Universidade de Santa Cruz do Sul – UNISC
Departamento de informática

COMPILADORES

Revisão – Linguagens formais – Parte 02

Prof. Geovane Griesang
geovanegriesang@unisc.br

Linguagens formais

Legenda:

$\sum$ = sigma (somatório)

δ = delta

ε = epsilon

λ = lambda

α = alfa

β = beta

γ = gamma

ξ = xi

Linguagens formais

Gramáticas livre de contexto (GLC):

As GLCs tem grande importância dentro das Linguagens Formais, pois através delas pode ser descrita a maior parte das construções sintáticas das linguagens de programação.

Uma GLC é uma notação formal para expressar tais definições recursivas de linguagens.

São capazes de tratar: Balanceamento de parênteses. Ex: (), [], ...
Construções em bloco. Ex: begin end, {}, ...

Uso em Linguagens Computacionais.

Linguagens formais

Gramáticas livre de contexto (GLC):

Usam símbolos auxiliares, chamados não terminais, que servem de “ponto de partida” para gerar alguma cadeia.

Exemplo: 0^n1^n, n >0

Devemos usar um símbolo não terminal, exemplo, X. Esse símbolo será o “ponto de partida” para gerar cadeias completas de linguagem.

Os símbolos 0 e 1, por sua vez, são chamados de símbolos terminais.

Linguagens formais

Gramáticas livre de contexto (GLC):

Pode-se usar um "receita" para gerar cadeias na forma 0^n1^n, cuja será usada para escrever as regras da gramática.

1º) ver a menor cadeia da linguagem: 01.

Precisa-se de uma regra para que essa cadeia possa ser gerada.

Se X for o símbolo escolhido para gerar as cadeias da linguagem, deve-se dizer que X pode gerar 01.

$$X \rightarrow 01$$

Linguagens formais

Gramáticas livre de contexto (GLC):

2º) Exemplo: 0^n1^n. Cada cadeia precisa possuir, para cada símbolo 0, um símbolo 1 correspondente.

Com isso, pode-se usar essa cadeia de linguagem para gerar outra cadeia maior da linguagem:

Portanto, pode-se inserir um 0 no início da cadeia e um 1 (correspondente ao 0) no fim da cadeia.

$$X \rightarrow 0X1$$

Linguagens formais

Gramáticas livre de contexto (GLC):

Portanto, para gerar essa cadeia intermediária, continua-se o processo recursivamente usando as regras de X.

Desta forma, consegue-se gerar todas as cadeias desejadas.

$$X \rightarrow 01$$

$$X \rightarrow 0X1$$

Linguagens formais

$X \rightarrow 01$

$X \rightarrow 0X1$

Gramáticas livre de contexto (GLC):

Para gerar uma palavra, deve-se iniciar no símbolo X e seguir em um processo de seguidas substituições, onde os demais símbolos continuam inalterados.

Esse processo é chamado de derivação de uma cadeia.

Palavra: 000111.

$$X \rightarrow 0X1 \Rightarrow X \rightarrow 0X1 = X \rightarrow 00X11$$

$$X \rightarrow 01 = X \rightarrow 000111$$

Linguagens formais

$X \rightarrow 01$

$X \rightarrow 0X1$

Gramáticas livre de contexto (GLC):

Outras cadeias, como: 1, 011, 0101 não podem ser derivadas pelas regras e portanto, não fazem parte desta linguagem.

O exemplo anterior é um exemplo clássico e de grande importância no estudo das Linguagens Livre de Contexto (LLC), pois permite estabelecer analogia entre 0^n1^n e as linguagens bloco-estruturados do tipo $BEGIN^n$ END^n, ou com expressões com parênteses balanceados na forma $(n)^n$.

$X \rightarrow$ begin end, $X \rightarrow$ begin$X$end

$X \rightarrow ()$, $X \rightarrow (X)$

Linguagens formais

$X \rightarrow 01$

$X \rightarrow 0X1$

Gramáticas livre de contexto (GLC):

De forma formal, pode-se dizer que uma determinada GLC é uma 4-tupla (quatro elementos ordenado) da forma:

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

V é o alfabeto que contém os símbolos não terminais ou variáveis, que são os símbolos auxiliares no processo de geração de cadeias (como o símbolo X do exemplo inicial).

T é o alfabeto dos símbolos terminais, que são os símbolos usados nas cadeias da linguagem. O conjunto **T** é equivalente ao alfabeto de entrada $\sum$ dos autômatos finitos.

Linguagens formais

$X \rightarrow 01$

$X \rightarrow 0X1$

Gramáticas livre de contexto (GLC):

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

P é o conjunto de regras de produção da gramática, que servem para gerar as palavras da linguagem.

Cada regra tem a forma $N \rightarrow a$, onde:

N é chamado de cabeça de produção e consiste simplesmente em um símbolo não terminal (ou seja, $N \in V$).

a é chamado de corpo da produção e consiste em qualquer cadeia de símbolos terminais e/ou não terminais, inclusive a cadeia vazia ε.

Linguagens formais

$X \rightarrow 01$

$X \rightarrow 0X1$

Gramáticas livre de contexto (GLC):

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

S é o símbolo não terminal de início ($S \in V$), pois é usado para iniciar as derivações de cadeia da linguagem.

