Gramáticas sensíveis ao contexto

IBM1088 Linguagens Formais e Teoria da Computação

Evandro Eduardo Seron Ruiz
evandro@usp.br

Departmento de Computação e Matemática
FFCLRP
Universidade de São Paulo

Homenagem do dia

Machado de Assis

O dinheiro não traz felicidade – para quem não sabe o que fazer com ele.

MA: Escritor e poeta brasileiro, 1839 – 1908.

Recordando. . . Definição: GLC

Definição de GLC

Uma gramática livre de contexto é uma 4-tupla (V, Σ, P, S), em que:

- V é um conjunto finito de símbolos denominado **variáveis**,
- Σ é um conjunto finito de símbolos, disjunto de V, denominado **terminais**,
- P é um conjunto finito de regras de produção da forma $A \to \alpha$, tal que $A \in V$ e $\alpha \in (V \cup \Sigma)^*$.
- $S \in V$ é a variável inicial

Portanto, uma GLC é uma gramática na qual o lado esquerdo das produções contém exatamente uma variável.

Motivação

- "Lema do Bombeamento" para LLC mostra que existem outras linguagens além destas
- As GLC restringem o lado esquerdo das regras de produção à uma única variável
- Todas as derivações são feitas dependendo apenas desta variável
- Estas gramáticas não consideram o contexto. Por exemplo, como os termos vizinhos desta variável
- Isso dificulta a representação de casos como a declaração de variáveis em LP e
- E a declaração de comandos válidos a partir de variáveis declaradas

Conteúdo

GSC: o que é

Uma Gramática Sensível ao Contexto $G = (V, \Sigma, P, S)$ é uma gramática cujas regras de produção P são formuladas como $\alpha \rightarrow \beta$, tal que:

- α e β são cadeias de terminais e não terminais
- $|\beta| \geq |\alpha|$

Observações:

- O conceito de GSC foi introduzido por Noam Chomsky em 1950
- Para toda derivação o comprimento da cadeia nunca diminui ($|\beta| \geq |\alpha|$)

LLC: premissas

Linguagem sensível ao contexto

Uma LSC é toda linguagem definida através de uma GSC.

- Esta definição estende-se a todas as linguagens que contenha a cadeia vazia
- Desde que $L - \{\epsilon\}$ possa ser gerada por uma GSC
- Estas são as linguagens do tipo 1

Veja próximo slide:

GSC: definição

Definição de GSC

Uma gramática sensível ao contexto é uma 4-tupla (V, Σ, P, S), em que:

- V é um conjunto finito de símbolos denominado **variáveis**,
- Σ é um conjunto finito de símbolos, disjunto de V, denominado **terminais**,
- P é um conjunto finito de regras de produção da forma $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$, tal que, $A \in V$ e $\alpha_1, \alpha_2, \beta \in (V \cup \Sigma)^+$.
- $S \in V$ é a variável inicial

LLC: explicações

- Uma GSC não pode sofrer redução na formação de suas palavras
- Por isso limita-se o ϵ à regra $S \to \epsilon$ desde que S não apareça do lado direito em outra regra de produção
- Assim garantimos que as linguagens geradas a partir de uma GSC com esta limitação são, de fato, GSC

Veremos mais detalhes adiante, ainda nesta aula.

Exemplo: Linguagem Estritamente Sensível ao Contexto

Uma LESC é uma LSC mas que não é uma LLC.

Exemplo

Seja a gramática $G_1 = \{\{a, b, c, S, B, C\}, \{a, b, c\}, P, S\}$, tal que

$$
\begin{aligned}
P = \{ & S \rightarrow aSBC, \\
& S \rightarrow aBC, \\
& CB \rightarrow BC, \\
& aB \rightarrow ab, \\
& bB \rightarrow bb, \\
& bC \rightarrow bc, \\
& cC \rightarrow cc\}
\end{aligned}
$$

