

# 2001–2010: Ten Years of Exact String Matching Algorithms

T. Lecroq  
joint work with S. Faro

University of Rouen, LITIS EA 4108, 76821 Mont-Saint-Aignan Cedex, France

Prague Stringology Conference  
29–31 August 2011 – Prague, Tcheque Republic



# Outline

- 1 Introduction
- 2 Classical solutions
- 3 New efficient solutions
- 4 Experimental framework

# Outline

- 1 Introduction
- 2 Classical solutions
- 3 New efficient solutions
- 4 Experimental framework

# Exact String Matching

## Definition

Find all the occurrences of a pattern  $x$  of length  $m$  in a text  $y$  of length  $n$ .  
 $x, y \in \Sigma^*$

## 2 instances

- $x$  is given first
- $y$  is given first

# Exact String Matching

## Interests

- basic components of many softwares
- theoretical problems

# Exact String Matching

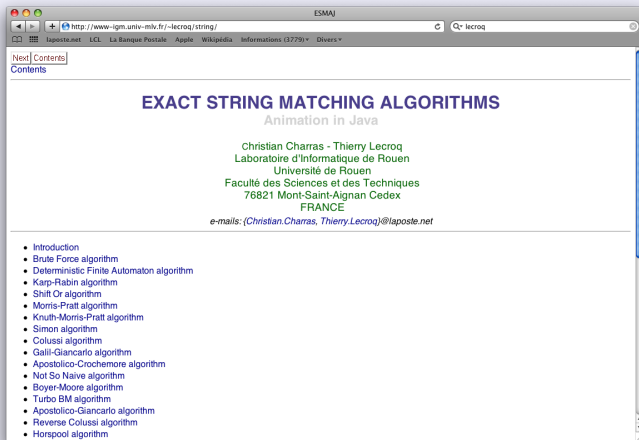
## Theory

- linear time since Morris and Pratt 1970
- linear time and constant space
- $O((n \log m)/m)$  in average [Yao 1979]

# Exact String Matching

## Solutions

Many!! see <http://monge.univ-mlv.fr/~lecroq/string>



The screenshot shows a web browser window with the address bar containing `http://www-igm.univ-mlv.fr/~lecroq/string/`. The page title is "ESMAJ". The browser's address bar also shows "lecroq". The page content includes a navigation bar with "Next" and "Contents" links. The main heading is "EXACT STRING MATCHING ALGORITHMS" in blue, followed by "Animation in Java" in grey. The author information is in green: "Christian Charras - Thierry Lecroq", "Laboratoire d'Informatique de Rouen", "Université de Rouen", "Faculté des Sciences et des Techniques", "76821 Mont-Saint-Aignan Cedex", "FRANCE". The contact email is "e-mails: {Christian.Charras, Thierry.Lecroq}@laposte.net". A list of algorithms is provided at the bottom, including Introduction, Brute Force algorithm, Deterministic Finite Automaton algorithm, Karp-Rabin algorithm, Shift Or algorithm, Morris-Pratt algorithm, Knuth-Morris-Pratt algorithm, Simon algorithm, Colussi algorithm, Gall-Giancarlo algorithm, Apostolico-Crochemore algorithm, Not So Naive algorithm, Boyer-Moore algorithm, Turbo BM algorithm, Apostolico-Giancarlo algorithm, Reverse Colussi algorithm, and Horspool algorithm.

Next Contents

## EXACT STRING MATCHING ALGORITHMS

Animation in Java

Christian Charras - Thierry Lecroq  
Laboratoire d'Informatique de Rouen  
Université de Rouen  
Faculté des Sciences et des Techniques  
76821 Mont-Saint-Aignan Cedex  
FRANCE

e-mails: {Christian.Charras, Thierry.Lecroq}@laposte.net

- Introduction
- Brute Force algorithm
- Deterministic Finite Automaton algorithm
- Karp-Rabin algorithm
- Shift Or algorithm
- Morris-Pratt algorithm
- Knuth-Morris-Pratt algorithm
- Simon algorithm
- Colussi algorithm
- Gall-Giancarlo algorithm
- Apostolico-Crochemore algorithm
- Not So Naive algorithm
- Boyer-Moore algorithm
- Turbo BM algorithm
- Apostolico-Giancarlo algorithm
- Reverse Colussi algorithm
- Horspool algorithm

# Outline

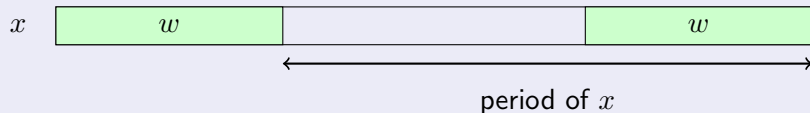
- 1 Introduction
- 2 Classical solutions**
- 3 New efficient solutions
- 4 Experimental framework



# Definitions

## Periods and borders

- $w$  is a border of  $x$  if it is both a prefix and a suffix of  $x$
- $|x| - |w|$  is a period of  $x$

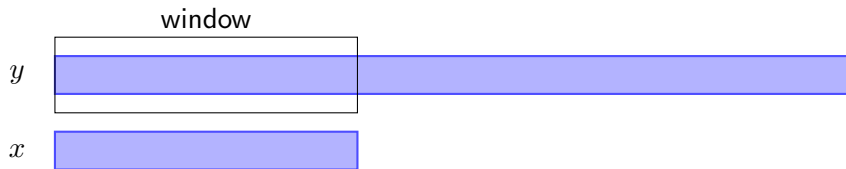


# Exact String Matching

## Classical solutions

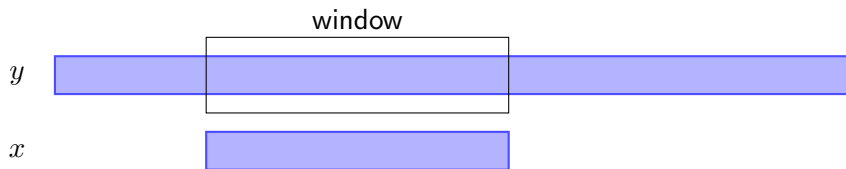
- comparisons
  - ▶ Knuth-Morris-Pratt
  - ▶ Boyer-Moore
- automata
  - ▶ Backward DAWG Matching (with suffix automaton or oracle)
- bit-parallelism
  - ▶ Shift Or
  - ▶ Backward Nondeterministic DAWG Matching
- filtering
  - ▶ Karp-Rabin

# Sliding Window



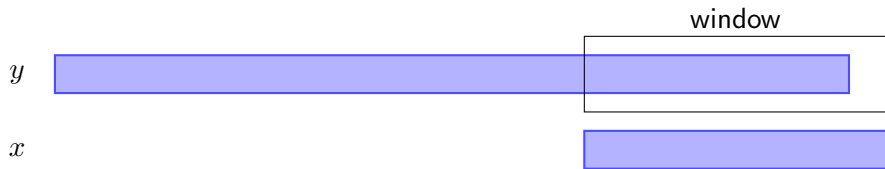
beginning

# Sliding Window



middle

# Sliding Window

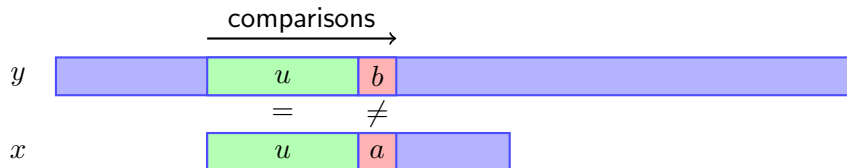


end

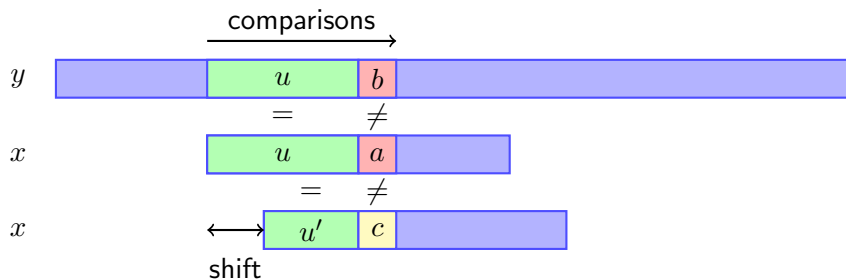
# Knuth-Morris-Pratt (1977)



