Context Free Grammar (CFG)
&
Context Free Language (CFL)

Recall

- A regular grammar has the form:

$$\mathbb{A} \Rightarrow \boldsymbol{a}\mathbb{A} \mid \boldsymbol{a} \mid \boldsymbol{\varepsilon}$$

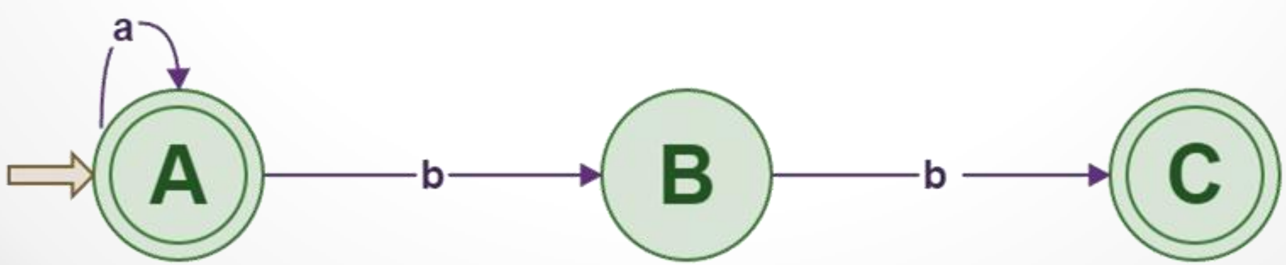
- A regular grammar generates a regular language.
- Regular languages can be accepted by finite acceptors.
- Regular languages can be denoted by regular expressions.

Example

Grammar:

$$\mathbb{A} \Rightarrow \boldsymbol{a}\mathbb{A} \mid \boldsymbol{b}\mathbb{B} \mid \boldsymbol{\varepsilon}$$
$$\mathbb{B} \Rightarrow \boldsymbol{b}$$

- L (G) = a* + bb = E // Expression
- An fa (finite automaton) [M], such that L (M) = L (E)



Context Free Grammar

Every rule has the form:

$$\mathbf{N} \Rightarrow (N+T)^*$$

Examples:

- $L_1 = \{a^n b^n \mid n \geq 1\}$
- $L_2 = \{w \mid w \in \{a,b\}^*, s.t.\ n_a(w) = n_b(w)\}$
- $L(G_3) = ?$

$$G_3: \quad \begin{aligned} \mathbb{S} &\Rightarrow z\mathbb{M}\mathbb{N}z \\ \mathbb{M} &\Rightarrow a\mathbb{M}a \mid z \\ \mathbb{N} &\Rightarrow b\mathbb{N}b \mid z \end{aligned}$$

- $L_4 = \{w \mid w \in \{0,1\}^*, s.t.\ w = xcx^R\}$
- $L_5 = \{w \mid w \in \{0,1\}^*, s.t.\ w = w^R\}$
- $L_6 = \{a^i b^j c^k \mid (i = j) or\ (j = k),\ i,j,k \geq 0\}$

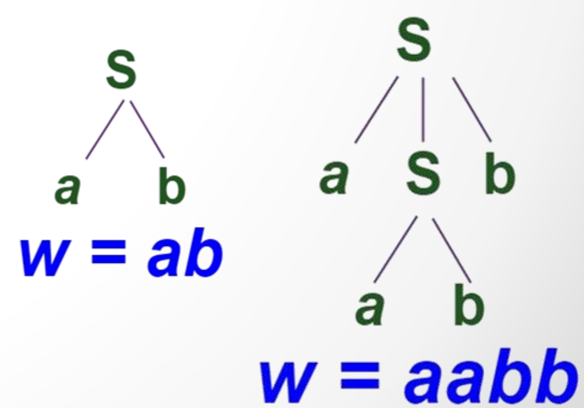
Example (1)

$L_1 = \{a^n b^n | \; n \geq 1\}$

- *'a' might represent a left parenthesis , i.e. , '(' and 'b' a right parenthesis, i.e. , ')'*
- *Every time an 'a' is generated, we must also generate a 'b'.*

- <u>*Consider*</u>

$$G_1: \quad (1)(2)\ \mathbb{S} \Rightarrow a\mathbb{S}b \mid ab$$



Illustration

- **Rule 1** allows us to replace the start symbol "S" with "aSb".

  (Notice: Rule 1 is recursive)

  After applying the first rule (n-1) times, we obtain a sentential string:

  $\mathbf{a^{n-1}\ S\ b^{n-1}}$

- Finally, we must eliminate non-terminals to obtain a word in the language – yielding (**Rule 2**) :

  $\mathbf{a^{n-1}\ a\ b^{n-1}\ b}$ or $\mathbf{a^{n}\ b^{n}}$

$$\mathbb{S} \underset{(\mathbf{1})}{\Rightarrow} \boldsymbol{a\mathbb{S}b} \underset{(\mathbf{1})}{\Rightarrow} \boldsymbol{aa\mathbb{S}bb} \underset{(\mathbf{2})}{\Rightarrow} \boldsymbol{aaabbb}$$

Notice: we generate the string from the outside ("of the onion") to its center.

