

Universidade Federal do Pará
Instituto de Ciências Exatas e Naturais
Programa de Pós-Graduação em Ciência da Computação

Casamento de Cadeias

Nelson Cruz Sampaio Neto
nelsonneto@ufpa.br

12 de junho de 2019

Definição

- Cadeia é uma sequência de elementos denominados caracteres.
- Os caracteres devem ser escolhidos em um conjunto denominado alfabeto.
Ex: Em uma cadeia de *bits*, o alfabeto é $\{0,1\}$.
- **Problema do casamento de cadeias:** encontrar todas as ocorrências de um padrão em um texto.
- Exemplos de aplicação: edição de texto; classificação de documentos; estudo de sequências de DNA; etc.

Notação

- Texto: é uma cadeia T $[0..n - 1]$ de tamanho n .
- Padrão: é uma cadeia P $[0..m - 1]$ de tamanho $m \leq n$.
- Os elementos de P e T são escolhidos de um alfabeto finito Σ de tamanho c .
Ex: $\Sigma = \{0,1\}$ ou $\Sigma = \{a, b, c, \dots, z\}$.
- Casamento de cadeias ou padrão: dadas duas cadeias, $|P| = m$ e $|T| = n$, onde $n \gg m$, deseja-se saber todas as ocorrências de P em T .

Casamento Exato

- Consiste em recuperar todas as ocorrências **exatas** do padrão no texto.
- Ocorrência exata do padrão **teste** no texto abaixo:
“os **testes** testam estes alunos”
- **Ideia:** Se o primeiro caractere do padrão é idêntico à um referente no texto, todos os sucessores devem ser idênticos também, até finalizar o padrão.

Categorias de Algoritmos

- Padrão e Texto não são pré-processados:
 - Algoritmos sequenciais para aplicações em tempo real.
 - Padrão e texto não são conhecidos *a priori*.
 - Complexidade no tempo $O(mn)$ e de espaço $O(1)$.
 - Exemplo: Algoritmo força bruta.
- Padrão é pré-processado:
 - Algoritmos sequenciais.
 - O padrão é conhecido *a priori*, permitindo seu pré-processamento. Usados tipicamente pelos programas de edição de texto.
 - Complexidade no tempo $O(n)$ e de espaço $O(m + c)$.
 - Exemplo: Knuth-Morris-Pratt (KMP), Shift-And e Boyer-Moore.

Categorias de Algoritmos

- Padrão e Texto são pré-processados:
 - Algoritmos que constroem um índice.
 - Padrão e texto são conhecidos *a priori*.
 - Complexidade no tempo depende da estrutura de dados empregada e de espaço $O(n)$.
 - É viável construir um índice quando pretende-se realizar muitas operações de pesquisa em uma base de dados grande e semi-estática.
 - Esse é o caso de bancos de dados constituídos de textos em linguagem natural, como máquinas de busca na *web* e bibliotecas digitais.

Algoritmos que constroem um
índice com texto e padrão
conhecidos

Arquivo Invertido

- Na categoria onde P e T são pré-processados, o tipo de índice mais conhecido é o **arquivo invertido**.
- Um arquivo invertido possui duas partes: vocabulário e ocorrências.
- O **vocabulário** é o conjunto de todas as palavras distintas do texto.
- Para cada palavra distinta, uma lista de posições onde ela ocorre no texto é armazenada.
- O conjunto dessas listas é chamado de **ocorrências**.

Exemplo

- Exemplo de texto em linguagem natural:

Texto exemplo. Texto tem palavras. Palavras exercem fascínio.