# Knuth-Morris-Pratt (1977)



# Knuth-Morris-Pratt (1977)



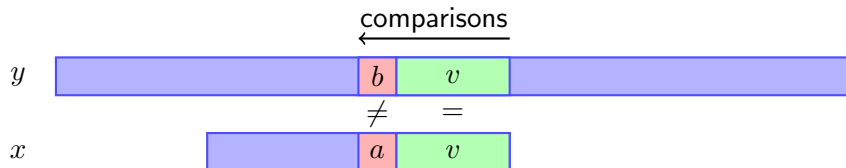
$u'$  is the longest border of  $u$  followed by a letter  $\neq a$



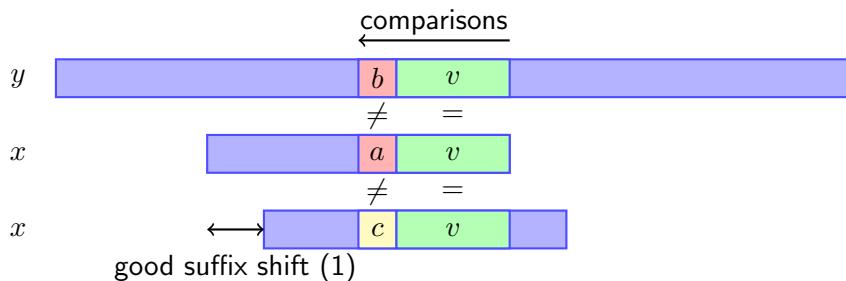
# Boyer-Moore (1977)



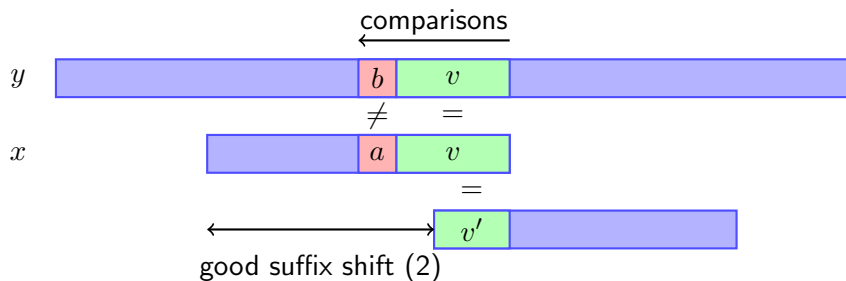
# Boyer-Moore (1977)



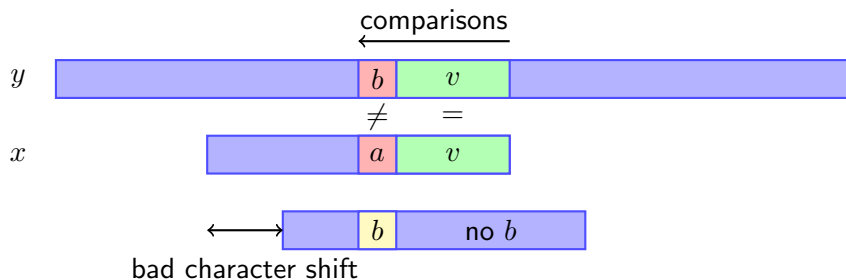
# Boyer-Moore (1977)



# Boyer-Moore (1977)



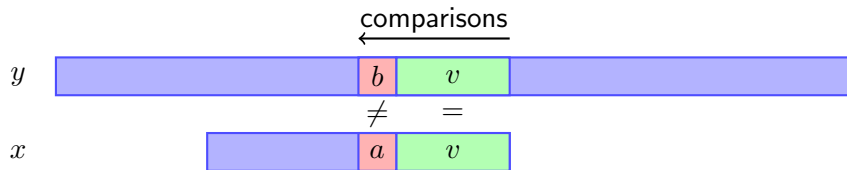
# Boyer-Moore (1977)



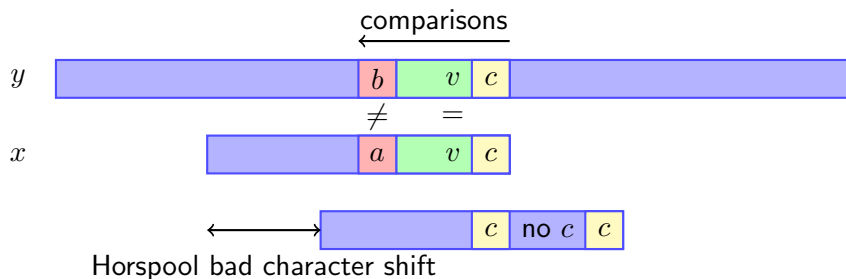
# Horspool (1981)



# Horspool (1981)



# Horspool (1981)

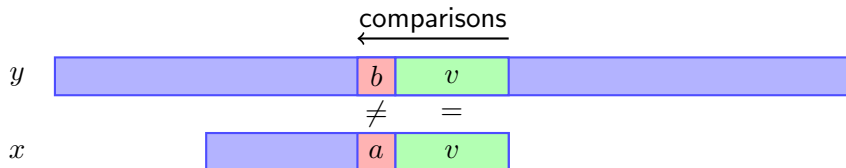




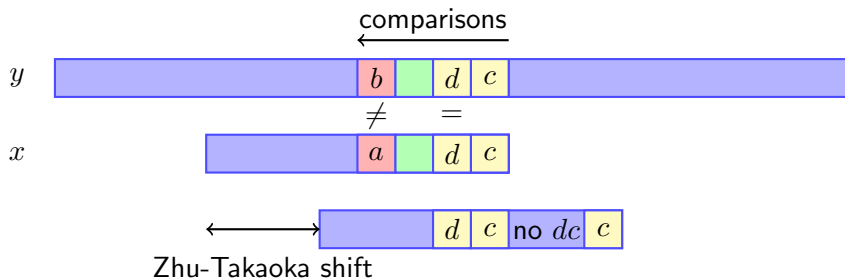
# Zhu-Takaoka (1987)



# Zhu-Takaoka (1987)



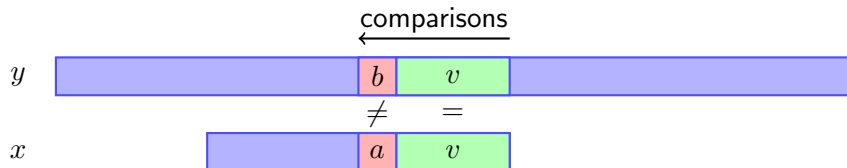
# Zhu-Takaoka (1987)



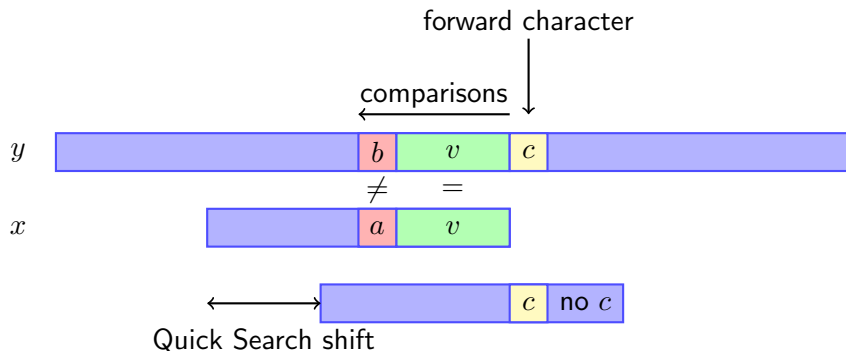
# Quick Search (Sunday, 1990)



