Cursos:	Bacharelado em Ciência da Computação e Bacharelado em Sistemas de Informação
Disciplinas:	(1493A) Teoria da Computação e Linguagens Formais, (4623A) Teoria da Computação e Linguagens Formais e (1601A) Teoria da Computação
Professora:	Simone das Graças Domingues Prado
e-mail:	simonedp@fc.unesp.br
home-page:	wwwp.fc.unesp.br/~simonedp/discipl.htm

Apostila 01

Assunto: Fundamentação da Teoria da Computação e Linguagens Formais

Objetivos:

⇒ Introduzir conceitos fundamentais da disciplina

Conteúdo:

1. Introdução
2. Alfabetos, Cadeias e Linguagens

1. Introdução

Estudar a teoria da computação providencia conceitos e princípios que ajudam a entender a natureza geral da computação. Para estudar esses princípios básicos constroem-se modelos de computadores abstratos para resolução de pequenos, mas não fúteis, problemas.

Para modelar o hardware de um computador é introduzido o conceito de autômatos. Um autômato é uma construção que possui todas as características indispensáveis de um computador digital: entrada, saída, armazenagem temporária, tomadas de decisão. Um autômato é um reconhecedor da linguagem. Através dos reconhecedores é possível identificar se a cadeia de símbolos pertence ou não a uma linguagem.

Linguagem formal é uma abstração das características gerais de uma linguagem de programação. Assim, possui um conjunto de símbolos, regras de formação de sentenças, etc.

Estudar computação do ponto de vista teórico é sinônimo de caracterizar o que é ou não é computável. Para tanto, é preciso obter um modelo matemático que represente o que se entende por computação. Existem vários modelos. Nessa disciplina será dado enfoque nas Máquinas de Turing. Com a Máquina de Turing podem-se verificar os limites da computação e a complexidade computacional de um sistema. A Máquina de Turing também é um reconhecedor.

Entre as linguagens existe uma ordem hierárquica chamada de Hierarquia de Chomsky (Figura 1). Noah Chomsky definiu estas classes como potenciais modelos para as linguagens naturais. Nem sempre as linguagens de programação são tratadas adequadamente nessa hierarquia. Existem linguagens que não são livres de contexto, para as quais os formalismos sensíveis ao contexto são excessivos, sendo inadequados principalmente no que se refere à complexidade computacional. Adicionalmente o conhecimento das linguagens sensíveis ao contexto é relativamente limitado, o que dificulta o seu tratamento. (Menezes, 2000)

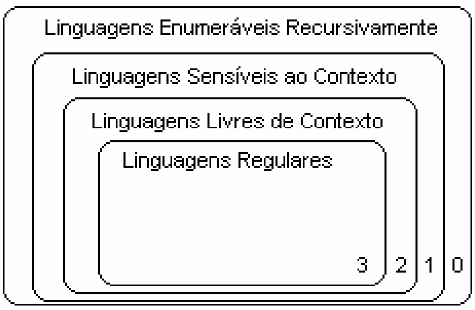

Figura 1. Hierarquia de Chomsky

A seguir tem uma tabela (Tabela 1) que traz uma correspondência entre as classes de linguagens, gramáticas e reconhecedores.

Tabela 1. Linguagem, gramática e reconhecedor

Linguagem	**Gramática**	**Reconhecedor**
Tipo 0: Ling Enumeráveis Recursivamente	Gramáticas Irrestritas	Máquinas de Turing
Tipo 1: Sensíveis ao Contexto	Gramáticas Sensíveis ao Contexto	Autômato Limitado Linearmente
Tipo 2: Ling Livres de Contexto	Gramáticas Livres de Contexto	Autômatos com Pilha
Tipo 3: Ling Regulares	Gramáticas Regulares	Autômatos Finitos

2. Alfabeto, Cadeias e Linguagens

Estamos acostumados com a noção de linguagens naturais como o Português, Inglês, etc. Uma definição informal de uma linguagem poderia ser dada como sendo um sistema capaz de expressar idéias, fatos, conceitos e que inclui um conjunto de símbolos e regras para sua manipulação. Ou ainda, defini-se linguagem como o uso da palavra articulada ou escrita como meio de expressão e comunicação entre pessoas. As definições são muitas, mas uma definição formal deve ser introduzida.

