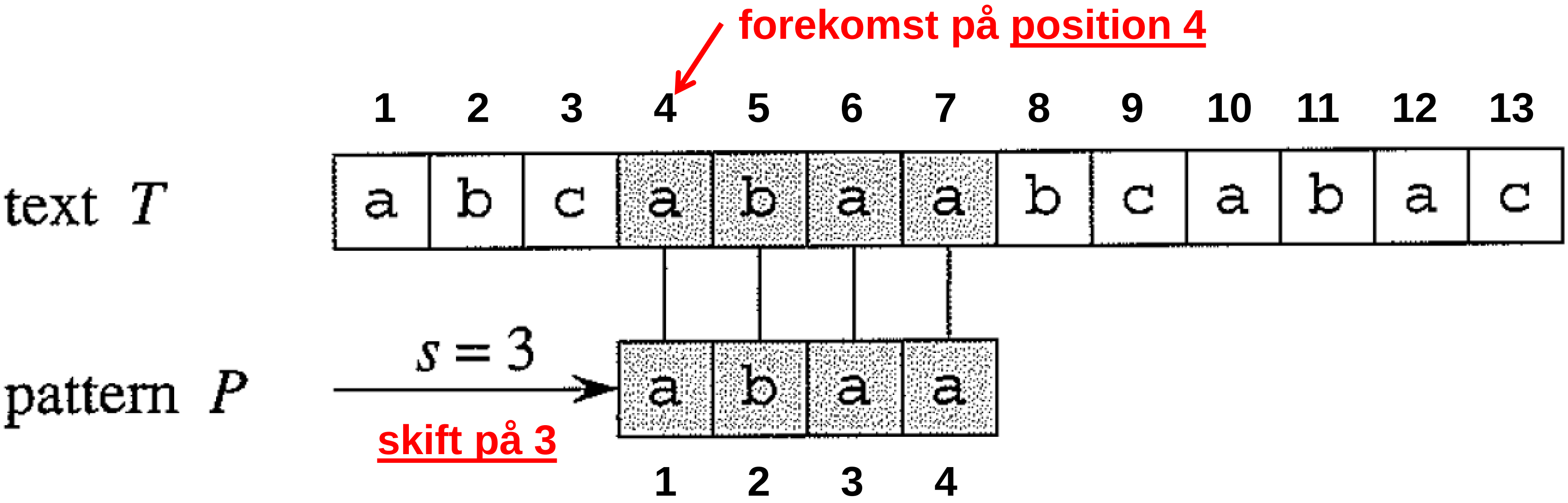
Algoritmer og Datastrukturer 2

Gerth Stølting Brodal

Mønstergenkendelse [CLRS, kapitel 32.1-32.2, 32.4]

AARHUS UNIVERSITET

Mønster genkendelse



forekomst på position 4

	1	2	3	4	5	6	7	8	9	10	11	12	13
text T	a	b	c	a	b	a	a	b	c	a	b	a	c

pattern P $s = 3$

skift på 3

1	2	3	4
a	b	a	a

Input: Tekst T af længde n og mønster P af længde m

Output: Alle positioner i T hvor P forekommer

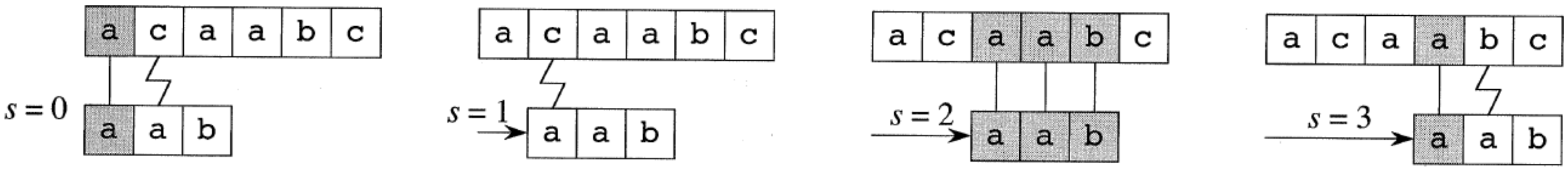
Antal forekomster af *P* = ”aba” i

$$T = \text{”ac}\underline{\overset{3}{\text{a}}\text{ba}}\text{bb}\underline{\overset{8}{\text{a}}\text{b}\overset{10}{\text{a}}\text{ba}}\underline{\overset{13}{\text{a}}\text{ba}}\text{” ?}$$

