Simplificação de GLC

É possível simplificar algumas produções de uma GLC sem reduzir o seu poder de geração de linguagens. A simplificação de gramáticas tem como objetivo reduzir a complexidade na geração de palavras, reduzindo a quantidade de derivações necessárias para algumas palavras, e eliminando derivações desnecessárias que não gerem nenhuma palavra.

A simplificação das GLC pode ser dividida em três etapas: simplificação das variáveis que geram a palavra vazia, simplificação das regras de produções unitárias e simplificação das regras de produção inúteis, que devem ser feitas nesta ordem (vazias, unitárias e inúteis).

Resumindo as três etapas têm-se:

- A simplificação das produções vazias pode ser definida como a exclusão de produções da forma $A \rightarrow \lambda$ (se a palavra vazia pertence à linguagem, é incluída uma produção vazia específica para tal fim, por exemplo, $S \rightarrow \lambda$)
- A simplificação das regras de produções unitárias, da forma $A \rightarrow B$, ou seja, que simplesmente substituem uma variável por outra e, consequentemente, não adicionam qualquer informação na geração de palavras.
- A simplificação das regras de produções unitárias visa eliminar as variáveis ou terminais não usados para gerar palavras.

Para a etapa de simplificação de regras de produções vazias, cria--se inicialmente um conjunto V_λ, formado pelas variáveis que geram λ diretamente ou indiretamente. Após esse conjunto V_λ definido, criam-se regras adicionais, considerando que as variáveis de V_λ podem ser vazias individualmente e em conjuntos. Ao fim, a regra de produção $S \rightarrow \lambda$ é incluída caso λ pertença à linguagem.

A simplificação de produções unitárias deve-se procurar as regras que substituem variáveis, no formato $A \rightarrow B$ (unitárias) e para cada regra $A \rightarrow B$, retira-se esta regra, incluindo no lugar regras do tipo $B \rightarrow X$, para todas as regras de B.

E por fim, a exclusão de variáveis inúteis procura variáveis com loops infinitos, variáveis não definidas ou improdutivas, variáveis inalcançáveis e estes são eliminados, eliminando também regras que contenham estas variáveis.

Dado a seguinte GLC abaixo, a simplificação segue os seguintes passos:

$G = \{V = \{S, A, B, C, D, E\}, T = \{a, b, c, d\}, P, S\}$, onde:

$P = \{ \quad S \rightarrow ABB \mid CB \mid BE$

$A \rightarrow aaA \mid \lambda$

$B \rightarrow bBb \mid A$

$C \rightarrow ccC$

$D \rightarrow dD \mid d\}$

Ao definir o conjunto V_λ, temos que a variável A chama λ diretamente, e B chama λ indiretamente, pois B gera A e A gera a palavra vazia. Definido o conjunto $V_\lambda = \{A, B\}$, revisam-se todas as regras de produção existente.

Para a regra $S \rightarrow ABB$, mantemos a regra e como temos um elemento de V_λ (variável A), consideramos que se a variável A for vazia, na regra de produção o restante da regra, o termo BB é inserido uma regra de produção $S \rightarrow BB$. Para a regra $S \rightarrow CB$ o elemento B faz parte de V_λ então repetimos a regra e inserimos uma nova regra de produção com $S \rightarrow C$. Para a terceira regra de produção de S, $S \rightarrow BE$, também temos que B faz parte de V_λ assim repetimos a regra e inserimos uma nova regra $S \rightarrow E$, considerando que B pode ser vazio.

Para as regras de A, a primeira regra de produção $A \rightarrow aaA$ é repetida nas produções e como temos uma variável pertencente ao conjunto V_λ inserimos uma nova regra considerando que esta variável (A) pode ser vazia, resultando na inserção da regra $A \rightarrow aa$. Para a segunda regra de produção de A, $A \rightarrow \lambda$, o objetivo desta etapa é eliminar regras de produções que gerem a palavra vazia, então esta regra é simplesmente eliminada.

Para as regras de B, em $B \rightarrow bBb$ como B faz parte do conjunto V_λ consideramos que ele pode ser vazio e inserimos uma nova regra com os símbolos restantes da regra de produção $B \rightarrow bb$, e em $B \rightarrow A$, se A, que faz parte de V_λ resultar em vazio, temos a palavra vazia, porém nesta etapa estamos eliminando as regras que geram vazias, e não iremos incluir uma regra que gere a palavra vazia, somente então repetindo a regra de produção $B \rightarrow A$.

Para a regra de C, $C \rightarrow ccC$, e para as regras de D, $D \rightarrow dD$ e $D \rightarrow d$, não temos nenhuma variável pertencente a V_λ do lado direito das regras de produções, portanto somente repetimos estas regras.

As regras de produções resultantes da etapa de eliminação das produções vazias podem ser vistas abaixo.