Comentários

- Todas as regras possuem pelo menos um não-terminal do lado esquerdo e uma seqüência arbitrária de terminais e não terminais do lado direito
- $|\beta|$ não é inferior ao comprimento do lado esquerdo
- Possíveis derivações:
  - $S \Rightarrow aBC \Rightarrow abC \Rightarrow abc$
  - $S \Rightarrow aSBC \Rightarrow aaBCBC \Rightarrow aaBBCC \Rightarrow aabBCC \Rightarrow aabbCC \Rightarrow aabbcC \Rightarrow aabbcc$
  - $S \Rightarrow aSBC \Rightarrow aaSBCBC \Rightarrow aaaBCBCBC \Rightarrow aaaBBCCBC \Rightarrow aaBBCBCC \Rightarrow aaaBBBCCC \Rightarrow \ldots \Rightarrow aaabbbccc$

$L(G_1) = \{a^n b^n c^n \mid n \geq 1\}$. Já demonstramos pelo Lema do Bombeamento que $L(G_1)$ não é uma LLC.

Observações importantes

- As LLC que não contém ϵ estão contidas no universo das LSC
- Nem todas as LSC são LLC
- Todas LSC são recursivas, decidíveis

Monotônicas

- As GSC são ditas monotônicas
- O comprimento das cadeias derivadas nunca sofre redução
- Veja exemplo anterior

Monotônicas

- As GSC são ditas monotônicas
- O comprimento das cadeias derivadas nunca sofre redução
- Veja exemplo anterior
- No entanto, nem todas as gramáticas do tipo 2 (LC) são também do tipo 1 (SC)
- GLC admitem ϵ, a cadeia vazia, como substituição ao lado esquerdo, o que viola a regra abaixo
- $|\beta| \geq |\alpha|$

De volta a definição

Rigorosamente uma linguagem é dita uma LLC see:

- $\epsilon \notin L$ e $L = L(G)$, tal que G é uma GSC; **ou**
- $\epsilon \in L$ e $L - \{\epsilon\}$ pode ser gerada por uma GSC.

De volta a definição

Rigorosamente uma linguagem é dita uma LLC see:

- $\epsilon \notin L$ e $L = L(G)$, tal que G é uma GSC; **ou**
- $\epsilon \in L$ e $L - \{\epsilon\}$ pode ser gerada por uma GSC.

Na última regra, aceita-se:

- Aceita-se a regra $S \rightarrow \epsilon$ desde que S não apareça do lado direito de nenhuma outra regra
- Assim, por derivações sucessivas, não ocorre contração da cadeia gerada

'Consertando' uma GLC

- Seja $L(G)$ uma LLC e $\epsilon \in L(G)$
- Pode-se transformar G em G' de modo que:
  - $S \to \epsilon$ seja a única regra vazia em G'
  - S não aparece do lado direito de outras regras de G'
  - $L(G) = L(G')$

Ou seja, uma transformação de uma LLC numa LSC é sempre possível.

‘Consertando’ uma GLC

- Seja $L(G)$ uma LLC e $\epsilon \in L(G)$
- Pode-se transformar G em G' de modo que:
  - $S \rightarrow \epsilon$ seja a única regra vazia em G'
  - S não aparece do lado direito de outras regras de G'
  - $L(G) = L(G')$

Ou seja, uma transformação de uma LLC numa LSC é sempre possível.

Conclusão

Toda LLC é também uma LSC.
As LLC são um subconjunto próprio das LSC.

Outro exemplo

Considere a gramática definida pelas produções

$$
\begin{aligned}
\{\textit{Programa} &\to \textit{Declaracoes}\ \ \textit{Comandos},\\
\textit{Declaracoes} &\to \textit{Declaracoes}\ \ \textit{Declaracao} \mid \epsilon,\\
\textit{Declaracao} &\to \text{`\%'}\textit{Identificador},\\
\textit{Comandos} &\to \textit{Comandos}\ \ \textit{Comando} \mid \epsilon,\\
\textit{Comando} &\to \text{`\#'}\textit{Identificador}\ \text{`='}\ \textit{Expressao},\\
\textit{Expressao} &\to \textit{Expressao}\ \text{`+'}\textit{Expressao}\\
&\mid \textit{Expressao}\ \text{`*'}\textit{Expressao}\\
&\mid \textit{Identificador},\\
\textit{Identificador} &\to \text{`a'} \mid \text{`b'} \mid \text{`c'}\}
\end{aligned}
$$