0%	a)	1	
0%	b)	2	
14%	c)	3	
80%	d)	4	☺
2%	e)	5	
0%	f)	6	
0%	g)	7	
2%	h)	8	
3%	i)	9	
0%	j)	Ved ikke	

Naive Algoritme

```
NAIVE-STRING-MATCHER$(T, P)$
$n = T.length$
$m = P.length$
for $s = 0$ to $n - m$
    if $P[1..m] == T[s+1..s+m]$
        print "Pattern occurs with shift" $s$
```

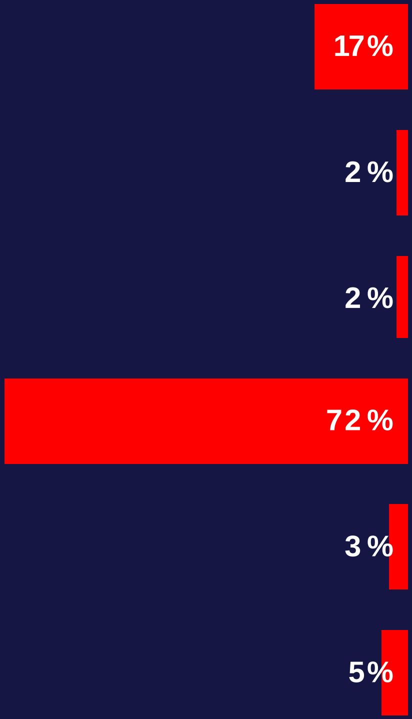


O(n·m)

Naive Algoritme - forventede tid ?