# Quick Search (Sunday, 1990)



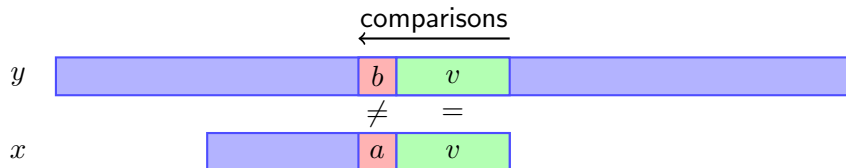
# Quick Search (Sunday, 1990)



# Berry-Ravindran (1999)

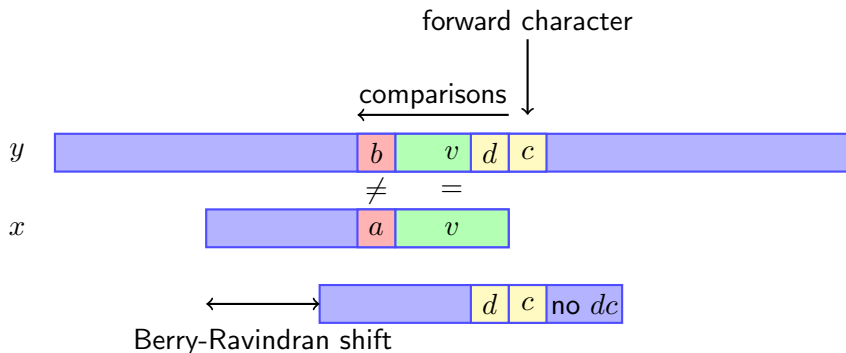


# Berry-Ravindran (1999)





# Berry-Ravindran (1999)



# Backward DAWG Matching (Crochemore & Rytter, 1994)

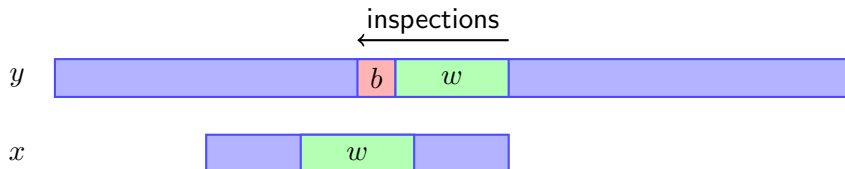
$y$



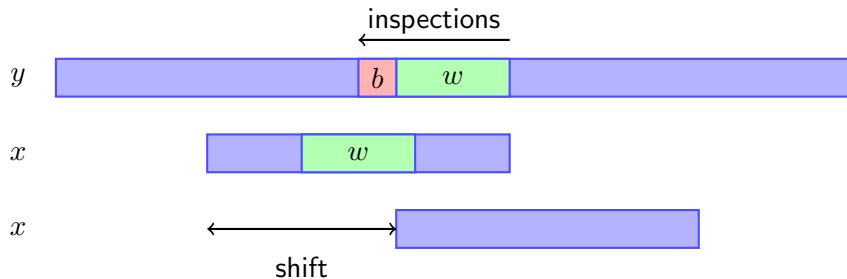
$x$



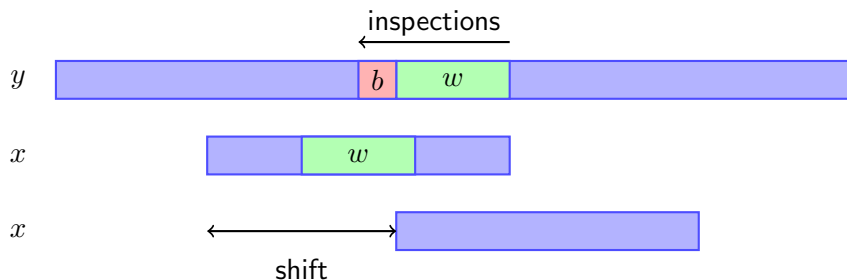
# Backward DAWG Matching (Crochemore & Rytter, 1994)



# Backward DAWG Matching (Crochemore & Rytter, 1994)



# Backward DAWG Matching (Crochemore & Rytter, 1994)

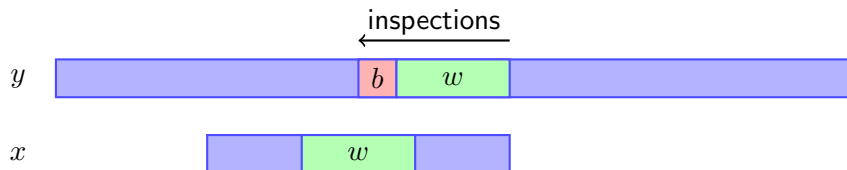


with Factor Oracle: BOM  
(Backward Oracle Matching, Allauzen, Crochemore & Raffinot, 1999)

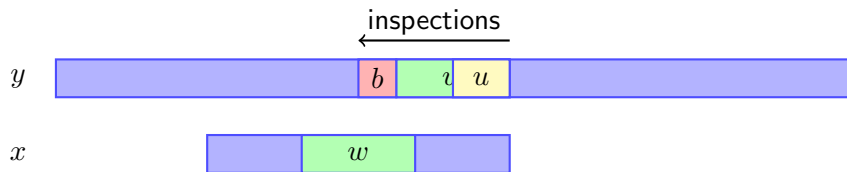
# Reverse Factor (Lecroq, 1992)



# Reverse Factor (Lecroq, 1992)

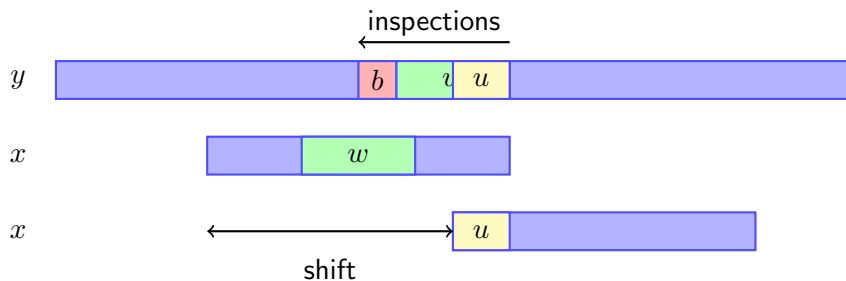


# Reverse Factor (Lecroq, 1992)

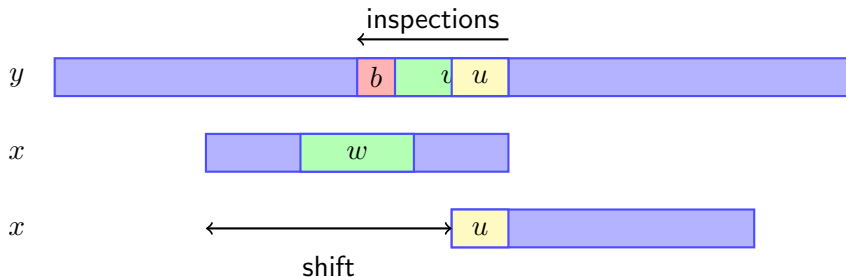




# Reverse Factor (Lecroq, 1992)



# Reverse Factor (Lecroq, 1992)



with Factor Oracle: BSOM

(Backward Suffix Oracle Matching, Allauzen, Crochemore & Raffinot, 1999)

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$     C A T A

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A
$S_A$	1	0	1	0
$S_C$	0	1	1	1
$S_G$	1	1	1	1
$S_T$	1	1	0	1

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0								
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	1	1		
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	1	1		
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	1	1		
$S_C$	0	1	1	1			0	1	1	1		
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

Shift



# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	1	1		
$S_C$	0	1	1	1			0	1	1	1		
$S_G$	1	1	1	1			0	1	1	1		
$S_T$	1	1	0	1								

Shift  
Or with  $S_C$

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	1	1		
$S_C$	0	1	1	1			0	1	1	1		
$S_G$	1	1	1	1			0	1	1	1		
$S_T$	1	1	0	1			0	1	1	1		