$P = \{ \quad S \rightarrow ABB \mid BB \mid CB \mid C \mid BE \mid E$

$A \rightarrow aaA \mid aa$

$B \rightarrow bBb \mid bb \mid A$

$C \rightarrow ccC$

$D \rightarrow dD \mid d\}$

Para a etapa de eliminar as regras de produções unitárias, procura-se nas regras de produções resultantes da primeira etapa as regras unitárias, ou seja, as regras nas quais temos somente uma variável do lado direito das produções. Para o exemplo acima, temos que S → C, S → E e B → A são exemplos de regras de produção unitárias. Estas regras são eliminadas e incluímos outras regras no lugar destas, de acordo com a variável do lado direito das produções que foram excluídas, por exemplo, eliminamos a regra S → C, e incluímos nas regras de S todas as regras que existem em C, assim uma nova regra S → ccC é incluída em S. Para a regra S → E, eliminamos esta regra e inserimos todas as regras de E em S, porém não existe nenhuma regra de E, ou seja, a variável E não está definida nas regras de produção sendo considerada então uma variável inútil na formação das palavras da linguagem. Na etapa de eliminação de variáveis inúteis verificamos se uma variável não está definida nas regras de produção, porém alguns casos pode-se perceber que uma variável não está definida já em etapas anteriores, neste caso, por exemplo, assim eliminamos a regra S → E somente e não inserimos nenhuma regra a mais. Na terceira regra unitária, B → A, eliminamos esta regra e adicionamos as duas regras de A em B, resultando em B → aaA e B → aa.

Todas as outras regras de produção que não tiveram problema na etapa de eliminação das unitárias são repetidas, resultando nas seguintes regras:

P = { S → ABB | BB | CB | ccC | BE

A → aaA | aa

B → bBb | bb | aaA | aa

C → ccC

D → dD | d}

A terceira etapa visa eliminar variáveis inúteis, ou seja, variáveis que não contribuem para a formação das palavras da linguagem. As variáveis inúteis podem ser divididas em três categorias, as variáveis que são inalcançáveis, as variáveis improdutivas e as variáveis que geram loops infinitos na derivação das palavras.

Uma variável é inalcançável quando nenhuma variável a gera por alguma derivação. Para procurar por variáveis inalcançáveis podemos verificar que a partir do símbolo de partida S, geramos A e B pela regra S → ABB, e geramos C pela regra S → CB. Ainda pela regra S → CE, geramos a variável E, ou seja, todas estas variáveis A, B, C e E são alcançáveis a partir de S. Ainda não sabemos se D é alcançável, porém podemos verificar que nenhuma das variáveis que já foi alcançada (A, B, C e E) gera D, isto indica que a variável D não é gerada em nenhum momento das derivações usando as regras de produção acima, Indicando que D é uma variável inalcançável e deve ser eliminada todas as regras de D, eliminando assim D → dD e D → d. As produções resultantes desta etapa são:

P = { S → ABB | BB | CB | ccC | BE

A → aaA | aa

B → bBb | bb | aaA | aa

C → ccC}

O próximo passo na eliminação das variáveis inúteis é verificar se existem variáveis improdutivas, ou seja, variáveis que não possuem nenhuma regra de produção, tal como a eliminação da variável E na etapa de unitárias. Nesta etapa da simplificação temos uma regra de produção que contém a variável E, S → BE, porém a variável E não está definida nas regras de produção da gramática, ou seja, ela é improdutiva, não produz nenhuma sentença, devendo ser eliminado todas as regras nas quais aparece a variável E, eliminando então a regra de produção S → BE, resultando nas seguintes regras de produção:

P = { S → ABB | BB | CB | ccC

A → aaA | aa

B → bBb | bb | aaA | aa

C → ccC}

Por fim, o último passo da eliminação das variáveis inúteis é verificar se alguma variável possui loops infinitos, ou seja, verificar que as derivações de uma variável devem terminar em algum momento. Lembrando que uma derivação termina quando só temos terminais gerados, ou seja, enquanto houver uma variável na sentença a derivação não terminou.

Como a derivação começa com o símbolo de partida S, se este símbolo tiver um loop infinito significa que nenhuma palavra será gerada pela gramática, resumindo, dificilmente temos uma gramática na qual o símbolo de partida é uma variável inútil. Portanto analisando os outros símbolos, para o símbolo A, a regra de produção A → aaA representa um loop que gera dois terminais aa, e mais uma variável A, sendo considerado um loop, porém para a derivação de A finalizar deve-se usar a regra de produção A → finalizando então a derivação de A, e não sendo então a variável A um loop infinito.

Para a variável B, temos que a regra B → bBb é um loop com B, as regras B → bb e B → aa finalizam a derivação de B, e a regra B → aaA gera um A, que pode ser finalizado com a derivação de dois a's pela regra A → aa, portanto B não é uma variável de loop infinito.

Já a variável C tem somente uma regra, C → ccC, na qual a cada derivação de C é gerado novamente a variável C, sendo um loop e, como não temos uma regra de produção que gere somente símbolos terminais a derivação de C fica em loop infinito nunca terminando, sempre com um C para ser derivado. Assim, a variável C é uma variável de loop infinito e deve ser removida da gramática, bem como todas as regras de produção que derivam a variável C, resultando então nas seguintes regras de produção:

P = { $S \rightarrow ABB \mid BB$

$A \rightarrow aaA \mid aa$

$B \rightarrow bBb \mid bb \mid aaA \mid aa$}

Ainda na eliminação de variáveis inúteis podemos eliminar também terminais que não fazem parte da linguagem gerada pela gramática, por exemplo, na gramática original o conjunto de terminais era formado por T = {a, b, c, d}, porém nas regras de produções resultante não aparecem os símbolos c e d, que foram eliminados junto com as variáveis C, D e E, assim podemos eliminar estes terminais do conjunto T.