Tekst T = streng af n uniformt tilfældige {0,1}

Mønster P = streng af m tegn fra {0,1}



a) $O(n \cdot m)$

b) $O(n \cdot \log n)$

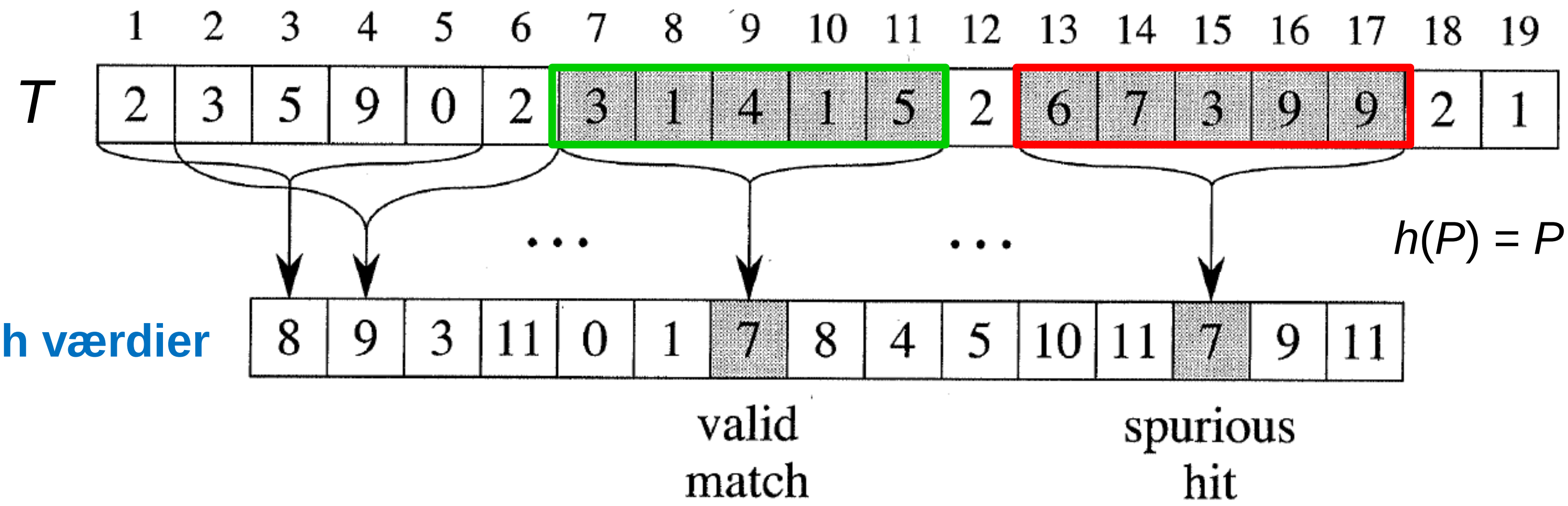
c) $O(m \cdot \log n)$

d) $O(n \cdot \log m)$

e) $O(n+m)$ ☺

f) Ved ikke

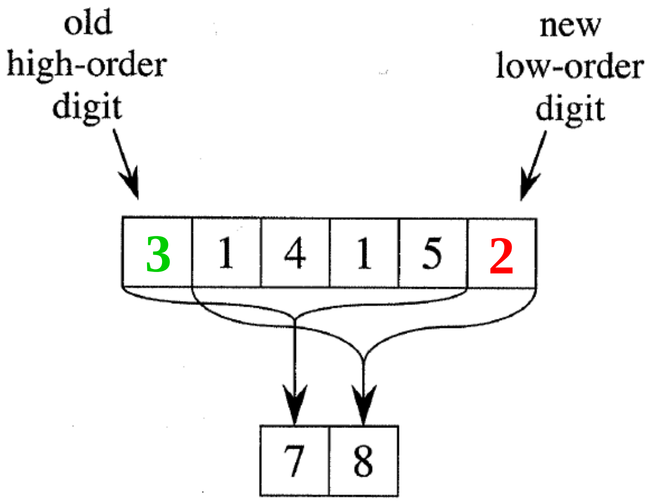
Rabin-Karp : Eksempel *P* = 31415



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
T	2	3	5	9	0	2	3	1	4	1	5	2	6	7	3	9	9	2	1

