

25 YEARS ANNIVERSARY  
SOKT

ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

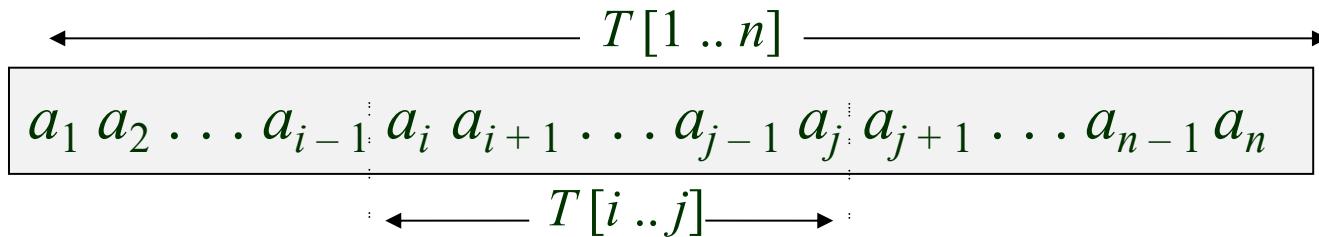
# Thuật Toán Xử Lý Xâu

## THUẬT TOÁN ỨNG DỤNG

1. Bài toán tìm kiếm xâu mẫu
2. Thuật toán trực tiếp
3. Thuật toán Boyer-Moore
4. Thuật toán Rabin-Karp
5. Thuật toán Knuth-Morris-Pratt (KMP)

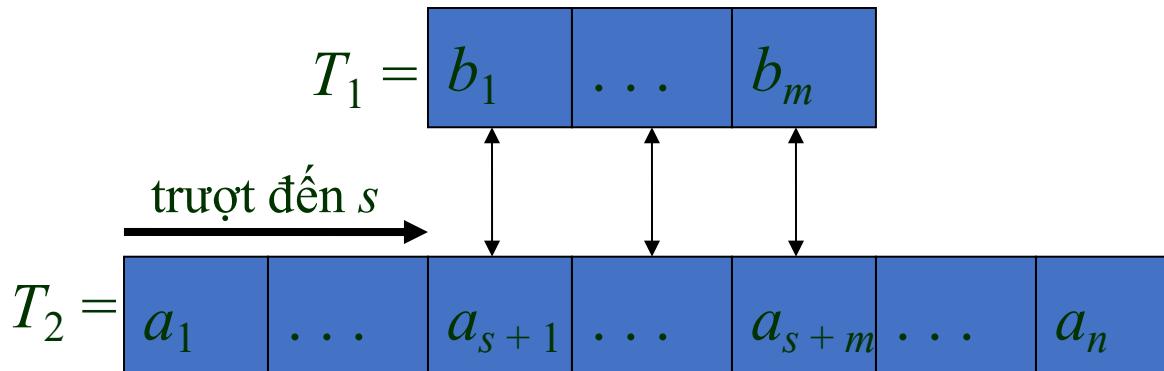
# Xâu (Strings)

- Xâu là dãy ký hiệu lấy từ bảng ký hiệu (alphabet)  $\Sigma$
- Ký hiệu  $T[i..j]$  là xâu con của xâu  $T$  bắt đầu từ vị trí  $i$  và kết thúc ở vị trí  $j$



# Trượt/đẩy (Shifts)

- Giả sử  $T_1$  và  $T_2$  là 2 xâu
- trong đó  $|T_1| = m$  và  $|T_2| = n$
- Ta nói  $T_1$  xuất hiện nhờ trượt (đẩy) đến  $s$  trong  $T_2$  nếu
- $T_1[1 .. m] = T_2[s + 1 .. s + m]$



# Vị trí khớp và không khớp

- Giả sử  $T_1$  và  $T_2$  là hai xâu
- Nếu  $T_1$  xuất hiện nhò trượt đến  $s$  trong  $T_2$  thì
- $s$  được gọi là **vị trí khớp** của  $T_1$  trong  $T_2$
- ngược lại
- $s$  được gọi là **vị trí không khớp**

# Bài toán tìm kiếm xâu mẫu

## (The String Matching Problem)

- Cho xâu  $T$  độ dài  $n$ 
  - $T$  được gọi là văn bản
- Cho xâu  $P$  độ dài  $m$ 
  - $P$  được gọi là xâu mẫu (*pattern*)
- **Bài toán: Tìm tất cả các vị trí khớp của  $P$  trong  $T$**
- **Ứng dụng:**
  - ☒ trong thu thập thông tin (information retrieval)
  - ☒ trong soạn thảo văn bản (text editing)
  - ☒ trong tính toán sinh học (computational biology)
  - ☒...

# Ví dụ

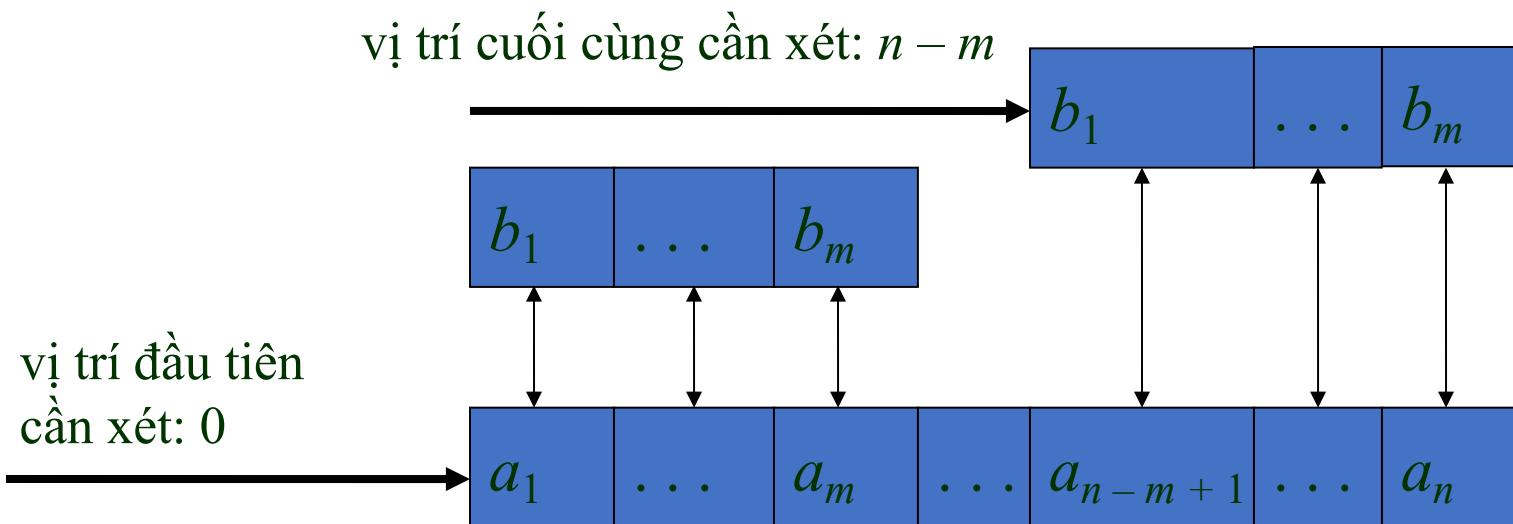
- Ví dụ:  $T = 000010001010001$  và  $P = 0001$ , các vị trí khớp là:
  - $s = 1$ 
    - $T = 0\textcolor{red}{0001}0001010001$
    - $P = \textcolor{blue}{0001}$
  - $s = 5$ 
    - $T = 0\textcolor{red}{0001}\textcolor{blue}{0001}010001$
    - $P = \textcolor{blue}{0001}$
  - $s = 11$ 
    - $T = 0\textcolor{red}{0001}\textcolor{blue}{0001}01\textcolor{cyan}{0001}$
    - $P = \textcolor{blue}{0001}$

1. Bài toán tìm kiếm xâu mẫu
- 2. Thuật toán trực tiếp**
3. Thuật toán Boyer-Moore
4. Thuật toán Rabin-Karp
5. Thuật toán Knuth-Morris-Pratt (KMP)

# Thuật toán trực tiếp

## Naïve algorithm

- Ý tưởng: Dịch chuyển từng vị trí  $s = 0, 1, \dots, n-m$ , với mỗi vị trí kiểm tra xem xâu mẫu có xuất hiện ở vị trí đó hay không.



# Thuật toán trực tiếp

## Naïve algorithm

- Ý tưởng: Dịch chuyển từng vị trí  $s=0, 1, \dots, n-m$ , với mỗi vị trí kiểm tra xem xâu mẫu có xuất hiện ở vị trí đó hay không.

```
void NaiveSM(char *P, int m, char *T, int n) {  
    int i, j;  
    /* Searching */  
    for (j = 0; j <= n - m; ++j) {  
        for (i = 1; i <= m && P[i] == T[i + j]; ++i);  
        if (i > m) OUTPUT(j);  
    }  
}
```

Thời gian tính trong tình huống tồi nhất =  $O(nm)$ .

