جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

قسم الامن السيبراني
DEPARTMENT OF CYBER SECURITY

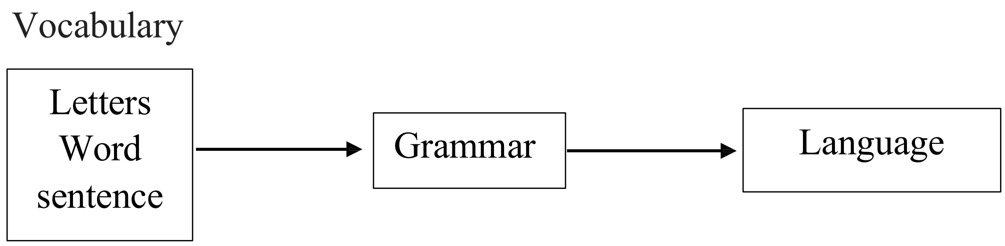
SUBJECT: COMPUTATION THEORY
CLASS: 3rd

LECTURER: MSC :MUNTATHER AL-MUSSAWEE

LECTURE: (5)
LANGUAGE GRAMMAR

Language Grammar

Language Grammar are the foundations and principles; through which we can link the vocabulary.



Example: Let the following grammar:

Sentence = Noun Phrase (NP) + Verb Phrase (VP) + Noun Phrase (NP)

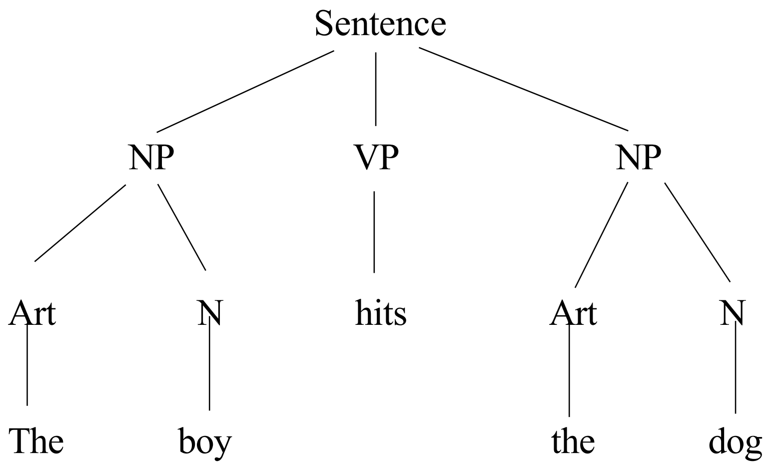
NP = Article (Art) + Noun (N)

Art = a or an or the

Noun = Man, Car, House, Dog, …

VP = eat, play, write, read, …

"The boy hits the dog"
NP VP NP



Phrase Tree

في المثال اعلاه الجملة مكونة من مفردات وقواعد تنتمي الى نفس اللغة، لذلك فأن هذه الجملة صحيحة ووفق قواعد هذه اللغة هي مقبولة (Accept).

Example: "The dog eats the house"

NP VP NP

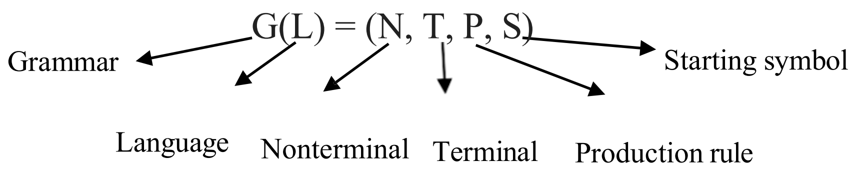
✍ الجملة اعلاه وفق القواعد صحيحة لكن من ناحية المعنى ليس لها معنى. من هذا نستنتج ان الجملة يجب ان تتكون من جزئين مترابطين هما القواعد (syntax) والمعنى (semantic).

Terminal Symbol (T): The words that cannot be replaced by anything are called terminals.
Non-terminal Symbol (N): The words that must be replaced by other things we call non-terminals.

Grammars

A ***grammar*** is a set of rules which are used to construct a language (combine words to generate sentences).
Grammars are containing four things:

1- A finite set of Nonterminal Symbols (N).
2- A finite set of Terminal Symbols (T).
3- A finite set of production rules (P) of the form $u \rightarrow v$; $(u, v) \in (N \cup T)^*$
4- Starting symbol (S).



Example: Let G(L) = ({S, A, B}, {a, b}, P, S) where P denoted as:

$S \rightarrow aA \mid bB \mid a \mid b$ rule 1
$A \rightarrow aA \mid a$ rule 2
$B \rightarrow bB \mid b$ rule 3

1- Is the string "aa" Accept or not?