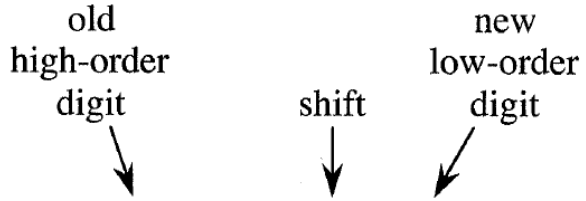
h(*P*) = *P* mod 13

hash værdier	8	9	3	11	0	1	7	8	4	5	10	11	7	9	11



3	1	4	1	5	2

7	8



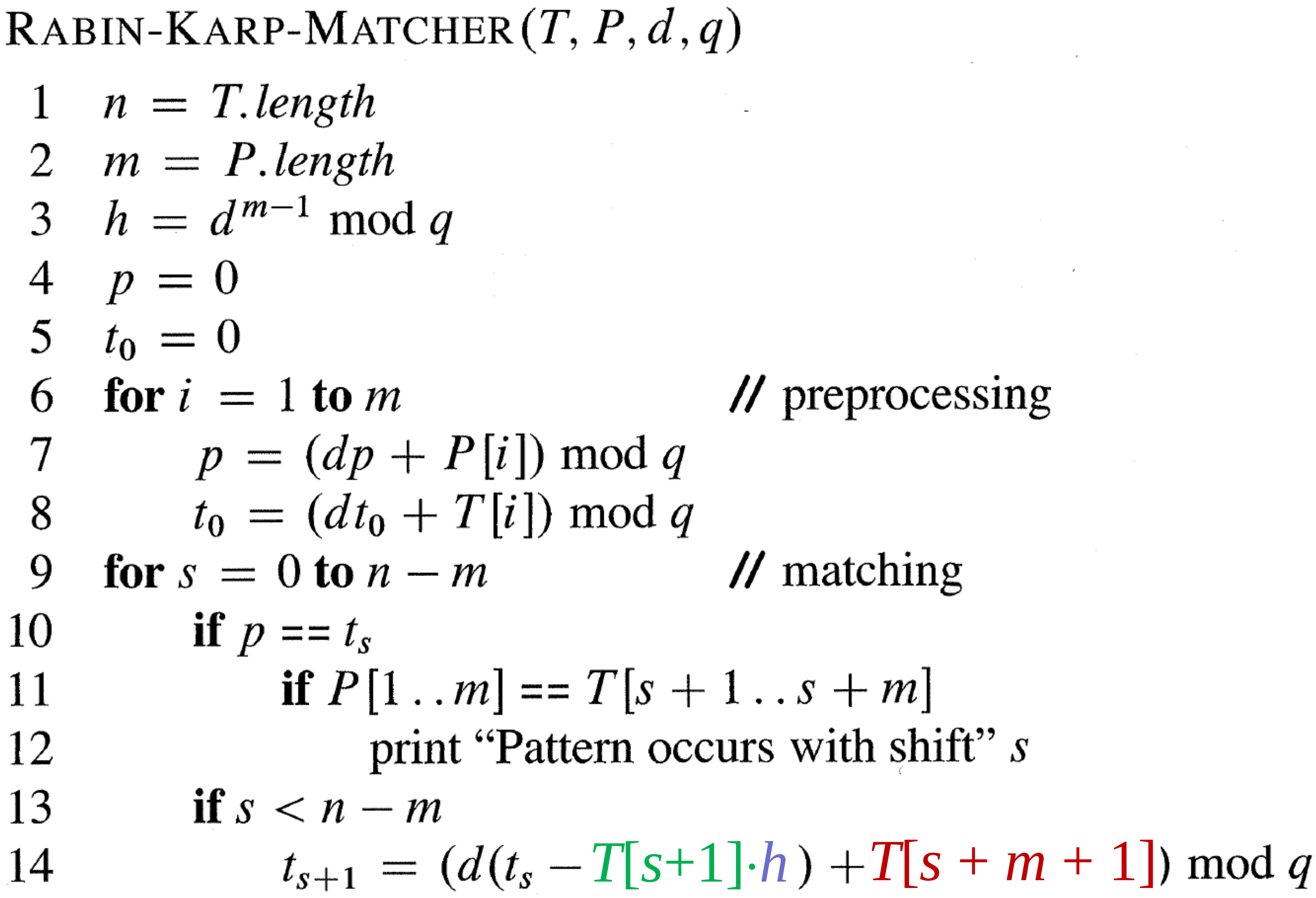
$$\begin{aligned} 14152 &\equiv (31415 - 3 \cdot 10000) \cdot 10 + 2 \pmod{13} \\ &\equiv (7 - 3 \cdot 3) \cdot 10 + 2 \pmod{13} \\ &\equiv 8 \pmod{13} \end{aligned}$$

$(a \cdot b) \bmod p = ((a \bmod p) \cdot b) \bmod p$ $(a+b) \bmod p = ((a \bmod p)+b) \bmod p$

$(a+p \cdot x) \bmod p = a \bmod p$, f.eks. 24 mod 13 = 11 = -2 mod 13

1981

Rabin-Karp



```
RABIN-KARP-MATCHER($T, P, d, q$)
$n = T.length$
$m = P.length$
$h = d^{m-1} \bmod q$
$p = 0$
$t_0 = 0$
for $i = 1$ to $m$                          // preprocessing
    $p = (dp + P[i]) \bmod q$
    $t_0 = (dt_0 + T[i]) \bmod q$
for $s = 0$ to $n - m$                      // matching
    if $p == t_s$
        if $P[1..m] == T[s+1..s+m]$
            print "Pattern occurs with shift" $s$
    if $s < n - m$
        $t_{s+1} = (d(t_s - T[s+1] \cdot h) + T[s+m+1]) \bmod q$
```

