Expressões Regulares

Uma expressão regular sobre um alfabeto Σ é uma cadeia sobre $\Sigma \cup \{), (, \Phi, \cup, *\}$, tal que:

a) Φ, assim como cada símbolo de Σ é uma expressão regular;
b) Se α e β são expressões regulares, $(\alpha\beta)$ também é;
c) Se α e β são expressões regulares, $(\alpha \cup \beta)$ também é;
d) Se α é expressão regular então α^* também é;
e) Nada será expressão regular se não seguir as regras de a) até d).

Cada expressão regular representa uma linguagem, seja L uma função tal que se α é expressão regular, então $L(\alpha)$ é a linguagem representada por α. L é definida como:

1) $L(\Phi) = \Phi$, $L(a) = \{a\}$ para cada $a \in \Sigma$;
2) Se α e β são expressões regulares, $L((\alpha\beta)) = L(\alpha)L(\beta)$;
3) Se α e β são expressões regulares, $L((\alpha \cup \beta)) = L(\alpha) \cup L(\beta)$;
4) Se α é expressão regular então $L(\alpha^*) = L(\alpha)^*$;

Hierarquia de Chomsky para Linguagens Recursivamente Enumeráveis

Tipo 0 (recursivamente enumeráveis)
Tipo 1 (sensíveis ao contexto)
Tipo 2 (livres de contexto)
Tipo 3 (regulares)

Gramáticas

Dispositivos geradores de linguagens, obedecendo à hierarquia de Chomsky, definidas como: G=(N, T, P, S), onde:
N – Alfabeto de símbolos não-terminais (ou auxiliares);
T – Alfabeto de símbolos terminais;
P – Conjunto de regras de produção, cada regra é representada da forma $\alpha \rightarrow \beta$, em que se aplicará uma substituição dos símbolos em α pelos símbolos de β, ao se efetuar uma derivação (α possui ao menos um símbolo não-terminal);
S – Símbolo inicial (a partir do qual derivam-se as cadeias da linguagem);

Derivação

Processo através do qual as cadeias de uma linguagem são geradas. Representa-se através do símbolo $\Rightarrow$, que representa uma substituição em uma forma sentencial (usando alguma regra $\alpha \rightarrow \beta$). A linguagem gerada por G é representada por L(G), e, tem-se:
$L(G) = \{w \mid S \Rightarrow^* w \in T^*\}$

Hierarquia de Chomsky

Gramáticas Irrestritas (Tipo 0) $\alpha \in (N \cup T)^*N(N \cup T)^*$, $\beta \in (N \cup T)^*$
Gramáticas Sensíveis ao Contexto (Tipo 1), restrição: $|\alpha| \leq |\beta|$
Gramáticas Livres de Contexto (Tipo 2), restrição: $\alpha \in N$
Gramáticas Regulares (Tipo 3), restrição:
$\alpha \in N$, e $\beta = xB$, $\beta = x$, ou
$\alpha \in N$, e $\beta = Bx$, $\beta = x$,
onde $x \in T^*$, $B \in N$.

Teorema: Toda linguagem sensível ao contexto é recursiva.

Demonstra-se através de um algoritmo de reconhecimento de linguagens sensíveis ao contexto.

Autômato Finito Determinístico

Um autômato finito é um modelo computacional composto de uma fita de entrada dividida em células, nas quais estarão os símbolos das cadeias (cada símbolo em uma célula). A principal parte do modelo é um controle finito, o qual indica em qual estado, dentre um número finito de estados distintos, o autômato estará. Há ainda um cabeçote de leitura, que é capaz de capturar o símbolo presente na posição corrente da fita, e, ao capturá-lo, avança o cabeçote para a próxima célula da fita.

Um modelo de autômato possui um estado inicial e um conjunto de estados finais (todos pertencentes ao conjunto de estados). O estado inicial designa o início de funcionamento do modelo, dependendo do símbolo presente na fita o próximo estado será escolhido, enquanto que os estados finais indicam a finalização do processo computacional com sucesso, isto é, se a cadeia presente na fita de entrada tiver sido completamente lida e o modelo não se encontrar em um estado final, considera-se a cadeia como não aceita, e, em caso contrário, como aceita.

A **linguagem aceita** por um modelo de autômato finito é composta pelas cadeias aceitas por este (isto é, aquelas cujo processo computacional termina em um estado final).
M=(K, Σ, δ, s, F), onde, K é conjunto de estados do autômato, Σ é o alfabeto aceito, s é o estado inicial (que é único), F é conjunto de estados finais ($F \subseteq K$), e δ é a função de transferência, $\delta: K \times \Sigma \rightarrow K$. A função de transferência determina o próximo estado do autômato, a partir do estado corrente e do símbolo encontrado na fita de entrada.

Como o modelo de autômato é determinístico, δ é uma função, a cada par estado e símbolo há apenas um único estado destino.