Shift  
Or with  $S_C$   
=

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1			0	0	1	1		
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

Shift

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1			0	0	1	1		
$S_G$	1	1	1	1			0	1	1	1		
$S_T$	1	1	0	1								

Shift  
Or with  $S_C$

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1			0	0	1	1		
$S_G$	1	1	1	1			0	1	1	1		
$S_T$	1	1	0	1			0	1	1	1		

Shift  
Or with  $S_C$   
=

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1			0	0	1	1		
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

Shift



# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1			0	0	1	1		
$S_G$	1	1	1	1			1	0	1	0		
$S_T$	1	1	0	1								

Shift  
Or with  $S_C$

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			0	1	1	1		
$S_C$	0	1	1	1			0	0	1	1		
$S_G$	1	1	1	1			1	0	1	0		
$S_T$	1	1	0	1			1	0	1	1		

Shift  
Or with  $S_C$   
=

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	0	1	1		
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	0	1	1		
$S_C$	0	1	1	1			0	1	0	1		
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

Shift

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	0	1	1		
$S_C$	0	1	1	1			0	1	0	1		
$S_G$	1	1	1	1			1	1	0	1		
$S_T$	1	1	0	1								

Shift  
Or with  $S_C$

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	0	1	1		
$S_C$	0	1	1	1			0	1	0	1		
$S_G$	1	1	1	1			1	1	0	1		
$S_T$	1	1	0	1			1	1	0	1		

Shift  
Or with  $S_C$   
=

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0				1	1	0	1	
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	0	1		
$S_C$	0	1	1	1			0	1	1	0		
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

Shift



# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	0	1		
$S_C$	0	1	1	1			0	1	1	0		
$S_G$	1	1	1	1			1	0	1	0		
$S_T$	1	1	0	1								

Shift  
Or with  $S_C$

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0			1	1	0	1		
$S_C$	0	1	1	1			0	1	1	0		
$S_G$	1	1	1	1			1	0	1	0		
$S_T$	1	1	0	1			1	1	1	0		

Shift  
Or with  $S_C$   
=

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0					1	1	1	0
$S_C$	0	1	1	1								
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0					1	1	1	0
$S_C$	0	1	1	1					0	1	1	1
$S_G$	1	1	1	1								
$S_T$	1	1	0	1								

Shift

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0					1	1	1	0
$S_C$	0	1	1	1					0	1	1	1
$S_G$	1	1	1	1					0	1	1	1
$S_T$	1	1	0	1								

Shift  
Or with  $S_C$

# Shift Or (Baeza-Yates & Gonnet 1992)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	1	0	1	0					1	1	1	0
$S_C$	0	1	1	1					0	1	1	1
$S_G$	1	1	1	1					0	1	1	1
$S_T$	1	1	0	1					0	1	1	1

Shift  
Or with  $S_C$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$     C A T A

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A
$S_A$	0	1	0	1
$S_C$	1	0	0	0
$S_G$	0	0	0	0
$S_T$	0	0	1	0



# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1								
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0			0	0	1	0		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_T$

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0			0	0	1	0		
$S_G$	0	0	0	0			0	0	1	0		
$S_T$	0	0	1	0								

And with  $S_T$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0			0	0	1	0		
$S_G$	0	0	0	0			0	0	1	0		
$S_T$	0	0	1	0			0	1	0	0		

And with  $S_T$   
=  
Shift

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_A$



# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0			0	1	0	0		
$S_T$	0	0	1	0								

And with  $S_A$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0			0	1	0	0		
$S_T$	0	0	1	0			1	0	0	0		

And with  $S_A$   
=  
Shift

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_C$

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0			1	0	0	0		
$S_T$	0	0	1	0								

And with  $S_C$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0			1	0	0	0		
$S_T$	0	0	1	0			0	0	0	0		

And with  $S_C$   
=  
Shit

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	0	0	0		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_C$



# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0			0	0	0	0		
$S_T$	0	0	1	0								

And with  $S_C$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_A$

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0			0	1	0	1		
$S_T$	0	0	1	0								

And with  $S_A$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	1	1	1		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0			0	1	0	1		
$S_T$	0	0	1	0			1	0	1	0		

And with  $S_A$   
=  
Shift

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	1	0		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	1	0		
$S_C$	1	0	0	0			0	0	1	0		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_T$

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	1	0		
$S_C$	1	0	0	0			0	0	1	0		
$S_G$	0	0	0	0			0	0	1	0		
$S_T$	0	0	1	0								

And with  $S_T$   
=



# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	1	0		
$S_C$	1	0	0	0			0	0	1	0		
$S_G$	0	0	0	0			0	0	1	0		
$S_T$	0	0	1	0			0	1	0	0		

And with  $S_T$   
=  
Shift

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_A$

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0			0	1	0	0		
$S_T$	0	0	1	0								

And with  $S_A$   
=

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			0	1	0	0		
$S_C$	1	0	0	0			0	1	0	1		
$S_G$	0	0	0	0			0	1	0	0		
$S_T$	0	0	1	0			1	0	0	0		

And with  $S_A$   
=  
Shift

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0								
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0								
$S_T$	0	0	1	0								

And with  $S_C$

# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0			1	0	0	0		
$S_T$	0	0	1	0								

And with  $S_C$   
=



# BNDM (Navarro & Raffinot, 1998)

$x$	C	A	T	A		$y$	C	C	A	T	A	C
$S_A$	0	1	0	1			1	0	0	0		
$S_C$	1	0	0	0			1	0	0	0		
$S_G$	0	0	0	0			1	0	0	0		
$S_T$	0	0	1	0								

And with  $S_C$   
=

# SO and BNDM

Easily adapt to approximate string matching

# Karp-Rabin (1987)

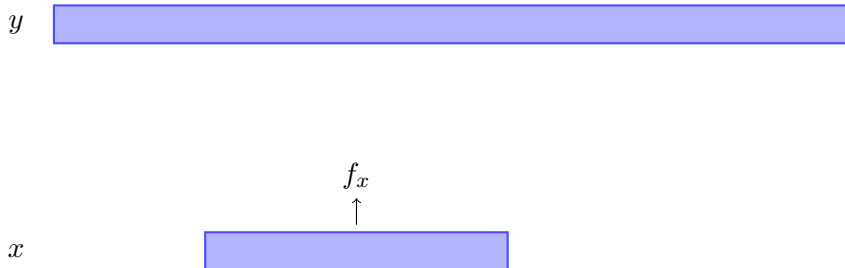
$y$



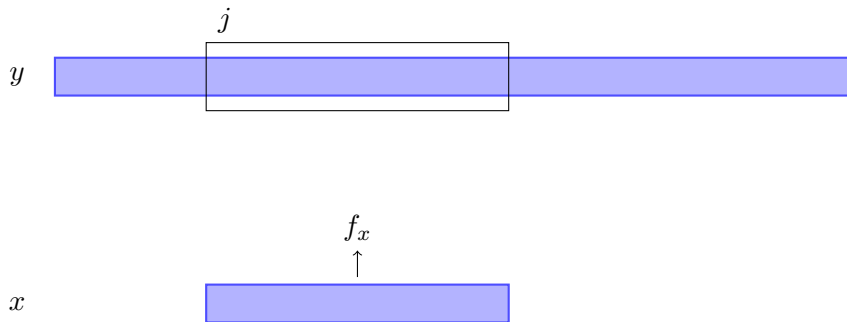
$x$



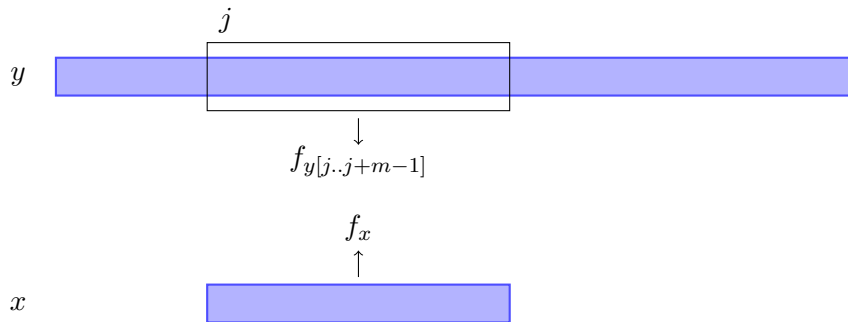
# Karp-Rabin (1987)