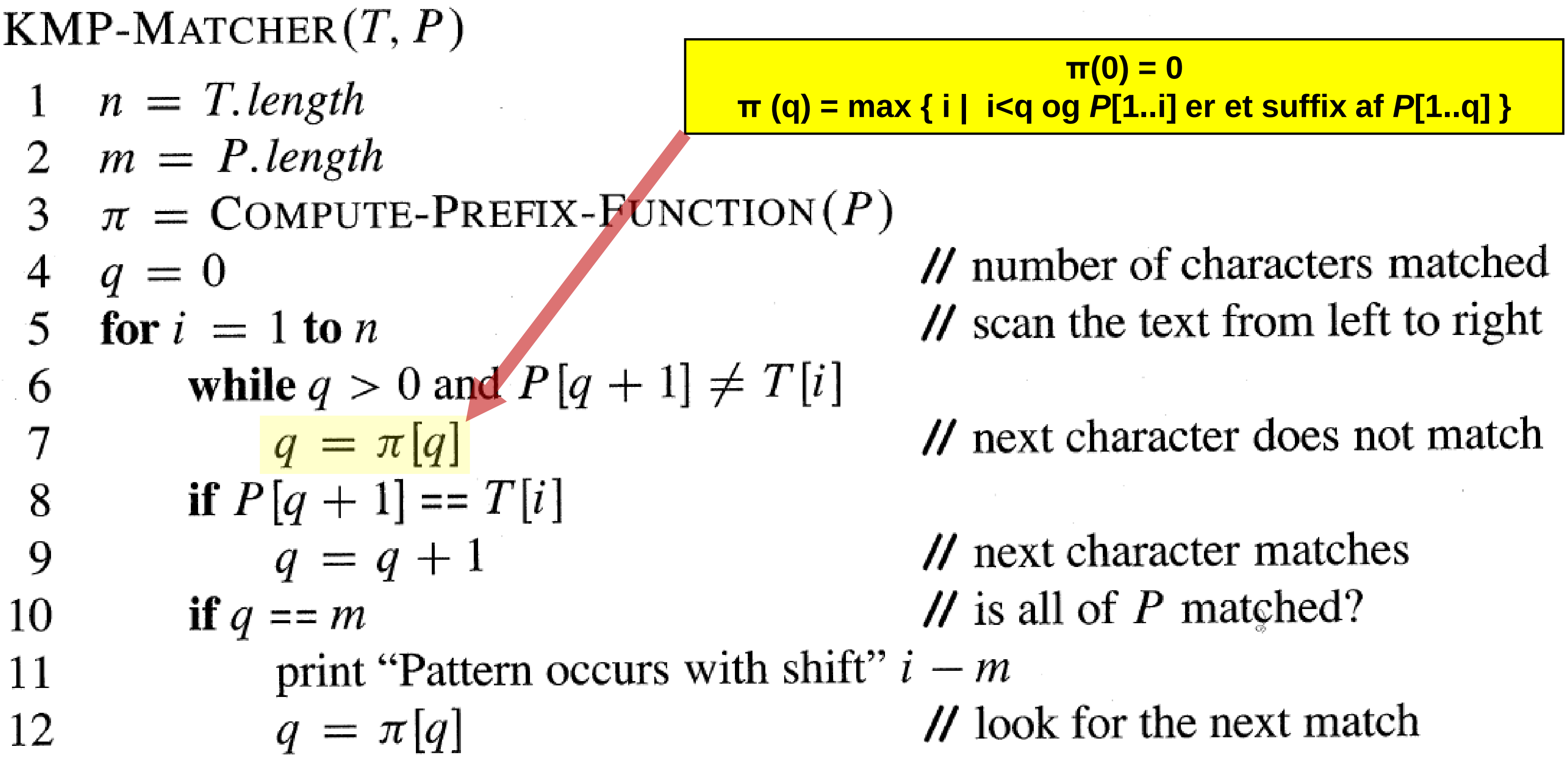
$p = P[1]d^{m-1} + P[2]d^{m-2} + \cdots + P[m-1]d^{1} + P[m]d^{0} \bmod q$