Desta forma, pode-se definir a gramática vista no exemplo inicial (0^n1^n) de maneira completa.

$$G = (\{X\}, \{0, 1\}, \{X \rightarrow 01, X \rightarrow 0X1\}, X)$$

Em casos de produções com a mesma "cabeça", pode separar os "corpos" pela barra vertical "|" (*pipe*), que pode ser lida como "ou".

$$G = (\{X\}, \{0, 1\}, \{X \rightarrow 01 \mid 0X1\}, X)$$

Linguagens formais

Gramáticas livre de contexto (GLC):

Extensão da definição GLC: $A \rightarrow \varepsilon$

Estas produções, cujo lado direito contém somente a sentença vazia, são chamadas de ε-produções.

$$P = \{ A \rightarrow \beta \mid A \in N, \beta \in (N \cup T)^* \}$$

GLC ε-livre

Uma GLC é ε-livre quando não possui ε-produções ou quando possui uma única ε-produção, $S \rightarrow \varepsilon$, onde S é o símbolo inicial da gramática e S não aparece do lado direito de nenhuma regra de produção.

Linguagens formais

Gramáticas livre de contexto (GLC):

Árvores de Derivação para GLC

São representações gráficas para as derivações nas GLC.

Considere a gramática G = ({S, A}, {a, b}, P, S), onde P consiste de:

$S \rightarrow aAS \mid a$

$A \rightarrow SbA \mid SS \mid ba$

Linguagens formais

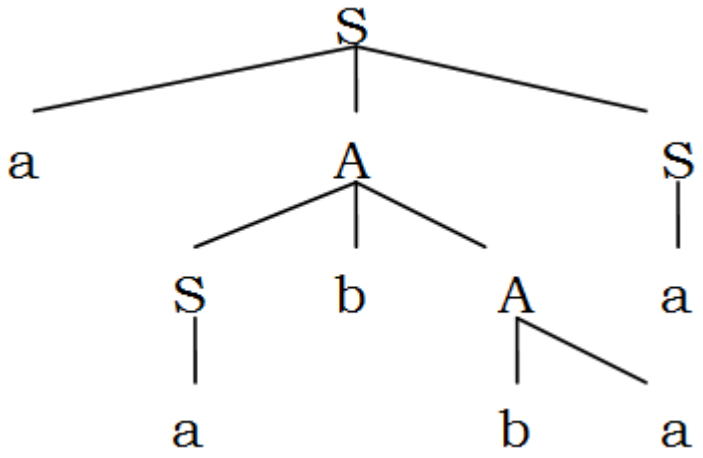
$S \rightarrow aAS \mid a$
$A \rightarrow SbA \mid SS \mid ba$

Gramáticas livre de contexto (GLC):

A derivação da sentença *aabbaa* é dada por:

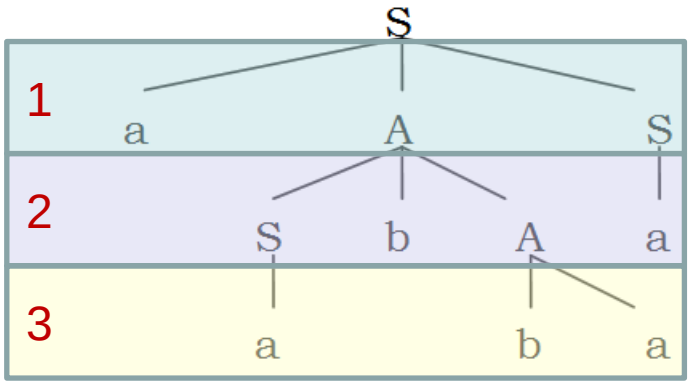
$$S \rightarrow aAS \rightarrow aSbAS \rightarrow aabAS \rightarrow aabbaS \rightarrow aabbaa$$

A árvore de derivação correspondente a sentença *aabbaa* seria:



Linguagens formais

Gramáticas livre de contexto (GLC):



Profundidade da Árvore de Derivação

É o comprimento do maior caminho entre a raiz e um nodo terminal. No exemplo anterior, a árvore de derivação da sentença tem profundidade 3.

Limite de uma Árvore de Derivação

É a sequência formada pela concatenação, da esquerda para a direita, das folhas da árvore de derivação: aabbaa.

Linguagens formais

Exercício 01

Mostre a árvore de derivação para a sentença aaabbabbba, levando em conta a gramática abaixo:

$$G = (\{S, A, B\}, \{a, b\}, P, S)$$

$$P = \{ \quad S \rightarrow aB \mid bA$$
$$A \rightarrow a \mid aS \mid bAA$$
$$B \rightarrow b \mid bS \mid aBB \}$$