0 6 15 21 25 35 44 52

exemplo 6

exercem 44

fascínio 52

palavras 25 35

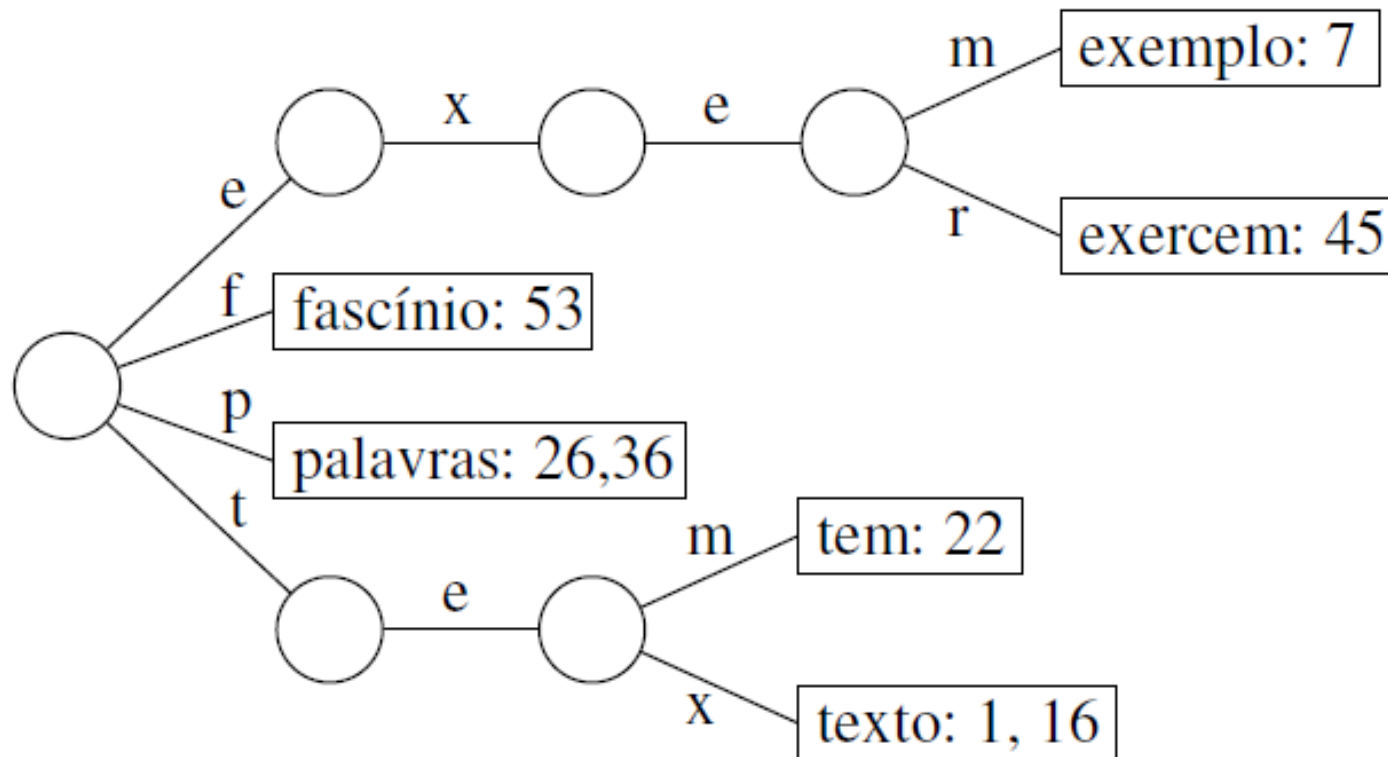
tem 21

texto 0 15

Arquivo Invertido

- A pesquisa tem geralmente três passos:
 - Pesquisa no vocabulário: palavras e padrões presentes na consulta são isolados e pesquisados no vocabulário.
 - Recuperação das ocorrências: as listas de ocorrências das palavras encontradas no vocabulário são recuperadas.
 - Manipulação das ocorrências: as listas de ocorrências são processadas para resolver frases, proximidade, etc.
- A pesquisa por palavras simples pode ser realizada usando qualquer estrutura de dados que a torne eficiente, como Tabela Hash, Árvore Trie ou Árvore B.

Arquivo Invertido usando Trie



Arquivo Invertido

- Já a pesquisa por frases usando índices é mais difícil de ser realizada.
- Cada elemento da frase é pesquisado separadamente e suas listas de ocorrências recuperadas.
- Em seguida, as listas têm de ser percorridas de forma sincronizada para encontrar as posições nas quais todas as palavras aparecem em sequência.

Algoritmo sequencial com texto e
padrão desconhecidos

Algoritmo Força Bruta

- O algoritmo força bruta é a mais simples estratégia de projeto: “somente faça”.
- Nenhuma fase de pré-processamento.
- O algoritmo consiste em analisar em todas as posições do texto entre 0 e $n - m$ se há ocorrência do padrão.
- Após cada tentativa sem sucesso, a janela é deslocada à direita uma posição.
- Também referenciado na literatura como Algoritmo Naive.

Algoritmo Força Bruta



t = ACTGATGACGTTAC...

p = TGAC



Algoritmo Força Bruta



t = ACTGATGACGTTAC...

p = TGAC

p = TGAC



Algoritmo Força Bruta



t = ACTGATGACGTTAC...

p = TGAC

p = TGAC

p = TGAC



Algoritmo Força Bruta

t = ACTGA**TGAC**GTTAC...

p = TGAC

p = TGAC

p = TGAC

p = TGAC

p = TGAC

p = TGAC

Padrão encontrado
no texto!
match!

Algoritmo Força Bruta

- O pior caso é “muito ruim”: o algoritmo tem que comparar todos os m caracteres antes de deslocar-se e isso pode ocorrer para cada uma das $n - m$ tentativas.

```
t = x x x x x x x x x x x x x x x x x x x x x x x x x x x x  
p = x x x x x x x x y  
  p = x x x x x x x x y  
    p = x x x x x x x x y  
      p = x x x x x x x x y  
        ...
```

- Logo, o pior caso para esse algoritmo é $O(nm)$.