Um exemplo de programa nesta linguagem

```
%a
%b
#a = a + b
#b = b * b
```

- Contém a declaração de duas variáveis e dois comandos de atribuição usam estas variáveis
- Imagina-se que os únicos identificadores possíveis nos comandos (depois do '#') devam ter sido declarados antes (depois do '%')

Um exemplo de programa nesta linguagem

```
%a
%b
#a = a + b
#b = b * b
```

- Contém a declaração de duas variáveis e dois comandos de atribuição usam estas variáveis
- Imagina-se que os únicos identificadores possíveis nos comandos (depois do '#') devam ter sido declarados antes (depois do '%')
- Mas... esta regra não está na GLC acima
- O que pode levar a escrevermos programas como este...

Programa que viola a regra esperada

```
%a
%c
#a = a + b
#b = b * b
```

A variável ‘b’ é referenciada indevidamente pois sua declaração foi substituída pela declaração da variável ‘c’.

Programa que viola a regra esperada

```
%a
%c
#a = a + b
#b = b * b
```

A variável ‘b’ é referenciada indevidamente pois sua declaração foi substituída pela declaração da variável ‘c’.

Próximo!

Vejamos agora exemplos de autômatos que reconhecem/geram LSC

MT com fita limitada

- *Linear Bounded Automata*
- ALL – Autômato Linearmente Limitado
- LSC podem ser reconhecidas por MT cujo comprimento da fita é função linear do tamanho da cadeia de entrada
- $|w| = n$, esta palavra necessita de $|\text{fita}| = n + 2$

Características da MT com fita limitada

- Fita de trabalho tem comprimento acrescido de duas posições: inicio da cadeia e final da cadeia
- Cursos de acesso aos símbolos pode mover para esquerda (E) e para direita (D)
- Cursor de acesso (cabeça de I/O) pode ler e escrever símbolos em substituição ao símbolo apontado

Diferentemente dos autômatos anteriores, na MT a fita que contém a cadeia de entrada não é de apenas leitura mas é uma fita de trabalho.

Formalização da MT

Formalização da Máquina de Turing

Uma MT é uma 7-upla $M = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, para Q, Σ e Γ conjuntos finitos

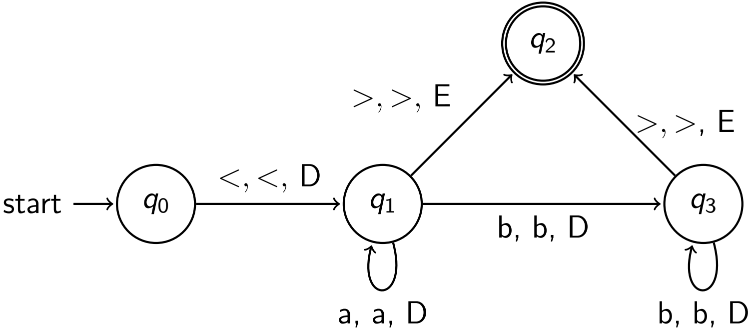
1. Q o conjunto dos estados
2. Σ o alfabeto de entrada (sem ␣)
3. Γ é o alfabeto da fita (␣$\in \Gamma$ e $\Sigma \subseteq \Gamma$)
4. $\delta : Q \times \Gamma \rightarrow Q \times \{E, D\}$ é a função de transição
5. q_0 é o estado inicial
6. $q_A \in Q$ é o estado de aceitação
7. $q_R \in Q$ é o estado de rejeição, $q_A \neq q_R$

Acrescentarei dois símbolos especiais: '$<$' início da cadeia e, '$>$' final da cadeia.