Linguagens formais

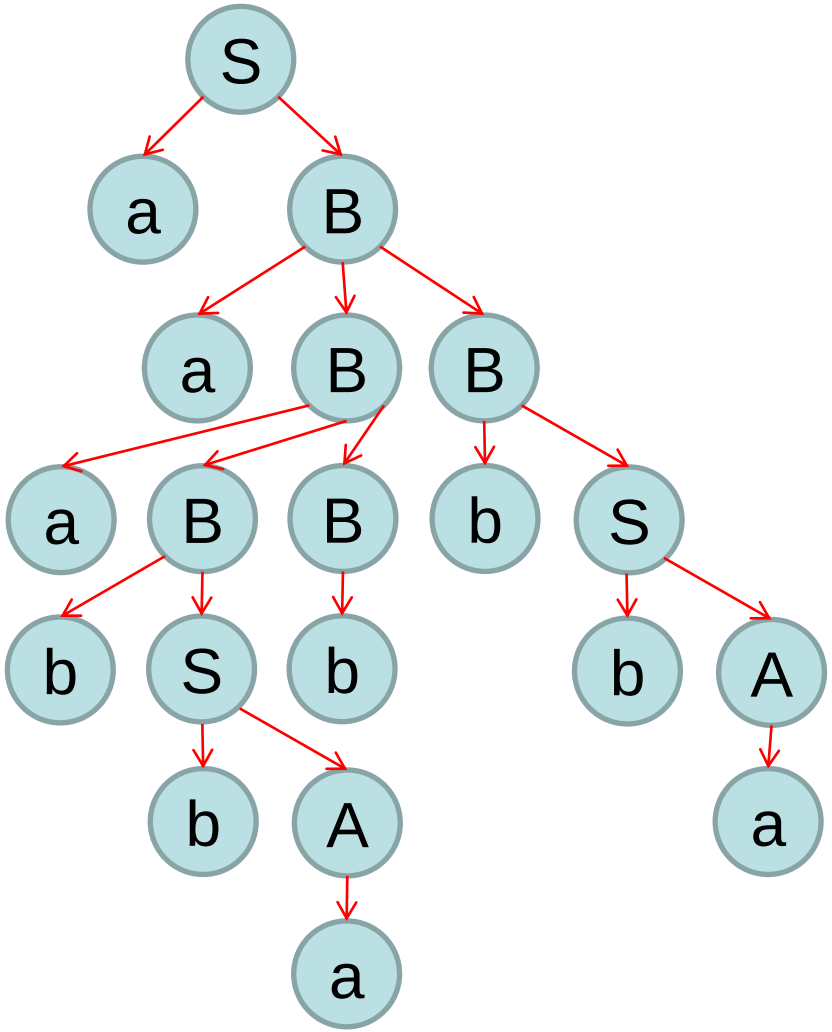
G = ({S, A, B}, {a, b}, P, S)
P = { S → aB | bA
A → a | aS | bAA
B → b | bS | aBB }

Exercício 01: aaabbabbba.

~~a~~aabbabbba

~~aa~~abbabbba

~~aaa~~bbabbba
~~aaab~~babbba
~~aaabb~~abbba
~~aaabba~~bbba
~~aaabbab~~bba
~~aaabbabb~~ba
~~aaabbabbb~~a
~~aaabbabbba~~



Linguagens formais

Gramáticas livre de contexto (GLC):

Derivação mais à esquerda e mais à direita

Uma árvore de derivação ignora variações na ordem em que os símbolos foram substituídos na derivação.

Por exemplo, na gramática de expressão aritmética abaixo:

G = ({E}, {+, *, (,), -, id}, P, E) onde,

P =
E → E + E
E → E * E
E → (E)
E → - E
E → id

Linguagens formais

P = $E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow (E)$
$E \rightarrow - E$
$E \rightarrow id$

Gramáticas livre de contexto (GLC):

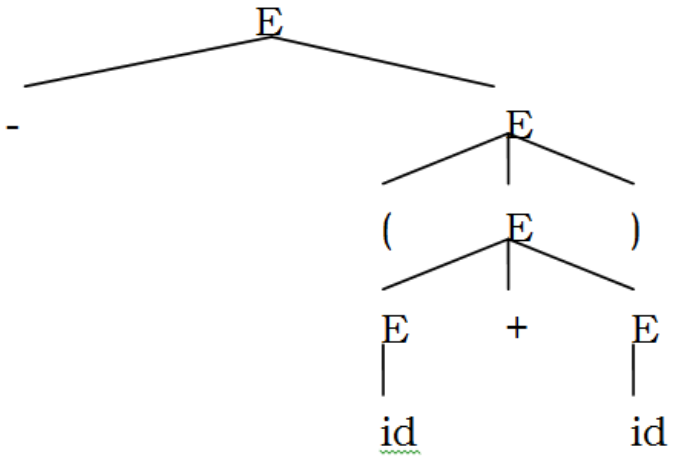
a sentença "- (id + id)" pode ser derivada de dois modos diferentes:

$E \rightarrow - E \rightarrow - (E) \rightarrow - (E + E) \rightarrow - (id + E) \rightarrow - (id + id)$

ou

$E \rightarrow - E \rightarrow - (E) \rightarrow - (E + E) \rightarrow - (E + id) \rightarrow - (id + id)$

As derivações acima correspondem à mesma árvore de derivação:



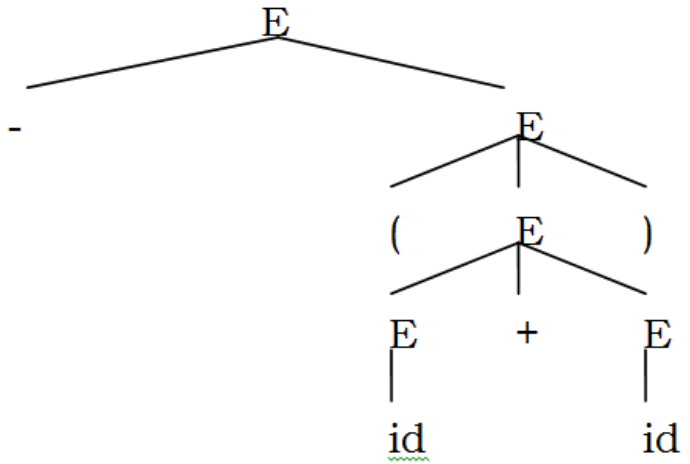
Linguagens formais

Gramáticas livre de contexto (GLC):

Uma derivação é chamada de mais à esquerda quando o símbolo substituído for o não terminal mais à esquerda da forma sentencial.

