Es)

$\Sigma = \{a, b, c\}$

$L_1 = \{a^i b^j c^k \mid i, j, k \geq 1\}$

$\rightarrow aa^*bb^*cc^*$

$S \rightarrow S_1 S_2 S_3$

$S_1 \rightarrow a \mid aS_1$ → at least 1 a

$S_2 \rightarrow b \mid bS_2$ → at least 1 b

$S_3 \rightarrow c \mid cS_3$ → at least 1 c

another approach:

$S \rightarrow S_1$

$S_1 \rightarrow aS_1 \mid aS_2$

$S_2 \rightarrow bS_2 \mid bS_3$

$S_3 \rightarrow cS_3 \mid c$

$L_2 = \{ a^i b^j c^k \mid i,j,k \geq 1,\ \ j \neq k \}$

$S \to S_1 F$

$S_1 \to a \mid aS_1$

$F \to G \mid H$

$G \to bG'c \mid bGc$ $\quad$ $G: \#_b = \#_c$

$G' \to b \mid bG'$ $\quad$ $G': \#_b > \#_c$

$H \to bH'c \mid bHc$ $\quad$ $H: \#_b = \#_c$

$H' \to c \mid cH'$ $\quad$ $H': \#_b < \#_c$

Sentential form

Let $G = (V, T, P, S)$ be a CFG and let $\alpha \in (V \cup T)^*$

- if $S \stackrel{*}{\Rightarrow} \alpha$ we say that α is a **sentential form**
- if $S \underset{lm}{\stackrel{*}{\Rightarrow}} \alpha$ we say that α is a **left sentential form**
- if $S \underset{rm}{\stackrel{*}{\Rightarrow}} \alpha$ we say that α is a **right sentential form**

Note : $L(G)$ contains the sentential forms in T^*

Examples

Consider previous CFG G for a fragment of arithmetic expressions. Then $E * (I + E)$ is a sentential form, since

$$E \Rightarrow E * E \Rightarrow E * (E) \Rightarrow E * (E + E) \Rightarrow E * (I + E)$$

This derivation is neither leftmost nor rightmost

$a * E$ is a leftmost sentential form, since

$$E \underset{lm}{\Rightarrow} E * E \underset{lm}{\Rightarrow} I * E \underset{lm}{\Rightarrow} a * E$$

$E * (E + E)$ is a rightmost sentential form, since

$$E \underset{rm}{\Rightarrow} E * E \underset{rm}{\Rightarrow} E * (E) \underset{rm}{\Rightarrow} E * (E + E)$$

Test

Define a CFG for each of the following languages, describing for each variable the set of generated strings

- $L = \{w \mid w = x2x^R,\ x \in \{0,1\}^*\}$ $S \to 1S1 \mid 0S0 \mid 2$
- $L = \{w \mid w = a^i b^j c^k,\ i,j,k \geqslant 1,\ j \neq k\}$ (done before)

Test

Describe in words the language generated by the following CFG

$$G = (\{S, Z\}, \{0, 1\}, P, S)$$

where

$$P = \{S \rightarrow 0S1 \mid 0Z1,\ Z \rightarrow 0Z \mid \epsilon\}$$

$$\Downarrow$$

$$L = \{w \mid w = 0^i 1^j,\ i \geq j \geq 1\}$$

Derivation composition

We can always compose two derivations $A \overset{*}{\Rightarrow} \alpha B \beta$ and $B \overset{*}{\Rightarrow} \gamma$ into a single derivation

$$A \overset{*}{\Rightarrow} \alpha B \beta \overset{*}{\Rightarrow} \alpha \gamma \beta$$

This follows from the hypothesis about rewriting being **independent** from the context (context-free)

Example

Consider our CFG for generating arithmetic expressions. Starting with

$$E \Rightarrow E + E \Rightarrow E + (E)$$
$$E \Rightarrow I \Rightarrow Ib \Rightarrow ab$$

we can compose at the rightmost occurrence of E, obtaining

$$E \Rightarrow E + E \Rightarrow E + (E) \Rightarrow E + (I) \Rightarrow E + (Ib) \Rightarrow E + (ab)$$

Derivation factorization

Assume $A \Rightarrow X_1 X_2 \cdots X_k \overset{*}{\Rightarrow} w$. We can **factorize** w as $w_1 w_2 \cdots w_k$ such that $X_i \overset{*}{\Rightarrow} w_i$, $1 \leqslant i \leqslant k$