$S \rightarrow aA$ using rule 1 ($S \rightarrow aA$)
$\rightarrow aa$ using rule 2 ($A \rightarrow a$)
The string is Accept

2- Is the string "bbb" Accept or not?

S → bB using rule 1 (S → bB)
→ bbB using rule 3 (B → bB)
→ bbb using rule 3 (B → b)

The string is Accept

3- Is the string "aaba" Accept or not?

S → aA using rule 1 (S → aA)
→ aaA using rule 2 (A → aA)

The string is not Accept

Example: Let G(L) = ({S, B, C}, {a, b, c}, P, S) where P denoted as:

S → aSBC | aBC rule 1
CB → BC rule 2
aB → ab rule 3
bB → bb rule 4
bC → bc rule 5
cC → cc rule 6

1- Is the string "abc" Accept or not?

S → aBC using rule 1 (S → aBC)
→ abC using rule 3 (aB → ab)
→ abc using rule 5 (bC → bc)

The string is Accept

2- Is the string "$a^2b^2c^2$" Accept or not?

S → aSBC using rule 1 (S → aSBC)
→ aaBCBC using rule 1 (S → aBC)
→ aaBBCC using rule 2 (CB → BC)
→ aabBCC using rule 3 (aB → ab)
→ aabbCC using rule 4 (bB → bb)
→ aabbcC using rule 5 (bC → bc)
→ aabbcc using rule 6 (cC → cc)

The string is Accept

Homework: Let G(L) = ({S, B, C}, {a, b, c}, P, S) where P denoted as:

$S \to aSBC \mid aBC$

$CB \to BC$

$aB \to ab$

$bB \to bb$

$bC \to bc$

$cC \to cc$

1- Is the string "$a^3b^3c^3$" Accept or not?
2- Is the string "a^3b^2" Accept or not?

Context-Free Grammar (CFG)

CFG stands for context-free grammar. It is a formal grammar which is used to generate all possible patterns of strings in a given formal language. Context-free grammar G can be defined by four tuples as:

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

Where,

1- **G** is the grammar, which consists of a set of the production rule. It is used to generate the string of a language.
2- **T** is the final set of a terminal symbol. It is denoted by lower case letters.
3- **N** is the final set of a non-terminal symbol. It is denoted by capital letters.
4- **P** is a set of production rules, which is used for replacing non-terminals symbols (on the left side of the production) in a string with other terminal or non-terminal symbols (on the right side of the production).
5- **S** is the start symbol which is used to derive the string. We can derive the string by repeatedly replacing a non-terminal by the right-hand side of the production until all non-terminal have been replaced by terminal symbols.

Example: Let G(L) = ({S}, {a}, P, S), where P is:

$S \to aS$ rule 1

$S \to \wedge$ rule 2

If we apply production ($S \to aS$) four times and then apply production ($S \to \wedge$) we generate the following string: $\mathbf{a^4}$

S → aS using rule 1
→ aaS using rule 1
→ aaaS using rule 1
→ aaaaS using rule 1
→ aaaa∧ using rule 2

The RE = a* can generate a set of string {∧, a, aa, aaa,. ....}. We can have a null string because S is a start symbol and rule 2 gives S →∧ .

✍ نلاحظ في هذه القواعد بأنه يمكن التكرار بأي عدد من الخطوات والتوقف في اي مرحلة من الاشتقاق، والصيغة العامة للكلمات الناتجة من هذه القواعد:

$$\{a^n, n \geq 0 \text{ by n steps}\}$$

Example: Construct a CFG for the regular expression (0+1)*= ^ 0 1 00 01 11

Solution:

The CFG can be given by,

Production rule (P):

S → 0S | 1S
S → ∧

The rules are in the combination of 0's and 1's with the start symbol. Since (0+1)* indicates {∧, 0, 1, 01, 10, 00, 11, …}.

Example: Construct a CFG for a language L = $\{wcw^R: w \in (a, b)^*\}$.

Solution:

The string that can be generated for a given language is {aacaa, bcb, abcba, bacab, abbcbba,.... }

The grammar could be:

S → aSa rule 1
S → bSb rule 2
S → c rule 3

Now if we want to derive a string "abbcbba", we can start with start symbols.

S → aSa
→ abSba using rule 2
→ abbSbba using rule 2
→ abbcbba using rule 3

Thus any of this kind of string can be derived from the given production rules.

Example: Construct a CFG for the language $\{a^n b^{2n}$ where $n>=1\}$.

Solution:

The string that can be generated for a given language is {abb, aabbbb, aaabbbbbb, ...}.

The grammar could be: **there is another grammar (H.W)**

S → aSbb | abb

Now if we want to derive a string "aabbbb", we can start with start symbols.

S → aSbb

→ aabbbb

Homework: Construct a CFG for the language $\{a^m\ b^n \mid m >= n\}$