$O(n \cdot m)$

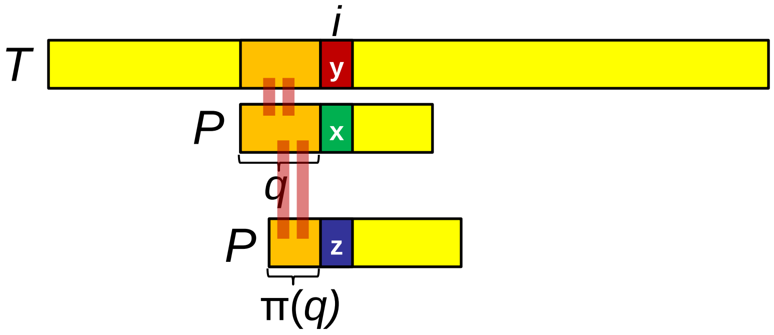
Knuth-Morris-Pratt

1977

π(0) = 0
π (q) = max { i | i<q og *P*[1..i] er et suffix af *P*[1..q] }

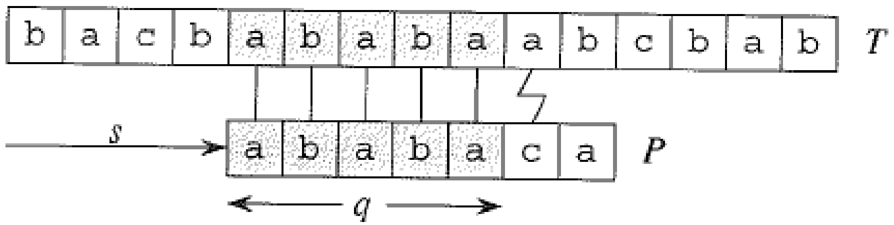


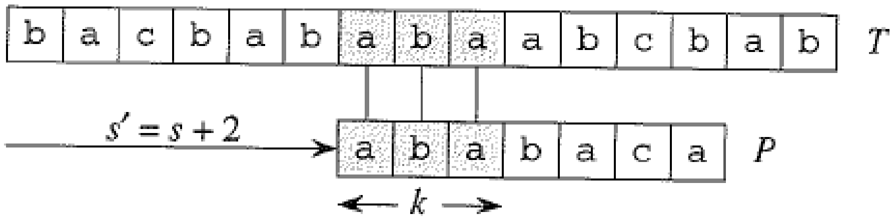
```
KMP-MATCHER(T, P)
n = T.length
m = P.length
π = COMPUTE-PREFIX-FUNCTION(P)
q = 0                                  // number of characters matched
for i = 1 to n                         // scan the text from left to right
    while q > 0 and P[q + 1] ≠ T[i]
        q = π[q]                       // next character does not match
    if P[q + 1] == T[i]
        q = q + 1                      // next character matches
    if q == m                          // is all of P matched?
        print "Pattern occurs with shift" i − m
        q = π[q]                       // look for the next match
```

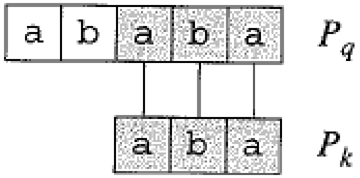


O(*n*)

Knuth-Morris-Pratt: Eksempel







π(0) = 0

π (q) = max { i | i<q og *P*[1..i] er et suffix af *P*[1..q] }

π(7) ?

5%	a)	1
3%	b)	2
41%	c)	3
2%	d)	4
7%	e)	5
5%	f)	6
7%	g)	7
2%	h)	8
0%	i)	9
28%	j)	Ved ikke