Configuração

Representa o estado do controle finito, da fita de entrada e do cabeçote de leitura, em momentos sucessivos, isto é, permite acompanhar a evolução do processo computacional. Uma configuração é um elemento do conjunto $K \times \Sigma^*$. Como é possível acompanhar a evolução do processo a cada momento, define-se sobre o conjunto das configurações a relação $\vdash_M$, que indica um par de configurações sucessivas durante o processo computacional, isto é, indica um passo computacional (um único movimento do modelo de autômato sobre o conjunto de configurações). O fecho transitivo e reflexivo da relação de passo é denotado por $\vdash_M^*$.

Usando a definição anterior pode-se definir a linguagem aceita por um autômato finito M=(K, Σ, δ, s, F) como: $L(M) = \{w \in \Sigma^* \mid (s, w) \vdash_M^* (q, \varepsilon), q \in F\}$, isto é, todas as cadeias que M aceite.

A representação pode ser feita através de um grafo orientado com as indicações dos estados finais e do estado inicial, como também pode ser através da descrição dos componentes da quíntupla, com uma tabela representando a função δ.

Autômato Finito Não-Determinístico

Indica a possibilidade de mudar de estado de uma forma que é apenas parcialmente determinada através do estado corrente e do símbolo capturado na fita de entrada. Isto indica que pode haver várias possibilidades para o próximo estado, isto é, a função de transferência pode retornar um conjunto de estados. Desta maneira o autômato pode "escolher", a cada passo computacional, qual o estado seguinte (dentro do conjunto retornado pela função), como a escolha não é determinada por qualquer parte do modelo, esta é dita não-determinística.

M=(K, Σ, Δ, s, F), onde a única distinção é que a função de transição torna-se neste modelo uma relação de transição, $\Delta \subseteq K \times \Sigma^* \times K$. Isto indica a possibilidade de aceitar uma cadeia a cada transição, assim, (q, u, p) denota uma transição do estado q para o estado p consumindo cadeia u.

A seqüência de configurações é, como antes, uma relação sobre $K \times \Sigma^*$. $(q, w) \vdash_M (p, v) \Leftrightarrow u \in \Sigma^* \mid w=uv$, $(q, u, p) \in \Delta$.
A linguagem aceita por M é o conjunto das cadeias aceitas por M, $L(M)=\{w \in \Sigma^* \mid (s,w) \vdash_M^* (q,\varepsilon), q \in F\}$.

Equivalência entre autômato finito determinístico e autômato finito não-determinístico

Teorema: Para cada modelo de autômato não-determinístico há um modelo de autômato determinístico equivalente.

Demonstra-se transformando o modelo não-determinístico em outro (primeiro eliminando-se as transições sobre cadeias, depois as transições vazias calculando-se os estados alcançáveis, e depois criando um modelo determinístico a partir dos conjuntos de estados gerados), e, a seguir, usando-se o princípio da indução sobre o tamanho da cadeia aceita.

Minimização de autômatos finitos determinísticos

Processo pelo qual se busca encontrar um outro modelo de autômato finito determinístico que aceite a mesma linguagem, e cujo conjunto de estados tenha tamanho mínimo.

Estados indistinguíveis: Inicia-se o processo separando estados finais de não-finais como distinguíveis (não-equivalentes). Estados que consomem símbolos diferentes são distinguíveis.
Equivalência entre estados: Os conjuntos de estados indistinguíveis constituem classes de equivalência.
Classes de equivalência: As classes de equivalência indicam os novos estados do modelo (mínimo), onde cada classe representa a unificação dos estados presentes na classe em um outro.

Propriedades da classe de linguagens aceita por um autômato finito

Teorema: A classe de linguagens aceita pelos autômatos finitos é fechada sobre:

a) União
b) Concatenação
c) Estrela de Kleene
d) Complementação
e) Interseção

Demonstra-se construindo modelos de autômato que tratem cada operação.

Teorema: Há algoritmos para tratar os seguintes problemas dos autômatos finitos, dado um autômato M e uma cadeia w:

a) $w \in L(M)$?
b) $L(M) = \Phi$?
c) $L(M) = \Sigma^*$?
d) Dados dois autômatos $L(M1) \subseteq L(M2)$?
e) $L(M1) = L(M2)$?

Demonstra-se construindo modelos de autômato.

Teorema: Uma linguagem é regular se é aceita por um modelo de autômato finito.

Demonstra-se construindo modelos de autômato para cada tipo de expressão regular.

Linguagens não-regulares

Teorema (bombeamento): Seja L uma linguagem regular infinita. Então há cadeias x, y, z tais que $y \neq \varepsilon$, $xy^nz \in L$ para cada $n \geq 0$.

Demonstra-se através do princípio da casa de pombos.

Autômatos Finitos e Gramáticas Regulares

Teorema: A toda gramática há um autômato finito equivalente.

Demonstra-se através da construção de um modelo de autômato a partir dos não-terminais da gramática.

Teorema: A todo autômato finito há uma gramática regular equivalente.

Demonstra-se através da construção de um modelo de uma gramática cujos não-terminais são os estados do autômato.