CONJUNTO FIRST E FOLLOW

Silvana Trindade e Maurício André Cinelli

29 de junho de 2014

Analisador Preditivo Tabular

A análise preditiva tabular implementa um autômato com pilha controlado por uma tabela de análise. O princípio do reconhecimento preditivo é a determinação da produção a ser aplicada, cujo lado direito irá substituir o símbolo não terminal que se encontra no topo da pilha. O analisador busca a produção a ser aplicada na tabela de análise, levando em conta o não terminal no topo da pilha e o token sob o cabeçote de leitura.

A fita de entrada contém a sentença a ser analisada seguida de $, símbolo que marca o fim da sentença. Inicialmente, a pilha contém $, que marca a sua base, seguido do símbolo inicial da gramática. A tabela de análise á uma matriz com n linhas e $(t+1)$ colunas, onde n é o número de símbolos não-terminais e t é o número de símbolos terminais.

Considere X o símbolo no topo da pilha e a o terminal da fita. O analisador executa uma das três possíveis ações:

1. se $X = a = \$$, o analisador para, aceitando a senteça
2. se $X = a \neq \$$, o analisador desempilha a e avança o cabeçote de leitura para o próximo símbolo na fita.
3. se X não terminal, o analisador consulta a entrada $M[X,a]$ da tabela de análise. Então o X é substituído pelo lado direito da regra de produção escolhida.

Conjuntos First e Follow

Dada uma palavra γ contendo símbolos não terminais e terminais, $FIRST(\gamma)$ é o conjunto de todos os símbolos terminais que podem começar cqualquer palavra derivada de γ .

Se duas produções $X \to \gamma_1$ e $X \to \gamma_2$ possuem o mesmo símbolo não terminal X à esquerda da produção e seus símbolos à direita possuem conjuntos $FIRST$ sobrepostos, então esta gramática não pode ser analisada utilizando algoritmo dscendente recursivo. Se um símbolo terminal I pertence a $FIRST(\gamma_1)$ e à $FIRST(\gamma_2)$, então a função que representa a produção X em um analisador descendente recursivo não saberá qual cláusula condicional executar quando o token de entrada é I. O calculo do conjunto $FIRST$ aparentemente é muto simples: se $\gamma = XYZ$, aparentemente Y e Z podem ser ignorados, e $FIRST(X)$ é a unica coisa que realmente interessa para o cálculo de $FIRST(\gamma)$. Infelizmente as coisas não são tão simples assim, como por exemplo na gramática apresentada abaixo:

$$\begin{array}{ll} Z \to d & Y \to \varepsilon \\ Z \to XYZ & X \to Y \\ Y \to c & X \to a \end{array}$$

Nesta gramática Y pode produzir a palavra vazia - ε portanto X pode produzir a palavra vazia - ε, descobre-se que $FIRST(XYZ)$ deve incluir $FIRST(Z)$.

Portanto, no cálculo dos conjuntos FIRST, é necessário armazenar quais símbolos poderão produzir palavras vazias; tais símbolos são denominados *nullable*. É necessário armazenar também o que pode seguir um símbolo *nullable*.

Dada uma cadeia particular w de uma gramática contendo símbolos terminais e não terminais.

- *nullable*(X) é verdadeiro se X pode derivar a palavra vazia
- $FIRST(w)$ é o conjunto de terminais que podem iniciar palavras derivadas de γ
- $FOLLOW(X)$ é o conjunto de terminais que podem imediatamente seguir X. Isto é, $t \in FOLLOW(X)$ se existe qualquer derivação contendo X_t. Isto pode ocorrer se a derivação XYZ_t onde Y e Z ambos derivem ε.

Algoritmo

O algoritmo de geração dos conjuntos first e follow tem como entrada o alfabeto da linguagem e a gramática a ser analisada.

O conjunto first, na primeira etapa, analisamos cada estado, varrendo as suas produções e verificando se o primeiro símbolo de cada produção é terminal, e o atribuímos ao conjunto first do estado em questão. Na segunda etapa, é verificado todos os símbolos não terminais da esquerda para a direita. Se existir um próximo símbolo, e esse for um símbolo terminal, então este símbolo entra no conjunto first do não-terminal sendo verificado. Se este próximo símbolo for um não-terminal, verifica-se se este estado possui ε em seu conjunto first, então copia-se o first deste próximo estado, para o estado sendo verificado, e o próximo estado passa a ser verificado, seguindo os mesmos passos.

No conjunto follow, primeiramente, atribui-se ao follow do estado inicial da gramática, o símbolo \$ ($S' ::= S\$$). Na primeira etapa do processamento, percorre-se todas as produções (S) da gramática, analisando os símbolos não-terminais. Para todos os não-terminais(S), $FOLLOW(A) = First(B)$.

$S ::= ABS|aA$
$A ::= \varepsilon|a$
$B ::= Bb|cd$

Caso o próximo símbolo do não-terminal sendo verificado for um símbolo terminal, este é adicionado ao conjunto follow to não-terminal em questão. Por exemplo, a produção Bb, e B é o não-terminal sendo verificado, e b é o próximo símbolo, então b é adicionado ao conjunto follow de B.

Na segunda etapa, é verificado se o último caractere de cada produção é um não-terminal. Se for um não-terminal, faz-se o seguinte:

$FOLLOW(A) = FOLLOW(A) \cup FOLLOW(S)$,

onde S é o estado onde a produção esta sendo vericada. Além disso verifica-se se este estado A possui em seu conjunto first o ε, se sim passa a verificar o anterior, seguindo os mesmos conjuntos de passos descritos acima.

Considerações Finais

Através do algoritmo o grupo conseguiu obter os conjuntos First e Follow das gramáticas livre de contexto. Podendo assim exercitar melhor nosso conhecimento deste tópico, tirando assim algumas dúvidas.