# Thuật toán trực tiếp

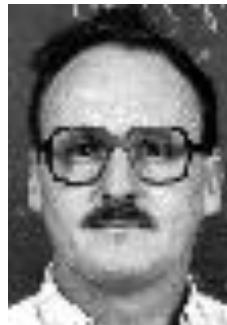
- Ví dụ:  $T = 000010001010001$  và  $P = 0001$ .

|       |                       |                          |
|-------|-----------------------|--------------------------|
|       | $T = 000010001010001$ |                          |
| $s=0$ | $P = 0001$            | $\Rightarrow$ không khớp |
| $s=1$ | $P = 0001$            | $\Rightarrow$ khớp       |
| $s=2$ | $P = 0001$            | $\Rightarrow$ không khớp |
| $s=3$ | $P = 0001$            | $\Rightarrow$ không khớp |
| $s=4$ | $P = 0001$            | $\Rightarrow$ không khớp |
| $s=5$ | $P = 0001$            | $\Rightarrow$ khớp       |
| $s=6$ | $P = 0001$            | $\Rightarrow$ không khớp |
| ...   |                       |                          |

1. Bài toán tìm kiếm xâu mẫu
2. Thuật toán trực tiếp
- 3. Thuật toán Boyer-Moore**
4. Thuật toán Rabin-Karp
5. Thuật toán Knuth-Morris-Pratt (KMP)

# Boyer-Moore Algorithm

- Làm việc tốt khi  $P$  dài và  $\Sigma$  lớn
- Chúng ta sẽ xét sự khớp nhau bằng cách duyệt từ phải qua trái.
- Trước hết hãy quan sát lại thuật toán trực tiếp làm việc theo thứ tự duyệt này...



Robert-Stephen Boyer

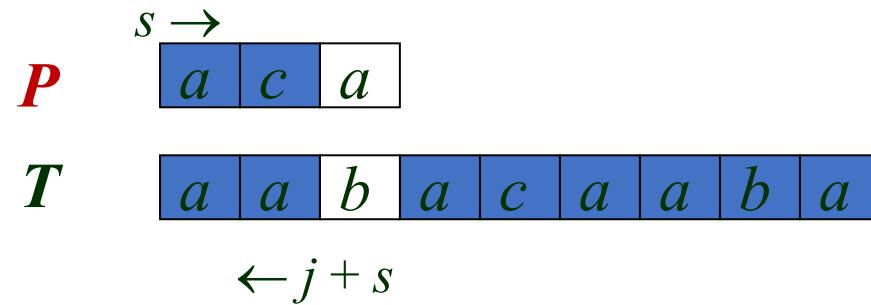


J. Strother Moore

# Thuật toán trực tiếp cải biên

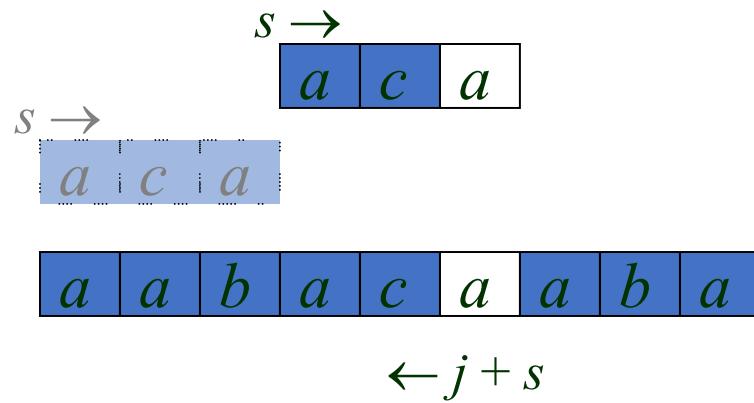
```
for  $s \leftarrow 0$  to  $n - m$  do  
     $j \leftarrow m$   
    while  $j > 0$  and  $T[j + s] = P[j]$  do  
         $j \leftarrow j - 1$   
    if  $j = 0$  then  
        print  $s$  “là vị trí khớp”
```

# Xét ví dụ



$b$  không xuất hiện ở bất cứ đâu trong  $P$   
vì thế có thể di chuyển  $P$  bỏ qua  $b$

# Bỏ qua được một đoạn



# Xét ví dụ khác

$s \rightarrow$

|   |   |   |   |   |
|---|---|---|---|---|
| a | c | b | a | b |
|---|---|---|---|---|

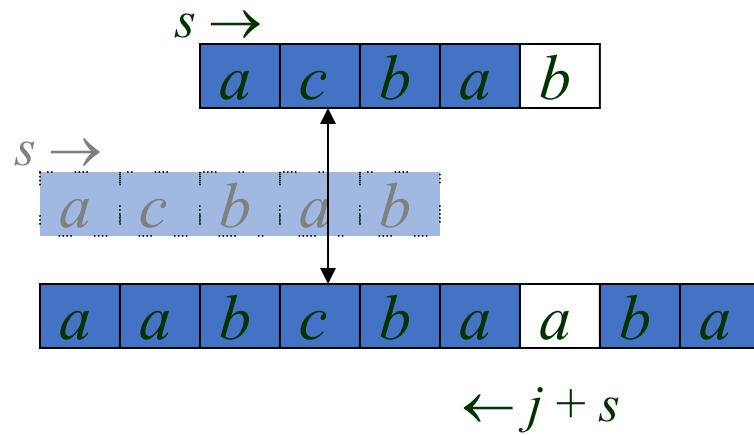
|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| a | a | b | c | b | a | a | b | a |
|---|---|---|---|---|---|---|---|---|

$\leftarrow j + s$

$c$  xuất hiện ở vị trí 2 trong  $P$

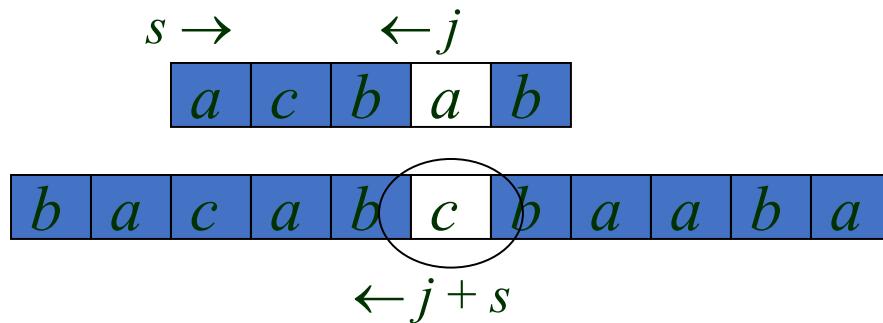
vì thế có thể di chuyển  $P$  sao cho các  $c$  được đóng thăng

# Bỏ qua được một đoạn



# Ký tự tồi

- Giả sử  $P[j] \neq T[j + s]$



- Ta gọi  $T[j + s]$  là ký tự tồi (**bad character**)
- Sử dụng ký tự tồi ta có thể tránh được việc dóng hàng không đúng

# Hàm Last

- $T[j + s]$  xuất hiện ở vị trí nào trong  $P$ ?
- Ta xác định hàm *last* như sau:

Nếu  $c$  xuất hiện trong  $P$  thì đặt

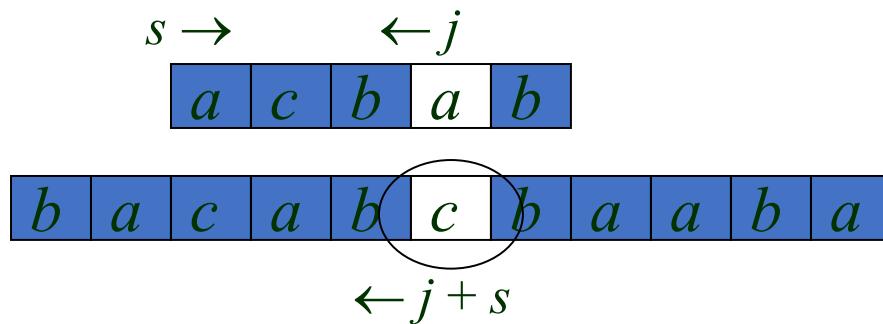
last( $c$ ) = chỉ số lớn nhất (bên phải nhất) của vị trí xuất hiện của  $c$  trong  $P$

Trái lại đặt

last( $c$ ) = 0

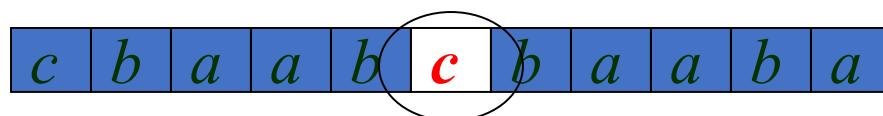
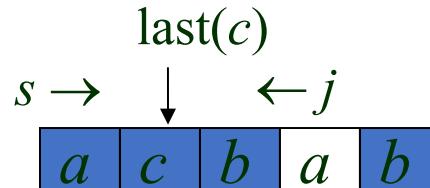
# Tăng vị trí dịch chuyen