Example (2)

- $L_2 = \{ w \mid w \in \{a, b\}^*, s.t.\ n_a(w) = n_b(w)\}$

Similar to L_1, every time we generate an ‘a’, we must also generate a ‘b’. However, now order doesn’t matter.

- ***Consider***

$$G_2 : (1)..(4)\ \mathbb{S} \Rightarrow a\mathbb{S}b \mid b\mathbb{S}a \mid \mathbb{S}\mathbb{S} \mid \varepsilon$$

Illustration

$G_2: (1)..(4)\ \mathbb{S} \Rightarrow \boldsymbol{a\mathbb{S}b} \mid \boldsymbol{b\mathbb{S}a} \mid \mathbb{S}\mathbb{S} \mid \boldsymbol{\varepsilon}$

- *Let* $w_1 = ab$ $\quad \mathbb{S} \underset{(1)}{\Rightarrow} a\mathbb{S}b \underset{(4)}{\Rightarrow} ab$
- *Let* $w_2 = ba$ $\quad \mathbb{S} \underset{(2)}{\Rightarrow} b\mathbb{S}a \underset{(4)}{\Rightarrow} ba$
- *Let* $w_3 = abba$

$$\mathbb{S} \underset{(3)}{\Rightarrow} \mathbb{S}\mathbb{S} \underset{(1)}{\Rightarrow} a\mathbb{S}b\mathbb{S} \underset{(2)}{\Rightarrow} a\mathbb{S}bb\mathbb{S}a \underset{(4)}{\Rightarrow} abb\mathbb{S}a \underset{(4)}{\Rightarrow} abba$$

- *Let* $w_4 = babbaa$

$$\mathbb{S} \underset{(2)}{\Rightarrow} b\mathbb{S}a \underset{(3)}{\Rightarrow} b\mathbb{S}\mathbb{S}a \underset{(1)}{\Rightarrow} ba\mathbb{S}b\mathbb{S}a \underset{(4)}{\Rightarrow} bab\mathbb{S}a \underset{(2)}{\Rightarrow} babb\mathbb{S}aa \underset{(4)}{\Rightarrow} babbaa$$

- *In each instance, a leftmost derivation was used.*

Example (3)

In this example, we are given a grammar and must discover what language it generates.

$$\boldsymbol{L(G_3)} = ?$$

$$\begin{array}{llcl} \boldsymbol{G_3}: & \mathbf{(1)} & \mathbb{S} & \Rightarrow \boldsymbol{z}\mathbb{M}\mathbb{N}\boldsymbol{z} \\ & \mathbf{(2)(3)} & \mathbb{M} & \Rightarrow \boldsymbol{a}\mathbb{M}\boldsymbol{a} \mid \boldsymbol{z} \\ & \mathbf{(4)(5)} & \mathbb{N} & \Rightarrow \boldsymbol{b}\mathbb{N}\boldsymbol{b} \mid \boldsymbol{z} \end{array}$$

- From **rule 1**, we know that every string in L_3 begins and ends with a single 'z'.

$$\mathbb{S} \Rightarrow \boldsymbol{z}\mathbb{M}\mathbb{N}\boldsymbol{z}$$

we use rules 2 and 3 here.

we use rules 4 and 5 here.

Illustration

- **Rule 2** yields a prefix and suffix consisting of some number of a's including none.
- **Rule 4** does much the same thing, however with b's
- So far, we have:

$$\boldsymbol{za^n za^n b^m zb^m z} \text{ with } \boldsymbol{m, n \geq 0}$$

- Finally, **rule 3** replaces M with z, and rule 5 does the same with N.

$$\boldsymbol{L(G)_3 = \{ za^n za^n b^m zb^m z \mid m, n \geq 0\}}$$

Example (4)

$$L_4 = \{\, \boldsymbol{w} \mid \boldsymbol{w} \in \{\mathbf{0}, \mathbf{1}\}^*, \boldsymbol{s.t.}\ \ \boldsymbol{w} = \boldsymbol{xcx^R} \,\}$$

- X^R is the reversal of the string X

  i.e., if X = 011 then $X^R = 110$

Strings accepted in L_4 : $\boldsymbol{c}, \mathbf{0}\boldsymbol{c}\mathbf{0}, \mathbf{1}\boldsymbol{c}\mathbf{1}, \mathbf{01}\boldsymbol{c}\mathbf{01}, \mathbf{01}\boldsymbol{c}\mathbf{10}, \ldots \ldots$

‘c’ is known as a center marker.

Example (5)

$$L_5 = \{\, \boldsymbol{w} \mid \boldsymbol{w} \in \{\mathbf{0}, \mathbf{1}\}^*, \boldsymbol{s.t.}\ \ \boldsymbol{w} = \boldsymbol{w}^R \,\}$$

This language generates one string that is the reversal of the input string in each instance (each input string).

Example (6)

$$L_6 = \{ a^i b^j c^k \mid (i = j) or (j = k), \quad i, j, k \geq 0 \}$$

- It might be that i = j and j = k !

$$L_6 = \{\varepsilon, ab, bc, aabb, aabbc, abbbccc, \ldots \ldots\}$$

- We want a CFG (G_6) for L_6.
  How might the "Divide and Conquer" paradigm help?