Busca de padrão

Dados

- uma palavra $P[1\,..\,m]$ e
- um texto $T[1\,..\,n]$,

uma ocorrência de P em T é um índice s tal que $T[s+j] = P[j]$ para $j = 1, \ldots, m$.

Busca de padrão

Dados

- uma palavra $P[1\,..\,m]$ e
- um texto $T[1\,..\,n]$,

uma ocorrência de P em T é um índice s tal que $T[s+j] = P[j]$ para $j = 1, \ldots, m$.

Exemplo:

	1	2	3
P	B	R	A

	1	2	3	4	5	6	7	8	9	10	11
T	A	B	R	A	C	A	D	A	B	R	A

Busca de padrão

Dados

- uma palavra $P[1\,..\,m]$ e
- um texto $T[1\,..\,n]$,

uma ocorrência de P em T é um índice s tal que $T[s+j] = P[j]$ para $j = 1, \ldots, m$.

Exemplo:

	1	2	3
P	B	R	A

	1	2	3	4	5	6	7	8	9	10	11
T	A	B	R	A	C	A	D	A	B	R	A

Problema: Dada uma palavra $P[1\,..\,m]$ e um texto $T[1\,..\,n]$, encontrar todas as ocorrências de P em T.

Busca de padrão

Dados

- uma palavra $P[1\,..\,m]$ e
- um texto $T[1\,..\,n]$,

uma ocorrência de P em T é um índice s tal que $T[s+j] = P[j]$ para $j = 1, \ldots, m$.

Exemplo:

	1	2	3
P	B	R	A

	1	2	3	4	5	6	7	8	9	10	11
T	A	B	R	A	C	A	D	A	B	R	A

Problema: Dada uma palavra $P[1\,..\,m]$ e um texto $T[1\,..\,n]$, encontrar todas as ocorrências de P em T.

No exemplo, P ocorre duas vezes em T: em 1 e em 8.

Algoritmo ingênuo

BUSCA-TRIVIAL (T, n, P, m)

para $s \leftarrow 0$ **até** $n - m$ **faça**
    $j \leftarrow 1$
    enquanto $j \leq m$ **e** $P[j] = T[s + j]$ **faça**
        $j \leftarrow j + 1$
    se $j > m$
        então imprima s

Algoritmo ingênuo

BUSCA-TRIVIAL (T, n, P, m)

para $s \leftarrow 0$ **até** $n - m$ **faça**
    $j \leftarrow 1$
    enquanto $j \leq m$ **e** $P[j] = T[s + j]$ **faça**
        $j \leftarrow j + 1$
    se $j > m$
        então imprima s

Consumo de tempo: $\mathrm{O}(mn)$

Algoritmo ingênuo

BUSCA-TRIVIAL (T, n, P, m)

```
para s ← 0 até n − m faça
    j ← 1
    enquanto j ≤ m e P[j] = T[s + j] faça
        j ← j + 1
    se j > m
        então imprima s
```

Consumo de tempo: $\Theta(mn)$

Exemplo: $T = a^n$ e $P = a^m$, ou $P = a^{m-1}b$.

Algoritmo ingênuo

BUSCA-TRIVIAL (T, n, P, m)

para $s \leftarrow 0$ **até** $n - m$ **faça**
    $j \leftarrow 1$
    enquanto $j \leq m$ **e** $P[j] = T[s + j]$ **faça**
        $j \leftarrow j + 1$
    se $j > m$
        então imprima s

Consumo de tempo: $\Theta(mn)$

Exemplo: $T = a^n$ e $P = a^m$, ou $P = a^{m-1}b$.

Vamos ver um algoritmo que consome tempo $\Theta(m + n)$.

Algoritmo KMP

KMP: Knuth, Morris e Pratt.

Algoritmo KMP

KMP: Knuth, Morris e Pratt.

P_q: prefixo de P de comprimento q

Algoritmo KMP

KMP: Knuth, Morris e Pratt.

P_q: prefixo de P de comprimento q

$P_k \sqsupseteq P_q$: P_k é sufixo de P_q

Algoritmo KMP

KMP: Knuth, Morris e Pratt.

P_q: prefixo de P de comprimento q

$P_k \sqsupseteq P_q$: P_k é sufixo de P_q

Função prefixo: $\Pi[q] \;=\; \max\{k : k < q$ e $P_k \sqsupseteq P_q\}$

Algoritmo KMP

KMP: Knuth, Morris e Pratt.

P_q: prefixo de P de comprimento q

$P_k \sqsupseteq P_q$: P_k é sufixo de P_q

Função prefixo: $\Pi[q] = \max\{k : k < q$ e $P_k \sqsupseteq P_q\}$

Em palavras, $\Pi[q]$ é o comprimento do maior prefixo próprio de P_q que é sufixo de P_q.

Algoritmo KMP

KMP: Knuth, Morris e Pratt.

