Chapter 12: Context-Free Grammar

I. Theory of Automata

→ II. Theory of Formal Languages

III. Theory of Turing Machines …

- programming languages
- compiling a program: an operation that generates an equivalent program in machine or assembler language.
- 2 phases:
  1. parsing ←
  2. translation to machine language

Example: AE (Arithmetic Expressions)

- Rule 1: Any number is in AE
- Rule 2: If x and y are in AE, then so are:

  (x) – (x) (x+y) (x–y) (x*y)

A different way for defining the set AE is to use a set of substitutions rules similar to the grammatical rules:

Substitution rules that define the AE's:

S → EA
EA → (EA + EA)
EA → (EA–EA)
EA → (EA*EA)
EA → (EA)
EA → –(EA)
EA → NUMBER

NUMBERS??

NUMBER → FIRST-DIGIT
FIRST-DIGIT → FIRST-DIGIT OTHER-DIGIT
FIRST-DIGIT → 1 2 3 4 5 6 7 8 9
OTHER-DIGIT → 0 1 2 3 4 5 6 7 8 9

S ⇒ EA ⇒ (EA*EA) ⇒ ((EA+EA)*EA) ⇒ ((EA+EA)*(EA+EA))
… ⇒ ((3+4)*(6+7))

EA:
S → EA
EA → (EA + EA)
EA → (EA–EA)
EA → (EA*EA)
EA → (EA)
EA → –(EA)
EA → NUMBER

How to generate the number 1066?

NUMBER ⇒ FIRST-DIGIT
⇒ FIRST-DIGIT OTHER-DIGIT
⇒ FIRST-DIGIT OTHER-DIGIT OTHER-DIGIT
⇒ FIRST-DIGIT OTHER-DIGIT OTHER-DIGIT OTHER-DIGIT
⇒ 1 0 6 6

NUMBERS
NUMBER → FIRST-DIGIT
FIRST-DIGIT → FIRST-DIGIT OTHER-DIGIT
FIRST-DIGIT → 1 2 3 4 5 6 7 8 9
OTHER-DIGIT → 0 1 2 3 4 5 6 7 8 9

Definition: A context free grammar (CFG) is:

1. an alphabet S of letters, called terminals.
2. a set of symbols, called nonterminals or variables. One symbol S is called the start symbol.
3. a finite set of productions of the form:

   A → a

   where A is a nonterminal and a is a finite sequence (word) of nonterminals and terminals.

- Examples:

Terminals: (,), +, -, *, numbers
Nonterminals: S, EA

EA:
S → EA
EA → (EA + EA)
EA → (EA–EA)
EA → (EA*EA)
EA → (EA)
EA → –(EA)
EA → NUMBER

NUMBERS

NUMBER	→ FIRST-DIGIT
FIRST-DIGIT	→ FIRST-DIGIT OTHER-DIGIT
FIRST-DIGIT	→ 1 2 3 4 5 6 7 8 9
OTHER-DIGIT	→ 0 1 2 3 4 5 6 7 8 9

Terminals : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Nonterminals : S, FIRST-DIGIT, OTHER-DIGIT

Definition: A sequence of applications of productions starting with the start symbol and ending in a sequence of terminals is called a derivation.

Definition: The language generated by a CFG is the set of all sequences of terminals produced by derivations. We also say language defined by, language derived from, or language produced by the CFG.

Definition: A language generated by a CFG is called a context-free language.

Examples: $S \to aS$ $S \to \Lambda$

$S \Rightarrow aS$
$\Rightarrow aaS$
$\Rightarrow aaaS$
$\Rightarrow aaaaS$
$\Rightarrow aaaaaS$
$\Rightarrow aaaaaaS$
$\Rightarrow aaaaaa\Lambda = aaaaaa$

Generated Language:
$\{\Lambda, a, aa, aaa, \dots\}$ = language(**a***).

$S \to SS$ $S \to a$ $S \to \Lambda$

S $\Rightarrow$ SS
$\Rightarrow$ SSS
$\Rightarrow$ SaS
$\Rightarrow$ SaSS
$\Rightarrow$ ΛaSS
$\Rightarrow$ ΛaaS
$\Rightarrow$ ΛaaΛ=aa

Generated Language:
{Λ, a, aa, aaa, …} = language(**a***).

(An infinite number of derivations for the word aa.)

In general: variables: Upper case letters
terminals: Lower case letters

The empty word:
is it a nonterminal? $\Lambda \to$ …
no
a terminal? ΛaaΛ = aa
not exactly, because it is erased.

$N \to \Lambda$ N can simply be deleted.

- $S \to aS$ $S \to bS$ $S \to a$ $S \to b$
  S $\Rightarrow$ aS $\Rightarrow$ abS $\Rightarrow$ abbS $\Rightarrow$ abba
- $S \to X$ $S \to Y$ $X \to \Lambda$ $Y \to aY$
  $Y \to bY$ $Y \to a$ $Y \to b$
- $S \to aS$ $S \to bS$ $S \to a$ $S \to b$ $S \to \Lambda$
  S $\Rightarrow$ aS $\Rightarrow$ abS $\Rightarrow$ abbS $\Rightarrow$ abbaS $\Rightarrow$ abba
- $S \to aS$ $S \to bS$ $S \to \Lambda$

S → XaaX   X → aX   X → bX   X → Λ

S ⇒ XaaX ⇒ aXaaX ⇒ abXaaX ⇒ abXaabX ⇒ abaab

How many derivations are possible for the word baabaab?

