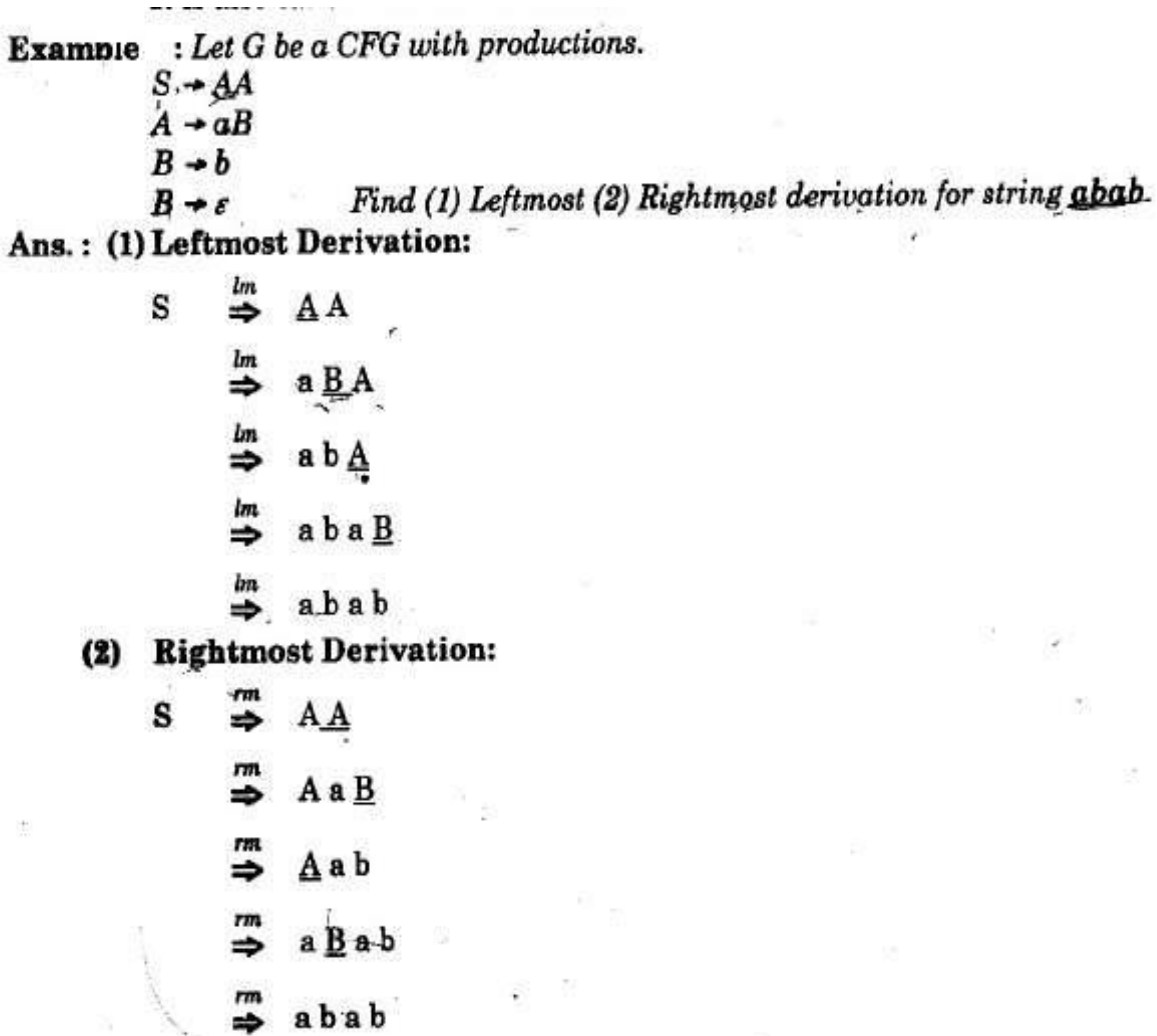

Step 2:


Step 3:


Step 4:


Step 5:

Example 3



Example : *Let G be a CFG with productions.*

$S \rightarrow AA$
$A \rightarrow aB$
$B \rightarrow b$
$B \rightarrow \varepsilon$