Algoritmos sequenciais com
padrão conhecido

Knuth-Morris-Pratt (KMP)

- O Knuth-Morris-Pratt (KMP) foi o primeiro algoritmo a apresentar complexidade de tempo linear no tamanho do texto no seu pior caso.
- Criado em 1977, tem uma implementação complicada e, na prática, perde em eficiência para os outros algoritmos que apresentaremos a seguir.
- Por isso, não vamos estudá-lo!

Algoritmo BM

- Publicado em 1977, a ideia do algoritmo Boyer-Moore (BM) é pesquisar o padrão no sentido da direita para a esquerda, o que o torna muito rápido.
- Em 1980, Horspool mostrou uma simplificação importante no algoritmo, tão eficiente quanto o original, ficando conhecida como Boyer-Moore-Horspool (BMH).
- De simples implementação e comprovada eficiência, o BMH deve ser escolhido em aplicações de uso geral que necessitam realizar casamento exato de cadeias.

Algoritmo BM

- O algoritmo Boyer-Moore (BM) faz a varredura dos símbolos do padrão da direita para a esquerda.
- Se o padrão não foi encontrado, o algoritmo BM utiliza duas heurísticas para deslocar o padrão para a direita. São elas:
 - **Heurística ocorrência:** alinha a posição no texto que causou a colisão com o primeiro caractere no padrão que casa com ele.
 - **Heurística casamento:** ao mover o padrão para a direita, ele casa com o pedaço do texto anteriormente casado.

Algoritmo BM

- Exemplo de funcionamento da heurística ocorrência para o padrão $P = \text{cacbac}$ no texto $T = \text{aabcaccacbac}$.

a a b c a c c a c b a c

c a c b a c

c a c b a c

c a c b a c

c a c b a c

c a c b a c



Padrão
encontrado

Algoritmo BM

- Exemplo de funcionamento da heurística casamento para o padrão $P = \text{cacbac}$ no texto $T = \text{aabcaccacbac}$.

a a b c a c c a c b a c

c a c b a c

c a c b a c

c a c b a c



Padrão
encontrado

Algoritmo BM

- O algoritmo BM escolhe a heurística que provoca o maior deslocamento do padrão.
- Essa escolha implica em realizar uma comparação entre dois inteiros para cada caractere lido do texto, penalizando o desempenho no tempo do algoritmo.
- Assim, várias propostas de simplificação ocorreram ao longo dos anos. As que produzem os melhores resultados são as que consideram apenas a heurística de ocorrência.

Exercícios

- Mostre os passos intermediários do algoritmo BM para obter a ocorrência do padrão “MOORE” no texto “BOYERMOORE”.
- Repita o exercício anterior para o texto “BOYER MOORE”.
- Mostre os passos intermediários do algoritmo BM para obter a ocorrência do padrão “teste” no texto “os testes testam”.

Algoritmo BMH

- A simplificação mais importante é creditada a Horspool (1980).
- É o algoritmo executado frequentemente em editores de texto para os comandos de “localizar” e “substituir”.
- **Ideia básica:** Ele parte da observação de que qualquer caractere já lido do texto a partir do último deslocamento pode ser usado para endereçar a tabela de deslocamentos. Assim, o algoritmo endereça a tabela com o caractere do texto correspondente ao último caractere do padrão.

Algoritmo BMH

- Para pré-computar o padrão, o valor inicial de todas as entradas na tabela de deslocamentos é feito igual a m .
- A seguir, apenas os $m - 1$ primeiros caracteres do padrão são usados para obter os outros valores da tabela.
- Formalmente:

$$d[x] = \min \{ j \text{ tal que } j = m \mid (1 \leq j < m \ \& \ P[m - j - 1] = x) \}$$

- Exemplo: Para o padrão $P = \text{"teste"}$, os valores de d são $d[t] = 1$, $d[e] = 3$ e $d[s] = 2$, e os outros valores são iguais a $m = 5$.

Algoritmo BMH

$\Sigma = \{a b c d e f g h\}$

$P = \{c a d e\}$

$T = \{h b a d e c a e d c a d e\}$

a	b	c	d	e	f	g	h
4	4	4	4	4	4	4	4
2	4	3	1	4	4	4	4

c	a	d	e
3	2	1	4

h b a d e c a e d c a d e

c a d e

Shift para [d] = 1

Algoritmo BMH

h b a d e c a e d c a d e

c a d e Shift para [e] = 4

h b a d e c a e d c a d e

c a d e Shift para [d] = 1

h b a d e c a e d c a d e

c a d e Shift para [c] = 3

h b a d e c a e d c a d e

c a d e Shift para [match] = 4

Algoritmo BMH

- **Análise:**

O deslocamento de ocorrência pode ser pré-computado com base apenas no padrão e no alfabeto, logo a complexidade de tempo e espaço para essa fase é $O(m + c)$.

Já para a fase de pesquisa, o pior caso do algoritmo é $O(nm)$, o melhor caso é $O(n/m)$ e o caso esperado é $O(n/m)$, se c não é pequeno e m não é muito grande.