- Giả sử ký tự tồi được đóng ở vị trí  $j$  của  $P$



# Tình huống 1

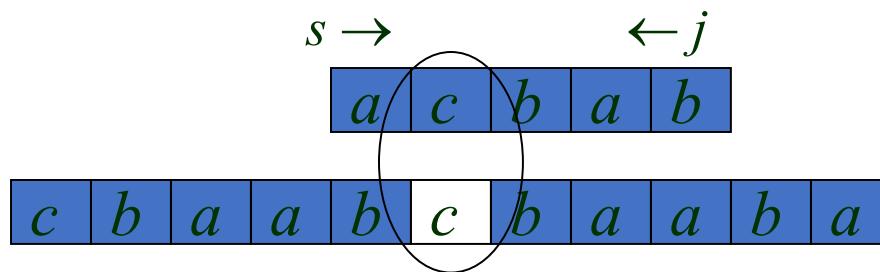
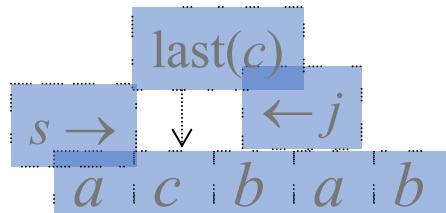
- Ký tự tồi có mặt trong  $P$  và  $\text{last}(c) < j$



**Khi đó có thể đẩy đến:**  $s \leftarrow s + (j - \text{last}(c))$

# Tình huống 1

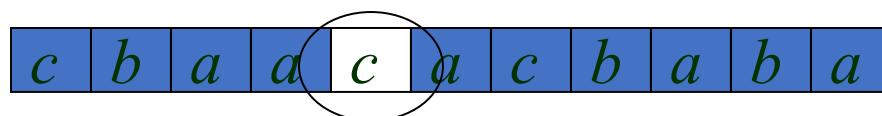
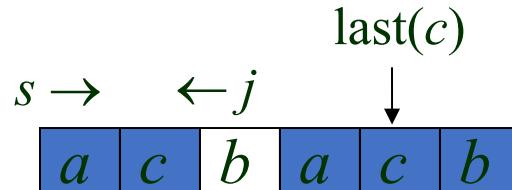
- Ký tự tồi có mặt trong  $P$  và  $\text{last}(c) < j$



**Khi đó có thể đẩy đến:**  $s \leftarrow s + (j - \text{last}(c))$

# Tình huống 2

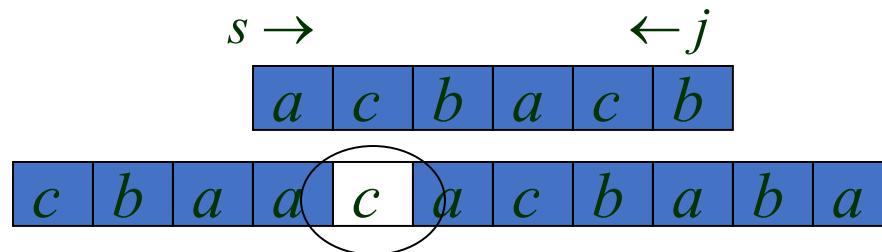
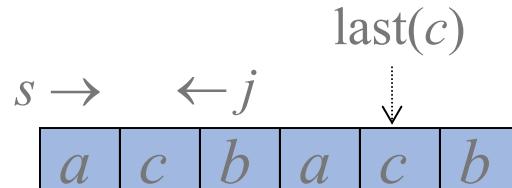
- Ký tự tồi có mặt trong  $P$  và  $\text{last}(c) > j$



**Khi đó:**  $s \leftarrow s + 1$

# Tình huống 2

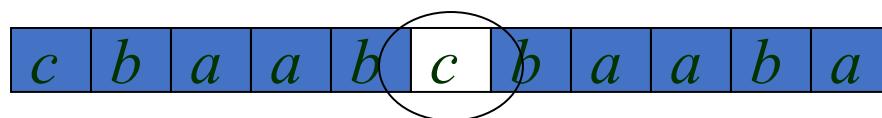
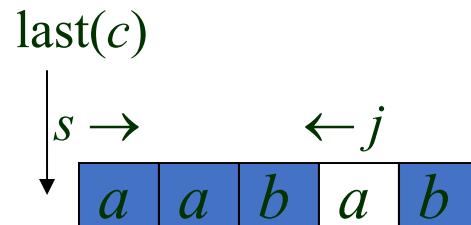
- Ký tự tối có mặt trong  $P$  và  $\text{last}(c) > j$



**Khi đó:**  $s \leftarrow s + 1$

# Tình huống 3

- Ký tự tôi không có mặt trong  $P$  và vì thế  $\text{last}(c) = 0$



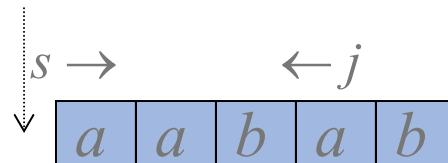
**Khi đó:**  $s \leftarrow s + (j - \text{last}(c))$

hay     $s \leftarrow s + j$

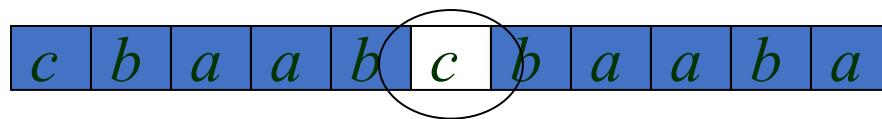
# Tình huống 3

- Ký tự tồi không có mặt trong  $P$  và do đó  $\text{last}(c) = 0$

$\text{last}(c)$



$s \rightarrow \quad \quad \quad \leftarrow j$



**Khi đó:**  $s \leftarrow s + (j - \text{last}(c))$

$$s \leftarrow s + j$$

# Boyer-Moore Algorithm

```
s ← 0
while  $s \leq n - m$  do
     $j \leftarrow m$ 
    while  $j > 0$  and  $T[j + s] = P[j]$  do
         $j \leftarrow j - 1$ 
    if  $j = 0$  then
        print  $s$  “là vị trí khớp”
         $s \leftarrow s + 1$ 
    else
         $k \leftarrow \text{last}(T[j + s])$ 
         $s \leftarrow s + \max(j - k, 1)$ 
```

# Ví dụ minh họa

pattern

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| a | c | a | b | a | c |
|---|---|---|---|---|---|

text

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | a | b | a | c | b | d | c | a | a | c | a | a | c | a | b | a | c |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Tính **last** cho mỗi ký hiệu trong bảng ký hiệu  $\Sigma = \{a, b, c, d\}$

# Ví dụ minh họa

$$\begin{aligned}\text{last}(a) &= 5 \quad \text{last}(c) = 6 \\ \text{last}(b) &= 4 \quad \text{last}(d) = 0\end{aligned}$$

pattern

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| a | c | a | b | a | c |
|---|---|---|---|---|---|

text

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | a | b | a | c | b | d | c | a | a | c | a | a | c | a | b | a | c |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

khởi tạo  $s$  bằng 0

```

 $s \leftarrow 0$ 
while  $s \leq n - m$  do
     $j \leftarrow m$ 
    while  $j > 0$  and  $T[j + s] = P[j]$  do  $j \leftarrow j - 1$ 
    if  $j = 0$  then
        print  $s$  “là vị trí khớp”
         $s \leftarrow s + 1$ 
    else
         $k \leftarrow \text{last}(T[j + s])$ 
         $s \leftarrow s + \max(j - k, 1)$ 

```

$s \rightarrow$

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| a | c | a | b | a | c |
|---|---|---|---|---|---|

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | a | b | a | c | b | d | c | a | a | c | a | a | c | a | b | a | c |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

đặt  $j = m$

```

 $s \leftarrow 0$ 
while  $s \leq n - m$  do
     $j \leftarrow m$ 
    while  $j > 0$  and  $T[j + s] = P[j]$  do  $j \leftarrow j - 1$ 
    if  $j = 0$  then
        print  $s$  “là vị trí khớp”
         $s \leftarrow s + 1$ 
    else
         $k \leftarrow \text{last}(T[j + s])$ 
         $s \leftarrow s + \max(j - k, 1)$ 

```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

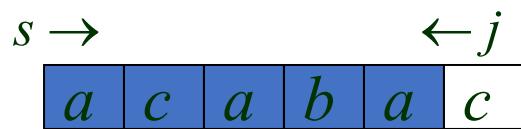


so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

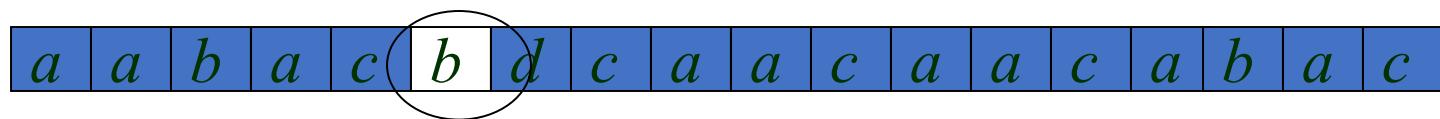
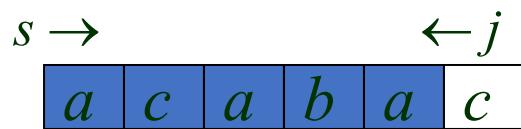