Passado então as 3 etapas da simplificação, etapa de eliminação de produções vazias, etapa de eliminação de regras de produção unitárias e eliminação das variáveis inúteis, temos a seguinte gramática resultante simplificada:

G = {V = {S, A, B}, T = {a, b}, P, S}, onde:

P = { $S \rightarrow ABB \mid BB$

$A \rightarrow aaA \mid aa$

$B \rightarrow bBb \mid bb \mid aaA \mid aa$}

Exemplo 2 – Faça a simplificação da GLC abaixo e apresente a gramática simplificada resultante

G = {V = {S, A, B, C, D}, T = {a, b, c, d}, P, S}, onde:

P = { $S \rightarrow AAC \mid BSB \mid A$

$A \rightarrow aB \mid a$

$B \rightarrow bB \mid \lambda$

$C \rightarrow cCc \mid c$

$D \rightarrow dD \mid d$ }

Para a etapa de eliminação das produções vazias, construímos o conjunto V_λ, que é formado pelos elementos que geram a palavra vazio diretamente e indiretamente, sendo então formado por B, V_λ = {B}. Assim incluímos novas regras de produções, considerando as regras nas quais os símbolos de V_λ aparecem, e considerando que eles podem ser vazios independentemente um do outro ou em conjunto quando aparecem mais de um símbolo em uma regra de produção. Assim temos que as regras de produção resultantes são:

P = {   S → AAC | BSB | SB | BS | S | A

A → aB | a

B → bB | b

C → cCc | c

D → dD | d }

Lembrando que para a etapa de eliminação de produções vazias nesta gramática, como B faz parte de V_λ, para a regra S → BSB, consideramos que os símbolos B podem ser vazios independentemente um do outro, e em conjunto, assim adicionamos a regra S → SB considerando que o primeiro B pode ser vazio, adicionamos a regra S → BS, considerando que o segundo B pode ser vazio, e adicionamos ainda a regra S → S, considerando que os dois B's podem ser vazios ao mesmo tempo.

Ainda na etapa de eliminação de vazias desta gramática, para a regra A → aB, considerando que B pode ser vazio, adicionaríamos a regra A → a, porém esta regra já existe nas produções de A, então não precisamos adicionar novamente.

Para a eliminação de produções unitárias, temos duas regras unitárias, S → S e S → A, para a segunda regra, realizamos a adição das regras de A em S normalmente, resultando em S → aB e S → a, porém para a primeira regra, S → S, deveríamos colocar todas as regras de S em S porém fazer isso repetiríamos regras de S em S novamente, assim somente eliminamos a regra S → S das regras de produção, sem adicionar nenhuma regra de produção. As regras de produção resultante desta etapa são:

P = {   S → AAC | BSB | SB | BS | aB | a

A → aB | a

B → bB | b

C → cCc | c

D → dD | d }

Por fim, a eliminação das regras de produção inúteis, dividindo em variáveis inalcançáveis, variáveis improdutivas e variáveis com loop infinito. Na análise de variáveis inalcançáveis, percebemos que D não é gerado em nenhum momento, eliminando assim todas as suas regras. Para variáveis improdutivas e loop infinito não encontra-se nenhuma na gramática, resultando então na seguinte gramática simplificada:

G = {V = {S, A, B, C, D}, T = {a, b, c, d}, P, S}, onde:

P = {   S → AAC | BSB | SB | BS | aB | a

A → aB | a

B → bB | b

C → cCc | c}

A simplificação de GLC é utilizada ainda para colocar as GLC em formas normais, sendo as mais utilizadas a Forma Normal de Chomsky e a Forma Normal de Greibach.

Bibliografia

VIEIRA, NEWTON JOSÉ. **Introdução aos Fundamentos da Computação**. São Paulo. Pioneira Thomson Learning. 2006;
MENEZES, P. B.; DIVERIO, T. A.; **Teoria da Computação: Máquinas Universais e Computabilidade**; 3ª edição Bookman 2011
MENEZES, P. B. ; **Linguagens Formais e Autômatos**. 6ª edição. Ed. Artmed. 2011
SIPSER M. **Introdução à Teoria da Computação**. 2 ed. Cengage Learning.2007
MORET, B. M. **"Theory of Computation".** Addison-Wesley, 1998.
HOPCROFT, JOHN E.; ULLMAN, JEFFREY D.; MOTWANI, RAJEEV **Introdução à Teoria de Autômatos, Linguagens e Computação** Ed.Campus 2002
SILVA, FLAVIO SOARES CORRÊA; MELO, ANA CRISTINA VIEIRA; **Modelos Clássicos de Computação** Ed. Thomson 1ª Edição 2006