

Mais busca de palavras

PF 13

<http://www.ime.usp.br/~pf/algoritmos/aulas/strma.html>

Busca de palavras em um texto

Problema: Dados $P[1..m]$ e $T[1..n]$, encontrar o número de ocorrências de P em T .

Exemplo: Para $n = 10$, $m = 4$, e

T

1	2	3	4	5	6	7	8	9	10
b	b	a	b	a	b	a	c	b	a

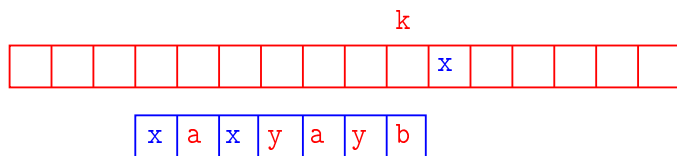
P

1	2	3	4
b	a	b	a

P ocorre 2 vezes em T.

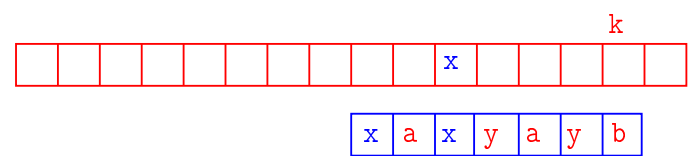
Primeiro algoritmo de Boyer-Moore

O primeiro algoritmo de R.S. Boyer e J.S. Moore (1977) é baseado na seguinte heurística.



Primeiro algoritmo de Boyer-Moore

O primeiro algoritmo de R.S. Boyer e J.S. Moore (1977) é baseado na seguinte heurística.



Boyer-Moore

P = a n d a n d o

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
 a s a n d o r i n h a s a n d a m a n d a n d o a l t o T
 1 a n d a n d o

Boyer-Moore

P = a n d a n d o

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
 a s a n d o r i n h a s a n d a m a n d a n d o a l t o T
 1 a n d a n d o
 2 a n d a n d o

Boyer-Moore

P = a n d a n d o
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
a s a n d o r i n h a s a n d a m a n d a n d o a l t o T
1 a n d a n d o
2 a n d a n d o
3 a n d a n d o

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ 🔍

Boyer-Moore

P = a n d a n d o
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
a s a n d o r i n h a s a n d a m a n d a n d o a l t o T
1 a n d a n d o
2 a n d a n d o
3 a n d a n d o
4 a n d a n d o

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ 🔍

Boyer-Moore

P = a n d a n d o
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
a s a n d o r i n h a s a n d a m a n d a n d o a l t o T
1 a n d a n d o
2 a n d a n d o
3 a n d a n d o
4 a n d a n d o
5 a n d a n d o

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ 🔍

Boyer-Moore

P = a n d a n d o
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
a s a n d o r i n h a s a n d a m a n d a n d o a l t o T
1 a n d a n d o
2 a n d a n d o
3 a n d a n d o
4 a n d a n d o
5 a n d a n d o
6 a n d a ...

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ 🔍

Boyer-Moore

P = a b a b b a b a b b a
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
a b a a b a b a b b a b a b a b b a b a b b a T
1 a b a b b a b a b b a

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ 🔍

Boyer-Moore

P = a b a b b a b a b b a
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
a b a a b a b a b b a b a b a b b a b a b b a T
1 a b a b b a b a b b a
2 a b a b b a b a b b a

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ 🔍

Boyer-Moore

P = a b a b b a b a b b a

1	a	b	a	a	b	a	b	a	b	b	a	b	a	b	a	b	b	a	b	a	b	b	a	T
1	a	b	a	b	b	a	b	a	b	b	a													
2		a	b	a	b	b	a	b	a	b	b	a												
3			a	b	a	b	b	a	b	a	b	b	a											

Navigation icons

Boyer-Moore

P = a b a b b a b a b b a

1	a	b	a	a	b	a	b	a	b	b	a	b	a	b	a	b	b	a	b	a	b	b	a	T
1	a	b	a	b	b	a	b	a	b	b	a													
2		a	b	a	b	b	a	b	a	b	b	a												
3			a	b	a	b	b	a	b	a	b	b	a											
4				a	b	a	b	b	a	b	a	b	b	a										

Navigation icons

Boyer-Moore

P = a b a b b a b a b b a

1	a	b	a	a	b	a	b	a	b	b	a	b	a	b	a	b	b	a	b	a	b	b	a	T
1	a	b	a	b	b	a	b	a	b	b	a													
2		a	b	a	b	b	a	b	a	b	b	a												
3			a	b	a	b	b	a	b	a	b	b	a											
4				a	b	a	b	b	a	b	a	b	b	a										
5					a	b	a	b	b	a	b	a	b	b	a									