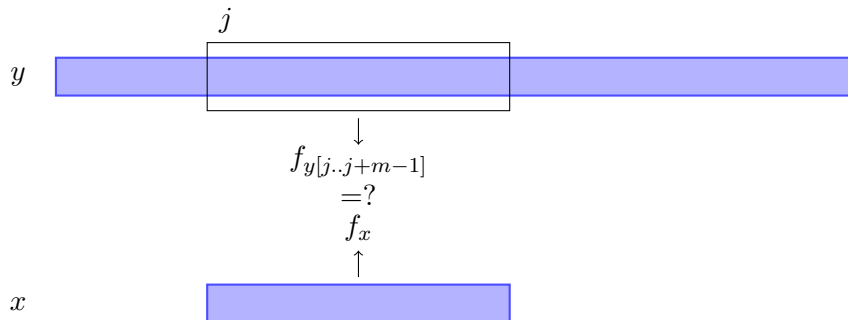
# Karp-Rabin (1987)



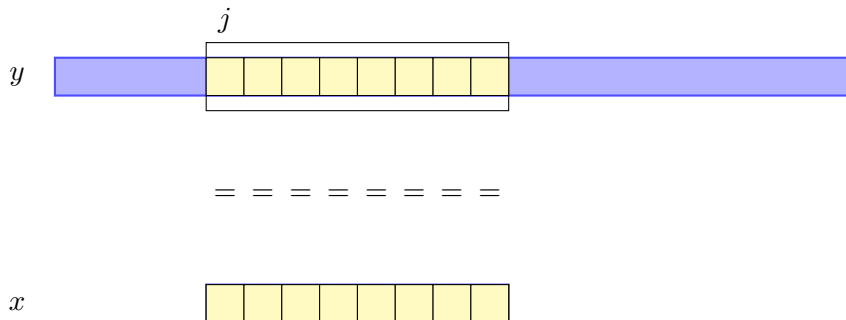
# Karp-Rabin (1987)



# Karp-Rabin (1987)

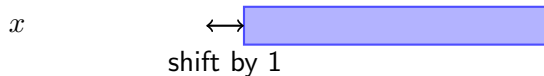
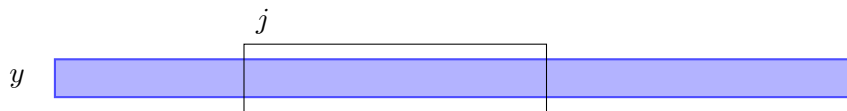


# Karp-Rabin (1987)





# Karp-Rabin (1987)



# Usual tricks

- sentinel: append  $x$  at the end of  $y$
- fast loop: locate first one character of  $x$

# Usual algorithm

## $SM(x, m, y, n)$

- 1 preprocessing on  $x$
- 2 position the window at the left end of  $y$
- 3 **while** the window is still on  $y$  **do**
- 4     **if**  $x =$  content of the window **then**
- 5         OUTPUT(1 occurrence)
- 6     shift the window

# Sentinel

## SM( $x, m, y, n$ )

```
1 preprocessing on  $x$ 
2 append  $x$  at the end of  $y$ 
3 position the window at the left end of  $y$ 
4 while true do
5     if  $x$  = content of the window then
6         if the window is still on  $y$  then
7             OUTPUT(1 occurrence)
8         else EXIT
9     shift the window
```

# Fast loop

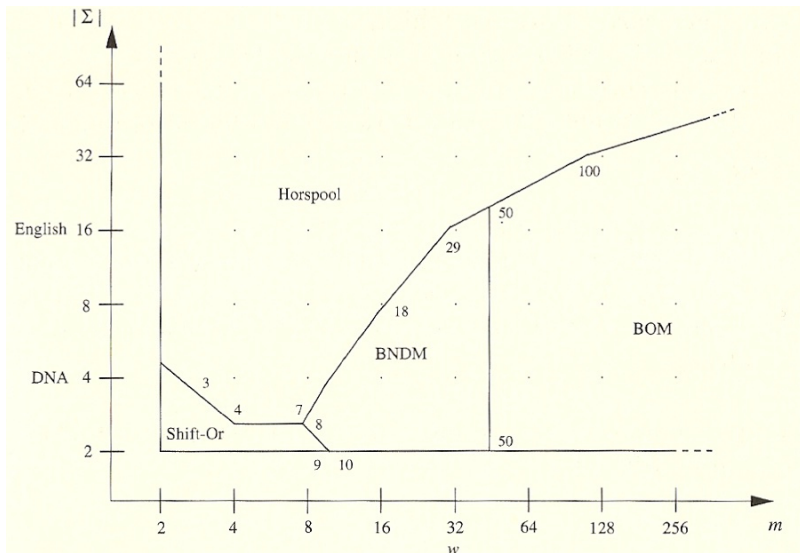
$SM(x, m, y, n)$

```
1 preprocessing on  $x$ 
2 append  $x$  at the end of  $y$ 
3 position the window at the left end of  $y$ 
4  $c \leftarrow$  1 character of  $x$ 
5 while true do
6   while  $c$  is not in the window do
7     shift the window
8   if  $x =$  content of the window then
9     if the window is still on  $y$  then
10      OUTPUT(1 occurrence)
11     else EXIT
12   shift the window
```

# Outline

- 1 Introduction
- 2 Classical solutions
- 3 New efficient solutions**
- 4 Experimental framework

# The XXth Century



G. Navarro & M. Raffinot, *Flexible Pattern Matching in Strings*, CUP, 2002

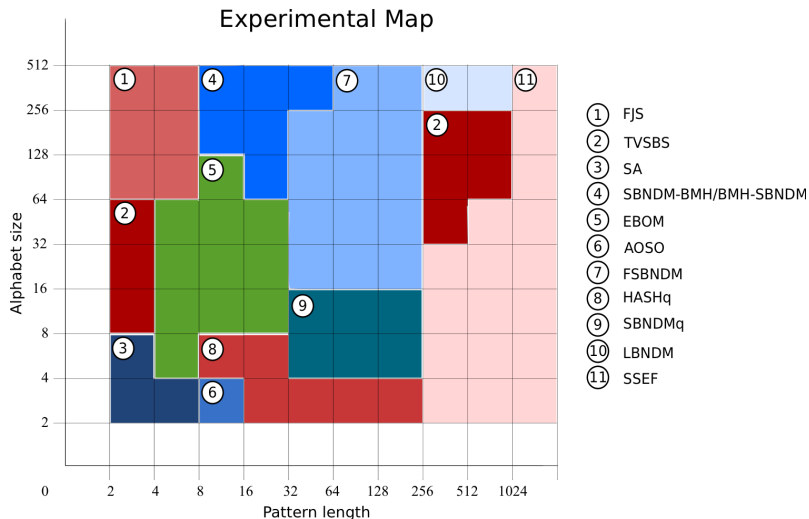
# The XXIst Century

## More than 50 algorithms

- Variants of:
  - ▶ BM
  - ▶ BOM
  - ▶ BNDM
- 2 windows
- 2 automata
- large window
- sample the pattern



# The XX1st Century



S. Faro & T. Lecroq, *The Exact String Matching Problem: a Comprehensive Experimental Evaluation*, arXiv:1012.2547, 2010

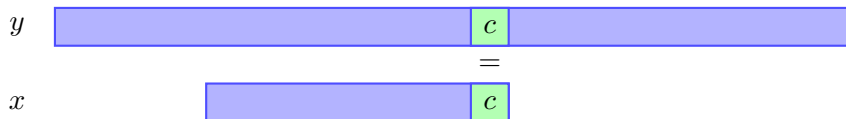
## FJS (Franek-Jennings-Smyth, 2007)