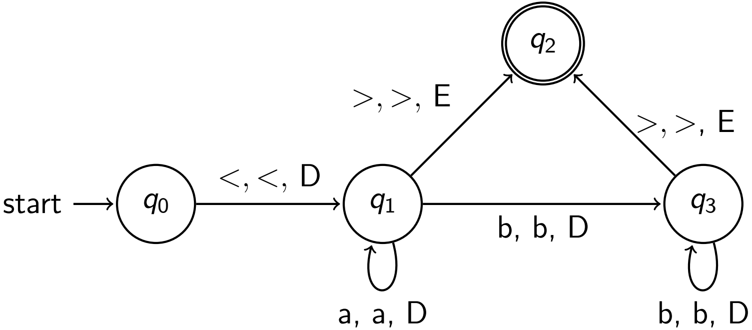
Comentários importantes

- δ é a função de transição que compreende:
  - $Q \times \Gamma \to 2^{Q \times \Gamma \times \{E,D\}}$
  - $Q \times \{<\} \to 2^{Q \times \{<\} \times \{D\}}$
  - $Q \times \{>\} \to 2^{Q \times \{>\} \times \{E\}}$
- Ou seja, os símbolos especiais só podem ser gravados na posição original
- O símbolo de espaço em branco ($\llcorner\!\lrcorner$) pode ser representado por $\square$

Exemplo: MT com fita limitada que aceita $L_1 = \{a^* b^+\}$

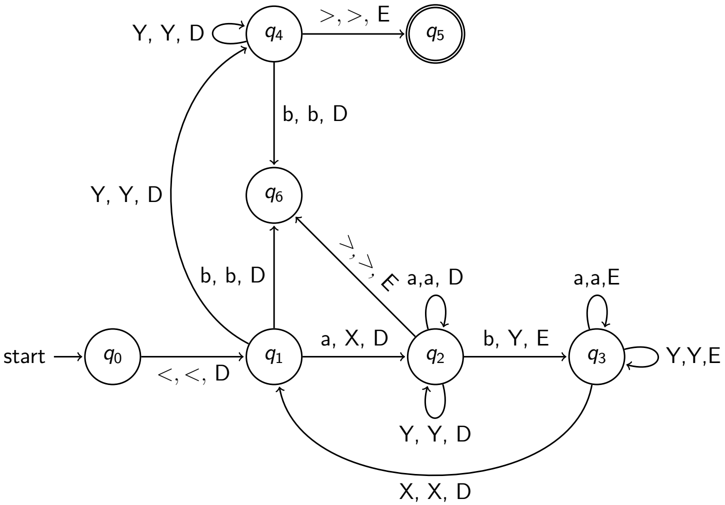


Exemplo: MT com fita limitada que aceita $L_1 = \{a^* b^+\}$



Este exemplo sugere que MT com fita limitada possam ser usadas em substituição aos autômatos finitos, a despeito do custo de gravação na fita.

Exemplo: MT com fita limitada que aceita $L_2 = \{a^n b^n \mid n \geq 1\}$



Como funciona:

1. O símbolo ‘a’ é substituído por ‘X’. Cursor ‘D’
2. Cursor continua ‘D’ até encontrar ‘b’ ou ‘$>$’
3. Se encontar ‘$>$’ cadeia rejeitada (mais ‘a’ que ‘b’). Se encontrar ‘b’ substitui por ‘Y’, desloca ‘E’ até ‘X’. (*) Desloca mais uma posição ‘D’e volta ao passo 1.
4. Se (*) encontrar ‘Y’ todos os ‘a’ foram considerados. Só restam ‘Y’ e a cadeia é aceita. Caso contrário a cadeia é rejeitada (mais ‘b’ que ‘a’)

Para casa

Desafio!

Escrever uma MT que reconheça $L_3 = \{wcw \in \{a, b\}\}$.

Notem que L_3 é realmente uma LSC.

Finalizando...

Caros estudantes,

Leiam os livros texto e façam os exercícios sugeridos.

Resposta ao desafio: ALL que aceita $L_3 = \{wcw \mid w \in \{a, b\}\}$