Definição 1.
Um **alfabeto** é um conjunto finito de símbolos.

Um símbolo de um alfabeto é uma entidade abstrata básica, podendo representar números, letras, desenhos etc, por exemplo: 1, 0, a, b, c.

Um alfabeto é representado por: Σ (lê-se sigma)
Assim são alfabetos:

$\Sigma = \{\}$
$\Sigma = \{0, 1\}$
$\Sigma = \{a, b\}$

Definição 2.
Uma **cadeia de caracteres** sobre um alfabeto á uma seqüência finita de símbolos (do alfabeto) justapostos.

Seja o alfabeto $\Sigma = \{a, b\}$
Então as cadeias (w) possíveis são:

$w = \lambda$ representa a cadeia vazia, ou seja, não possui símbolos
$w = a$
$w = b$
$w = ab$
$w = ba$
$w = aab$

Definição 3.

Uma **linguagem** é um conjunto de palavras sobre um alfabeto.

Se Σ representa um alfabeto,

então Σ^* representa o conjunto de todas as palavras possíveis sobre Σ.

e Σ^+ representa o conjunto de todas as palavras excetuando a palavra vazia.

Assim pode-se dizer que

$\Sigma^* = \{ \lambda, a, b, ab, ba, aab, abb, aaaa, ...\}$

$\Sigma^+ = \Sigma^* - \{\lambda\} = \{a, b, ab, ba, aab, abb, aaaa, ...\}$

Uma linguagem (L) é geralmente definida como um subconjunto de Σ^*.

Uma sentença é geralmente definida como uma cadeia pertencente à linguagem L.

Definição 4.

O **tamanho ou comprimento** de uma cadeia w, representado por $|w|$, é o número de símbolos que compõem a cadeia.

Por exemplo:

$w = \lambda$	$\|w\| = 0$
$w = a$	$\|w\| = 1$
$w = ab$	$\|w\| = 2$
$w = abb$	$\|w\| = 3$
$w = bbaa$	$\|w\| = 4$

Definição 5.

Um **prefixo** de uma cadeia é qualquer seqüência de símbolos iniciais da cadeia.

Um **sufixo** de uma cadeia é qualquer seqüência de símbolos do final da cadeia.

Uma **subcadeia** de uma cadeia é qualquer seqüência contígua de símbolos da cadeia.

Por exemplo:

Seja w = abcde, então os prefixos podem ser: λ, a, ab, abc, abcd, abcde

os sufixos podem ser: λ, e, de, cde, bcde, abcde

as subcadeias são quaisquer cadeias apresentadas como prefixos e sufixos, além de outras como bc, bcd, cd etc.

Definição 6.

O **reverso** de uma cadeia é obtido pela ordem inversa dos símbolos da cadeia.

Se $w = a_1a_2...a_n$

Então o reverso, $w^R = a_{n...}a_2a_1$

Por exemplo:

Seja w = ababab então w^R = bababa

Seja v = cdcdcdcd então v^R = dcdcdcdc

Definição 7.

A **concatenação** de duas cadeias w e v é a cadeia obtida pela adição dos símbolos de v no final da cadeia w. Nenhum símbolo pode ser mudado de lugar no momento da concatenação.

Se $w = a_1a_2...a_n$ e $v = b_1b_2....b_m$

Então a concatenação, $w \text{ o } v = a_1a_2...a_n\, b_1b_2....b_m$

Por exemplo:

Seja w = ababab e v = cdcdcdcd

Então w o v = abababcdcdcdcd

A operação de concatenação satisfaz as seguintes propriedades:

(a) associatividade
(b) elemento neutro à direita e à esquerda

Ou seja, se w, v, e t são cadeias

Pela associatividade, tem-se que:	v o (w o t) = (v o w) o t
Pelo elemento neutro à direita, tem-se que:	$v \text{ o } \lambda = v$
Pelo elemento neutro à esquerda, tem-se que:	$\lambda \text{ o } v = v$

Obs: Uma operação de concatenação definida sobre a linguagem L não é, necessariamente, fechada sobre L, ou seja, a concatenação de duas cadeias de L não é, necessariamente, uma cadeia de L.