- uses KMP and QS
- $O(m + \sigma)$  time for preprocessing phase
- $3n - 2m$  character comparisons during the searching phase

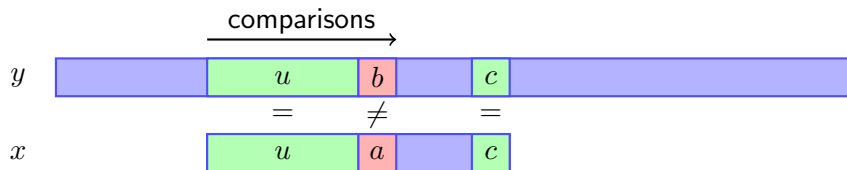
# FJS (Franek-Jennings-Smyth, 2007)



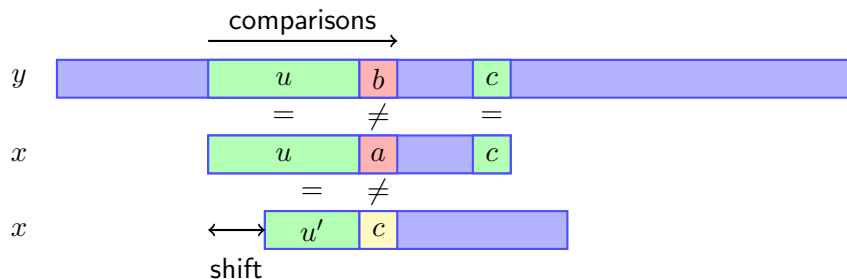
# FJS (Franek-Jennings-Smyth, 2007)



# FJS (Franek-Jennings-Smyth, 2007)



# FJS (Franek-Jennings-Smyth, 2007)



$u'$  is the longest border of  $u$  followed by a letter  $\neq a$

# TVSBS

(Thathoo-Virmani-Lakshmi-Balakrishnan-Sekar, 2006)

- compares first the rightmost character of the window then the leftmost, then all the others
- uses the Berry-Ravindran bad character shift
- $O(m + \sigma^2)$  time for the preprocessing phase
- $O(nm)$  time for the searching phase

# TVSBS

(Thathoo-Virmani-Lakshmi-Balakrishnan-Sekar,  
2006)

$y$



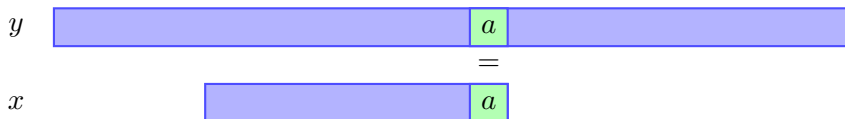
$x$





# TVSBS

(Thathoo-Virmani-Lakshmi-Balakrishnan-Sekar,  
2006)



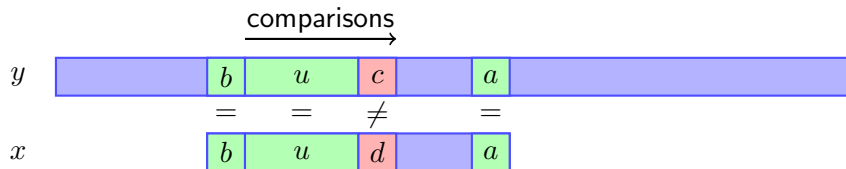
# TVSBS

(Thathoo-Virmani-Lakshmi-Balakrishnan-Sekar,  
2006)



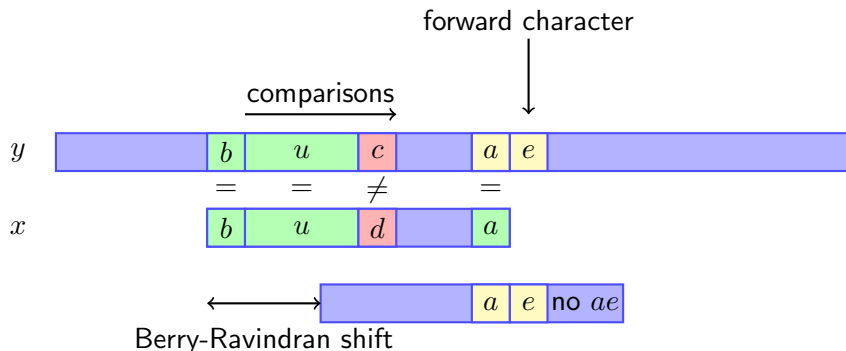
# TVSBS

(Thathoo-Virmani-Lakshmi-Balakrishnan-Sekar,  
2006)



# TVSBS

(Thathoo-Virmani-Lakshmi-Balakrishnan-Sekar, 2006)



## Shift-And (Wu & Manber, 1992)

- Shift-Or with 1 for match and 0 for mismatch

# SBNDM-BMH and BMH-SBNDM (Holub & Durian, 2005)

- alternates SBNDM and Horspool strategies

## Extended BOM (Faro & Lecroq, 2008)

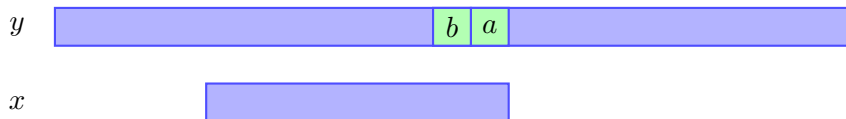
- fast loop with 2 transitions in the factor oracle with a 2-dimensional table
- $O(\sigma^2)$  additional space
- $O(mn)$  worst case time searching phase

# Extended BOM (Faro & Lecroq, 2008)

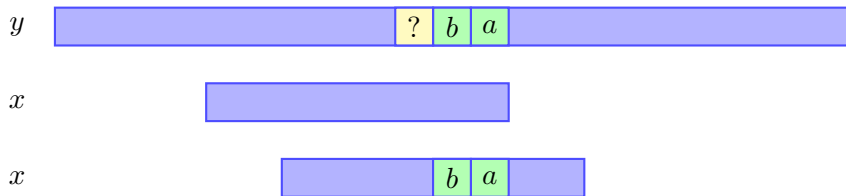




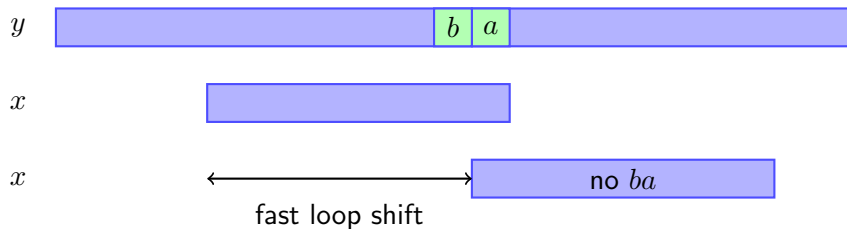
# Extended BOM (Faro & Lecroq, 2008)



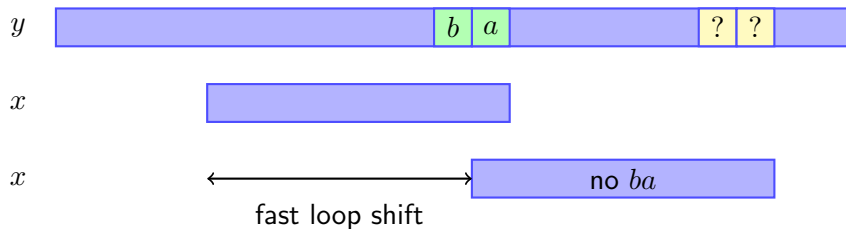
# Extended BOM (Faro & Lecroq, 2008)