$P[j] \neq T[j + s]$  do đó phát hiện ký tự  $b$  là tồi

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last(T[j + s])  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



exit **while** loop với  $j > 0$

đặt  $k = \text{last}(b) = 4$

tăng  $s$  thêm  $\max(j - k, 1) = 2$

```
s ← 0  
while  $s \leq n - m$  do
```

```
j ← m  
while  $j > 0$  and  $T[j + s] = P[j]$  do  $j \leftarrow j - 1$   
if  $j = 0$  then  
    print  $s$  "là vị trí khớp"  
     $s \leftarrow s + 1$   
else  
     $k \leftarrow \text{last}(T[j + s])$   
     $s \leftarrow s + \max(j - k, 1)$ 
```

$$\begin{aligned}\text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0\end{aligned}$$

$s \rightarrow$



đặt  $j = m$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

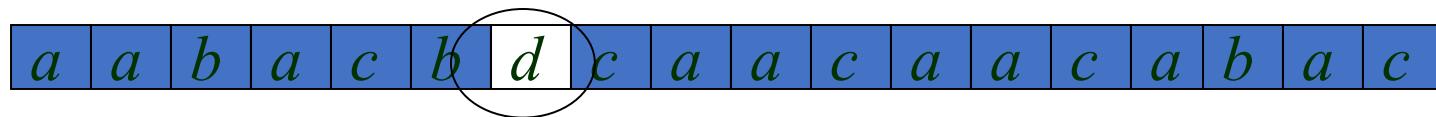


$P[j] \neq T[j + s]$  ký tự  $d$  là tồi

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last(T[j + s])  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



exit **while** loop với  $j > 0$

đặt  $k = \text{last}(d) = 0$

tăng  $s$  lên  $\max(j - k, 1) = 5$

```
s ← 0  
while s ≤ n – m do
```

```
j ← m  
while j > 0 and T[j + s] = P[j] do j ← j – 1  
if j = 0 then  
    print s “là vị trí khớp”  
    s ← s + 1  
else  
    k ← last( T[j + s] )  
    s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

$s \rightarrow$



đặt  $j = m$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{array}{l} \text{last}(a) = 5 \quad \text{last}(c) = 6 \\ \text{last}(b) = 4 \quad \text{last}(d) = 0 \end{array}$$



so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

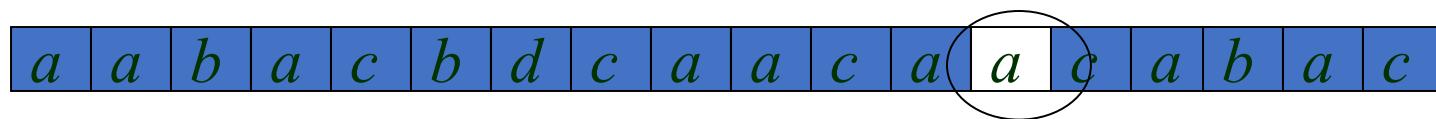


$P[j] \neq T[j + s]$  : ký tự  $a$  là tồi

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last(T[j + s])  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



exit **while** loop với  $j > 0$

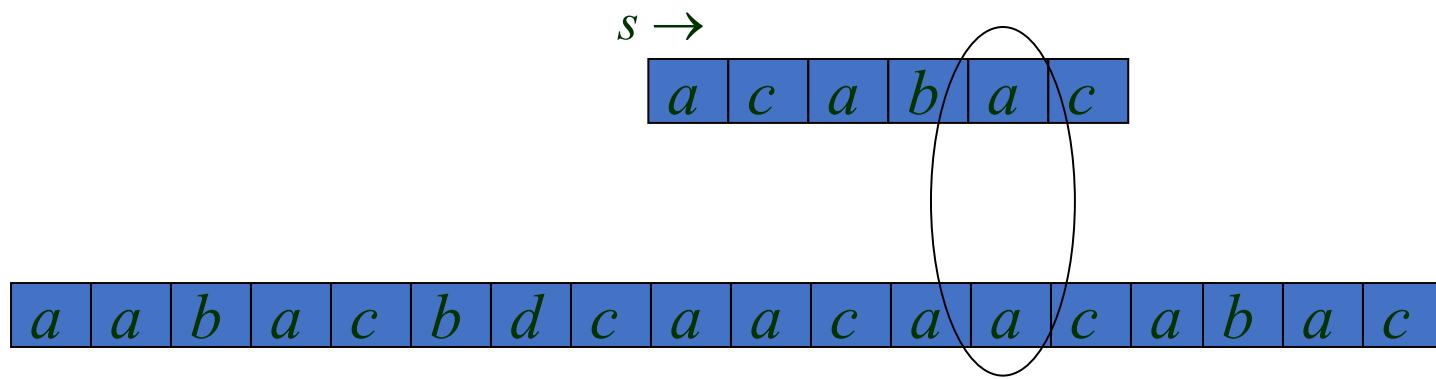
đặt  $k = \text{last}(a) = 5$

tăng  $s$  lên  $\max(j - k, 1) = 1$

```
s ← 0  
while  $s \leq n - m$  do
```

```
    j ← m  
    while  $j > 0$  and  $T[j + s] = P[j]$  do  $j \leftarrow j - 1$   
    if  $j = 0$  then  
        print  $s$  "là vị trí khớp"  
         $s \leftarrow s + 1$   
    else  
         $k \leftarrow \text{last}(T[j + s])$   
         $s \leftarrow s + \max(j - k, 1)$ 
```

$$\begin{aligned}\text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0\end{aligned}$$



đặt  $j = m$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{array}{l} \text{last}(a) = 5 \quad \text{last}(c) = 6 \\ \text{last}(b) = 4 \quad \text{last}(d) = 0 \end{array}$$

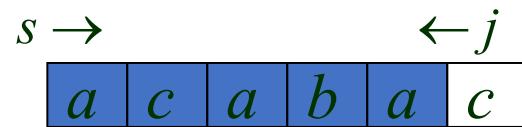


so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{array}{l} \text{last}(a) = 5 \quad \text{last}(c) = 6 \\ \text{last}(b) = 4 \quad \text{last}(d) = 0 \end{array}$$



$P[j] = T[j + s]$ , vì thế giảm  $j$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

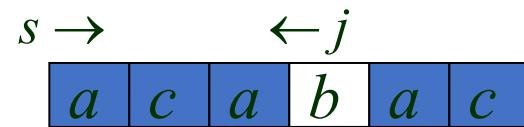


$P[j] = T[j + s]$  vì thế giảm  $j$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

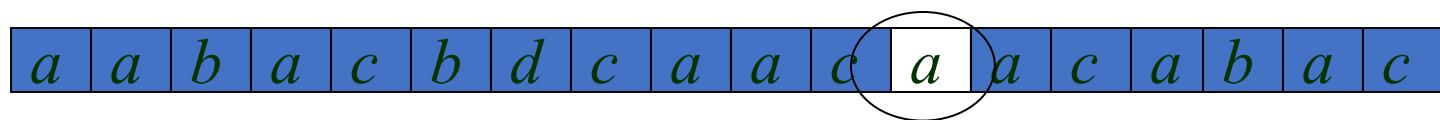
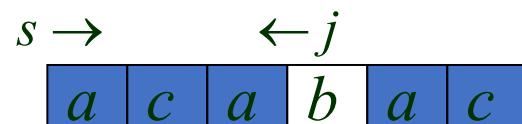


$P[j] \neq T[j + s]$  nên  $a$  là ký tự tồi

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last(T[j + s])  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



exit **while** loop với  $j > 0$

đặt  $k = \text{last}(a) = 5$

tăng  $s$  lên  $\max(j - k, 1) = 1$

```
s ← 0  
while s ≤ n – m do
```

```
j ← m  
while j > 0 and T[j + s] = P[j] do j ← j – 1  
if j = 0 then  
    print s “là vị trí khớp”  
    s ← s + 1  
else  
    k ← last( T[j + s] )  
    s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



đặt  $j = m$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{array}{l} \text{last}(a) = 5 \quad \text{last}(c) = 6 \\ \text{last}(b) = 4 \quad \text{last}(d) = 0 \end{array}$$