Algoritmo BMHS

- Sunday (1990) apresentou outra simplificação importante para o algoritmo BM, ficando conhecida como BMHS.
- É uma variante do algoritmo BMH.
- **Ideia básica:** endereçar a tabela com o caractere no texto correspondente ao próximo caractere após o último caractere do padrão, em vez de deslocar o padrão usando o último caractere como no algoritmo BMH.

Algoritmo BMHS

- Para pré-computar o padrão, o valor inicial de todas as entradas na tabela de deslocamentos é feito igual a $m + 1$.
- A seguir, os m primeiros caracteres do padrão são usados para obter os outros valores da tabela.
- Formalmente:

$$d[x] = \min \{ j \text{ tal que } j = m + 1 \mid (1 \leq j \leq m \ \& \ P[m - j] = x) \}$$

- Exemplo: Para o padrão $P = \text{"teste"}$, os valores de d são $d[t] = 2$, $d[e] = 1$ e $d[s] = 3$, e os outros valores são iguais a $m + 1 = 6$.

Algoritmo BMHS

$\Sigma = \{a\ b\ c\ d\ e\ f\ g\ h\}$

$P = \{c\ a\ d\ e\}$

$T = \{h\ b\ a\ d\ e\ c\ a\ e\ d\ c\ a\ d\ e\}$

a	b	c	d	e	f	g	h
5	5	5	5	5	5	5	5
3	5	4	2	1	5	5	5

c	a	d	e
4	3	2	1

h b a d e c a e d c a d e

c a d e

Shift para [e] = 1

Algoritmo BMHS

h b a d e c a e d c a d e

c a d e

Shift para [c] = 4

h b a d e c a e d c a d e

c a d e

Shift para [c] = 4

h b a d e c a e d c a d e

match!

c a d e

Algoritmo BMHS

- Na variante BMHS, seu comportamento assintótico é igual ao do algoritmo BMH.
- Porém, os deslocamentos são mais longos (podendo ser iguais a $m + 1$), levando a saltos relativamente maiores para padrões curtos.
- Por exemplo, para um padrão de tamanho $m = 1$, o deslocamento é igual a $2m$ quando não há casamento.

Algoritmo Shift-And

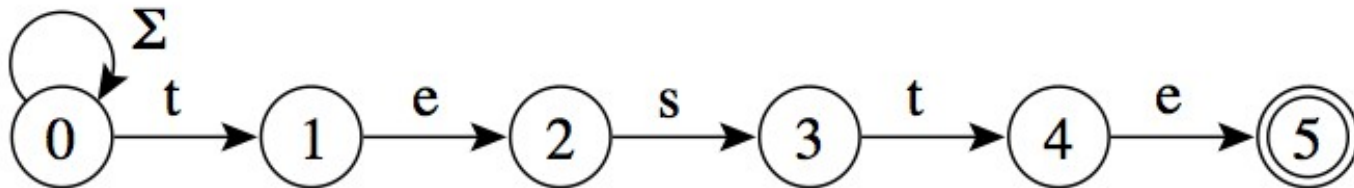
- O Shift-And foi proposto por Baeza-Yates e Gonnet (1989).
- Ele é aproximadamente duas vezes mais rápido e muito mais simples do que o algoritmo KMP; e pode ser estendido para permitir casamento aproximado de cadeia de caracteres.
- Usa o conceito de **paralelismo de *bit***:
 - Técnica que usa o paralelismo intrínseco das operações sobre *bits* dentro de uma palavra do computador a seu favor.
 - É possível empacotar muitos valores em uma única palavra e atualizar todos eles em uma única operação.

Algoritmo Shift-And

- Mantém um conjunto de todos os prefixos do padrão P que casam com o texto já lido.
- Utiliza o paralelismo de *bit* para atualizar o conjunto a cada caractere lido do texto.
- Este conjunto é representado por uma máscara de *bits*:
 $R = (b_0, b_1, \dots, b_{m-1})$.
- O algoritmo Shift-And pode ser visto como a simulação de um autômato que pesquisa pelo padrão no texto (não-determinista para simular o paralelismo de *bit*).

Algoritmo Shift-And

- Exemplo: Autômato não-determinista que reconhece todos os prefixos de $P = \{\text{teste}\}$.



Algoritmo Shift-And

- O primeiro passo é a construção de uma tabela M para armazenar uma máscara de *bits* b_0, \dots, b_{m-1} para cada caractere de P . Etapa de pré-processamento do padrão.
- Exemplo: Máscaras de *bits* para os caracteres presentes em $P = \{\text{teste}\}$. Por exemplo, a máscara em $M[t]$ é 10010, pois o caractere “t” aparece nas posições 0 e 3 de P .