Na derivação mais à direita, o símbolo substituído é o não terminal mais à direita.

Nas duas derivações da sentença "- (id + id)", a primeira é mais à esquerda e a segunda mais à direita.



Linguagens formais

P = $E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow (E)$
$E \rightarrow - E$
$E \rightarrow id$

Gramáticas livre de contexto (GLC):

Exemplo: Para a mesma gramática anterior, obtenha as derivações mais à esquerda e mais à direita da sentença:

id + id * id

Solução:

Derivação mais à esquerda

$E \rightarrow E + E \rightarrow id + E \rightarrow id + E * E \rightarrow id + id * E \rightarrow id + id * id$

Derivação mais à direita

$E \rightarrow E + E \rightarrow E + E * E \rightarrow E + E * id \rightarrow E + id * id \rightarrow id + id * id$

Linguagens formais

P = $E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow (E)$
$E \rightarrow - E$
$E \rightarrow id$

Gramáticas livre de contexto (GLC):

Gramática Ambígua

Uma GLC é ambígua quando, para alguma sentença da linguagem gerada, existe mais de uma árvore de derivação.

Exemplo:

A gramática de expressão aritmética apresentada antes é ambígua. Isto pode ser visto através de duas árvores de derivação diferentes para a sentença vista:

id + id * id

Linguagens formais

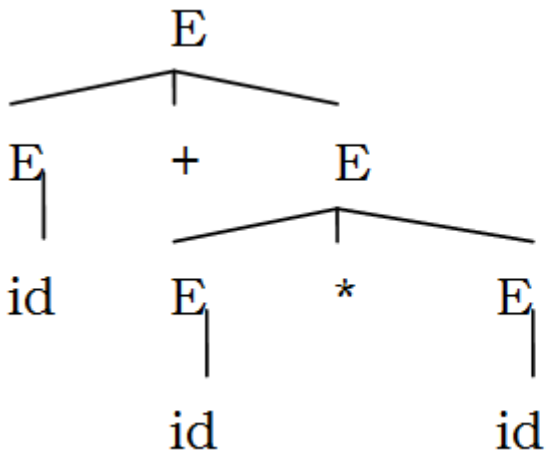
P = $E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow (E)$
$E \rightarrow - E$
$E \rightarrow id$

Gramáticas livre de contexto (GLC):

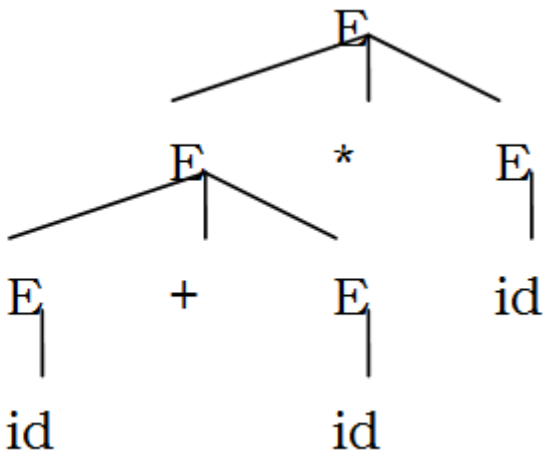
Gramática Ambígua

$E \rightarrow E + E \rightarrow id + E \rightarrow id + E * E \rightarrow id + id * E \rightarrow id + id * id$ Esquerda

$E \rightarrow E + E \rightarrow E + E * E \rightarrow E + E * id \rightarrow E + id * id \rightarrow id + id * id$ Direita



id + id * id



Linguagens formais

Gramáticas livre de contexto (GLC):

Transformações em G.L.C

Algumas transformações podem ser efetuadas em GLC's com o objetivo de torná-las mais simples ou de prepará-las para posteriores aplicações.

É importante destacar que, qualquer que seja a transformação efetuada, a linguagem gerada deverá ser sempre a mesma.

Linguagens formais

Gramáticas livre de contexto (GLC):

Eliminação de Símbolos Inúteis

Em uma GLC, um símbolo (terminal ou não-terminal) é inútil se ele não aparece na derivação de nenhuma sentença.

Ou seja, um símbolo é inútil se ele é:

estéril: não gera nenhuma sequência de terminais pertencente a uma sentença, ou,

inalcançável: não aparece em nenhuma forma sentencial da gramática.

Linguagens formais

Gramáticas livre de contexto (GLC):

Felizmente, é possível eliminar todas as variáveis e todas as produções inúteis de uma gramática.

