GRAMMAR

Dr. A. Beulah
AP/CSE

LEARNING OBJECTIVE

- To Understand the need of formal languages, and grammars (K3)

INTRODUCTION

- The theory of formal languages is used in the field of Linguistics- to define valid sentences and give structural descriptions of sentences.

S→<noun> <verb> <adverb>

S→<noun><verb>

<noun>→ Andrew

<noun> → Joe

<verb>→ ran

<verb> → ate

<adverb> → slowly

<adverb> → quickly

T, NT

Variables

NT={S, N, V, A}

S, NT, T, P

T={ . . . }

S→ NVA | NV

INTRODUCTION

- S variable to denote a sentence
- $\rightarrow$ represents a rule meaning that the word on the right side of the arrow can replace the word on the left side of the arrow.
- P collection of rules (or) productions.
- The sentences are derived from the above mentioned productions by:
  - Starting with S
  - Replacing words using the productions
  - Terminating when a string of terminals is obtained.

EXAMPLE

- S→ <noun><verb><adverb>
- S → Joe ate slowly

- S → <noun><verb>
- S → Andrew ran

BACKUS-NAUR FORM

- Backus-Naur Form or Backus Normal Form → BNF
- BNF is formal and precise
  - BNF is a notation for context-free grammars
- BNF is essential in compiler construction
- Example

  <number> ::= <digit> | <number> <digit>

  <digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

  $\Rightarrow$

$n \rightarrow d$

$n \rightarrow nd$

52

$n \Rightarrow nd$

$\Rightarrow dd$

$\Rightarrow 52$

FORMAL GRAMMAR - DEFINITION

A formal grammar G = (N, Σ, P, S) consists of:

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

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

- A finite set *N* of non terminal symbols. $S \in N$
- A finite set Σ of terminal symbols that is disjoint from *N*.
- A finite set *P* of production rules where a rule is of the form
  - string in (Σ U *N*)* → string in (Σ U *N*)*
  - the left-hand side of a rule must contain at least one non terminal symbol.
- A symbol *S* in *N* that is indicated as the start symbol.

$\underbrace{(N \cup T)^*}_{LHS} \rightarrow \underbrace{(T \cup N)^*}_{RHS}$

EXAMPLE

- G = (N, Σ, P, S)
- N = {<sentence>,<noun>,<verb>,<adverb>}

  S
- Σ = {Andrew, Joe, ate, ran ,slowly, quickly}
- S = <sentence>
- P   8

EXAMPLE

- The grammar G with $N = \{S, B\}$, $\Sigma = \{a, b, c\}$, P consisting of the following productions
  - $S \rightarrow aBSc$
  - $S \rightarrow abc$
  - $Ba \rightarrow aB$
  - $Bb \rightarrow bb$

'S'

NOTATIONS

Names Beginning with	Represent Symbols In	Examples
Uppercase	N	A, B, C, Prefix
Lowercase and punctuation	Σ	a, b, c, if, then, (, ;
X, *Y*	N∪Σ	X_i, Y_3
Other Greek letters	(N∪Σ)*	α, β, γ

DERIVATION

- If $\alpha \rightarrow \beta$ is a production in a grammar G and γ, δ are any two strings on $N \cup \Sigma$, then we say $\gamma\alpha\delta$ directly derives $\gamma\beta\delta$ in G.
  - (i.e.) $\gamma\alpha\delta \Rightarrow \gamma\beta\delta$
- This process is called *one-step derivation.*
- In particular, if $\alpha \rightarrow \beta$ is a production, then $\alpha \ {}_G\!\Rightarrow \beta$

DERIVATION

- The purpose of a grammar is to derive strings in the language defined by the grammar
- $\alpha \Rightarrow \beta$, β can be derived from α in one step
- $\Rightarrow^{+}$ derived in one or more steps
- $\Rightarrow^{*}$ derived in any number of steps
- $\Rightarrow_{lm}$ leftmost derivation $\Rightarrow_{lm}$
  - Always substitute the leftmost non-terminal $\Rightarrow_{rm}$
- $\Rightarrow_{rm}$ rightmost derivation
  - Always substitute the rightmost non-terminal

EXAMPLE

- G = ({S}, {0,1}, {S→0S1, S→01}, S) then the derivation is:

N T P S

- S⇒ 0S1

  ⇒ 0011

S → 0S1
S → 01

is a one step derivation, where S is replaced by 01.