Find (1) Leftmost (2) Rightmost derivation for string ***abab***

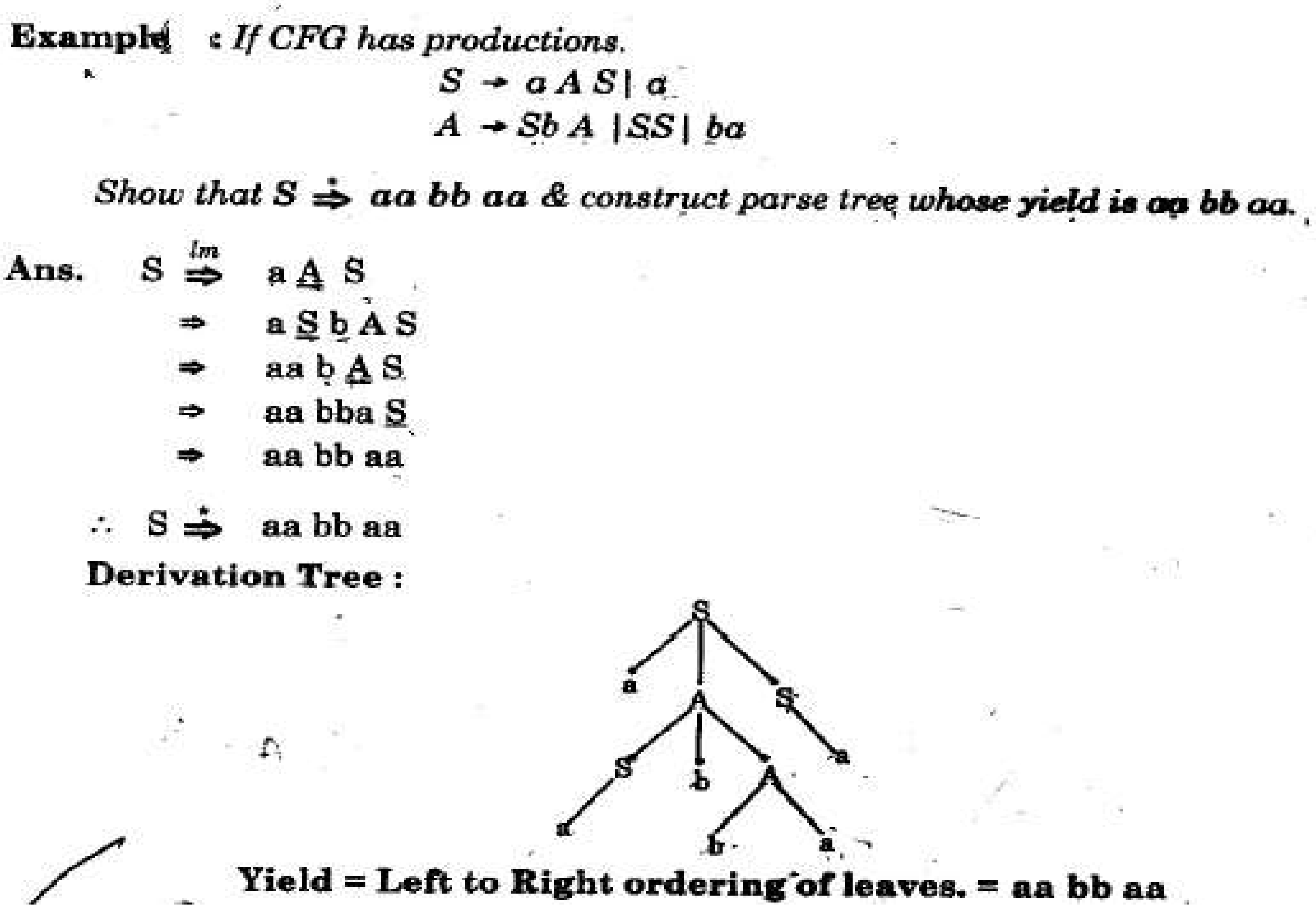
Ans. : (1) Leftmost Derivation:

$$\begin{aligned} S &\overset{lm}{\Rightarrow} \underline{A}\,A \\ &\overset{lm}{\Rightarrow} a\,\underline{B}\,A \\ &\overset{lm}{\Rightarrow} a\,b\,\underline{A} \\ &\overset{lm}{\Rightarrow} a\,b\,a\,\underline{B} \\ &\overset{lm}{\Rightarrow} a\,b\,a\,b \end{aligned}$$

(2) Rightmost Derivation:

$$\begin{aligned} S &\overset{rm}{\Rightarrow} A\,\underline{A} \\ &\overset{rm}{\Rightarrow} A\,a\,\underline{B} \\ &\overset{rm}{\Rightarrow} \underline{A}\,a\,b \\ &\overset{rm}{\Rightarrow} a\,\underline{B}\,a\,b \\ &\overset{rm}{\Rightarrow} a\,b\,a\,b \end{aligned}$$

Example 4



Example : *If CFG has productions.*

$S \rightarrow aAS \mid a$

$A \rightarrow SbA \mid SS \mid ba$

Show that $S \overset{*}{\Rightarrow}$ *aa bb aa & construct parse tree whose yield is aa bb aa.*

Ans. $S \overset{lm}{\Rightarrow} a\underline{A}\,S$

$\Rightarrow a\underline{S}\,\underline{b}\,A\,S$

$\Rightarrow aa\,\underline{b}\,\underline{A}\,S$

$\Rightarrow aa\,bba\,\underline{S}$

$\Rightarrow aa\,bb\,aa$

$\therefore\ S \overset{*}{\Rightarrow} aa\,bb\,aa$

Derivation Tree :