Para qualquer LVC existe uma gramática sem variáveis ou produções inúteis, então:

Se G = (V, T, S, P) é uma GLC com símbolos inúteis, significa que existe uma gramática equivalente G' = (V', T', S, P') que não contém qualquer variável ou produção inútil.

Linguagens formais

Gramáticas livre de contexto (GLC):

Determinação do conjunto de símbolos férteis

Pode ser efetuada através do seguinte algoritmo:

a) Construir o conjunto $N_0 = \varnothing$ e fazer i = 1

b) Repetir

$N_i = N_{i-1} \cup \{ A \mid A \rightarrow \alpha \in P \text{ e } \alpha \in (N_{i-1} \cup T)^* \}$

i = i + 1

até que $N_i = N_{i-1}$

c) N_i é o conjunto de símbolos férteis.

Se o símbolo inicial não fizer parte do conjunto de símbolos férteis, a linguagem gerada pela gramática é vazia.

Linguagens formais

Gramáticas livre de contexto (GLC):

Determinação do conjunto de símbolos férteis

Etapa A. Construção de uma gramática intermédia cujas variáveis são todas úteis. Seja N conjunto das variáveis úteis. Inicialmente está vazio,‡ ∅, e vai-se enchendo em voltas sucessivas.

Exemplo - Retirar os símbolos estéreis da gramática:

G = ({S,A,B,C,D}, {a,b,c,d}, P, S)

P:
- $S \rightarrow aA$
- $A \rightarrow a \mid bB$
- $B \rightarrow b \mid dD$
- $C \rightarrow cC \mid c$
- $D \rightarrow dD$

Linguagens formais

P: $S \rightarrow aA$
$A \rightarrow a \mid bB$
$B \rightarrow b \mid dD$
$C \rightarrow cC \mid c$
$D \rightarrow dD$

Gramáticas livre de contexto (GLC):

1º) Inicialização, $N_0 = \varnothing$

2º) Na 1ª volta só entram no conjunto N_1 as variáveis que tenham do lado direito apenas símbolos terminais. $N_1 = \{A, B, C\}$

Na 2ª volta e nas voltas seguintes, entram as que tenham do lado direito símbolos terminais e/ou variáveis já incluídas em N_1 nas voltas anteriores. $N_2 = \{S, A, B, C\}$

O processo é repetido até que não haja mais variáveis para serem inseridas em N. $N_3 = \{S, A, B, C\} = N_2$

Linguagens formais

P: S → aA
A → a | bB
B → b | dD
C → cC | c
D → dD

Gramáticas livre de contexto (GLC):

3º) Se houver uma variável que não esteja em N, ela não produz qualquer cadeia final, e será portanto inútil e prejudicial.

Portanto, todas as produções que têm estas variáveis inúteis são eliminadas na etapa A. Eliminar D → dD, pois é estéril.

Solução:

$N_0 = \varnothing$
$N_1 = \{A, B, C\}$
$N_2 = \{S, A, B, C\}$
$N_3 = \{S, A, B, C\} = N_2$

Conjunto de símbolos férteis: {S, A, B, C}

Linguagens formais

P: $S \rightarrow aA$
$A \rightarrow a \mid bB$
$B \rightarrow b \mid dD$
$C \rightarrow cC \mid c$
$D \rightarrow dD$

Gramáticas livre de contexto (GLC):

Exemplo: Retirar os símbolos estéreis da gramática:

G = ({S,A,B,C,~~D~~}, {a,b,c,~~d~~}, P, S)
P: $S \rightarrow a\,A$
$A \rightarrow a \mid b\,B$
$B \rightarrow b \mid$ ~~$d\,D$~~
$C \rightarrow cC \mid c$
~~$D \rightarrow d\,D$~~

Solução:
G' = ({S, A, B, C}, {a, b, c}, P', S)
P': $S \rightarrow aA$
$A \rightarrow a \mid bB$
$B \rightarrow b$
$C \rightarrow cC \mid c$

Observação: Se o símbolo inicial não fizer parte do conjunto de símbolos férteis, a linguagem gerada pela gramática é vazia.

Linguagens formais

Gramáticas livre de contexto (GLC):

Determinação do conjunto de símbolos alcançáveis

Além das produções já eliminadas na etapa A, pode haver outras que sejam inúteis, não por serem prejudiciais, mas por serem inalcançáveis.

Etapa B. Construção de uma gramática intermediaria cujas variáveis são todas alcançáveis. Seja V conjunto das variáveis alcançáveis.

Encontrar e eliminar todas as variáveis e produções que não podem ser alcançadas a partir de S.

Para isso usa-se o grafo de dependências como ferramenta auxiliar.

Linguagens formais

P: $S \rightarrow aA$
$A \rightarrow a \mid bB$
$B \rightarrow b \mid dD$
$C \rightarrow cC \mid c$
$D \rightarrow dD$

Gramáticas livre de contexto (GLC):

Continuação do exemplo, agora, eliminando símbolos inalcançáveis.