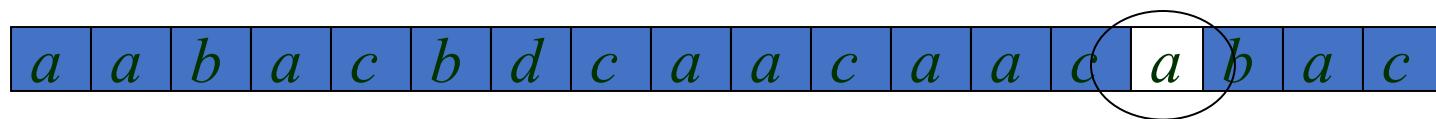


$P[j] \neq T[j + s]$  nên  $a$  là ký tự tồi

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last(T[j + s])  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



exit **while** loop với  $j > 0$

đặt  $k = \text{last}(a) = 5$

tăng  $s$  lên  $\max(j - k, 1) = 1$

```
s ← 0  
while s ≤ n – m do
```

```
j ← m  
while j > 0 and T[j + s] = P[j] do j ← j – 1  
if j = 0 then  
    print s “là vị trí khớp”  
    s ← s + 1  
else  
    k ← last( T[j + s] )  
    s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



đặt  $j = m$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

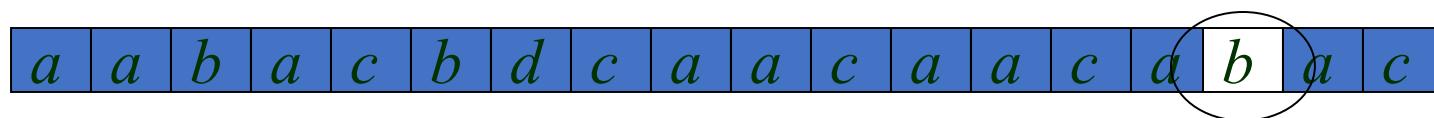


$P[j] \neq T[j + s]$  vì thế  $b$  là ký tự tồi

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last(T[j + s])  
        s ← s + max(j - k, 1)
```

$$\begin{array}{l} \text{last}(a) = 5 \quad \text{last}(c) = 6 \\ \text{last}(b) = 4 \quad \text{last}(d) = 0 \end{array}$$



exit **while** loop với  $j > 0$

đặt  $k = \text{last}(b) = 4$

tăng  $s$  lên  $\max(j - k, 1) = 2$

```
s ← 0  
while s ≤ n – m do
```

```
j ← m  
while j > 0 and T[j + s] = P[j] do j ← j – 1  
if j = 0 then  
    print s “là vị trí khớp”  
    s ← s + 1  
else  
    k ← last( T[j + s] )  
    s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



đặt  $j = m$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



so sánh  $P[j]$  với  $T[j + s]$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

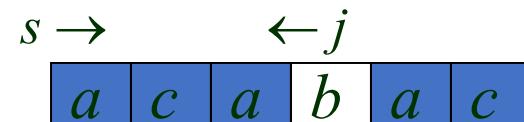


$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$

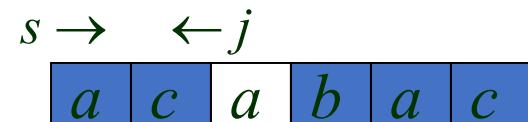


$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0 \end{aligned}$$



$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while s ≤ n – m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j – 1  
    if j = 0 then  
        print s “là vị trí khớp”  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j – k, 1)
```

$$\begin{array}{l} \text{last}(a) = 5 \quad \text{last}(c) = 6 \\ \text{last}(b) = 4 \quad \text{last}(d) = 0 \end{array}$$

$s \rightarrow\leftarrow j$

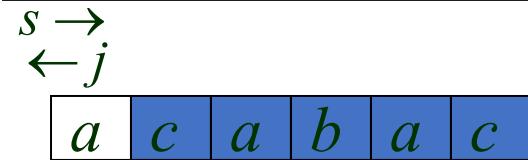


$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while s ≤ n - m do
```

```
    j ← m  
    while j > 0 and T[j + s] = P[j] do j ← j - 1  
    if j = 0 then  
        print s "là vị trí khớp"  
        s ← s + 1  
    else  
        k ← last( T[j + s] )  
        s ← s + max(j - k, 1)
```

$$\begin{aligned} \text{last}(a) &= 5 \quad \text{last}(c) = 6 \\ \text{last}(b) &= 4 \quad \text{last}(d) = 0 \end{aligned}$$



|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | a | b | a | c | b | d | c | a | a | c | a | a | c | a | b | a | c |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

$P[j] = T[j + s]$  do đó giảm  $j$

```
s ← 0  
while  $s \leq n - m$  do
```

```
    j ← m  
    while  $j > 0$  and  $T[j + s] = P[j]$  do  $j \leftarrow j - 1$   
    if  $j = 0$  then  
        print  $s$  "là vị trí khớp"  
         $s \leftarrow s + 1$   
    else  
         $k \leftarrow \text{last}(T[j + s])$   
         $s \leftarrow s + \max(j - k, 1)$ 
```

$$\begin{aligned}\text{last}(a) &= 5 & \text{last}(c) &= 6 \\ \text{last}(b) &= 4 & \text{last}(d) &= 0\end{aligned}$$

$\leftarrow j$   $\xrightarrow{s}$

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| a | c | a | b | a | c |
|---|---|---|---|---|---|

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | a | b | a | c | b | d | c | a | a | c | a | a | c | a | b | a | c |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

$j = 0$  do đó  $s$  là vị trí khớp

# Thời gian tính

- Việc tính hàm **last** đòi hỏi thời gian  $O(m + |\Sigma|)$
- Tình huống tồi nhất không khác gì thuật toán trực tiếp, nghĩa là đòi hỏi thời gian  $O(nm + |\Sigma|)$
- Ví dụ tình huống tồi nhất xảy ra khi
  - Pattern:  $ba^{m-1}$
  - Text:  $a^n$
- Thuật toán làm việc kém hiệu quả đối với bảng  $\Sigma$  nhỏ

1. Bài toán tìm kiếm xâu mẫu
2. Thuật toán trực tiếp
3. Thuật toán Boyer-Moore
- 4. Thuật toán Rabin-Karp**
5. Thuật toán Knuth-Morris-Pratt (KMP)

# Rabin-Karp Algorithm

- **Ý tưởng:**
  - Coi mẫu  $P[0..m-1]$  như là khoá, và chuyển đổi nó thành số nguyên  $p$
  - Tương tự như vậy ta cũng chuyển đổi các xâu con của  $T[]$  thành các số nguyên
    - For  $s=0,1,\dots,n-m$ , chuyển  $T[s..s+m-1]$  thành số nguyên tương đương  $t_s$
    - **Mẫu xuất hiện ở vị trí s khi và chỉ khi  $p=t_s$**
- Nếu ta có thể tính  $p$  và  $t_s$  nhanh chóng, thì bài toán tìm kiếm xâu mẫu qui dẫn về bài toán so sánh  $p$  với  $n-m+1$  số nguyên

# Rabin-Karp Algorithm ...

- Tính  $p$  như thế nào?

$$p = 2^{m-1} P[0] + 2^{m-2} P[1] + \dots + 2 P[m-2] + P[m-1]$$

- Sử dụng sơ đồ Horner

$$p = P[m-1] + 2*(P[m-2] + 2*(P[m-3] + \dots + 2*(P[1] + 2*P[0]) \dots)),$$
  
ta có thể cài đặt trên C đoạn chương trình tính  $p$  như sau

```
p = 0;  
for (i=0; i<m; i++)  
    p = 2*p + P[i];
```

Việc này đòi hỏi thời gian  $O(m)$ , nếu giả thiết các phép toán số học được thực hiện với thời gian  $O(1)$ .

# Rabin-Karp Algorithm ...

- Tương tự, tính  $(n-m+1)$  số nguyên  $t_s$  từ xâu văn bản

```
for (s = 0; s <= n-m; s++) {  
    t[s] = 0;  
    for (i = 0; i < m; i++)  
        t[s] = 2*t[s] + T[s+i];  
}
```

- Việc này đòi hỏi thời gian  $O((n - m + 1) m)$ , với giả thiết các phép toán số học được thực hiện với thời gian  $O(1)$ .
- Công đoạn này là công đoạn tốn kém !

# Rabin-Karp Algorithm

- Để ý rằng có thể sử dụng  $t[s-1]$  để tính  $t[s]$ :

$$t[s-1] = 2^{m-1} T[s-1] + 2^{m-2} T[s] + \dots + 2 T[s+m-3] + T[s+m-2]$$

$$t[s] = 2^{m-1} T[s] + 2^{m-2} T[s-2] + \dots + 2 T[s+m-2] + T[s+m-1]$$

nên có thể thực hiện việc tính các số  $t_s$  hiệu quả hơn như sau:

```
t[0] = 0;  
offset = 1;  
for (i = 0; i < m; i++)  
    offset = 2*offset;  
for (i = 0; i < m; i++)  
    t[0] = 2*t[0] + T[i];  
for (s = 1; s <= n-m; s++)  
    t[s] = 2*(t[s-1] - offset*T[s-1]) + T[s+m-1];
```

Việc này đòi hỏi thời gian  $O(n + m)$ , với giả thiết các phép toán số học được thực hiện với thời gian  $O(1)$ .

# Khó khăn

- Khó khăn này sinh khi thực hiện thuật toán là nếu  $m$  là lớn thì các phép toán số học cần thực hiện là rất tốn thời gian, chứ không phải là  $O(1)$  như đã giả thiết.
  - Trong trường hợp  $m$  lớn ta thường phải cài đặt phép tính số học với số nguyên lớn.
- Để giải quyết khó khăn này có thể sử dụng tính toán theo modulo. Giả sử  $q$  là số nguyên tố sao cho  $2q$  có thể cất giữ trong một từ máy.
  - Điều này đảm bảo tất cả các tính toán đều được thực hiện nhờ sử dụng tính toán số học với độ chính xác đơn.

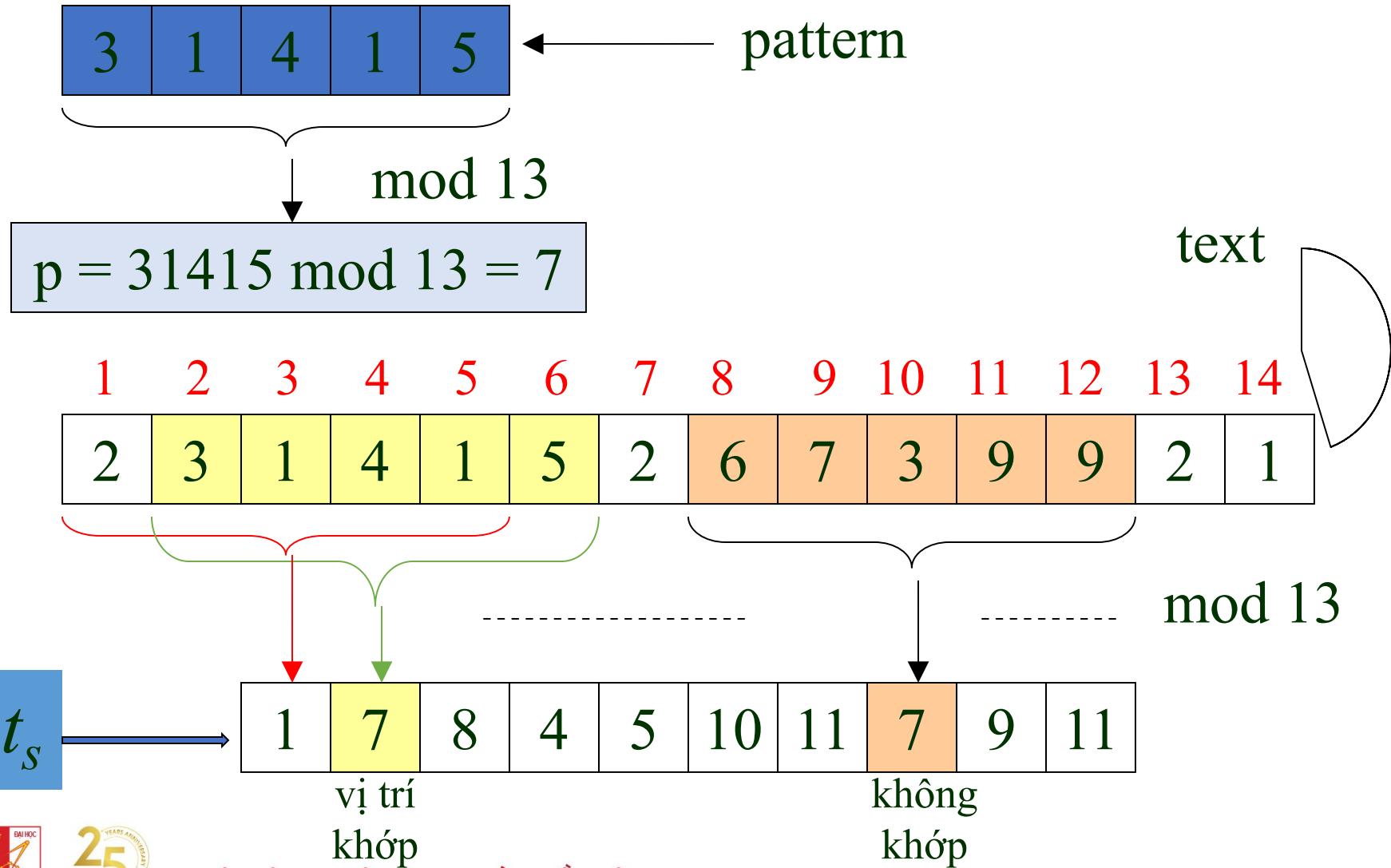
# Tính $t_s$