Navigation icons

Boyer-Moore

P = a b a b b a b a b b a

1	a	b	a	a	b	a	b	a	b	b	a	b	a	b	a	b	b	a	b	a	b	b	a	T
1	a	b	a	b	b	a	b	a	b	b	a													
2		a	b	a	b	b	a	b	a	b	b	a												
3			a	b	a	b	b	a	b	a	b	b	a											
4				a	b	a	b	b	a	b	a	b	b	a										
5					a	b	a	b	b	a	b	a	b	b	a									
6						a	b	a	b	b	a	b	a	b	b	a								

Navigation icons

Boyer-Moore

P = a b a b b a b a b b a

1	a	b	a	a	b	a	b	a	b	b	a	b	a	b	a	b	b	a	b	a	b	b	a	T
1	a	b	a	b	b	a	b	a	b	b	a													
2		a	b	a	b	b	a	b	a	b	b	a												
3			a	b	a	b	b	a	b	a	b	b	a											
4				a	b	a	b	b	a	b	a	b	b	a										
5					a	b	a	b	b	a	b	a	b	b	a									
6						a	b	a	b	b	a	b	a	b	b	a								
7							a	b	a	b	b	a	b	a	b	b	a							

Navigation icons

Boyer-Moore

P = a b a b b a b a b b a

1	a	b	a	a	b	a	b	a	b	b	a	b	a	b	a	b	b	a	b	a	b	b	a	T
1	a	b	a	b	b	a	b	a	b	b	a													
2		a	b	a	b	b	a	b	a	b	b	a												
3			a	b	a	b	b	a	b	a	b	b	a											
4				a	b	a	b	b	a	b	a	b	b	a										
5					a	b	a	b	b	a	b	a	b	b	a									
6						a	b	a	b	b	a	b	a	b	b	a								
7							a	b	a	b	b	a	b	a	b	b	a							
8								a	b	a	b	b	a	b	a	b	b	a						

Navigation icons

Primeiro algoritmo de Boyer-Moore

Ideia: calcular um deslocamento de modo que $T[k+1]$ fique emparelhado com a última ocorrência do caractere $T[k+1]$ em P .

Suponha que o conjunto a que pertencem todos os elementos de P e de T é conhecido de antemão. Este conjunto é o alfabeto do problema.

Suponha que o alfabeto é o conjunto de todos os 256 caracteres.

Navigation icons

Primeiro algoritmo de Boyer-Moore

Recebe vetores $P[1..m]$ e $T[1..n]$ de caracteres, com $m \geq 1$ e $n \geq 0$, e devolve o número de ocorrências de P em T .

```
int BoyerMoore (unsigned char P[], int m,
               unsigned char T[], int n) {
    int ult[256];
    int i, r, k, ocorre;

    /* pre-processamento da palavra P */
    for (i=0; i < 256; i++) ult[i] = 0;
    for (i=1; i <= m; i++) ult[P[i]] = i;
}
```

Navigation icons

Pior caso

```
P = a a a a a a a a a a
1  a a a a a a a a a a
2  a a a a a a a a a a
3  a a a a a a a a a a
4  a a a a a a a a a a
5  a a a a a a a a a a
6  a a a a a a a a a a
7  a a a a a a a a a a
8  a a a a a a a a a a
9  a a a a a a a a a a
10 a a a a a a a a a a
11 a a a a a a a a a a
12 a a a a a a a a a a
13 a a a a a a a a a a
```

Navigation icons

Primeiro algoritmo de Boyer-Moore

Para implementar essa ideia fazemos um pré-processamento de P , determinando para cada símbolo x do alfabeto a posição de sua última ocorrência em P .

	1	2	3	4	5	6	7
P	a	n	d	a	n	d	o

ult	0	...	'a'	'b'	'c'	'd'	'n'	'o'	'p'	...	255
	0	...	4	0	0	6	5	7	0

Navigation icons

Primeiro algoritmo de Boyer-Moore

```
/* busca da palavra P no texto T */
3  ocorre = 0; k = m;
4  while (k <= n) {
5      r = 0;
6      while (r < m && P[m-r] == T[k-r])
7          r += 1;
8      if (r == m) ocorre += 1;
9      if (k == n) k += 1;
10     else k += m - ult[T[k+1]] + 1;
11 }
12 return ocorre;
13 }
```

Navigation icons

Melhor caso

```
P = a a a a b
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
? ? ? ? a c ? ? ? ? a c ? ? ? ? a c ? ? ? ? a T
1 a a a a b
```

