DETERMINIZAÇÃO DE GRAMÁTICAS REGULARES

Silvana Trindade e Maurício André Cinelli

30 de maio de 2014

Linguagens Regulares

As linguagens regulares são constituídas de um conjunto de linguagens decidíveis simples e com propriedades bem definidas e compreendidas. Essas linguagens podem ser reconhecidas por autômatos finitos e são facilmente descritas por expressões simples, chamadas expressões regulares (ER).

O estudo das linguagens regulares pode ser abordado através de três diferentes formalismos:

- *operacional ou reconhecedor*: Autômato Finito, que pode ser determinístico, não determinístico ou com movimento vazio (com ϵ-transição).
- *axiomático ou gerador*: Gramática Regular.
- *denotacional*: Expressão Regular (também pode ser considerado gerador).

Autômatos Finitos

Um autômato finito, pode ser vista como uma máquina composta basicamente por três partes:

- *Fita*: Dispositivo de entrada que contém a informação a ser processada. A fita é finita à esquerda e à direita. É dividida em células onde cada uma armazena um símbolo. Os símbolos pertencem a um alfabeto de entrada. Não é possível gravar sobre a fita. Não existe memória auxiliar. Inicialmente a palavra a ser processada, isto é, a informação de entrada ocupa toda a fita.
- *Unidade de Controle*: Reflete o estado corrente da máquina. Possui uma unidade de leitura (cabeça de leitura), que acessa uma unidade da fita de cada vez. Pode assumir um número finito e pré-definido de estados. Após cada leitura a cabeça move-se uma célula para a direita.
- *Programa ou Função de Transição*: Função que comanda as leituras e define o estado da máquina. Dependendo do estado corrente e do símbolo lido determina o novo estado do autômato. Usa-se o conceito de estado para armazenar as informações necessárias à determinação do próximo estado, uma vez que não há memória auxiliar.

Autômato Finito Determinístico

Um autômato finito determinístico (AFD), ou simplesmente autômato finito, M é uma quíntupla:

$$M = (\Sigma, Q, \delta, q_0, F)$$

onde:

Σ - Alfabeto de símbolos de entrada

Q - Conjunto finito de estados possíveis do autômato.

δ - Função programa ou função de transição δ: $Q \times \Sigma \rightarrow Q$.

q_0 - Estado inicial tal que $q0 \in Q$.

F - Conjunto de estados finais, tais que $F \subseteq Q$.

Autômato Finito Não-Determinístico (AFND)

- Não-determinismo é uma importante generalização dos AF's, essencial para a teoria da computação e para a teoria das linguagens formais.
- Qualquer AFND pode ser simulado por um autômato finito determinístico. Em AFNDs, a função programa leva de um par (estado, símbolo) a um conjunto de estados possíveis.
- Em AFNDs, a função programa leva de um par (estado, símbolo) a um conjunto de estados possíveis.
- Pode-se entender que o AFND assume simultaneamente todas as alternativas de estados possíveis $\{ p_0, p_1, ..., p_n \}$ a partir do estado atual ($q \in Q$) e do símbolo recebido ($a \in \Sigma$), como se houvesse uma unidade de controle para processar cada alternativa independentemente, sem compartilhar recursos com as demais.
- Assim o processamento de um caminho não influi no estado, símbolo lido e posição da cabeça dos demais caminhos alternativos.

Processo de Determinização

Por definição, todo AFD é um caso especial de AFND no qual a relação de transição é uma função. Assim, a classe de linguagens reconhecidas por um AFND inclui as linguagens regulares (aquelas que são reconhecidas por AFD's). Entretanto, pode-se provar que as linguagens regulares são as únicas linguagens reconhecidas por um AFND. Para isto, basta mostrar que para qualquer AFND pode-se construir um AFD que reconhece a mesma linguagem. Um método de transformação é dado a seguir.

Algoritmo

O algoritmo de determinização tem como entrada o alfabeto da linguagem e a gramática a ser determinizada.

O algoritmo verifica se a gramatica é regular e se é deterministica ou indeterministica. Se a mesma for deterministica ou não regular, então o programa para, pois não há razão para continuar a execução do algoritmo.

Do contrário, o processo de determinização é iniciado, percorrendo todos os estados da AFND, verificando se há mais de um caminho para estados a partir de cada simbolo do alfabeto. Se houver mais de um caminho, significa que há uma indeterminização. Para resolver isso, é necessário criar um novo estado, onde as transiç oes de ambos os estados anteriores serão inseridas.

Este processo é executado até que todos os estados já visitados pelo algoritmo sejam inseridos na nova gramática determinizada. Sendo assim, é possível que alguns estados passem a não mais existir, e que muitos outros sejam criados durante o processo.

Estratégias

Optamos por fazer a leitura da gramática e não o autômato diretamente, pois é mais genérica, mas também porque é mais simples de o usuário digitar a gramática. Além disso, o usuário consegue ver os processos no qual a mesma passa para ser determinizada, que é determinizar usando o autômato finito.

Antes de determinizarmos, verificamos se a gramática é regular e se é indeterminística, pois nesses casos não é necessário ou mesmo não é possível realizar o processo de determinização com o nosso algoritmo.

Considerações Finais

Através do algoritmo desenvolvido o grupo conseguiu determinizar gramáticas regulares, além disso, conseguimos identificar se a mesma é ou não regular e indeterminística.

Devido a escolha da linguagem de programação *C* o grupo teve algumas dificuldades, tornando o trabalho mais complexo. Entretanto obtivos êxito na tarefa proposta, além de obtermos mais experiência na linguagem.