G' = ({S, A, B, ~~C~~}, {a, b, ~~c~~}, P', S)
P': $S \rightarrow aA$
$A \rightarrow a \mid bB$
$B \rightarrow b$
~~$C \rightarrow cC \mid c$~~

Solução da gramática simplificada:
G'' = ({S, A, B}, {a, b}, P'', S)
P'': $S \rightarrow aA$
$A \rightarrow a \mid bB$
$B \rightarrow b$

Observação: Note que aparentemente os dois passos são independentes e podem ser aplicados em qualquer ordem. Isto não é verdadeiro. Isto ocorre porque a eliminação de símbolos inativos (estéreis) pode ocasionar o surgimento de símbolos inatingíveis. O contrário não é verdadeiro.

Linguagens formais

Gramáticas livre de contexto (GLC):

Exercício 02: Considerando a GLC a seguir:

$$G = (\{S, A, B, C\}, \{0, 1, 2, 3\}, P, S)$$
$$P = \{S \rightarrow A \mid B \mid AB$$
$$A \rightarrow 0B \mid 1S \mid 2$$
$$B \rightarrow 1B \mid B0 \mid 2$$
$$C \rightarrow AS \mid 3B \mid 0\}$$

Eliminar símbolos inúteis e exibir a gramática simplificada, assim como, os conjuntos de símbolos férteis e o conjunto de símbolos alcançáveis.

Linguagens formais

Gramáticas livre de contexto (GLC):

Transformação de uma GLC qualquer para uma GLC ε-Livre

Esta transformação sempre é possível e pode ser efetuada pelo algoritmo:

a) Reunir em um conjunto os não terminais que derivam direta ou indiretamente a sentença vazia:

$Ne = \{A \mid A \in N \text{ e } A \rightarrow^{+} \varepsilon\}$

$G = (\{S, B\}, \{a, b\}, P, S)$

P: $S \rightarrow aB$

$B \rightarrow bB \mid \varepsilon$

Passo A) $Ne = \{B\}$

Linguagens formais

G = ({S, B}, {a, b}, P, S)
P: $S \rightarrow aB$
$B \rightarrow bB \mid \varepsilon$

Gramáticas livre de contexto (GLC):

b) Construir o conjunto de regras P' como segue:

b1) incluir em P' todas as regras de P, com exceção daquelas da forma $A \rightarrow \varepsilon$

Passo b1) P': $S \rightarrow aB$
$B \rightarrow bB$

b2) para cada ocorrência de um símbolo Ne do lado direito de alguma regra de P, incluir em P' mais uma regra, substituindo este símbolo por ε. Isto é, para regra de P do tipo:

$A \rightarrow \alpha B \beta$, $B \in Ne$ e $\alpha, \beta \in V^*$
incluir em P' a regra: $A \rightarrow \alpha \beta$

Passo b2) P': $S \rightarrow aB$
$B \rightarrow bB \mid b$

Linguagens formais

Gramáticas livre de contexto (GLC):

c) Se S $\in$ Ne, adicionar a P' as regras S' $\rightarrow$ S e S' $\rightarrow \varepsilon$, sendo que N' ficará igual a N $\cup$ S'. Caso contrário trocar os nomes de S por S' e N por N'.

G = ({S}, {a}, P, S)
P: S $\rightarrow$ aS | ε

Solução: Ne = {S}
P ': S $\rightarrow$ aS | a

P ': S' $\rightarrow$ S | ε
S $\rightarrow$ aS | a

d) A nova gramática será definida por:

G' = (N', T, P', S')

Solução:
G' = ({S', S}, {a}, P', S')
P ': S' $\rightarrow$ S | ε
S $\rightarrow$ aS | a

Linguagens formais

Gramáticas livre de contexto (GLC):

Exemplo – GLC transformada em GLC ε-Livre:

G = ({S, D, C}, {b, c, d, e}, P, S)

P:

$S \rightarrow bDCe$

$D \rightarrow dD \mid \varepsilon$

$C \rightarrow cC \mid \varepsilon$

$S \rightarrow bDCe$

$D \rightarrow dD \mid \cancel{\varepsilon}$

$C \rightarrow cC \mid \cancel{\varepsilon}$

Ne = {D, C}

Reescrever eliminando as variáveis em Ne (D, C).

S: bDCe, b~~D~~Ce, bD~~C~~e, b~~DC~~e;

D: d~~D~~;

C: c~~C~~;

Solução:

Ne = {D, C}

P': $S \rightarrow bDCe \mid bCe \mid bDe \mid be$

$D \rightarrow dD \mid d$

$C \rightarrow cC \mid c$

G': ({S, D, C}, {b, c, d, e}, P', S)

Linguagens formais

Gramáticas livre de contexto (GLC):

Exercício 03: GLC transformada em GLC ε-Livre:

G = ({S, A, B, C}, {a, b, c}, P, S)
P = {S → Ac | ABa | AbA
A → Aa | ε
B → Bb | BC | c | C
C → CB | CAc | bBc }

Linguagens formais

Gramáticas livre de contexto (GLC):

Remoção de Produções Simples (unitárias)

Produções simples são produções da forma $A \rightarrow B$ onde A e $B \in N$. Estas produções podem ser removidas de uma GLC através do seguinte algoritmo:

a) Transformar a GLC em uma GLC ε-livre, se necessário

b) Para todo não terminal de N, construir um conjunto com os não terminais que ele pode derivar, em um ou mais passos. Isto é, para todo $A \in N$, construir $N_A = \{ B \mid A \ ^*\!\rightarrow B \}$

Linguagens formais

Gramáticas livre de contexto (GLC):

c) Construir P' como segue:

se $B \rightarrow \alpha \in P$ e não é uma produção simples, adicione a P' as produções:

$$A \rightarrow \alpha \text{ para todo } A \mid B \in N_A$$

d) A GLC equivalente, sem produções simples, será definida por:

G' = (N, T, P', S)

Linguagens formais

Gramáticas livre de contexto (GLC):

Exemplo: Transformar as GLC abaixo em gramáticas equivalentes que não apresentem produções simples.