0011

$S \Rightarrow 0S1$
$\Rightarrow 0011$

000111

$S \Rightarrow 0S1$
$\Rightarrow 00S11$
$\Rightarrow 000111$

$S \overset{*}{\Rightarrow} 0011$

$S \overset{*}{\Rightarrow} 000111$

$S \overset{*}{\Rightarrow} w$

EXAMPLE

$$S \rightarrow AB$$
$$B \rightarrow b$$
$$A \rightarrow aA \mid c$$

- Derivation

$$
\begin{aligned}
S &\Rightarrow AB \\
&\Rightarrow aAB && A \rightarrow aA \\
&\Rightarrow aAb && B \rightarrow b \\
&\Rightarrow aaAb && A \rightarrow aA \\
&\Rightarrow aacb && A \rightarrow c
\end{aligned}
$$

LANGUAGE

- The language generated by a grammar G, L(G) is defined as $\{w \in \Sigma^* \mid S \Rightarrow^* w\}$.
- The elements of L(G) are called sentences.
- Stated in simple way, L(G) is the set of all terminal strings derived from the start symbol S.

$$L(G) = \{w \mid w \in T^*, S \xRightarrow{*} w\}$$

LANGUAGE

- G_1 and G_2 are equivalent if $L(G_1) = L(G_2)$
- $A \rightarrow \alpha_1$, $A \rightarrow \alpha_2$....... $A \rightarrow \alpha_m$ said to be A-productions, rewritten as

$$A \rightarrow \alpha_1 \mid \alpha_2 \mid .....\alpha_m$$

CHOMSKY HIERARCHY OF LANGUAGES

Ms. A. Beulah

AP/CSE

LEARNING OBJECTIVE

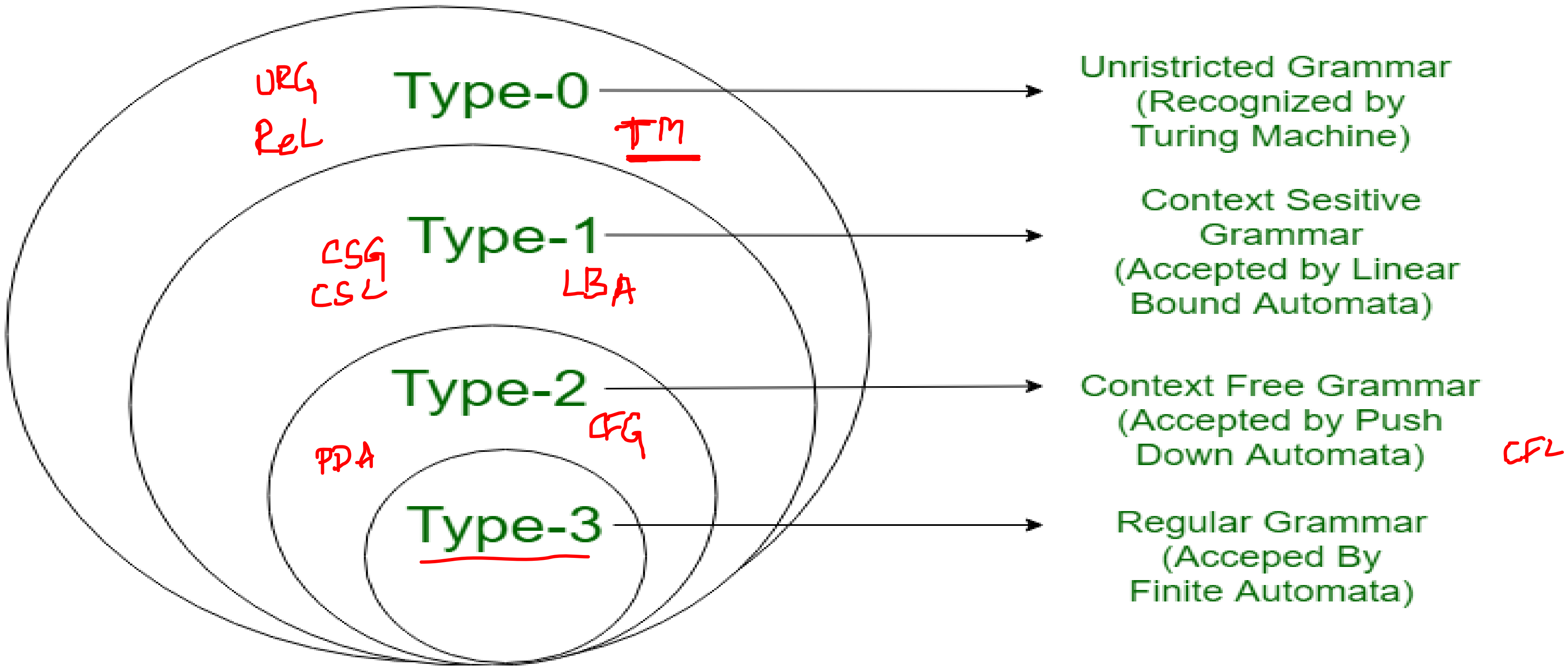
- To Understand the need of formal languages, and grammars (K3)

