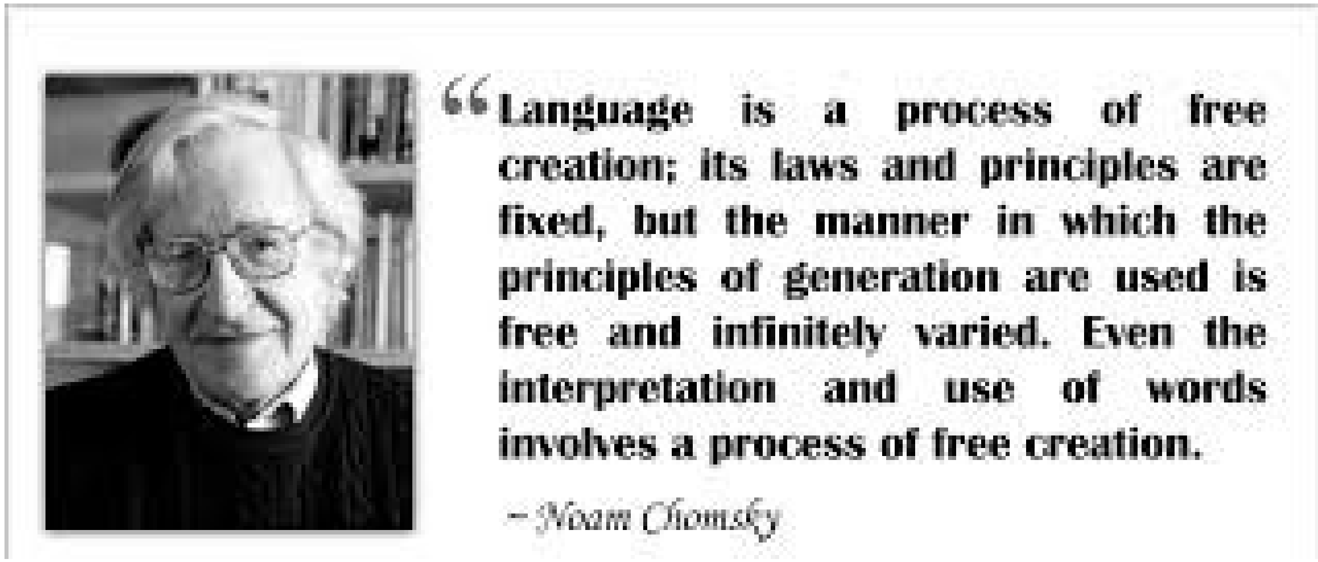
# Chomsky Hierarchy

Dr G Sudha Sadasivam



# Agenda

- Language notation
- Grammar
- Chomsky hierarchy of languages

# Recap

- ______________deals with computational problems that can be solved by abstract machines
- __________ solves problems through mathematical models and algorithms
- _________________classifies the problems on their degree of difficulty
- ___________classifies problems as being solvable or unsolvable

# Language – an Introduction

- **Noam Chomsky** is an American linguist, philosopher, cognitive scientist and social activist. He is one of the fathers of modern linguistics
- **Symbol** – An atomic unit, such as a digit, character, lower-case letter, for example 1,2,a, b…
- **Alphabet** – A finite set of symbols, usually denoted by Σ.

  $\Sigma = \{0, 1\}$ $\Sigma = \{0, a, 9, 4\}$ $\Sigma = \{a, b, c, d\}$
- **String** – A finite length sequence of symbols from some alphabet. Ex, 011011

- **Example** Consider strings w = 0110 y = 0aa x = aabcaa z = 111

  Special string: ε (also denoted by λ)

  Concatenation: wz = 0110111

  Length: |w| = 4 |ε| = 0 |x| = 6

  Reversal: $y^R$ = aa0

- Special strings:

  Σ* All strings of symbols from Σ

  $\Sigma^+$ is Σ* - {ε}

  Example : Consider Σ = {0,1}

{ ε, 0, 1, 00, 01, 10, 11, 000, 100, 010, 001, 110 ,011, 101, 111, ….}

{ x | x is in Σ* }

Language (L) is

1) A set of strings from some alphabet
2) In other words, any subset L of $\Sigma^*$

- Special languages:
  - {} The empty set / language, containing no string.
  - {ε} A language containing one string, the empty string.