# Extended BOM (Faro & Lecroq, 2008)



# Extended BOM (Faro & Lecroq, 2008)



## AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

- considers a set of  $q$  patterns  $X = \{x^0, x^1, \dots, x^{q-1}\}$  where  $|x^j| = \lfloor m/q \rfloor$  and  $x^j[i] = x[j + iq]$
- construct a new pattern  $x' = x^0x^1 \dots x^{q-1}$
- search for  $x'$  in  $y$  using the Shift-Or algorithm by considering every  $q$ -th character
- if a  $x^j$  is found  $x$  is naively checked

# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

$x$



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

$x$



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

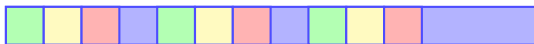
$x$





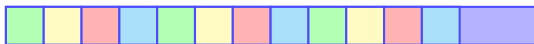
# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

$x$

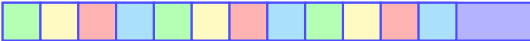


# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

$x$

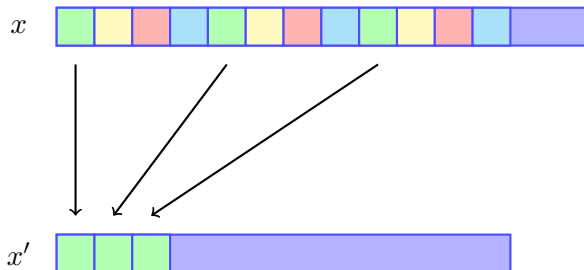


# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

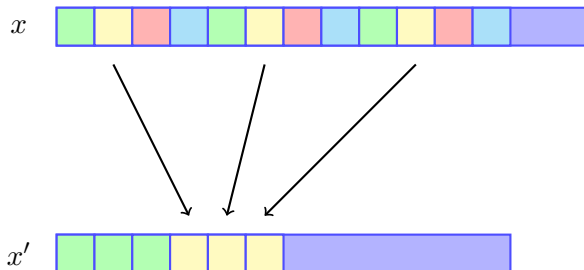
$x$  

$x'$  

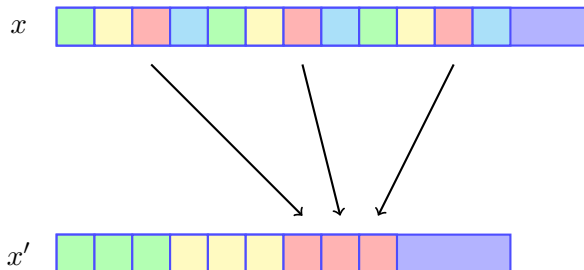
# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



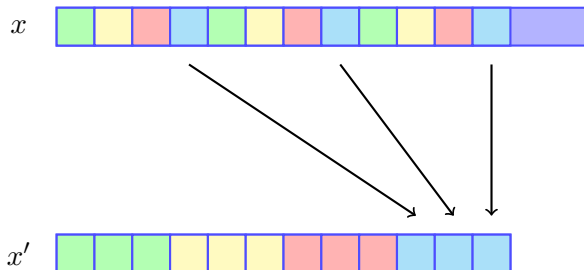
# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

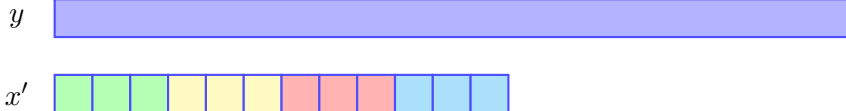
$y$



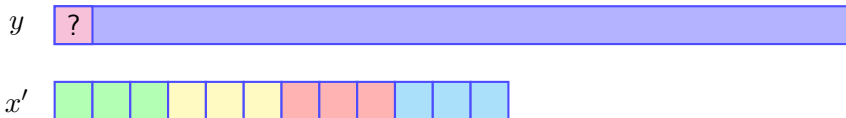
$x'$



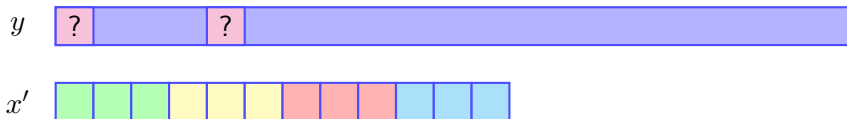
# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)



# AOSO (Average Optimal Shift Or, Fredriksson & Grabowski, 2005)

- with  $q = O(m/\log_{\sigma} m)$  average time is  $O((n \log_{\sigma} m)/m)$



## Hashing $q$ -grams (Lecroq, 2007)

- Compute a hash value in  $[0; 255]$  for every  $q$ -grams of the pattern  $x$
- Compute a shift for every hash value
- Unroll the loops as much as possible

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	<b>c</b>	<b>a</b>	<b>t</b>	<b>a</b>	<b>c</b>	<b>a</b>	<b>t</b>	<b>a</b>	<b>a</b>	<b>a</b>	<b>t</b>	<b>a</b>

$shift[i] \leftarrow 10$

$\forall i \in [0; 255]$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	<b>c</b>	<b>a</b>	<b>t</b>	<b>a</b>	<b>c</b>	<b>a</b>	<b>t</b>	<b>a</b>	<b>a</b>	<b>a</b>	<b>t</b>	<b>a</b>

$$h(\mathbf{cat}) = ((\mathit{rank}(\mathbf{c}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{t})) = 194$$

$$\mathit{shift}[194] = 10$$

$$\mathit{shift}[194] \leftarrow 9$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{ata}) = ((\mathit{rank}(\mathbf{a}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{a})) = 205$$

$$\mathit{shift}[205] = 10$$

$$\mathit{shift}[205] \leftarrow 8$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{tac}) = ((\mathit{rank}(\mathbf{t}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{c})) = 245$$

$$\mathit{shift}[245] = 10$$

$$\mathit{shift}[245] \leftarrow 7$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{aca}) = ((\mathit{rank}(\mathbf{a}) \times 2 + \mathit{rank}(\mathbf{c})) \times 2 + \mathit{rank}(\mathbf{a})) = 171$$

$$\mathit{shift}[171] = 10$$

$$\mathit{shift}[171] \leftarrow 6$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	<b>c</b>	<b>a</b>	<b>t</b>	<b>a</b>	<b>c</b>	<b>a</b>	<b>t</b>	<b>a</b>	<b>a</b>	<b>a</b>	<b>t</b>	<b>a</b>

$$h(\mathbf{cat}) = ((\mathit{rank}(\mathbf{c}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{t})) = 194$$

$$\mathit{shift}[194] = 9$$

$$\mathit{shift}[194] \leftarrow 5$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{ata}) = ((\mathit{rank}(\mathbf{a}) \times 2 + \mathit{rank}(\mathbf{t})) \times 2 + \mathit{rank}(\mathbf{a})) = 205$$

$$\mathit{shift}[205] = 8$$

$$\mathit{shift}[205] \leftarrow 4$$



# Hashing $q$ -grams (Lecroq, 2007)

## Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{taa}) = ((\mathit{rank}(\mathbf{t}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{a})) = 243$$

$$\mathit{shift}[243] = 10$$

$$\mathit{shift}[243] \leftarrow 3$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{aaa}) = ((\mathit{rank}(\mathbf{a}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{a})) = 167$$

$$\mathit{shift}[167] = 10$$

$$\mathit{shift}[167] \leftarrow 2$$

## Hashing $q$ -grams (Lecroq, 2007)

### Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{aat}) = ((\mathit{rank}(\mathbf{a}) \times 2 + \mathit{rank}(\mathbf{a})) \times 2 + \mathit{rank}(\mathbf{t})) = 186$$

$$\mathit{shift}[186] = 10$$

$$\mathit{shift}[186] \leftarrow 1$$

# Hashing $q$ -grams (Lecroq, 2007)

## Example

$i$	0	1	2	3	4	5	6	7	8	9	10	11
$x[i]$	c	a	t	a	c	a	t	a	a	a	t	a

$$h(\mathbf{ata}) = ((\mathit{rank}(\mathbf{a}) \times 2 + \mathit{rank}(\mathbf{t})) \times 2 + \mathit{rank}(\mathbf{a})) = 205$$