```
p =0;  
for (i=0; i<m; i++)  
    p = (2*p + P[i]) % q;  
t[0] = 0;  
offset = 1;  
for (i = 0; i < m; i++)  
    offset = 2*offset % q;  
for (i = 0; i < m; i++)  
    t[0] = (2*t[0]+T[i]) % q;  
for (s = 1; s <= n-m; s++)  
    t[s] = (2*(t[s-1]-offset*T[s-1])+T[s+m-1])%q;
```

# Chú ý

- Nếu sử dụng tính toán theo modulo, khi  $p=t_s$  với  $s$  nào đó, chúng ta không còn chắc chắn được rằng  $P[0 .. m-1]$  là bằng  $T[s..s+m-1]$
- Vì thế, sau khi kiểm tra được là  $p = t_s$ , ta cần tiếp tục so sánh  $P[0..m-1]$  với  $T[s..s+m-1]$  để có thể khẳng định là  $s$  đúng là vị trí khớp.
- Thời gian trong tình huống tồi nhất của thuật toán vẫn là  $O(nm)$ : khi  $P=a^m$  còn  $T = a^n$ . Tuy nhiên trong thực tế ứng dụng, ta loại được rất nhiều phép so sánh xâu mẫu.

# Ví dụ



1. Bài toán tìm kiếm xâu mẫu
2. Thuật toán trực tiếp
3. Thuật toán Boyer-Moore
4. Thuật toán Rabin-Karp
- 5. Thuật toán Knuth-Morris-Pratt (KMP)**

# Thuật toán Knuth-Morris-Pratt

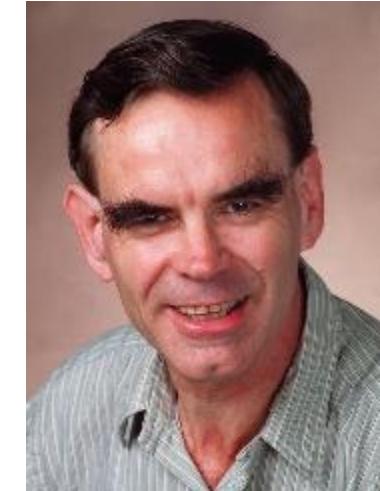
---



Donald Knuth



James Morris



Vaughan Pratt

Ý tưởng chính là:

- ★ Sử dụng hàm bù trợ (*prefix function*).
- ★ Đạt được thời gian tính  $O(n+m)$ !

# Prefix & Suffix

---

Xâu  $W$  được gọi là **tiền tố (prefix)** của xâu  $X$  nếu  $X = WY$  với một xâu  $Y$  nào đó, ký hiệu là  $W \sqsubset X$ .

**Ví dụ**  $W = \text{ab}$  là a prefix của  $X = \text{abefac}$  trong đó  $Y = \text{efac}$ .

Xâu rỗng  $\epsilon$  là prefix của mọi xâu.

Xâu  $W$  được gọi là **hậu tố (suffix)** của xâu  $X$  nếu  $X = YW$  với một xâu  $Y$  nào đó, ký hiệu là  $W \sqsupset X$ .

**Ví dụ:**  $W = \text{cdaa}$  là suffix của  $X = \text{acbecdaa}$  trong đó  $Y = \text{acbe}$

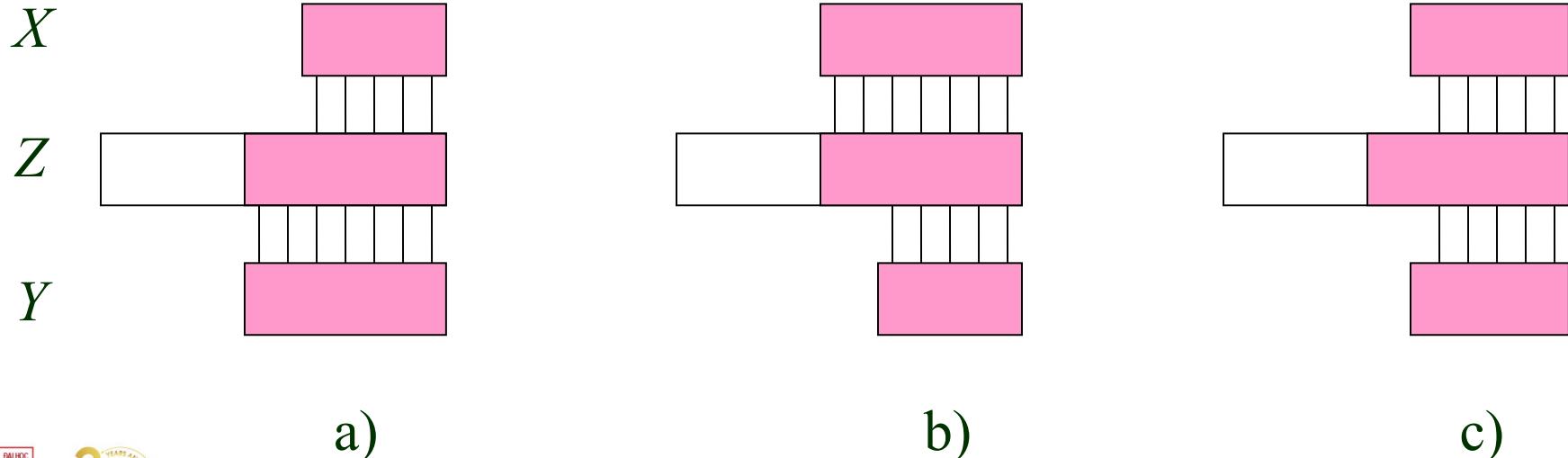
Xâu rỗng  $\epsilon$  là suffix của mọi xâu.

# Hậu tố đè nhau (Overlapping Suffix)

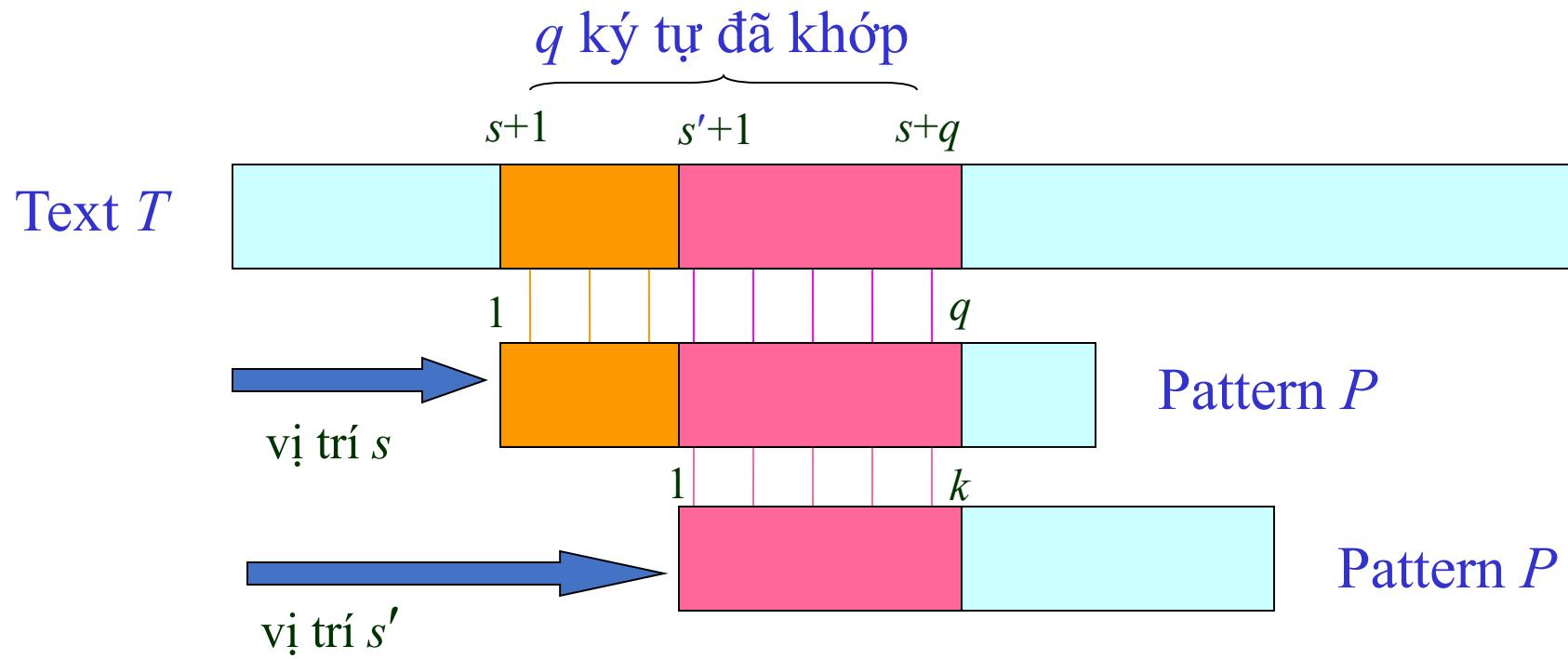
Bổ đề

Giả sử  $X \sqsupset Z$  và  $Y \sqsupset Z$ .

- a) nếu  $|X| \leq |Y|$ , thì  $X \sqsupset Y$ ;
- b) nếu  $|X| \geq |Y|$ , thì  $Y \sqsupset X$ ;
- c) nếu  $|X| = |Y|$ , thì  $X = Y$ .



# Dịch chuyển tối thiểu



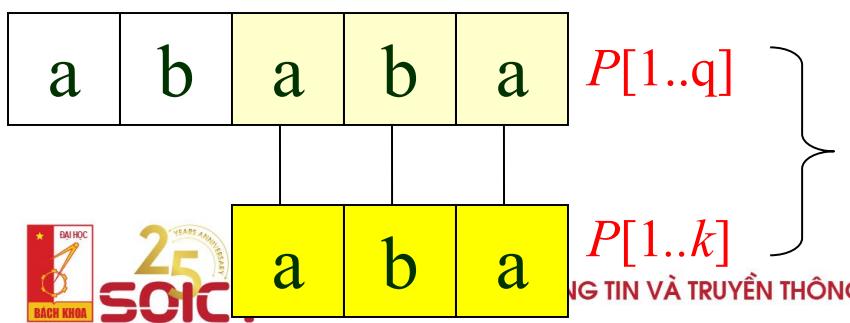
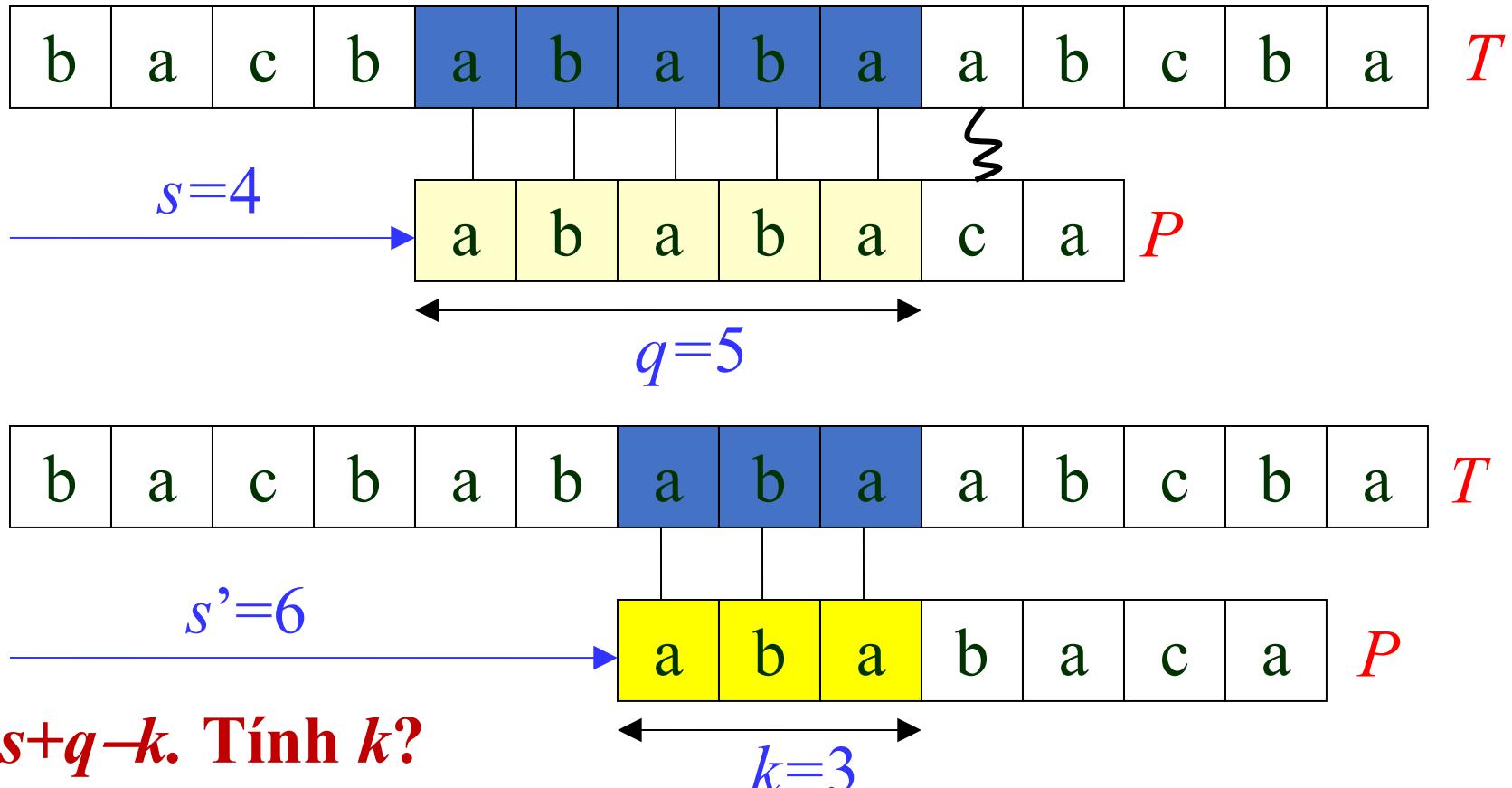
## Vấn đề đặt ra là:

Biết rằng prefix  $P[1..q]$  của xâu mẫu là khớp với đoạn  $T[(s+1)..(s+q)]$ , tìm giá trị nhỏ nhất  $s' > s$  sao cho:

$$P[1..k] = T[(s'+1)..(s'+k)], \text{ trong đó } s'+k=s+q.$$

Khi đó, tại vị trí  $s'$ , không cần thiết so sánh  $k$  ký tự đầu của  $P$  với các ký tự tương ứng của  $T$ , bởi vì ta biết chắc rằng chúng là khớp nhau.

# Prefix Function: Ví dụ



So sánh xâu mẫu với chính nó;  
prefix dài nhất của  $P$  đồng thời là  
suffix của  $P[1..5]$  là  $P[1..3]$ ; do đó  
 $\pi[5]=3$

# Hàm tiền tố (Prefix Function)

**Prefix function:**  $\pi[q]$  là độ dài của prefix dài nhất của  $P[1..m]$  đồng thời là suffix **thực sự** của  $P[1..q]$ .

$$\pi[q] = \max \{ k: \mathbf{k < q} \text{ và } P[1..k] \text{ là suffix của } P[1..q] \}$$

|          |   |   |   |   |   |   |   |   |   |    |
|----------|---|---|---|---|---|---|---|---|---|----|
| $i$      | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $P[i]$   | a | b | a | b | a | b | a | b | c | a  |
| $\pi[i]$ | 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 0 | 1  |

# Tính giá trị của Prefix Function

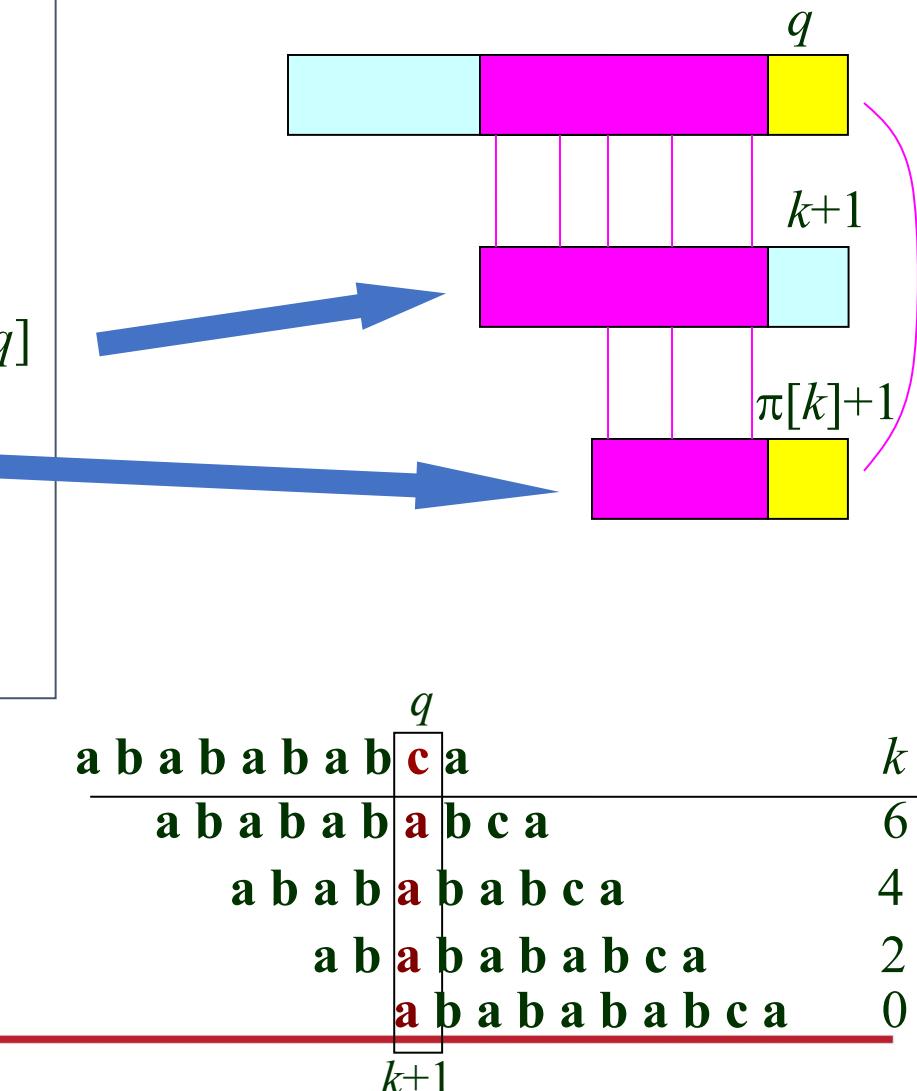
Compute-Prefix-Function( $P$ )