**Examples:**

- $\Sigma = \{0, 1\}$

  $L = \{x \mid x \text{ is in } \Sigma^* \text{ and x contains an even number of 0's}\}$

  $= \{010,001,..1001010...\}$

- $\Sigma = \{0, 1, 2,..., 9, .\}$

  $L = \{x \mid x \text{ is in } \Sigma^* \text{ and x forms a finite length real no}\}$

  $= \{0, 1.5, 9.326,...\}$

- Languages are sets. Therefore, all set operations like Union, Intersection, Difference, and Complement can be applied.

## Language rules

- $L\{\varepsilon\} = \{\varepsilon\} L = L$
- $(L1L2)L3 = L1(L2L3)$
- $L^+ = LL^*$
- $L^+ = L^* - \{\varepsilon\}$
- $L = \{\} = \Phi$
- $L\Phi = \Phi L = \Phi$
- Union operation $L1 \cup L2 = \{x \mid x \in L1 \text{ or } x \in L2\}$
- Intersection operation $L1 \cap L2 = \{x \mid x \in L1 \text{ and } x \in L2\}$
- Difference operation $L1 - L2 = \{x \mid x \in L1 \text{ and } x \notin L2\}$
- Complement : $\sim(L1) = \Sigma^* - L1$

- Non-Terminals or Variables (V) are capital letters A to Z
- Terminals are symbols from alphabet Σ given by a to z letters
- α, β, γ represent strings containing terminals and non-terminals
- Grammar is 4 tuple G = (V,Σ,P, S)

  V is a finite set of variables

  Σ is a finite set of terminals

  P is a finite set of rules

  S is the start symbol

Grammars are categorized based on the form of the rules/ productions

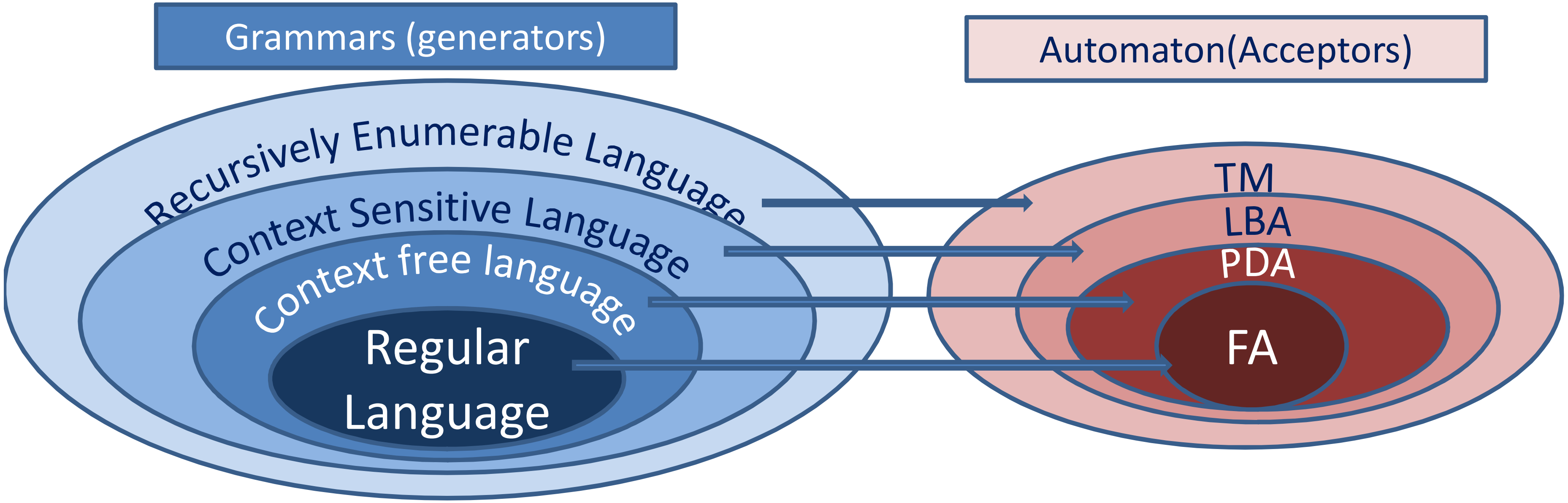
**Example**: Write the grammar to recognise L= $\{a^n b^n \mid n \geq 0\}$

Language L has ϵ, ab, aabb....