Navigation icons

Melhor caso

P = a a a a b

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
?	?	?	?	a	c	?	?	?	?	a	c	?	?	?	?	a	c	?	?	?	?	a	T

1 a a a a b

2 a a a a b

Navigation icons

Melhor caso

P = a a a a b

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
?	?	?	?	a	c	?	?	?	?	a	c	?	?	?	?	a	c	?	?	?	?	a	T

1 a a a a b

2 a a a a b

3 a a a a b

Navigation icons

Melhor caso

P = a a a a b

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
?	?	?	?	a	c	?	?	?	?	a	c	?	?	?	?	a	c	?	?	?	?	a	T

1 a a a a b

2 a a a a b

3 a a a a b

4 a a a a b

Navigation icons

Conclusões

O consumo de tempo da função **BoyerMoore** no **pior caso** é $O((n - m + 1)m)$.

O consumo de tempo da função **BoyerMoore** no **melhor caso** é $O(n/m)$.

Isto significa que no **pior caso** o consumo de tempo é essencialmente proporcional a **mn**.

Navigation icons

Comentários finais

MACO122 Princípios de Desenvolvimento de Algoritmos

Edição 2012

Navigation icons

Livros

Nossa referência básica **foi** o livro *PF = Paulo Feofiloff, Algoritmos em linguagem C,*



Este livro é baseado no material do sítio *Projeto de Algoritmos em C.*

Outros livros **foram** *S = Robert Sedgewick, Algorithms in C, vol. 1* e *CLRS = Cormen-Leiserson-Rivest-Stein, Introductions to Algorithms*

Navigation icons

MAC0122

MAC0122 Princípios de Desenvolvimento de Algoritmos **foi** uma disciplina introdutória em:

- ▶ **projeto de algoritmos**: recursão, algoritmos de enumeração, divisão-e-conquista, pré-processamento, heurísticas.
- ▶ **correção de algoritmos**: relações invariantes.
- ▶ **eficiência de algoritmos**: consumo de tempo, notação assintótica O , análise experimental.
- ▶ **estruturas de dados**: listas lineares encadeadas, listas encadeadas ciculares, listas com e sem cabeça, filas, pilhas, heaps.

◀ ▶ ↺ ↻ 🔍

Principais tópicos

Alguns dos tópicos de **MAC0122** foram:

- ▶ recursão;
- ▶ busca em um vetor;
- ▶ busca (binária) em vetor ordenado;
- ▶ listas encadeadas;
- ▶ listas lineares: filas e pilhas;
- ▶ algoritmos de enumeração;
- ▶ busca de palavras em um texto;
- ▶ algoritmos de ordenação: bubblesort, heapsort, mergesort,...; e

Tudo isso regado a muita **análise de eficiência de algoritmos e invariantes**.

◀ ▶ ↺ ↻ 🔍

Pausa para nossos comerciais

- ▶ **EP5**: segunda-feira, 3/DEZ
- ▶ **Prova 3**: terça-feira, 4/DEZ
- ▶ **Aula do Will**: quinta-feira, 6/DEZ (**cancelar?**)
- ▶ **Linux**: vocês devem usar
- ▶ **Página do BCC**: <http://bcc.ime.usp.br/>

◀ ▶ ↺ ↻ 🔍

MAC0122

MAC0122 **combinou** conceitos e recursos de programação:

- ▶ **recursão**
- ▶ **strings**
- ▶ **endereços e ponteiros**
- ▶ **registros e structs**
- ▶ **alocação dinâmica de memória**
- ▶ **interfaces**

que nasceram de aplicações cotidianas em ciência da computação.

◀ ▶ ↺ ↻ 🔍

C

“... There is an important pedagogical issue in choosing a language for our examples. Just as **no language** solves all problems equally well, no single language is best for presenting all topics. Higher-level languages preempt some design decisions. If we use a lower-level language, we get to consider alternative answers to the questions; by exposing more of the details, we can talk about them better. Experience shows that even when we use the facilities of high-level languages, it's invaluable to know how they relate to lower-level issues; without that insight, it's easy to run into performance problems and mysterious behavior. **So we will often use C for our examples, even though in practice we might choose something else...**”

The Practice of Programming
Brian W. Kernighan e Rob Pike

◀ ▶ ↺ ↻ 🔍

Próximos anos

MAC0122 **foi** um primeiro passo para

- ▶ **MAC0323** Estruturas de Dados
- ▶ **MAC0328** Algoritmos de Grafos
- ▶ **MAC0338** Análise de Algoritmos

Entretanto, várias outras disciplinas se apoiam em **MAC0122**.

◀ ▶ ↺ ↻ 🔍