	0	1	2	3	4
M[t]	1	0	0	1	0
M[e]	0	1	0	0	1
M[s]	0	0	1	0	0

Algoritmo Shift-And

- O valor do conjunto é inicializado como $R = 0^m$, *que* significa 0 repetido m vezes.
- Para cada novo caractere i lido do texto o valor do conjunto R' é atualizado:

$$R' = ((R \gg 1) \mid 10^{m-1}) \& M[T[i]]$$

Algoritmo Shift-And

- Pesquisa do padrão $P = \{\text{teste}\}$ no texto $T = \{\text{os testes}\}$.

Texto	$(R \gg 1) 10^{m-1}$					R'				
o	1	0	0	0	0	0	0	0	0	0
s	1	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0
t	1	0	0	0	0	1	0	0	0	0
e	1	1	0	0	0	0	1	0	0	0
s	1	0	1	0	0	0	0	1	0	0
t	1	0	0	1	0	1	0	0	1	0
e	1	1	0	0	1	0	1	0	0	1
s	1	0	1	0	0	0	0	1	0	0
	1	0	0	1	0	0	0	0	0	0



Casamento
exato

Algoritmo Shift-And

- **Análise:** O custo de pesquisa do algoritmo Shift-And é $O(n)$, dado pela atualização de R' a cada caractere lido do texto.
- Isso considerando que as operações para o cálculo de R' possam ser realizadas em $O(1)$ e o padrão caiba em umas poucas palavras do computador.

Casamento Aproximado

- **Definição:** É o problema de encontrar um padrão P em um texto T quando um número limitado k de operações (erros) de inserção, de substituição, ou de retirada é permitido entre P e suas ocorrências em T .
- Abaixo três ocorrências do padrão “teste” em que os casos de inserção, substituição e retirada acontecem:

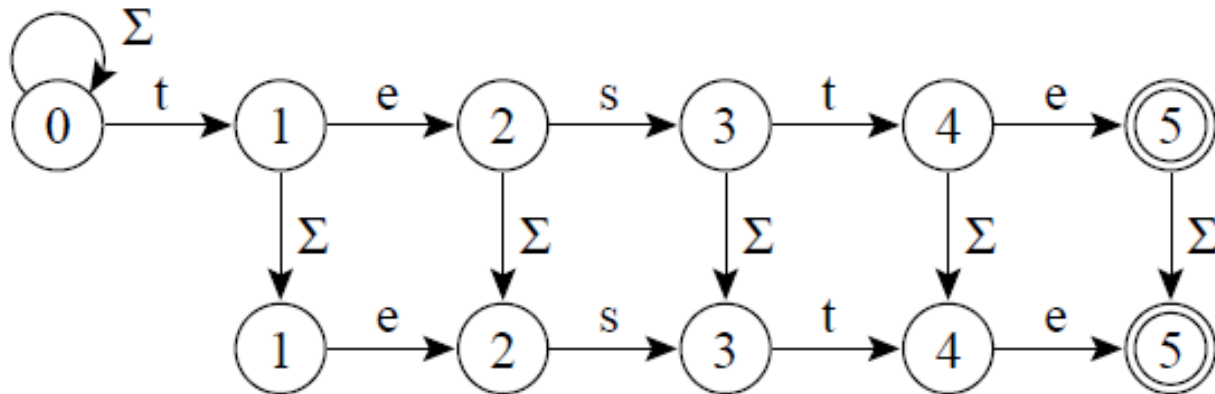
```
tes te
      testa
                este
os testes testam estes alunos . . .
```

Shift-And Aproximado

- Os melhores algoritmos para casamento aproximado de cadeias utilizam **paralelismo de *bit***.
- O algoritmo Shift-And Aproximado foi proposto por Wu e Manber (1992) e é uma extensão do Shift-And original.
- A busca com casamento aproximado também pode ser modelada por um autômato não-determinista.
- O custo da simulação do autômato é $O(kn)$ para padrões típicos na pesquisa em textos.

Shift-And Aproximado

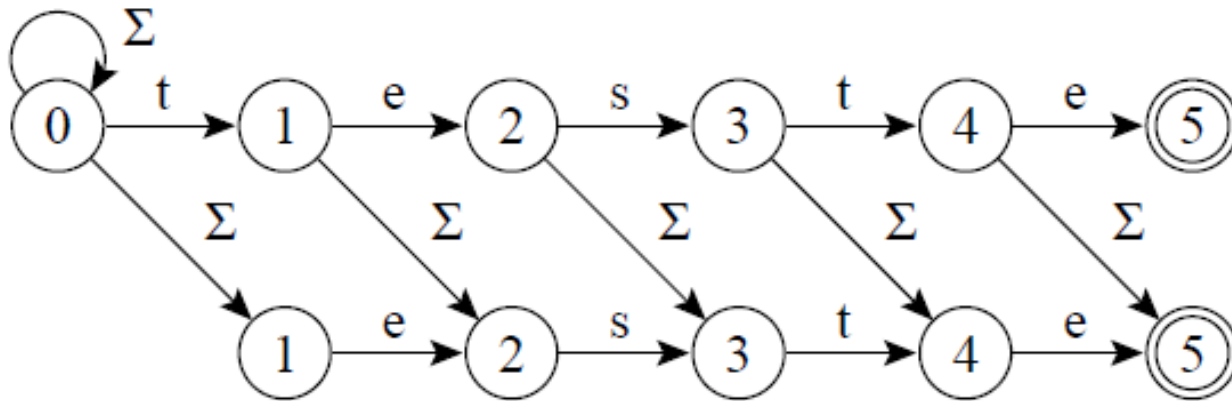
- Autômato que reconhece o padrão $P = \text{“teste”}$, permitindo uma **inserção**:



- Uma aresta horizontal representa um casamento de caractere, avançando-se no texto e no padrão.
- Uma aresta vertical insere um caractere no padrão, avançando-se no texto mas não no padrão.

Shift-And Aproximado

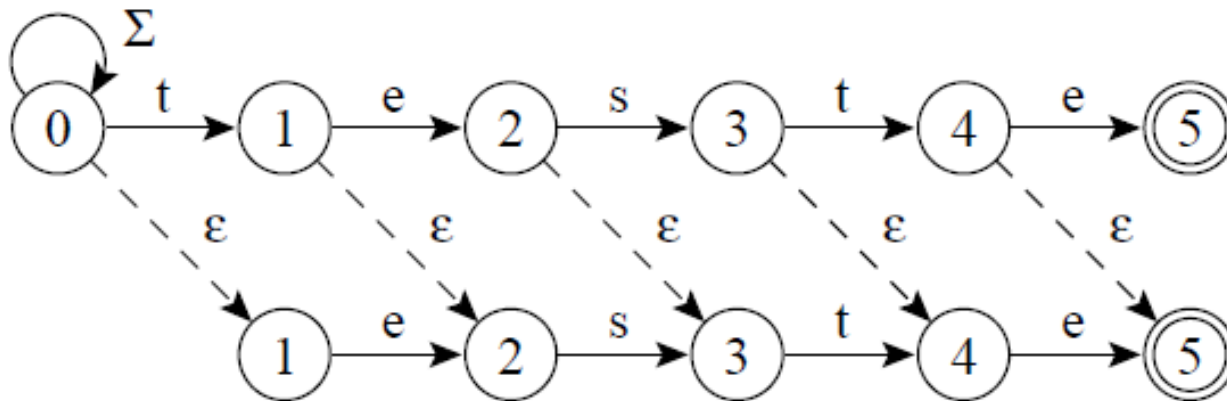
- Autômato que reconhece o padrão $P = \text{"teste"}$, permitindo uma **substituição**:



- Uma aresta horizontal representa um casamento de caractere, avançando-se no texto e no padrão.
- Uma aresta diagonal cheia substitui um caractere no padrão, avançando-se no texto e no padrão.

Shift-And Aproximado

- Autômato que reconhece o padrão $P = \text{"teste"}$, permitindo uma **retirada**:



- Uma aresta horizontal representa um casamento de caractere, avançando-se no texto e no padrão.
- Uma aresta diagonal tracejada retira um caractere no padrão, avançando-se no padrão mas não no texto.


Shift-And Aproximado


- A pesquisa inicia com $R_j = 1^j 0^{m-j}$ e para cada novo caractere i lido do texto, o valor do conjunto R'_j é atualizado de acordo com as fórmulas:

$$R'_0 = ((R_0 \gg 1) \mid 10^{m-1}) \& M[T[i]] \quad \text{e}$$

$$R'_j = ((R_j \gg 1) \& M[T[i]]) \mid R_{j-1} \mid (R_{j-1} \gg 1) \mid (R'_{j-1} \gg 1) \mid 10^{m-1}$$


inserção


substituição


retirada

Exemplo

- Padrão: “teste”.

Texto: “os testes testam”.

Permitindo um erro ($k = 1$) de inserção.

$$R'_0 = ((R_0 \gg 1) \mid 10^{m-1}) \& M[T[i]] \text{ e}$$

$$R'_1 = ((R_1 \gg 1) \& M[T[i]]) \mid R_0 \mid 10^{m-1}$$

- Uma ocorrência exata na posição 7 (“e”) e duas inserções, nas posições 8 e 11 (“s” e “e”, respectivamente).

