7 Context Free Grammars

Wednesday, February 14, 2024 12:07 PM

Why? Because Regex cannot specify programms in a programming language.

([]{¿?}) BEGIN... END.

Another attempt at the specification problem

- GRAMMARS:

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

N: set of non-terminal symbols
T: set of terminal symbols Σ
N and T are disjoint.
$S \in N$: the "start" non terminal
P: a set of "production rules"
rules of the form:

$A \to \alpha$ where $A \in N$

$\alpha \in (N \cup T)^*$

i.e. α is a string of terminals and non-terminals.
Note α could be the empty string λ

- in a production rule $A \to \alpha$ (A: head, α: body)

- shorthand

$A \to \alpha_1$
$A \to \alpha_2$
$A \to \alpha_3$

$\Big\}$ $A \to \alpha_1 \mid \alpha_2 \mid \alpha_3$

E.G.

$N = \{A, B, C\}$
$T = \{a, b\}$
$S = A$

$P \begin{cases} A \to a \\ A \to B \\ C \to AbC \\ B \to aCBbbb \end{cases}$ $A \to \lambda$

- Intuition: Production Rules are rules used to rewrite Strings.

e.g.

$aAbbBCba \xrightarrow{B \to aCBbbb} aAbbaCBbbbCba$

- Definitions:
  - sentence :- a string of terminals and non-terminals
    a string in $(N \cup T)^*$
  - Application of rule P in sentence β
    $P: A \to \alpha$
    replacing one occurrance of A in β with α
  - Derivation:-
    A sequence of sentences each obtained from the previous one by application of a rule
  - The language specified by a Grammar:-

    $w \in L$ iff there exists a derivation from S to w using the rules of the grammar.
    $w \in T^*$

E.G.

$N = \{A, B, C\}$
$T = \{a, b\}$
$S = A$

P:
- $A \to a$
- $A \to B$
- $C \to AbC$
- $B \to aCBbbb$
- $A \to \lambda$
- $B \to bb \mid C$
- $C \to aa \mid b$

what words belong to the Language specified by this grammar?

$\{a, b, aa, bbb\}$

$A \to a$

$A \to B \to C \to b$

$A \to B \to C \to aa$

$A \to B \to C \to AbC \to bC \to bAbC \to bbC \to bbb$

E.G.

$N = \{S, \quad\}$
$T = \{(\ )\ a\ b\ ¿\ ?\}$
$S = S$

P:
- $S \to AS \mid \lambda$
- $A \to BA \mid C \mid D \mid \lambda$
- $B \to a \mid b$
- $C \to (S)$
- $D \to ¿S?$

} every time a (is used, a) is also introduced.

$aa(ba)a¿a(b)?$

$S \to AS \to BAS \to aAS \to aBAS \to aaAS \to aaCS \to aa(S)S \to$
$aa(AS)S \to aa(BAS)S \to aa(bAS)S \to aa(b \ldots\ldots$

- MORE ON DERIVATIONS

Leftmost Derivation :- a derivation where the leftmost non-terminal is the one re-written.

Rightmost Derivation:- a derivation where the rightmost non-terminal is the one re-written.

E.G.

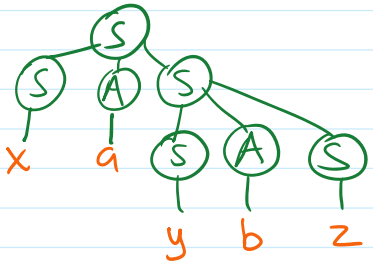
$S \to SAS \mid x \mid y \mid z$
$A \to a \mid b$

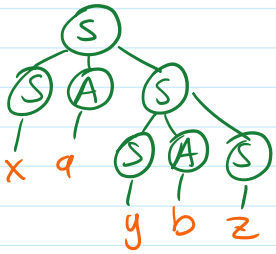
Prove: $xaybz \in L$

Leftmost:
S
SAS
xAS
xaS
xaSAS
xayAS
xaybS
xaybz

RightMost:
S
SAS
SASAS
SASAz
SASbz
SAybz
Saybz
xaybz

Parse Tree.- A tree representation of a derivation
- root : the start symbol
- leafs: terminal symbol
- intermediate nodes are labeled by non-terminal symbols
- a node has its rewriten terms as its children.





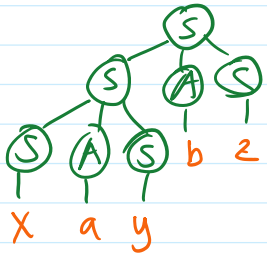
- AMBIGUITY IN GRAMMARS

A grammar is called ambiguous when:

- AMBIGUITY IN GRAMMARS

A grammar is called ambiguous when:
there exists a word w in the language that has
- more than one left-most derivations
or
- more than one right-most derivations
or
- more than one distinct parse tree

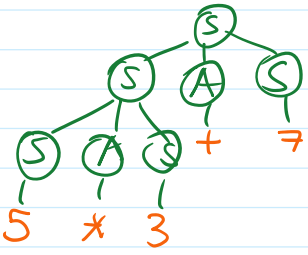
E.G.
A different Parse tree

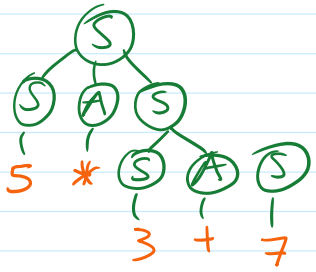


E.G.

$S \to SAS \mid 5 \mid 3 \mid 7$
$A \to + \mid *$

$w = 5 * 3 + 7$





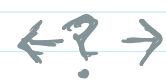
This is preferable!!

E.G. "the dangling else" – Algol '60

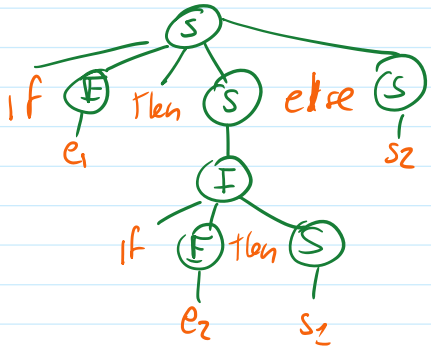
$S \to I \mid P \mid s$
$I \to$ if E then S | if E then S else S
$E \to e$

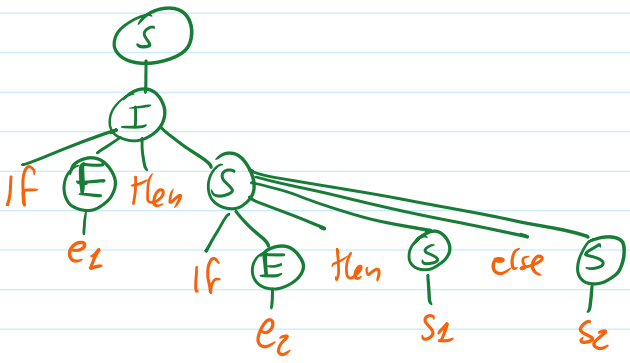
w = if e_1 then if e_2 then s_2 else s_2

if e_2 then
  if e_2 then
    s_1
  else
    s_2



if e_1 then
  if e_2 then
    s_1
  else
    s_2





Note: "an 'else' statement is paired with the closest 'then'."

—0— EOF