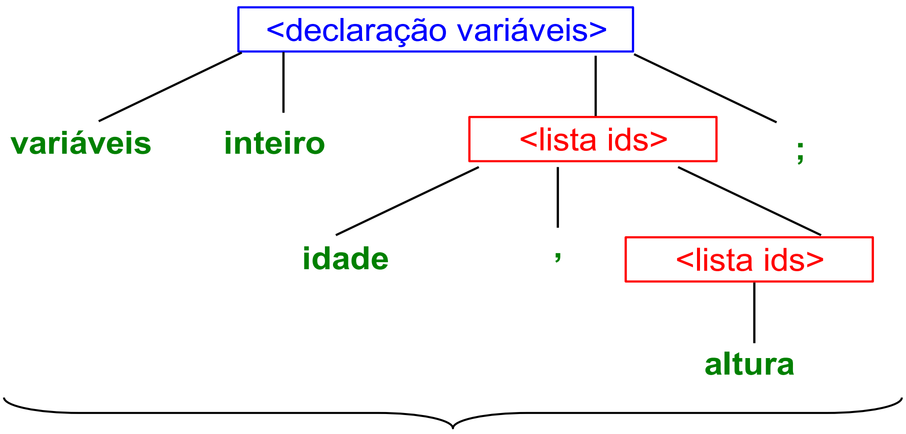
- **ÁRVORE DE DERIVAÇÃO OU ÁRVORE SINTÁTICA: representação gráfica** de derivações, que permitem visualizar estruturas hierárquicas das sentenças das linguagens geradas; **saída da análise sintática**, constituindo uma representação intermediária utilizada na análise semântica.

  - ✓ **DEFINIÇÃO:**
    1. todo **nó** de uma árvore de derivação é rotulado ou **com terminal ou com não-terminal** ou mesmo **com a sentença vazia**;
    2. **nó raiz** é rotulado com **símbolo inicial** S;
    3. **nós não-folha** são rotulados apenas com símbolos **não-terminais**;
    4. se um nó n tem rótulo R e os nós $n_1, n_2, ..., n_k$ são descendentes de n, da esquerda para a direita, com rótulos $X_1,X_2,...,X_k$, respectivamente, então $\mathbf{R \rightarrow X_1X_2...X_k}$ é uma produção em P; se um nó n tem rótulo R e o nó n_1 é descendente de n com rótulo ε, então $\mathbf{R \rightarrow \varepsilon}$ é uma produção em P.

  - ✓ **EXEMPLO:**



limite de uma árvore de derivação

- **DERIVAÇÃO MAIS À ESQUERDA E DERIVAÇÃO MAIS À DIREITA**

  ✓ **derivação mais à esquerda:** a cada passo na produção de uma derivação é aplicado o não-terminal mais à esquerda.

  ✓ **derivação mais à direita:** a cada passo na produção de uma derivação é aplicado o não-terminal mais à direita.

EXEMPLO:

<expressão> → <expressão> **+** <expressão>
| <expressão> ***** <expressão>
| **-** <expressão>
| **(** <expressão> **)**
| **número**

Apresente uma árvore de derivação e as derivações mais à esquerda e mais à direita para a sentença: *- (123 * 456)*

✓ uma sentença pode ter várias derivações mais à esquerda e várias derivações mais à direita, já que pode haver mais de uma árvore de derivação para a sentença. Entretanto, **para cada árvore de derivação, apenas uma derivação mais à esquerda e uma derivação mais à direita podem ser obtidas**.

- **SIMPLIFICAÇÕES DE GRAMÁTICAS LIVRES DE CONTEXTO:**

  ✓ **símbolos inúteis (inalcançáveis ou inférteis):** um símbolo terminal ou não-terminal é inútil em uma GLC se não for gerados a partir do símbolo inicial; ou se não gerar nenhuma sentença de símbolos terminais

  EXEMPLO:

$S \rightarrow A\ B \mid C$
$A \rightarrow a\ A \mid \varepsilon$
$B \rightarrow b\ B \mid \varepsilon$
$\mathbf{C \rightarrow c\ C}$

- **SIMPLIFICAÇÕES DE GRAMÁTICAS LIVRES DE CONTEXTO:**

  ✓ **GLC ε-livre:** uma GLC é dita ε-livre quando não possui produções cujo lado direito contém a sentença vazia ou quando possui uma única na forma $S \rightarrow \varepsilon$, onde S é o símbolo inicial da gramática e S não aparece do lado direito de nenhuma regra de produção.