Yield = Left to Right ordering of leaves. = aa bb aa

Example 5

Consider the following grammar

S → bB/Aa

A→b/ bS/ aAA

B → a/ aS/ bBB

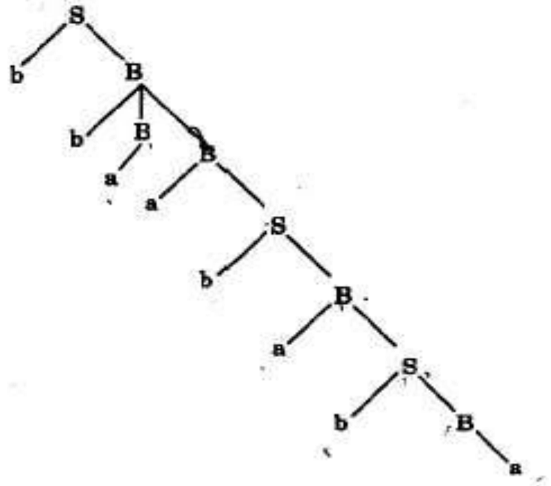
Find: Leftmost and right most derivation For string bbaababa and Also find derivation tree

Solution :

Ans. (*a*) Leftmost Derivation :

S ⇒ b B
⇒ bbBB
⇒ bbaB
⇒ bbaaS
⇒ bb aabB
⇒ bb aa b aS
⇒ bb aa bab B
⇒ bb aa ba ba

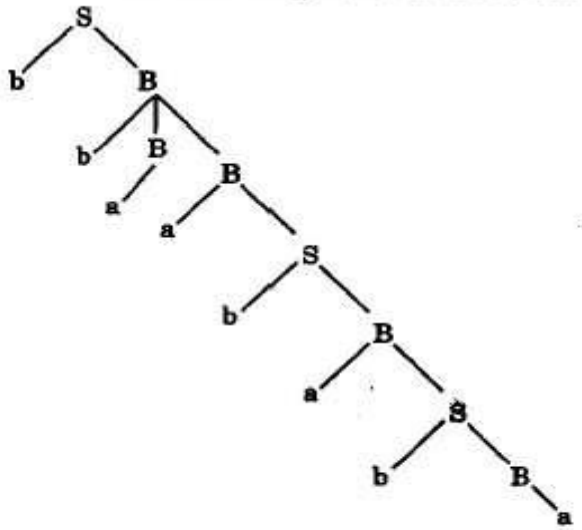
Derivation Tree for leftmost Derivation:



(*b*) Rightmost Derivation :

S ⇒ bB
⇒ bbBB
⇒ bbBaS
⇒ bbBabB
⇒ bbBabaS
⇒ bbBababB
⇒ bbBabab a
⇒ bbaababa

Derivation Tree for Rightmost Derivation:



Ambiguity in Context-Free Grammars

If a context free grammar G has more than one derivation tree(more than 1 LMDs or 1 RMDs) for some string w ∈ L(G), it is called an ambiguous grammar. There exist multiple right-most or left-most derivations for same string generated from that grammar.

Problem

Check whether the grammar G with production rules:

X → X+X | X*X |X| a is ambiguous or not.

Solution

Let's find out the derivation tree for the string "a+a*a". It has two derivations

Solution

Derivation 1:Parse tree1:Derivation 2:Parse tree2:

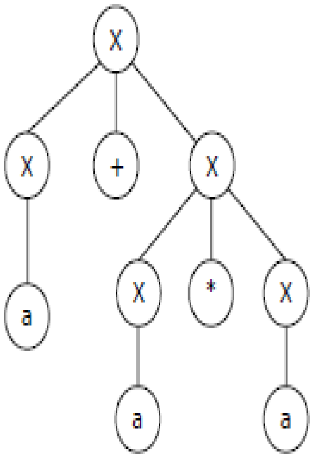
X → X+X

X→ a +X

X→ a+ X*X

X →a+a*X

X→ a+a*a



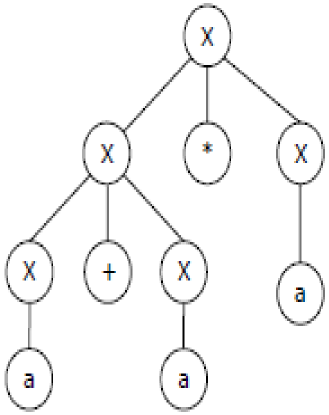
X→ X*X

X→ X+X*X

X→ a+X*X

X→ a+ a*X

X→ a+a*a



Since there are two parse trees for a single string "a+a*a", the grammar *G* is ambiguous

2. Consider the following CFG,

S -> aS|aSbS|ε

Show that derivation for the string " aab" is ambiguous.

3. CFG : S->SS| aSb|bSa|€

Given string w=aabb find whether this G is ambigeouse or not