As a special case, we can have $X_i = w_i \in \Sigma$ → (T)

Substring w_i can be identified from derivation $A \overset{*}{\Rightarrow} w$ by considering **only** those derivation steps that rewrite X_i

Example

Consider $E \Rightarrow E * E \overset{*}{\Rightarrow} a * b + a$

We have

$$\underbrace{a}_{E} \; \underbrace{*}_{*} \; \underbrace{b + a}_{E}$$

and we can write

$$E \overset{*}{\Rightarrow} a$$

$$* \overset{*}{\Rightarrow} *$$

$$E \overset{*}{\Rightarrow} b + a$$

Parse trees

Parse trees are a graphical representation alternative to derivations

Intuitively, parse trees represent the **syntactic structure** of a sentence according to the grammar

In compilers, parse trees are the structure of choice when **translating** into executable code

Parse trees

Let $G = (V, T, P, S)$ be a CFG. An ordered tree is a **parse tree** of G if :

- each internal node is labeled with a variable in V
- each leaf node is labeled with a symbol in $V \cup T \cup \{\epsilon\}$; each leaf labeled with ϵ is the only child of its parent
- if an internal node is labeled A and its children (from left to right) are labeled

  $$X_1, X_2, \ldots, X_k$$

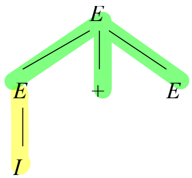
  then $A \rightarrow X_1 X_2 \cdots X_k \in P$

Example

CFG for arithmetic expressions and parse tree associated with the derivation $E \Rightarrow E + E \Rightarrow I + E$

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$

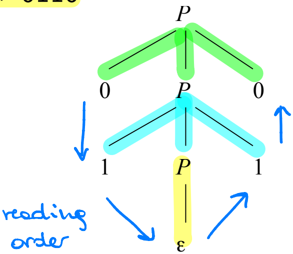
$\vdots$



Example

CFG for palindrome strings and parse tree associated with the derivation $P \Rightarrow 0P0 \Rightarrow 01P10 \Rightarrow 0110$

1. $P \rightarrow \epsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$



Yeld of a parse tree

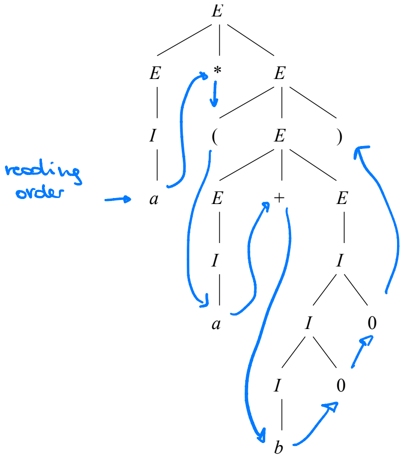
The **yield** of a parse tree is the string obtained by reading the leaves from left to right

Of special importance are the **complete** parse trees, where :

- the yield is a string of terminal symbols
- the root is labeled by the initial symbol

The set of yields of all complete parse trees is the language generated by the CFG

Example



Complete parse tree. The yield is $a * (a + b00)$

Derivations and parse trees

Let $G = (V, T, P, S)$ be a CFG, $A \in V$ and $w \in T^*$. The following statements are equivalent (statements must all be true or must all be false) :

- $A \overset{*}{\Rightarrow} w$
- $A \underset{lm}{\overset{*}{\Rightarrow}} w$
- $A \underset{rm}{\overset{*}{\Rightarrow}} w$
- there exists a parse tree for G with root label A and yield w

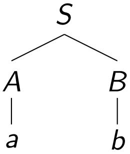
Proof not required for these theorems

Relation between derivations and parse trees **is not** one-to-one (see next slides)

Derivations and parse trees

A parse tree can be associated with **several** derivations

Example : Consider the CFG with productions $S \rightarrow AB$, $A \rightarrow a$, $B \rightarrow b$. The parse tree → Without recursion the language will be finite



$S \rightarrow AB$
$B \rightarrow b$
$L = \phi$
because A is not a terminal symbol

is associated with two derivations

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

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

Derivations and parse trees

A derivation can be associated with **several** parse trees

Example : Consider the CFG with productions $S \to SS \mid a$ The derivation

$$S \Rightarrow SS \Rightarrow SSS \Rightarrow aSS \Rightarrow aaS \Rightarrow aaa$$

is associated with two parse trees