P: S → ϵ | aSb ; V={S}, Σ ={a,b}, G: {V, Σ, P, S}

Noam Chomsky divided Grammars / Languages into four classes:

- **Recursively enumerable grammars** –recognizable by a Turing machine
- **Context-sensitive grammars** –recognizable by the linear bounded automaton
- **Context-free grammars** - recognizable by the pushdown automaton
- **Regular grammars** –recognizable by the finite state automaton




$$\mathrm{RG} \subset \mathrm{CFG} \subset \mathrm{CSG} \subset \text{recursively enumerable}$$

$$\mathrm{Type}\,3 \subset \mathrm{Type}\,2 \subset \mathrm{Type}\,1 \subset \mathrm{Type}\,0$$

# Type 0 or Recursively Enumerable Languages

- Unrestricted grammars include all formal grammars that generate language accepted by Turing Machines
- Production / Rules for Type 0 grammar is of the form $\alpha \rightarrow \beta$

Where $\alpha , \beta \in \{ V \cup \Sigma \}^*$

- Any decision problem is a formal language L, where $x \in L$ iff the answer on input x is "yes".

Grammar for $L = \{(ba)^n \mid n>=1\}$

$A \rightarrow SCA \mid \epsilon$
$C \rightarrow ab$
$Sab \rightarrow ba$

$A \rightarrow SCA \rightarrow SabA \rightarrow baA \rightarrow baSCA \rightarrow baSab\varepsilon \rightarrow baba$

# Type 1 or Context Sensitive Language

- CSG generate language accepted by Linear Bounded Automaton
- Production / Rules for Type 1 grammar is of the form $\alpha \rightarrow \beta$ and $|\alpha| \le |\beta|$ Where $\alpha, \beta \in \{V \cup \Sigma\}^*$

**Grammar for $L = \{(bc)^n b^m c : n >= 0\}$**

$$AB \rightarrow AbBc$$
$$A \rightarrow bcA$$
$$B \rightarrow b$$

**Grammar for $L = \{a^n b^n c^n : n > 0\}$**

$$S \rightarrow abc \mid aAbc$$

$$Ab \rightarrow bA$$

$$Ac \rightarrow Bbcc$$

$$bB \rightarrow Bb$$

$$aB \rightarrow aa \mid aaA$$

# Type 2 or Context Free Language

- CFG generate language accepted by Push Down Automaton
- Production / Rules for Type 2 grammar is of the form $\alpha \rightarrow \beta$ and $\alpha \in \{V\}$ Where $\beta \in \{V \cup \Sigma\}^*$

**Grammar for $L = \{w\, c\, w^r : w \in \{a,b\}^*\}$**

$$S \rightarrow aSa \mid bSb \mid c$$

**Generating string "abacaba"**

$$S \rightarrow aSa \rightarrow abSba \rightarrow abaSaba \rightarrow abacaba$$

# Type 3 or Regular Language

- RG generate language accepted by Finite Automaton
- Production / Rules for Type 3 grammar is of the form

  A → a | aB

Grammar for L generating all strings belonging to Σ = {a,b}

$$S \rightarrow aS \mid bS \mid \varepsilon$$

Generating string "abb"

$$S \rightarrow aS \rightarrow abS \rightarrow abbS \rightarrow abb$$

Every Type i Language is of Type i-1

| Type of Language | Language Defined By | Acceptor | Rules | Example |
|---|---|---|---|---|
| regular language | regular expression | finite automaton | $A \to a \mid aB$ | $a^*$ |
| context-free language | context-free grammar | pushdown automaton | $\alpha \to \beta$ & $\alpha \in \{V\}$ Where $\beta \in \{V \cup \Sigma\}^*$ | $a^n b^n$ |
| recursive language CSG | phrase-structure grammar | Turing machine that never loops | $\alpha \to \beta$ and $\lvert\alpha\rvert \le \lvert\beta\rvert$ Where $\alpha, \beta \in \{V \cup \Sigma\}^*$ | $a^n b^n c^n$ |
| recursively enumerable language | phrase-structure grammar | Turing machine | $\alpha \to \beta$ Where $\alpha, \beta \in \{V \cup \Sigma\}^*$ | $(ab)^n$ |

# Work Out

- $LL^*$ =?
- $L^*$-{Є} = ?
- LΦ = ?
- Identify the language type and string generated
  - Sab –> ba
  - S –> AB
    A –> a
    B –> b
  - S –> AB
    AB –> abc
  - S –> a