G = ({S, A}, {a, b}, P, S)

P: $S \rightarrow bS \mid A$

$A \rightarrow aA \mid a$

Solução:

Ns = {A}

$N_A = \{\}$

G' = ({S, A}, {a, b}, P', S)

P': $S \rightarrow bS \mid aA \mid a$

$A \rightarrow aA \mid a$

Linguagens formais

Gramáticas livre de contexto (GLC):

Exercício 4: Transformar as GLC abaixo em gramáticas equivalentes que não apresentem produções simples.

G = ({S, A, B}, {a, b, c}, P, S)

P: $S \rightarrow aSb \mid A$

$A \rightarrow aA \mid B$

$B \rightarrow bBc \mid bc$

Linguagens formais

Gramáticas livre de contexto (GLC):

Fatoração de GLC

Uma GLC está fatorada se ela é determinística, ou seja, não possui produções cujo lado direito inicie com o mesmo conjunto de símbolos ou com símbolos que gerem sequências que iniciem com o mesmo conjunto de símbolos.

Ex., a gramática fatorada não deverá apresentar as seguintes regras:

$$A \rightarrow aB \mid aC$$

pois as duas iniciam com o mesmo terminal a.

Linguagens formais

Gramáticas livre de contexto (GLC):

Para fatorar uma GLC devemos alterar as produções envolvidas no não determinismo da seguinte maneira:

a) as produções que apresentam não-determinismo direto, da forma

$$A \rightarrow \alpha\ \beta \mid \alpha\ \delta$$

serão substituídas por

$$A \rightarrow \alpha\ A'$$

$$A' \rightarrow \beta \mid \delta$$

sendo A' um novo não-terminal

Exemplo: $A \rightarrow aB \mid aC$

P': $A \rightarrow aA'$

$A' \rightarrow B \mid C$

Linguagens formais

Gramáticas livre de contexto (GLC):

b) O não-determinismo indireto é retirado fazendo, nas regras de produção, as derivações necessárias para torná-lo determinismo direto, resolvido posteriormente como no item anterior.

Exemplo de não-determinismo indireto:

P: $S \rightarrow A \mid B$
$A \rightarrow ac$
$B \rightarrow ab$

Solução:

P': $S \rightarrow A \mid B$
$A \rightarrow ac$
$B \rightarrow ab$

P': $S \rightarrow aA \mid aB$
$A \rightarrow c$
$B \rightarrow b$

P': $S \rightarrow aS'$
$S' \rightarrow A \mid B$
$A \rightarrow c$
$B \rightarrow d$

Linguagens formais

Gramáticas livre de contexto (GLC):

Exemplo 01: Fatorar as GLC abaixo.

G = ({S, A, B}, {a, b}, P, S)

P: $S \rightarrow aA \mid aB$

$A \rightarrow aA \mid a$

$B \rightarrow b$

G = ({S, S', A, A', B}, {a, b}, P', S)

Solução:

P': $S \rightarrow aS'$

$S' \rightarrow A \mid B$

$A \rightarrow aA'$

$A' \rightarrow A \mid \varepsilon$

$B \rightarrow b$

Linguagens formais

Gramáticas livre de contexto (GLC):

Exemplo 02: Fatorar as GLC abaixo.

G = ({S, A}, {a, b}, P, S)
P: S → Ab | ab | baA
A → aab | b

G' = ({S, A}, {a, b}, P', S)
Solução:
P': S → aabb | bb | ab | baA
A → aab | b

Ainda precisa ser fatorado...

G'' = ({S, S', S'', B}, {a, b}, P'', S)
Solução:
P'': S → aS' | bS''
S' → abb | b
S'' → b | aA
A → aab | b

Linguagens formais

Gramáticas livre de contexto (GLC):

Exercício 05: Fatorar a gramática abaixo.

Questão de prova (2005/02).

$$G = (\{S, C, D\}, \{a, b, c, d\}, P, S)$$
$$P = \{S \rightarrow abC \mid abD \mid a$$
$$C \rightarrow cC \mid cD \mid ba$$
$$D \rightarrow dS \mid dC \mid bc\}$$

Linguagens formais

Gramáticas livre de contexto (GLC):

Eliminação de Recursão à Esquerda (e a Direita)

Uma gramática G = (N, T, P, S) tem recursão à esquerda se existe $A \in N$ tal que $A +\rightarrow A\ \alpha$, $\alpha \in (N \cup T)^*$

Uma gramática G = (N, T, P, S) tem recursão à direita se existe $A \in N$ tal que $A +\rightarrow \alpha\ A$, $\alpha \in (N \cup T)^*$

A recursão é dita direta se a derivação acima for em um passo, isto é:

- G tem recursão direta à esquerda se existe produção $A \rightarrow A\alpha \in P$
- G tem recursão direta à direita se existe produção $A \rightarrow \alpha A \in P$

Linguagens formais

Gramáticas livre de contexto (GLC):

Para eliminar a recursão direta à esquerda usa-se o seguinte algoritmo:

Seja G = (N, T, P, S) uma GLC sem produções ε e sem produções do tipo $A \rightarrow A$.

a) Para eliminar recursão direta à esquerda envolvendo um símbolo não terminal A dividimos inicialmente o conjunto das produções de P do tipo $A \rightarrow \alpha$ em subconjuntos:

C1 = conjunto das produções $A \rightarrow \alpha$ que apresentam recursão direta à esquerda, ou seja, $A \rightarrow A\,\alpha \in P$ onde $\alpha\ (N \cup T)^*$.
C2 = conjunto das produções $A \rightarrow \beta$ que não apresentam recursão direta à esquerda, ou seja, $A \rightarrow \beta \in P \mid \beta \neq A\,\alpha$ para qualquer α.

Linguagens formais

Gramáticas livre de contexto (GLC):

b) Cria-se um novo não terminal B

c) Substitua as produções $A \rightarrow \alpha$ da gramática original de acordo com os seguintes passos:

c.1) para cada produção $A \rightarrow \beta_i \in C2$ criar a produção

$$A \rightarrow \beta_i B$$

c.2) para cada produção $A \rightarrow A\alpha_i \in C1$ criar as produções

$$B \rightarrow \alpha_i B$$

$$B \rightarrow \varepsilon$$

Linguagens formais

Gramáticas livre de contexto (GLC):

Observação: A recursão direta à esquerda foi transferida para recursão à direita.

Se a gramática original tiver produções $A \rightarrow A$ poderão surgir produções ε após a execução do algoritmo.

Além disso, podem aparecer produções simples após a execução do algoritmo.

O algoritmo para eliminação de recursão direta à direita é análogo.

Linguagens formais

Gramáticas livre de contexto (GLC):

Para eliminar a recursão geral à esquerda usa-se o seguinte algoritmo:

Método: Parte-se de uma gramática G sem produções ε

a) Estabeleça uma ordenação qualquer no conjunto de não terminais da gramática original G, isto é, faça N = { A_1, A_2, A_3, ..., A_n}, sendo que $A_1 < A_2 < A_3 < ... < A_n$

b) Gere uma GLC G' = (N', T, P', S) equivalente em que para toda regra do tipo $A_i \rightarrow A_j \alpha$ onde $\alpha \in (N \cup T)^*$, deve ocorrer $A_i < A_j$.

Linguagens formais

Gramáticas livre de contexto (GLC):

c) k = 0

enquanto existir k tal que existe produção $A_k \rightarrow A_j \alpha$ tal que k >= j, isto é, existe produção fora do padrão $A_i \rightarrow A_j \alpha$, i < j

faça: - procure o menor inteiro k tal que exista produção $A_k \rightarrow A_j \alpha$ tal que k >= j

OBS: Isto significa que para este k

- existe produção "incorreta" $A_k \rightarrow A_j \alpha$ onde k >= j
  - para todo i < k, todas as produções $A_i \rightarrow A_j \alpha$ estão "corretas", isto é, i < j
- enquanto houver produções do tipo $A_k \rightarrow A_j \alpha$, onde k > j

faça: - eliminar a produção $A_k \rightarrow A_j \alpha$

- para cada produção $A_j \rightarrow \beta$ existentes criar a produção $A_k \rightarrow \beta \alpha$

Linguagens formais

Gramáticas livre de contexto (GLC):

- se houver recursão direta à esquerda com o símbolo A_k, isto é, existe a produção $A_k \rightarrow A_k \alpha$, elimine-a pelo algoritmo dado anteriormente.

Linguagens formais

Gramáticas livre de contexto (GLC):

Exemplo: Elimine as recursões à esquerda da GLC abaixo

G = (N, T, P, S)

P: $S \rightarrow Aa$

$A \rightarrow Sb \mid cA \mid a$

Solução:

P': $S \rightarrow Aa$

$A \rightarrow Aab \mid cA \mid a$

P'': $S \rightarrow Aa$

$A \rightarrow cAA' \mid aA'$

$A' \rightarrow abA' \mid \varepsilon$

Linguagens formais

Gramáticas livre de contexto (GLC):

Exercício 06: Elimine as recursões à esquerda da GLC abaixo

G = ({S, A, B},{0, 1, 2}, P, S)

P: $S \rightarrow 0B \mid 1S \mid 10 \mid 1B \mid B0 \mid 2 \mid AB$

$A \rightarrow 0B \mid 1S \mid 10$

$B \rightarrow 1B \mid B0 \mid 2$

Linguagens formais

Próxima aula - Lembrete

2. Análise Léxica:

2.1 especificação de analisadores léxicos;

2.2 implementação de analisadores léxicos;

Atualizar e-mail no EAD.

Trazer material para escrita e exercícios (caderno, lápis, caneta, ...).

Início dos trabalhos por EAD, onde a média de todos os trabalhos irão representar ¼ da nota final. Os outros ¾ serão formados pela notas das 3 provas.

COMPILADORES

Obrigado!!

Prof. Geovane Griesang
geovanegriesang@unisc.br