CHOMSKY HIERARCHY OF LANGUAGES

- According to Chomsky hierarchy, grammars are divided of 4 types:
  - Type 0 → Unrestricted grammar. → Recursively enumerable language (REL)
  - Type 1 → Context sensitive grammar. → CSL
  - Type 2 → Context free grammar. → CF Language CFL
  - Type 3 → Regular Grammar. → Regular language (RL)

CHOMSKY HIERARCHY OF LANGUAGES

- $L_{rl} \subseteq L_{cfl} \subseteq L_{csl} \subseteq L_{rel}$



TYPE 0: UNRESTRICTED GRAMMAR

- Type 0 grammar language are recognized by TM.
- Languages → Recursively Enumerable languages.
- Grammar Production in the form of

$\gamma A \delta \rightarrow \gamma \alpha \delta$

$(T \cup N)^* \longrightarrow (T \cup N)^*$

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

P

LHS → RHS

where

γ is left content, δ is right content

γ, δ is (V / T)*

V : Variables or Non Terminals

T : Terminals.

$\gamma A \delta \longrightarrow \gamma \alpha \delta$

$\rightarrow \delta$

$(T \cup N)^*$

In type 0 there must be at least one variable or Non-terminal on left side of production.

TYPE 0: UNRESTRICTED GRAMMAR

abAbcd→abABbcd

where ab-left context

bcd-right context

α is AB

$\underset{\gamma}{ab}\,A\,\underset{\delta}{bcd}$

ab AB bcd

AC→A ε

where α ε

A - left context

ε - right context

α is ε

C → ε

A→γδα

A→γα

ε

ε - left and right context

α is ε

TYPE 1: CONTEXT SENSITIVE GRAMMAR

- Type-1 grammars generate the context-sensitive languages.
- The language generated by the grammar are recognized by the Linear Bound Automata

  In Type 1

  I. All Type 1 grammar should be Type 0.

  II. Grammar Production in the form of

$\gamma A\delta \rightarrow \gamma\alpha\delta$

$S \rightarrow \varepsilon$ × Type 1

$S \rightarrow \varepsilon$

S

RHS

where

γ is left content, δ is right content

γ, δ is (V / T)*

V : Variables or Non Terminals

T : Terminals.

$\alpha \neq \varepsilon$

The rule **$S \rightarrow \varepsilon$** is allowed if S does not appear on the right side of any rule

TYPE 1: CONTEXT SENSITIVE GRAMMAR

$aAbcD \rightarrow abcDbcD$

where a - left context

bcD - right context

A is replaced by bcD $\neq \varepsilon$

$AB \rightarrow AbBC$

where A - left context

ε - right context

B is replaced by bBC $\neq \varepsilon$

TYPE 1: CONTEXT SENSITIVE GRAMMAR

$A \rightarrow abA$

where ε - left & right context.

