ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Gramáticas Livres do Contexto:
2.1 Forma Normal de Chomsky
7.2 Algoritmo CYK (Cocke–Younger–Kasami)

Marcelo Lauretto
(Parte do material gentilmente cedido
pela Profa. Ariane Machado Lima)

Forma Normal de Chomsky

Uma GLC está na Forma Normal de Chomsky se:

a) Toda regra de substituição é da forma

$A \rightarrow BC$ ou $A \rightarrow a$

onde B,C são variáveis, a é símbolo terminal;

b) A variável inicial S não pode aparecer no lado direito de nenhuma regra;

c) Somente a variável inicial pode ter a regra

$S \rightarrow \varepsilon$.

Forma Normal de Chomsky

- Conversão de uma GLC G = (V, Σ, R, S) para FNC:

  a) Adicionar nova variável inicial S_0 e adicionar a regra $S_0 \rightarrow S$;

  b) Eliminação de regras $A \rightarrow \varepsilon$

  b_1) Remover a regra;

  b_2) Para toda regra $R \rightarrow u\ A\ v$, adicionar $R \rightarrow u\ v$;

  Nota: Fazer isso para cada ocorrência de A.

  Ex: se $R \rightarrow u\ A\ v\ A\ w$, deve-se acrescentar 3 regras:

  $$R \rightarrow u\ v\ A\ w, \quad R \rightarrow u\ A\ v\ w, \quad R \rightarrow u\ v\ w$$

  b_3) Se tivermos a regra $R \rightarrow A$ e se $R \rightarrow \varepsilon$ não tiver sido previamente eliminado, adicionar $R \rightarrow \varepsilon$

  (posteriormente, essa regra também será removida se $R \neq S_0$)

  b_4) Repetir até eliminar todas as ocorrências.

Forma Normal de Chomsky

- Conversão de uma GLC G = (V, Σ, R, S) para FNC (cont.):

  a) Remoção de regras unitárias A → B:

  c1) Remover a regra;

  c2) Para toda regra B → u, acrescentamos A → u, a menos que essa seja uma regra unitária já removida.

  c3) Repetir para todas as regras unitárias.

Forma Normal de Chomsky

- Conversão de uma GLC G = (V, Σ, R, S) para FNC (cont.):

  a) Converter todas as regras remanescentes para a forma apropriada $A \rightarrow BC$ ou $A \rightarrow a$:

  d1) Se $A \rightarrow u_1u_2 \ldots u_k$, onde $k \geq 3$ e u_i é variável ou símbolo terminal, então substituir esta regra por:

$A \rightarrow u_1A_1$, $A_1 \rightarrow u_2A_2$, $A_2 \rightarrow u_3A_3$, ..., $A_{k-2} \rightarrow u_{k-1}u_k$.

  d2) Se k=2, então substituir qualquer terminal u_i na(s) regra(s) precedente(s) por uma nova variável U_i, e adicionar a regra $U_i \rightarrow u_i$.

Algoritmos para Processamento de Gramáticas Livres do Contexto

- Problema:

  A_{GLC} = { (G, w) | G é uma GLC que gera a cadeia w }

- Restrição:
  - Se G for uma gramática qualquer:
    - Testar todas as possíveis derivações: loop infinito
  - Se G estiver na Forma Normal de Chomsky:
    - Teorema: qualquer derivação de uma cadeia não vazia *w* tem exatamente 2*n*-1 passos, onde *n* = |*w*|.
    - Método de força bruta (para n>0): custo exponencial
      - Teste todas as derivações possíveis com 2n-1 passos.
      - Se alguma das derivações gerar w, aceite; senão, rejeite

Algoritmos para Processamento de Gramáticas Livres do Contexto

- Algoritmo Polinomial (G na forma normal de Chomsky)
- **Programação dinâmica**: uso de soluções de subproblemas menores para resolver subproblemas maiores (até chegar à solução do problema original)
- Tabela n×n:
  - Para i ≤ j, a (i, j) entrada da tabela contém todas as varíaveis que geram a subcadeia $w_i\, w_{i+1} \ldots w_j$
  - Tratam-se subcadeias de tamanhos crescentes (começando de 1)

Algoritmo CYK – programação dinâmica

$D =$ "Sobre a entrada $w = w_1 \cdots w_n$:

1. Se $w = \varepsilon$ e $S \to \varepsilon$ for uma regra, *aceite*. [[trata o caso $w = \varepsilon$]]
2. Para $i = 1$ até n: [[examina cada subcadeia de comprimento 1]]
3. Para cada variável A:
4. Teste se $A \to \mathtt{b}$ é uma regra, onde $\mathtt{b} = w_i$.
5. Se for, coloque A em $tabela(i, i)$.
6. Para $l = 2$ até n: [[l é o comprimento da subcadeia]]
7. Para $i = 1$ até $n - l + 1$: [[i é a posição inicial da subcadeia]]
8. Faça $j = i + l - 1$, [[j é a posição final da subcadeia]]
9. Para $k = i$ até $j - 1$: [[k é a posição em que ocorre a divisão]]
10. Para cada regra $A \to BC$:
11. Se $tabela(i, k)$ contém B e $tabela(k + 1, j)$ contém C, ponha A em $tabela(i, j)$.
12. Se S estiver em $tabela(1, n)$, *aceite*. Caso contrário, *rejeite*."

Exemplo

Conversão de uma GLC para FNC:

Gramática original:
$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

1) Criação de nova variável inicial:
$S_0 \rightarrow S$
$S \rightarrow aSb \mid bSa \mid SS \mid \varepsilon$

2) Eliminação de substituições ε :
$S_0 \rightarrow S \mid \varepsilon$
$S \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba \mid S$

3) Eliminação de regras unitárias:
$S_0 \rightarrow \varepsilon \mid aSb \mid bSa \mid SS \mid ab \mid ba$
$S \rightarrow aSb \mid bSa \mid SS \mid ab \mid ba$

4) Padronização:
$S_0 \rightarrow \varepsilon \mid AT \mid BU \mid SS \mid AB \mid BA$
$S \rightarrow AT \mid BU \mid SS \mid AB \mid BA$
$T \rightarrow SB$
$U \rightarrow SA$
$A \rightarrow a$
$B \rightarrow b$

Exemplo

Aplicação algoritmo CYK:

Grámática na FNC:
$S_0 \rightarrow \varepsilon \mid AT \mid BU \mid SS \mid AB \mid BA$
$S \rightarrow AT \mid BU \mid SS \mid AB \mid BA$
$T \rightarrow SB$
$U \rightarrow SA$
$A \rightarrow a$
$B \rightarrow b$

Cadeia:
a b a a b b

Tabela:

	a	b	a	a	b	b
a	A	S_0, S	U	∅	∅	S_0, S
b		B	S_0, S	U	S_0, S	T
a			A	∅	∅	S_0, S
a				A	S_0, S	T
b					B	∅
b						B