7

P = abcbabcdef

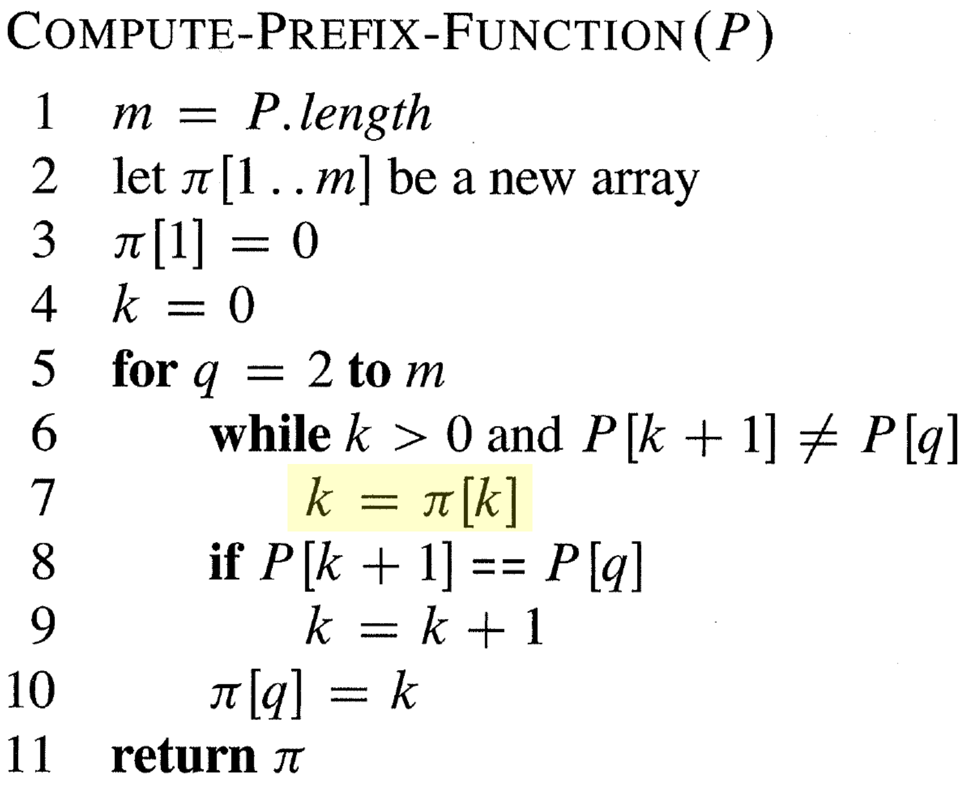
abcbabcdef

3

π(0) = 0

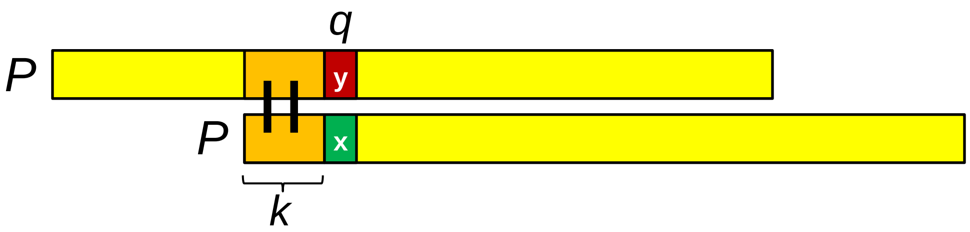
π (q) = max { i | i<q og *P*[1..i] er et suffix af *P*[1..q] }

Knuth-Morris-Pratt: Beregning af prefix funktionen



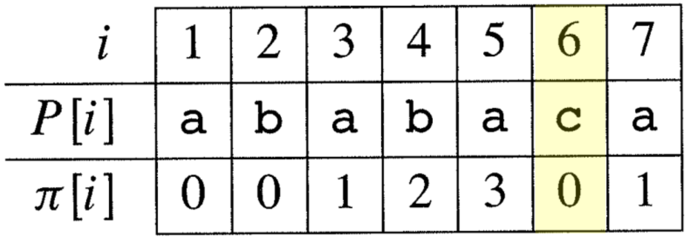
COMPUTE-PREFIX-FUNCTION(P)

```
$m = P.length$
let $\pi[1 \,..\, m]$ be a new array
$\pi[1] = 0$
$k = 0$
for $q = 2$ to $m$
    while $k > 0$ and $P[k+1] \neq P[q]$
        $k = \pi[k]$
    if $P[k+1] == P[q]$
        $k = k + 1$
    $\pi[q] = k$
return $\pi$
```