A - $abA \neq \varepsilon$

$A \rightarrow \varepsilon$ is allowed but A does not appear on the right handside of any production of the grammar

$S \rightarrow \varepsilon$.

$A \rightarrow \varepsilon$ A RHS

TYPE 2: CONTEXT FREE GRAMMAR

- Type-2 grammars generate the context-free languages.
- The language generated by the grammar is recognized by a Pushdown automata.
- In Type 2,

  1. All Type 2 grammar should be Type 1.
  2. Left hand side of production can have only one variable NT.

A→α

$A \rightarrow \alpha$

$\downarrow$

NT

$(N \cup T)^*$

$NT \rightarrow (N \cup T)^*$

TYPE 2: CONTEXT FREE GRAMMAR

$S \rightarrow Aa$

$A \rightarrow a$

$B \rightarrow abc$

$A \rightarrow \varepsilon$

TYPE 3: REGULAR GRAMMAR

- Type-3 grammars generate regular languages.
- The language generated by the regular grammar is accepted by a finite automata.
- Type 3 is most restricted form of grammar.
- Type 3 should be in the given form only :

V → VT / T (left linear grammar) –NT left

(or)

V → TV /T (right linear grammar)-NT

L

N → T / NT   TT ✗

N → T / TN

A → ε   A x RHS

TYPE 3: REGULAR GRAMMAR

$S \rightarrow b \mid c$

$S \rightarrow bA$

$S \rightarrow a$

$A \rightarrow \varepsilon$ is allowed but A does not appear on the right handside of any production of the grammar

IDENTIFY THE TYPE

S→ Aa 3

A→c | BaB 3 2

B→abc 2

CFG

3 RG

TO

- A→BaB, B→abc - type 2.

  (because productions are of the form A→α)

- S→Aa, A→ c - type 3 (because production of the form A→a)

∴ the type number is 2 (CFG).

IDENTIFY THE TYPE

S→ ASB^{2} | d$_{3}$

A→aA$_{3}$

CFG

- S→ ASB - type 2 (i.e. A→α)
- S→ d, A→aA - type 3 (i.e. A→a, A→aB)

∴ the highest type number is 2.

IDENTIFY THE TYPE

$S \rightarrow 0SA2$	2
$S \rightarrow 012$	2
$2A \rightarrow 2A12$	1
$1A \rightarrow 11$	1

IDENTIFY THE TYPE

$S \rightarrow 0SA2$

$S \rightarrow 012$

$2A \rightarrow A12$

$1A \rightarrow 11$

- Context sensitive grammar - type 1

IDENTIFY THE TYPE

$S \rightarrow aSBC \mid aBC$

$CB \rightarrow BC$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$

IDENTIFY THE TYPE

$S \rightarrow aSBC \mid aBC$

$CB \rightarrow BC$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$

Context sensitive grammar - type 1

IDENTIFY THE TYPE

$S \rightarrow aSa \mid bSb$

$S \rightarrow a \mid b$

$S \rightarrow \lambda$

IDENTIFY THE TYPE

$S \rightarrow aSa \mid bSb$

$S \rightarrow a \mid b$

$S \rightarrow \lambda$

Context free grammar - type 2

SUMMARY

- Definition of Grammar
- Notations followed in grammar
- Different types of grammar
- Language of a grammar

TEST YOUR KNOWLEDGE

- The entity which generate Language is termed as:

  a) Automata

  b) Tokens

  c) Grammar

  d) Data

- The minimum number of productions required to produce a language consisting of palindrome strings over ∑={a,b} is

  a) 3

  b) 7 c) 5 d) 6

TEST YOUR KNOWLEDGE

- The Grammar can be defined as: G=(V, ∑, p, S)
  In the given definition, what does S represents?
  a) Accepting State
  b) Starting Variable
  c) Sensitive Grammar
  d) None of these

LEARNING OUTCOME

On successful completion of this topic, the student will be able to:

- To Understand the need of formal languages, and grammars (K3)

REFERENCE

- Hopcroft J.E., Motwani R. and Ullman J.D, “Introduction to Automata Theory, Languages and Computations”, Second Edition, Pearson Education, 2008