Texto	$(R_0 \gg 1) 10^{m-1}$	R'_0	$R_1 \gg 1$	R'_1
o	1 0 0 0 0	0 0 0 0 0	0 1 0 0 0	1 0 0 0 0
s	1 0 0 0 0	0 0 0 0 0	0 1 0 0 0	1 0 0 0 0
	1 0 0 0 0	0 0 0 0 0	0 1 0 0 0	1 0 0 0 0
t	1 0 0 0 0	1 0 0 0 0	0 1 0 0 0	1 0 0 0 0
e	1 1 0 0 0	0 1 0 0 0	0 1 0 0 0	1 1 0 0 0
s	1 0 1 0 0	0 0 1 0 0	0 1 1 0 0	1 1 1 0 0
t	1 0 0 1 0	1 0 0 1 0	0 1 1 1 0	1 0 1 1 0
e	1 1 0 0 1	0 1 0 0 1	0 1 0 1 1	1 1 0 1 1
s	1 0 1 0 0	0 0 1 0 0	0 1 1 0 1	1 1 1 0 1
	1 0 0 0 0	0 0 0 0 0	0 1 1 1 0	1 0 1 0 0
t	1 0 0 0 0	1 0 0 0 0	0 1 0 1 0	1 0 0 1 0
e	1 1 0 0 0	0 1 0 0 0	0 1 0 0 1	1 1 0 0 1
s	1 0 1 0 0	0 0 1 0 0	0 1 1 0 0	1 1 1 0 0
t	1 0 0 1 0	1 0 0 1 0	0 1 1 1 0	1 0 1 1 0
a	1 1 0 0 1	0 0 0 0 0	0 1 0 1 1	1 0 0 1 0
m	1 0 0 0 0	0 0 0 0 0	0 1 0 0 1	1 0 0 0 0

Exemplo

- Padrão: “teste”.

Texto: “os testes testam”. Permitindo um erro ($k = 1$) de inserção, um de retirada, ou um de substituição.

$$R'_0 = ((R_0 \gg 1) \mid 10^{m-1}) \& M[T[i]] \text{ e}$$

$$R'_1 = ((R_1 \gg 1) \& M[T[i]]) \mid R_0 \mid (R'_0 \gg 1) \mid (R_0 \gg 1) \mid 10^{m-1}$$

- Uma ocorrência exata na posição 7 (“e”) e cinco, permitindo um erro, nas posições 6, 8, 11, 13 e 14 (“t”, “s”, “e”, “t” e “a”, respectivamente).

Texto	$(R_0 \gg 1) 10^{m-1}$	R'_0	$R_1 \gg 1$	R'_1
o	1 0 0 0 0	0 0 0 0 0	0 1 0 0 0	1 0 0 0 0
s	1 0 0 0 0	0 0 0 0 0	0 1 0 0 0	1 0 0 0 0
	1 0 0 0 0	0 0 0 0 0	0 1 0 0 0	1 0 0 0 0
t	1 0 0 0 0	1 0 0 0 0	0 1 0 0 0	1 1 0 0 0
e	1 1 0 0 0	0 1 0 0 0	0 1 1 0 0	1 1 1 0 0
s	1 0 1 0 0	0 0 1 0 0	0 1 1 1 0	1 1 1 1 0
t	1 0 0 1 0	1 0 0 1 0	0 1 1 1 1	1 1 1 1 1
e	1 1 0 0 1	0 1 0 0 1	0 1 1 1 1	1 1 1 1 1
s	1 0 1 0 0	0 0 1 0 0	0 1 1 1 1	1 1 1 1 1
	1 0 0 0 0	0 0 0 0 0	0 1 1 1 1	1 0 1 1 0
t	1 0 0 0 0	1 0 0 0 0	0 1 0 1 1	1 1 0 1 0
e	1 1 0 0 0	0 1 0 0 0	0 1 1 0 1	1 1 1 0 1
s	1 0 1 0 0	0 0 1 0 0	0 1 1 1 0	1 1 1 1 0
t	1 0 0 1 0	1 0 0 1 0	0 1 1 1 1	1 1 1 1 1
a	1 1 0 0 1	0 0 0 0 0	0 1 1 1 1	1 1 0 1 1
m	1 0 0 0 0	0 0 0 0 0	0 1 1 0 1	1 0 0 0 0

Exercícios

1) Sobre o algoritmo Boyer-Moore.

a) Preencha a tabela de deslocamento $d[]$ dos algoritmos BMH e BMHS para o padrão MOORE, para um texto contendo o vocabulário $\Sigma = \{B, E, M, O, R, Y\}$.

b) Mostre os passos intermediários para obter a ocorrência do padrão MOORE no texto BOYERMOORE para os algoritmos BMH e BMHS.

Exercícios

2) Sobre o algoritmo Shift-And para casamento exato ou aproximado de padrões.

a) Desenhe um autômato de busca que reconhece o padrão MOORE permitindo uma inserção ou uma retirada.

b) Mostre os passos intermediários para obter o casamento exato do padrão MOORE no texto MOORMOORE.

c) Apresente o funcionamento do algoritmo para pesquisar o padrão MOORE no texto MOORMOORE permitindo uma inserção ou uma retirada.