  ✓ **produções simples (ou produções unitárias):** uma produção da forma $A \rightarrow \alpha$ em uma GLC é uma produção simples se α é um não-terminal.

  EXEMPLO:

$S \rightarrow A\ B \mid \mathbf{C}$
$A \rightarrow a\ A \mid \varepsilon$
$B \rightarrow b\ B \mid \varepsilon$
$C \rightarrow c\ C \mid c$

- **SIMPLIFICAÇÕES DE GRAMÁTICAS LIVRES DE CONTEXTO:**

  ✓ **recursão à esquerda:** a recursividade pode ser direta ou indireta. Uma GLC possui recursão à esquerda direta se P contém pelo menos uma produção da forma $A \rightarrow A\ \alpha$. Uma GLC possui recursão à esquerda indireta se existe em G uma derivação da forma $A \Rightarrow^n A\ \beta$, para algum $n \geq 2$.

  EXEMPLO:

G_1 recursão à esquerda direta

$S \rightarrow \mathbf{S}\ a \mid b$

G_2 recursão à esquerda indireta

$S \rightarrow A\ b \mid B\ c$
$A \rightarrow \mathbf{S}\ a \mid a$
$B \rightarrow b\ B \mid \varepsilon$

- **SIMPLIFICAÇÕES DE GRAMÁTICAS LIVRES DE CONTEXTO:**

  ✓ **gramática ambígua:** uma GLC é ambígua quando, para alguma sentença da linguagem gerada, existe mais de uma árvore de derivação.

  EXEMPLO:

  G_1 ambígua

  <comando> → **IF** <expressão> **THEN** <comando> **ELSE** <comando>
  | **IF** <expressão> **THEN** <comando>
  | <outro>

  G_2 não ambígua.

  <comando> → <associado> | <não associado>
  <associado> → **IF** <expressão> **THEN** <associado> **ELSE** <associado>
  | <outro>
  <não associado> → **IF** <expressão> **THEN** <comando>
  | **IF** <expressão> **THEN** <associado> **ELSE** <não associado>

- **SIMPLIFICAÇÕES DE GRAMÁTICAS LIVRES DE CONTEXTO:**

  ✓ **gramática fatorada (ou determinística):** uma GLC está fatorada se não possui produções para um mesmo não-terminal no lado esquerdo cujo lado direito inicie com o mesmo conjunto de símbolos ou com símbolos que derivam sequências que iniciem com o mesmo conjunto de símbolos.

  EXEMPLO:

  G_1 não fatorada

  A → **a** B | **a** C
  B → **b** B | **b**
  C → **c** C | **c**

  G_2 não fatorada

  A → B | C
  B → **a b** B | **a b**
  C → **a c** C | **a c**

  G_3 fatorada

  A → a A'
  A' → B | C
  B → b B'
  B' → B | ε
  C → c C'
  C' → C | ε

ALGORITMO nº 1: eliminação de símbolos inúteis

ENTRADA: uma GLC $G = (V_N, V_T, P, S)$
SAÍDA: uma GLC $G = (V_N', V_T', P', S')$ sem símbolos inúteis

PASSO 1: eliminar os símbolos inférteis da seguinte forma:

$i \leftarrow 0$
$V_N[i] \leftarrow \{\}$
REPITA
  $i \leftarrow i + 1$
  $V_N[i] \leftarrow V_N[i-1] \cup \{A \mid A \rightarrow \alpha \in P,\ \alpha \in (V_N[i-1] \cup V_T)^*\}$
ATÉ $V_N[i] = V_N[i-1]$
$V_{N1} \leftarrow V_N[i]$ {conjunto de símbolos não-terminais férteis}
P_1 {possui as mesmas regras de produção de P, exceto aquelas cujos símbolos não-terminais não pertencem a $V_{N\,1}$}

PASSO 2: eliminar os símbolos inalcançáveis da seguinte forma:

$i \leftarrow 0$
$V_T[i] \leftarrow \{\}$
$V_N[i] \leftarrow \{S\}$
REPITA
  $i \leftarrow i + 1$
  $V_N[i] \leftarrow V_N[i-1] \cup \{X \mid A \rightarrow \alpha X \beta \in P_1, \alpha \text{ e } \beta \in (V_{N\,1} \cup V_T)^*, A \in V_N[i-1]\}$
  $V_T[i] \leftarrow V_T[i-1] \cup \{x \mid A \rightarrow \alpha x \beta \in P_1, \alpha \text{ e } \beta \in (V_{N\,1} \cup V_T)^*, A \in V_N[i-1]\}$
ATÉ $(V_N[i] = V_N[i-1])$ E $(V_T[i] = V_T[i-1])$
$V_N' \leftarrow V_{N\,1} \cap V_N[i]$ {conjunto de símbolos não-terminais férteis e alcançáveis}
$V_T' \leftarrow V_T \cap V_T[i]$ {conjunto de símbolos terminais alcançáveis}
P' {possui as mesmas regras de produção de P_1, exceto aquelas cujos símbolos não pertencem a $V_N' \cup V_T'$}
$S' \leftarrow S$