P_q: prefixo de P de comprimento q

$P_k \sqsupseteq P_q$: P_k é sufixo de P_q

Função prefixo: $\Pi[q] = \max\{k : k < q$ e $P_k \sqsupseteq P_q\}$

Em palavras, $\Pi[q]$ é o comprimento do maior prefixo próprio de P_q que é sufixo de P_q.

Exemplo:

	1	2	3	4	5	6	7	8	9	10	11
P	a	b	a	b	a	a	b	a	b	c	a
Π	0	0	1	2	3	1	2	3	4	0	1

Algoritmo KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0$
$k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
    enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
        $k \leftarrow \Pi[k]$
    se $P[k+1] = P[q]$
        então $k \leftarrow k + 1$
    $\Pi[q] \leftarrow k$
devolva Π

Algoritmo KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0$
$k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
    enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
        $k \leftarrow \Pi[k]$
    se $P[k+1] = P[q]$
        então $k \leftarrow k+1$
    $\Pi[q] \leftarrow k$
devolva Π

Consumo de tempo: $\mathrm{O}(m^2)$

Algoritmo KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0$
$k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
  enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
    $k \leftarrow \Pi[k]$
  se $P[k+1] = P[q]$
    então $k \leftarrow k+1$
  $\Pi[q] \leftarrow k$
devolva Π

Consumo de tempo: $\Theta(m^2)$??

Invariantes do Algoritmo KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
  enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
    $k \leftarrow \Pi[k]$
  se $P[k+1] = P[q]$
    então $k \leftarrow k+1$
  $\Pi[q] \leftarrow k$
devolva Π

Invariantes do Algoritmo KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
    enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
        $k \leftarrow \Pi[k]$
    se $P[k+1] = P[q]$
        então $k \leftarrow k+1$
    $\Pi[q] \leftarrow k$
devolva Π

(1) $k < q - 1$

(2) $P_k \sqsupseteq P_{q-1}$

(3) k é o maior possível tal que valem (1) e (2)

(4) $k = \Pi[q-1]$

(5) $\Pi[1 \,.\,.\, q-1]$ está calculado corretamente

Consumo de tempo do KMP

CALCULA-PREFIXO (P, m)

```
1    Π[1] ← 0     k ← 0
2    para q ← 2 até m faça
3        enquanto k > 0 e P[k + 1] ≠ P[q] faça
4            k ← Π[k]
5        se P[k + 1] = P[q]
6            então k ← k + 1
7        Π[q] ← k
8    devolva Π
```

Método do potencial: Tome $\Phi_q = k$ no fim da iteração q.

Consumo de tempo do KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
    enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
        $k \leftarrow \Pi[k]$
    se $P[k+1] = P[q]$
        então $k \leftarrow k + 1$
    $\Pi[q] \leftarrow k$
devolva Π

Método do potencial: Tome $\Phi_q = k$ no fim da iteração q.

Valor inicial: $\Phi_1 = 0$

Consumo de tempo do KMP

CALCULA-PREFIXO (P, m)

$\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
para $q \leftarrow 2$ **até** m **faça**
    enquanto $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
        $k \leftarrow \Pi[k]$
    se $P[k+1] = P[q]$
        então $k \leftarrow k+1$
    $\Pi[q] \leftarrow k$
devolva Π

Método do potencial: Tome $\Phi_q = k$ no fim da iteração q.

$\Phi_1 = 0 \quad$ e $\quad \Phi_q \geq 0$ para $q = 1, \ldots, m$.

Consumo de tempo do KMP

CALCULA-PREFIXO (P, m)

1 $\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
2 **para** $q \leftarrow 2$ **até** m **faça**
3   **enquanto** $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
4         $k \leftarrow \Pi[k]$
5   **se** $P[k+1] = P[q]$
6   **então** $k \leftarrow k+1$
7     $\Pi[q] \leftarrow k$
8 **devolva** Π

Método do potencial: Tome $\Phi_q = k$ no fim da iteração q.

$\Phi_1 = 0$ e $\Phi_q \geq 0$ para $q = 1, \ldots, m$.

c_q: $1+$ número de execuções da linha 4 na iteração q.

Consumo de tempo do KMP

```
CALCULA-PREFIXO (P, m)
1    Π[1] ← 0    k ← 0
2    para q ← 2 até m faça
3        enquanto k > 0 e P[k+1] ≠ P[q] faça
4            k ← Π[k]
5        se P[k+1] = P[q]
6            então k ← k+1
7        Π[q] ← k
8    devolva Π
```

Método do potencial: Tome $\Phi_q = k$ no fim da iteração q.

$\Phi_1 = 0$ e $\Phi_q \geq 0$ para $q = 1, \ldots, m$.

c_q: $1+$ número de execuções da linha 4 na iteração q.