Solução – Exercício 1

Tabela de deslocamento - BMH:

$d[M]$	$d[O]$	$d[O]$	$d[R]$	$d[E]$	$d[B]$	$d[Y]$
$5 - 1 = 4$	$5 - 2 = 3$	$5 - 3 = 2$	$5 - 4 = 1$	5	5	5

Tabela de deslocamento - BMHS:

$d[M]$	$d[O]$	$d[O]$	$d[R]$	$d[E]$	$d[B]$	$d[Y]$
$5 + 1 - 1 = 5$	$5 + 1 - 2 = 4$	$5 + 1 - 3 = 3$	$5 + 1 - 4 = 2$	1	6	6

Passos intermediários - BMH:

```

1 2 3 4 5 6 7 8 9 0
M O O R E
B O Y E R M O O R E
  M O O R E
    M O O R E
    
```

Tabela de deslocamento: $d[R] = 1$, $d[M] = 4$.

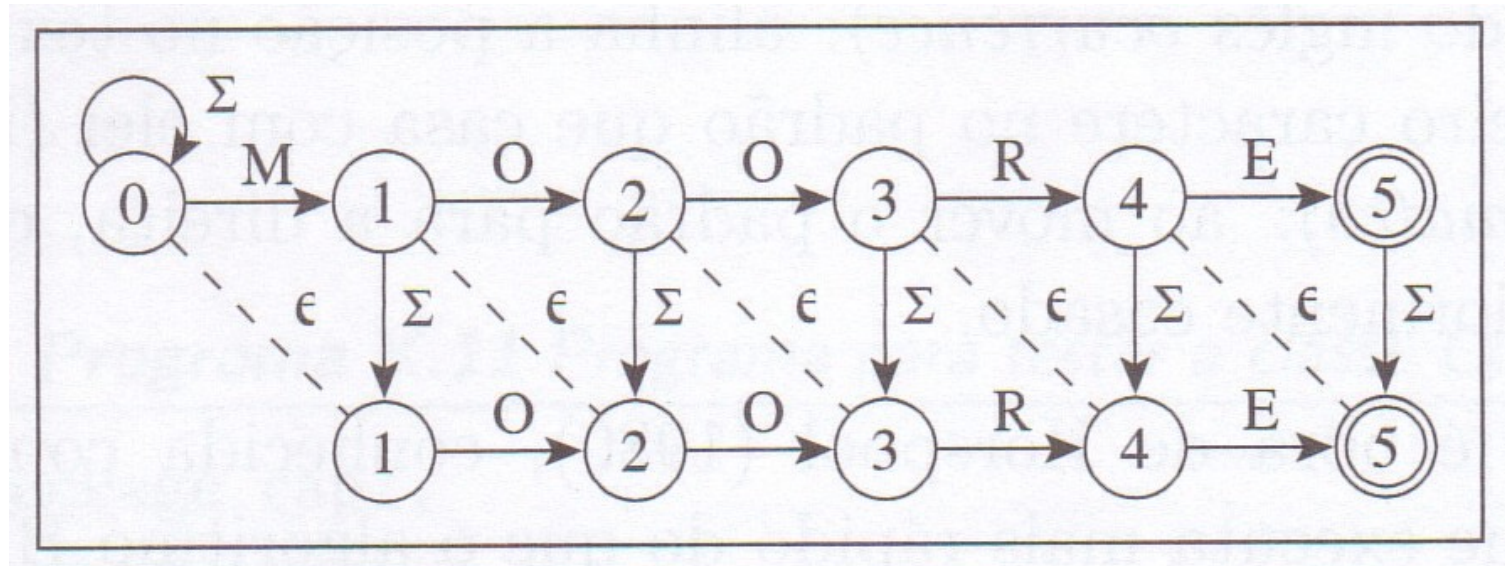
Passos intermediários - BMHS:

```

1 2 3 4 5 6 7 8 9 0
M O O R E
B O Y E R M O O R E
    M O O R E
    
```

Tabela de deslocamento: $d[M] = 5$.

Solução – Exercício 2a



Solução – Exercícios 2b e 2c

Texto	$(R_0 \gg 1) 10^{m-1}$					R'_0					$R_1 \gg 1$					R'_1				
M	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1	0	0	0
O	1	1	0	0	0	0	1	0	0	0	0	1	1	0	0	1	1	1	0	0
O	1	0	1	0	0	0	0	1	0	0	0	1	1	1	0	1	1	1	1	0
R	1	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1	0	1	1	1
M	1	0	0	0	1	1	0	0	0	0	0	1	0	1	1	1	1	0	1	0
O	1	1	0	0	0	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0
O	1	0	1	0	0	0	0	1	0	0	0	1	1	1	0	1	1	1	1	0
R	1	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1	0	1	1	1
E	1	0	0	0	1	0	0	0	0	1	0	1	0	1	1	1	0	0	1	1

$$R'_1 = R_1 \gg 1 \ \& \ M[T[i]] \mid R_0 \mid R'_0 \gg 1 \mid 10^{m-1}$$

$$R'_0 = ((R_0 \gg 1) \mid 10^{m-1}) \ \& \ M[T[i]]$$