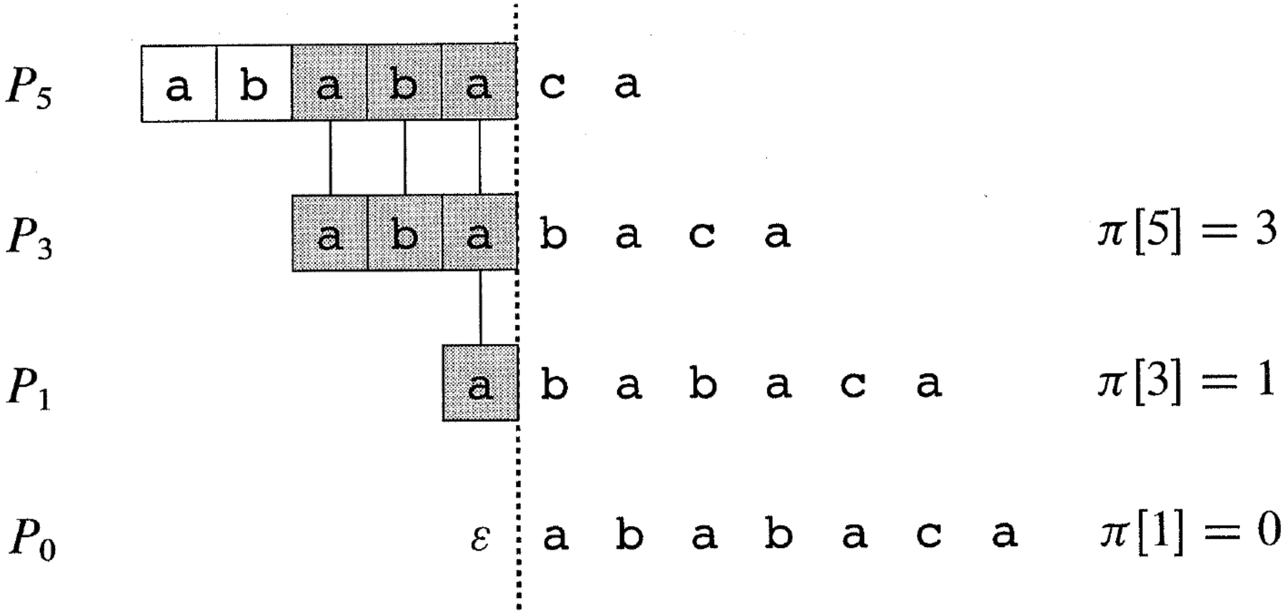


O(m)

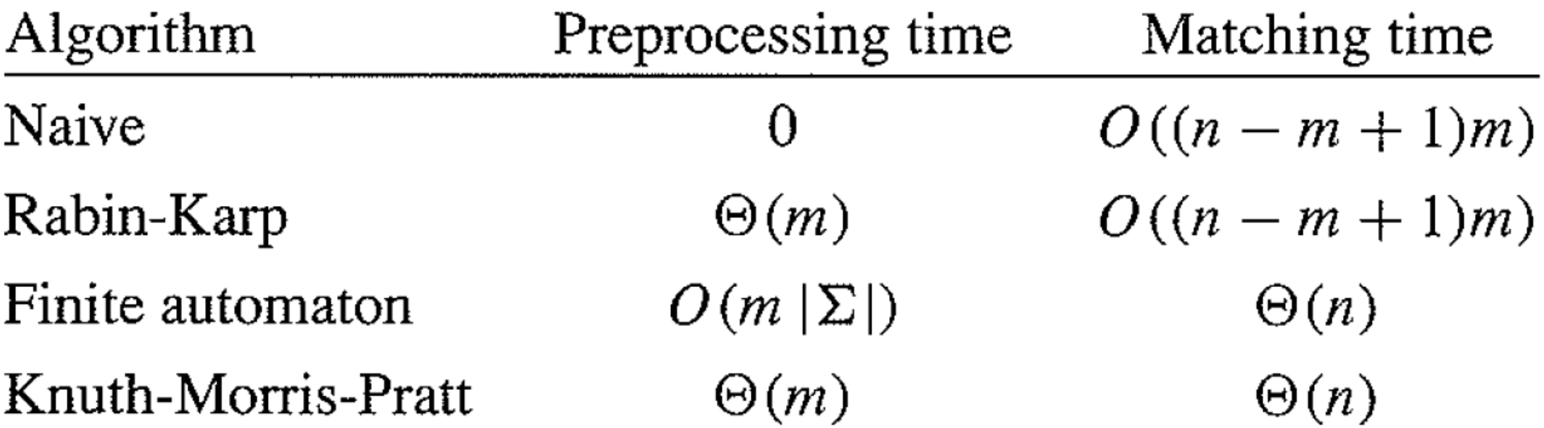
Knuth-Morris-Pratt: Beregning af prefix funktionen



i	1	2	3	4	5	6	7
$P[i]$	a	b	a	b	a	c	a
$\pi[i]$	0	0	1	2	3	0	1



Worst-case tider



Algorithm	Preprocessing time	Matching time	[CLRS]
Naive	0	$O((n-m+1)m)$	32.1
Rabin-Karp	$\Theta(m)$	$O((n-m+1)m)$	32.2
Finite automaton	$O(m\lvert\Sigma\rvert)$	$\Theta(n)$	(32.3)
Knuth-Morris-Pratt	$\Theta(m)$	$\Theta(n)$	32.4