$\hat{c}_q = c_q + \Phi_q - \Phi_{q-1}$

Consumo de tempo do KMP

```
CALCULA-PREFIXO (P, m)
1    Π[1] ← 0     k ← 0
2    para q ← 2 até m faça
3        enquanto k > 0 e P[k+1] ≠ P[q] faça
4            k ← Π[k]
5        se P[k+1] = P[q]
6            então k ← k+1
7        Π[q] ← k
8    devolva Π
```

$\Phi_q = k$ no fim da iteração q e $\Phi_1 = 0$.

c_q: $1+$ número de execuções da linha 4 na iteração q.

$\hat{c}_q = c_q + \Phi_q - \Phi_{q-1}$

Consumo de tempo do KMP

CALCULA-PREFIXO (P, m)

1 $\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
2 **para** $q \leftarrow 2$ **até** m **faça**
3   **enquanto** $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
4         $k \leftarrow \Pi[k]$
5   **se** $P[k+1] = P[q]$
6   **então** $k \leftarrow k + 1$
7     $\Pi[q] \leftarrow k$
8 **devolva** Π

$\Phi_q = k$ no fim da iteração q e $\Phi_1 = 0$.

c_q: $1+$ número de execuções da linha 4 na iteração q.

$\hat{c}_q = c_q + \Phi_q - \Phi_{q-1}$

Cada execução da linha 4 faz k diminuir de pelo menos 1.

Logo $\Phi_q \leq \Phi_{q-1} - (c_q - 1) + 1$, e $\hat{c}_q \leq c_q - (c_q - 1) + 1 = 2$.

Consumo de tempo do KMP

```
CALCULA-PREFIXO (P, m)
1   Π[1] ← 0    k ← 0
2   para q ← 2 até m faça
3       enquanto k > 0 e P[k+1] ≠ P[q] faça
4           k ← Π[k]
5       se P[k+1] = P[q]
6           então k ← k + 1
7       Π[q] ← k
8   devolva Π
```

$\Phi_q = k$ no fim da iteração q e $\Phi_1 = 0$.

c_q: $1+$ número de execuções da linha 4 na iteração q.

$\hat{c}_q = c_q + \Phi_q - \Phi_{q-1}$

Cada execução da linha 4 faz k diminuir de pelo menos 1.

Logo $\Phi_q \leq \Phi_{q-1} - (c_q - 1) + 1$, e $\hat{c}_q \leq c_q - (c_q - 1) + 1 = 2$.

Custo amortizado por iteração: 2

Consumo de tempo do KMP

CALCULA-PREFIXO (P, m)

1 $\Pi[1] \leftarrow 0 \quad k \leftarrow 0$
2 **para** $q \leftarrow 2$ **até** m **faça**
3   **enquanto** $k > 0$ **e** $P[k+1] \neq P[q]$ **faça**
4         $k \leftarrow \Pi[k]$
5   **se** $P[k+1] = P[q]$
6   **então** $k \leftarrow k+1$
7     $\Pi[q] \leftarrow k$
8 **devolva** Π

$\Phi_q = k$ no fim da iteração q e $\Phi_1 = 0$.

c_q: $1+$ número de execuções da linha 4 na iteração q.

$\hat{c}_q = c_q + \Phi_q - \Phi_{q-1}$

Cada execução da linha 4 faz k diminuir de pelo menos 1.

Logo $\Phi_q \leq \Phi_{q-1} - (c_q - 1) + 1$, e $\hat{c}_q \leq c_q - (c_q - 1) + 1 = 2$.

Consumo de tempo total: $\sum_q c_q \leq \sum_q \hat{c}_q \leq 2m = \Theta(m)$

Algoritmo KMP

KMP (T, n, P, m)

$\Pi \leftarrow$ CALCULA-PREFIXO (P, m)
$q \leftarrow 0$
para $i \leftarrow 1$ **até** n **faça**
    enquanto $q > 0$ **e** $P[q+1] \neq T[i]$ **faça**
        $q \leftarrow \Pi[q]$
    se $P[q+1] = T[i]$
        então $q \leftarrow q + 1$
    se $q = m$
        então imprima $i - q$
                $q \leftarrow \Pi[q]$

Algoritmo KMP

Invariantes:

(1) $q < m$

(2) $P_q \sqsupseteq T_{i-1}$

(3) q é o maior possível tal que valem (1) e (2)

Algoritmo KMP

Invariantes:

(1) $q < m$

(2) $P_q \sqsupseteq T_{i-1}$

(3) q é o maior possível tal que valem (1) e (2)

Método do potencial: Tome $\Phi_i = q$ no fim da iteração i.

Algoritmo KMP

Invariantes:

(1) $q < m$

(2) $P_q \sqsupseteq T_{i-1}$

(3) q é o maior possível tal que valem (1) e (2)

Método do potencial: Tome $\Phi_i = q$ no fim da iteração i.

Exercício: Complete a análise de correção e a análise do consumo de tempo.