Obs: A operação de concatenação também vale para as linguagens:

$$L_1 \text{ o } L_2 = \{ x \text{ o } y \mid x \in L_1 \text{ e } y \in L_2\}$$

Exemplo:

$L_1 = \{ a, bc \}$ e $L_2 = \{ aa, cb, bb \}$

$L_1 \text{ o } L_2 = \{ x \text{ o } y \mid x \in L_1 \text{ e } y \in L_2\} = \{$ a o aa, a o cb, a o bb, bc o aa, bc o cb, bc o bb $\}$

$L_1 \text{ o } L_2 = \{$aaa, acb, abb, bcaa, bccb, bcbb$\}$

Definição 8.

A **concatenação sucessiva** de uma cadeia w, representada por w^n, onde n é o número de concatenações sucessivas, é definida a partir da concatenação binária, como segue:

(a) para $w \neq \lambda$

$w^0 = \lambda$

$w^n = w \text{ o } w^{n-1}$, para n> 0

(b) para $w = \lambda$

w^0 = indefinida

$w^n = \lambda$, para n> 0

Exemplo:

Seja w uma cadeia (w = ab) e a um símbolo

w^3 = w o w o w = (ab) o (ab) o (ab) = ababab

w^1 = w = ab

a^5 = aaaaa

a^2 = aa

Obs: O mesmo vale para as linguagens.

Seja L = { 0, 11, 10}

$L^0 = \{\lambda\}$

$L^1 = L \text{ o } L^0 = \{ 0, 11, 10\} \text{ o } \{\lambda \} = \{ 0, 11, 10\}$

$L^2 = L \text{ o } L^1 = \{ 0, 11, 10\} \text{ o } \{ 0, 11, 10\} = \{ 00, 011, 010, 110, 1111, 1110, 100, 1011, 1010\}$

Definição 09.

Defini-se, para uma linguagem L qualquer, o seu **fechamento**, L*, como sendo:

$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup \ldots$$

Uma outra notação usada é $L^+ = \bigcup_{i=1}^{\infty} L^i = L^1 \cup L^2 \cup \ldots$

Por exemplo:

Seja a linguagem L = {0, 11}

$L^* = \{ \lambda, 0, 11, 00, 011, 110, 1111, 000, 0011, 0110, ... \}$

$L^+ = \{0, 11, 00, 011, 110, 1111, 000, 0011, 0110, ... \}$

Definição 10.

Desde que uma linguagem é definida como um conjunto (Definição 3), as operações de **união**, **intersecção** e **diferença** podem ser aplicadas.

União: $L_1 \cup L_2 = \{w: w \in L_1 \text{ ou } w \in L_2\}$

Intersecção: $L_1 \cap L_2 = \{w: w \in L_1 \text{ e } w \in L_2\}$

Diferença : $L_1 - L_2 = \{w: w \in L_1 \text{ e } w \notin L_2\}$

Por exemplo:

Sejam L_1 = {a, b, aa, ab, abb, aab, aaa} e L_2 = { a, aa, aaa, aaaa }, então

União: $L_1 \cup L_2$ = {a, b, aa, ab, abb, aab, aaa, aaaa }

Intersecção: $L_1 \cap L_2$ = { a, aa, aaa }

Diferença : $L_1 - L_2$ = { b, ab, abb, aab}

Definição 11.

O **complemento** de um conjunto consiste em todos os elementos que não pertencem a esse conjunto.

Complemento: $\overline{L} = \{w: w \in \Sigma^*, \ w \notin L\} = \Sigma^* - L$

Por exemplo:

Seja $\Sigma = \{a,b\}$.

Seja $L = \{a^n: n >= 0 \}$, então L = { λ, a, aa, aaa, aaaa, ...}

Como $\overline{L} = \{w: w \in \Sigma^*, \ w \notin L\}$, então $\overline{L}$ = {b, ab, aab, abb, ...}