ALGORITMO nº 2: fatoração de GLC (ou eliminação do não determinismo)

ENTRADA: uma GLC $G = (V_N, V_T, P, S)$
SAÍDA: uma GLC $G = (V_N, V_T, P', S)$ fatorada

PASSO 1: eliminar não determinismo direto da seguinte forma: substituir as regras de produção na forma

$$A \rightarrow \alpha\, \beta \mid \alpha\, \gamma$$

por

$$A \rightarrow \alpha\, A'$$
$$A' \rightarrow \beta \mid \gamma$$

PASSO 2: eliminar não determinismo indireto da seguinte forma: transformar em não determinismo direto através de derivações sucessivas e então aplicar o PASSO 1. P' possui as regras de produção anteriormente especificadas que não possuem não determinismo e as regras de produção modificadas pela aplicação do algoritmo.

ALGORITMO nº 3: eliminação da recursão à esquerda

ENTRADA: uma GLC $G = (V_N, V_T, P, S)$
SAÍDA: uma GLC $G = (V_N, V_T, P', S)$ sem recursão à esquerda

PASSO 1: eliminar a recursão à esquerda direta da seguinte forma: substituir as regras de produção na forma

$A \rightarrow A\, \alpha_1 \mid \ldots \mid A\, \alpha_n \mid \beta_1 \mid \ldots \mid \beta_m$, onde nenhum β_i começa com A

por

$A \rightarrow \beta_1\, A' \mid \ldots \mid \beta_m\, A'$
$A' \rightarrow \alpha_1\, A' \mid \ldots \mid \alpha_n\, A' \mid \varepsilon$, onde A' é um novo não-terminal

PASSO 2: eliminar a recursão à esquerda indireta da seguinte forma:

ordenar os não-terminais de G em uma ordem qualquer ($A_1, \ldots A_n$)
PARA I DE 1 ATÉ n FAÇA
  PARA J DE 1 ATÉ i - 1 FAÇA
    substituir as regras de produção na forma
    $A_I \rightarrow A_J\, \gamma$
   por
    $A_I \rightarrow \alpha_1\, \gamma \mid \ldots \mid \alpha_k\, \gamma$, onde $\alpha_1, \ldots, \alpha_k$ são os lados direitos das regras de produção A_J, ou seja, $A_J \rightarrow \alpha_1 \mid \ldots \mid \alpha_k$
  FIM PARA
  eliminar as recursões à esquerda diretas das A_I produções (PASSO 1)
FIM PARA

ALGORITMO nº 4: eliminação de ε – produções

ENTRADA: uma GLC $G = (V_N, V_T, P, S)$
SAÍDA: uma GLC $G = (V_N, V_T, P', S)$ sem ε – produções

PASSO 1: Determinar os símbolos não-terminais que geram (direta ou indiretamente) a sentença vazia.

PASSO 2: Eliminar as produções que geram sentenças vazias
Para cada produção cujo lado direito possui um símbolo não-terminal que gera a sentença vazia, criar uma nova produção sem esse símbolo.

PASSO 3: Incluir a sentença vazia, se necessário.
Se a sentença vazia pertence à linguagem, então é incluída uma produção para gerar a sentença vazia.

ALGORITMO nº 5: eliminação de produções simples

ENTRADA: uma GLC $G = (V_N, V_T, P, S)$
SAÍDA: uma GLC $G = (V_N, V_T, P', S)$ sem produções simples

PASSO 1: Construir o fecho transitivo de cada símbolo não-terminal
Se $A \rightarrow B$ e $B \rightarrow C$ então Fecho(A) = {B,C}

PASSO 2: Excluir as regras simples $A \rightarrow B$
Substituir as produções da forma $A \rightarrow B$ por produções da forma $A \rightarrow \alpha$, para toda $B \rightarrow \alpha$, onde $\alpha \in (V_N \cup V_T)^*$ e $B \in$ Fecho(A).