```
m ← length[P]  
 $\pi[1] \leftarrow 0$   
 $k \leftarrow 0$   
for  $q \leftarrow 2$  to  $m$   
do while  $k > 0$  and  $P[k+1] \neq P[q]$   
    do  $k \leftarrow \pi[k]$   
    if  $P[k+1] = P[q]$   
        then  $k \leftarrow k+1$   
 $\pi[q] \leftarrow k$   
return  $\pi$ 
```

Ví dụ.  $q = 9$  và  $k = 6$

$p[k+1] = a \neq c = p[q]$

$\pi[9] = \pi[\pi[\pi[6]]] = 0$



# Thời gian tính

```
1 Compute-Prefix-Function( $P$ )
2    $m \leftarrow \text{length}[P]$ 
3    $\pi[1] \leftarrow 0$ 
4    $k \leftarrow 0$ 
5   for  $q = 2$  to  $m$ 
6     do while  $k > 0$  and  $P[k+1] \neq P[q]$ 
7       do  $k \leftarrow \pi[k]$  // decrease  $k$  by at least 1
8       if  $P[k+1] = P[q]$ 
9         then  $k \leftarrow k+1$  //  $\leq m - 1$  increments, each by 1
10       $\pi[q] \leftarrow k$ 
11    return  $\pi$ 
```

số lần decrements  $\leq$  số lần increments,  
như vậy, tổng cộng dòng 7 thực hiện nhiều nhất  $m - 1$  lần.

Thời gian tính  $\Theta(m)$ .

# KMP Algorithm

```
KMP-Matcher( $T, P$ ) //  $n = |T|$  and  $m = |P|$ 
```

```
     $\pi \leftarrow \text{Compute-Prefix-Function}(P)$  //  $\Theta(m)$  time.
```

```
     $q \leftarrow 0$ 
```

```
    for  $i \leftarrow 1$  to  $n$ 
```

```
        do while  $q > 0$  and  $P[q+1] \neq T[i]$ 
```

```
            do  $q \leftarrow \pi[q]$ 
```

```
            if  $P[q+1] = T[i]$ 
```

```
                then  $q \leftarrow q+1$  //  $\leq n$  total increments
```

```
            if  $q = m$ 
```

```
                then print “Pattern xuất hiện ở vị trí”  $i - m$ 
```

```
                 $q \leftarrow \pi[q]$ 
```

//  $\Theta(n)$  time

Thời gian tổng cộng:  $\Theta(m+n)$ .

# Ví dụ: Thực hiện thuật toán KMP

|           |   |   |   |   |   |   |   |   |   |    |    |
|-----------|---|---|---|---|---|---|---|---|---|----|----|
| $i$       | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| $P[1..i]$ | a | b | a | b | b | a | b | a | b | a  | a  |
| $\pi[i]$  | 0 | 0 | 1 | 2 | 0 | 1 | 2 | 3 | 4 | 3  | 1  |

abab**a**bbbababbaababbababaa  
abab**b**ababaa

↓

đẩy đi  $q - \pi[q] = 4 - 2$

abab**babab**baababbababaa  
ababbababaa

↓

đẩy đi  $9 - 4 = 5$

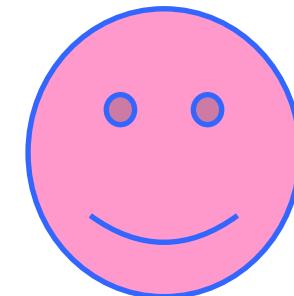
abababb**ababba**ababbababaa  
ababbababaa

abababbababba**a**bababbababaa  
**a**bababbababaa

↓

đẩy đi  $1 - 0 = 1$

abababbababba**ababbababaa**  
**ababbababaa**





25  
YEARS ANNIVERSARY  
**SOICT**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**  
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

**Thank you  
for your  
attentions!**