S → XY
X → aX
X → bX
X → a
Y → Ya
Y→ Yb
Y → a

Abbreviation:

S → XY
X → aX | bX | a
Y → Ya | Yb | a

S → SS / ES / SE / Λ /DSD
E → aa | bb
D → ab | ba

EVEN-EVEN=language([**aa+bb+(ab + ba)(aa+bb)*(ab+ba)]***)

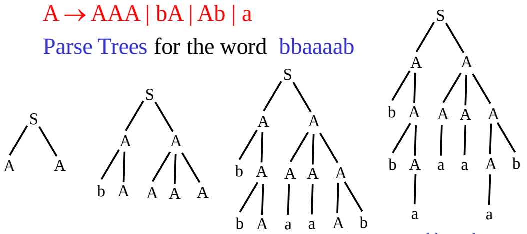
S → aSb / Λ

S ⇒ aSb ⇒ aaSbb ⇒ aaaSbbb ⇒ aaaaSbbbb
⇒ aaaaaSbbbbb ⇒ aaaaabbbbb

S → aSa / bSb / Λ

S → AA
A → AAA | bA | Ab | a

Parse Trees for the word bbaaaab



bbaaaab

Parse trees are also called syntax trees,generation trees, production trees, or derivation trees.

<u>Remark:</u> In a parse tree every internal nodes is labelled with a variable (nonterminal) and every leaf is labelled with a terminal.

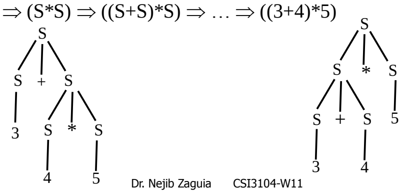
Example:

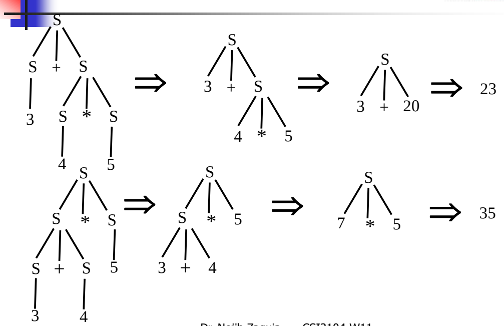
S → (S+S) | (S*S) | NUMBER

NUMBER → …

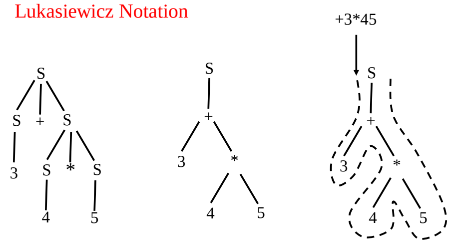
$S \Rightarrow (S+S) \Rightarrow (S+(S*S)) \Rightarrow \ldots \Rightarrow (3+(4*5))$

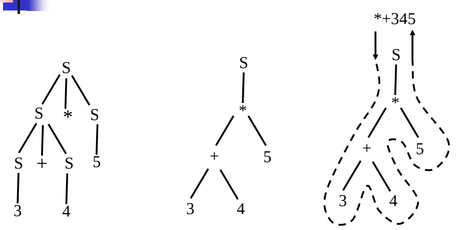
$S \Rightarrow (S*S) \Rightarrow ((S+S)*S) \Rightarrow \ldots \Rightarrow ((3+4)*5)$

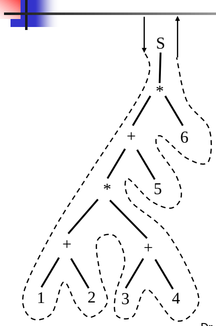




Lukasiewicz Notation





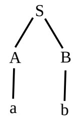


* + * + 1 2 + 3 4 5 6 + 1 2
⇓
* + * 3 + 3 4 5 6 + 3 4
⇓
* + * 3 7 5 6 * 3 7
⇓
* + 21 5 6 + 21 5
⇓
* 26 6 * 26 6
⇓
156

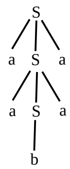
Example: $S \to AB \quad A \to a \quad B \to b$

$S \Rightarrow AB \Rightarrow aB \Rightarrow ab$

$S \Rightarrow AB \Rightarrow Ab \Rightarrow ab$



- *A CFG is ambiguous if there is at least one word in the language that has at least two derivation trees. It is called unambiguous otherwise.*

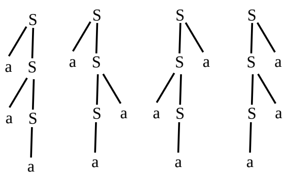


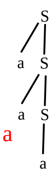
- Example: $S \to aSa \quad S \to bSb$
  $S \to a \quad S \to b \quad S \to \Lambda$

$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabaa$

Language(**a+**)

Example: $S \to aS \mid Sa \mid a$



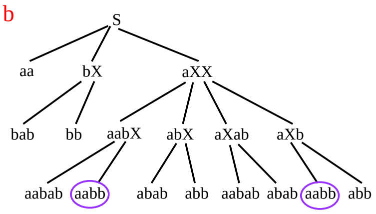


Example: $S \to aS \mid a$

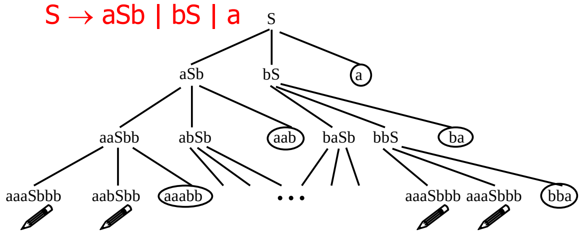
Total Language Trees

$S \to aa \mid bX \mid aXX$

$X \to ab \mid b$



$S \rightarrow aSb \mid bS \mid a$



$S \rightarrow X \mid b$

$X \rightarrow aX$