$$\mathit{shift}[205] = 4 \implies \mathit{sh1} \leftarrow 4$$

$$\mathit{shift}[205] \leftarrow 0$$

# Hashing $q$ -grams

**Algorithm**  $\text{Hash}_q(x, m, y, n)$  **for**  $q = 3$

▷ Preprocessing

**for**  $i \leftarrow 0$  **to** 255 **do**  $\text{shift}[i] \leftarrow m - 2$

**for**  $i \leftarrow 2$  **to**  $m - 2$  **do**

$h \leftarrow ((x[i - 2] \times 2 + x[i - 1]) \times 2) + x[i]$

$\text{shift}[h \bmod 256] \leftarrow m - 1 - i$

$h \leftarrow ((x[m - 3] \times 2 + x[m - 2]) \times 2) + x[m - 1]$

$\text{sh1} \leftarrow \text{shift}[h \bmod 256]$

$\text{shift}[h \bmod 256] \leftarrow 0$

# Hashing $q$ -grams

**Algorithm Hash $q(x, m, y, n)$  for  $q = 3$**

▷ Searching

$y[n..n + m - 1] \leftarrow x$      ▷ Sentinel

$j \leftarrow m - 1$

**while** TRUE **do**

$sh \leftarrow 1$

**while**  $sh \neq 0$  **do**

$h \leftarrow ((y[j - 2] \times 2 + y[j - 1]) \times 2) + y[j]$

$sh \leftarrow \text{shift}[h \bmod 256]$

$j \leftarrow j + sh$

**if**  $j < n$  **then**

**if**  $x = y[j - m + 1..j]$  **then** OUTPUT( $j - m + 1$ )

$j \leftarrow j + sh1$

**else** RETURN

# FSBNDM (Forward Simple BNDM, Faro & Lecroq, 2008)

- incorporate the forward character at each attempt

## SBNDM<sub>q</sub> (Simple BNDM with $q$ grams, Durian, Holub, Peltola & Tarhio, 2009)

- Simple BNDM that considers consecutive  $q$ -grams
- shifts of length  $m - q + 1$  if the  $q$ -gram is not present



## LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)

- partition  $x = x^0 x^1 \dots x^{\lfloor m/k \rfloor - 1}$  where  $|x^j| = k = \lfloor (m-1)/\omega \rfloor + 1$
- construct a new pattern  $x'$  where  $x'[j]$  is the set of characters of  $x^j$
- search for  $x'$  in  $y$  using the BNDM algorithm by considering every  $k$ -th character

# LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)

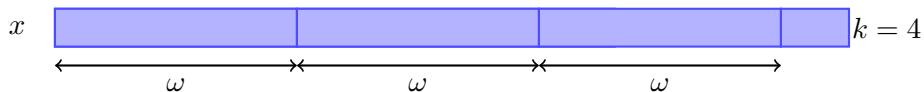
$x$



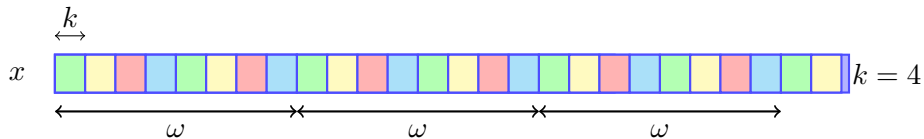
# LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)



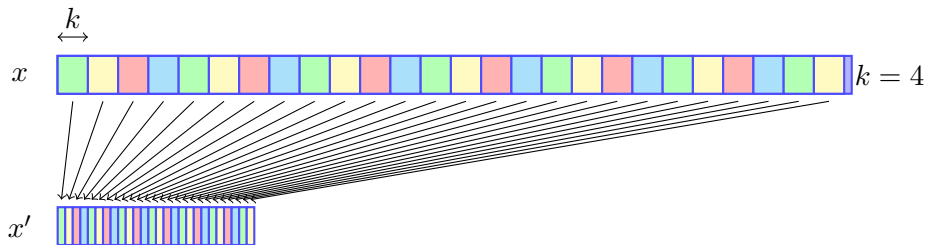
# LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)



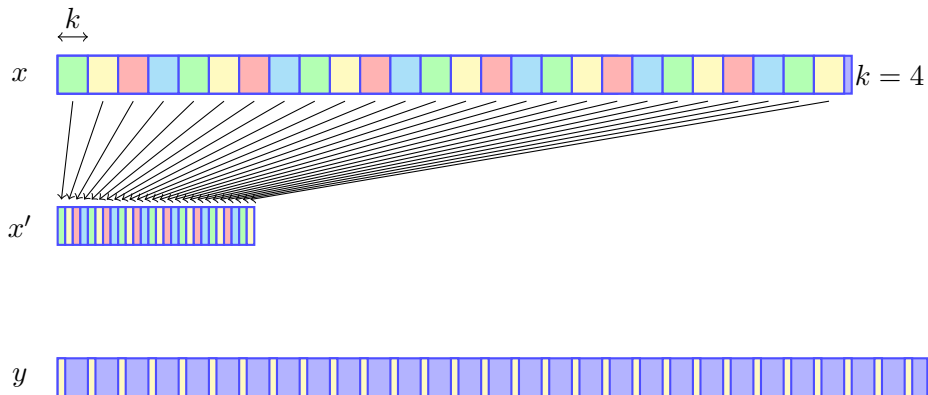
# LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)



# LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)

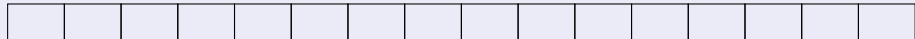


# LBNDM (Long patterns BNDM, Peltola & Tarhio, 2003)



## SSEF (Kulekci, 2009)

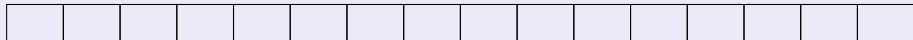
16-byte block





# SSEF (Külekci, 2009)

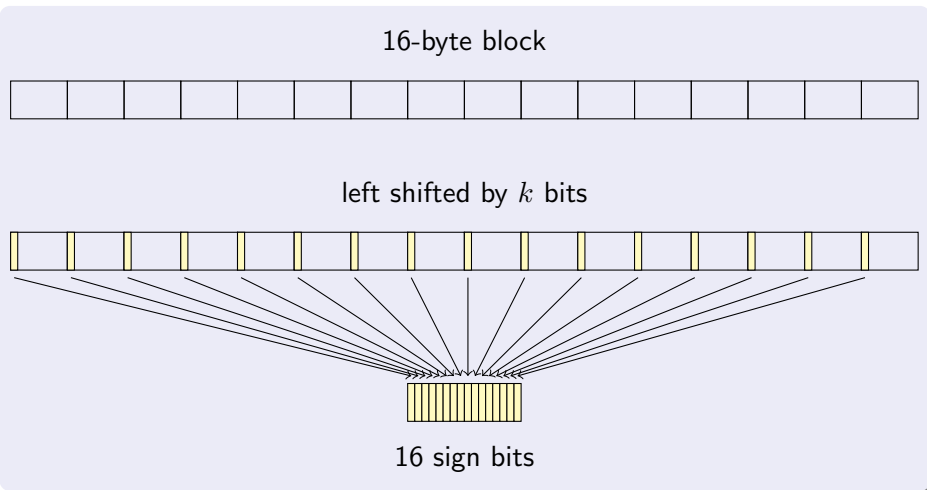
16-byte block



left shifted by  $k$  bits



# SSEF (Kulekci, 2009)

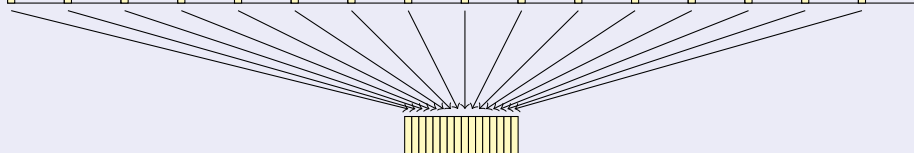


# SSEF (Kulekci, 2009)

16-byte block

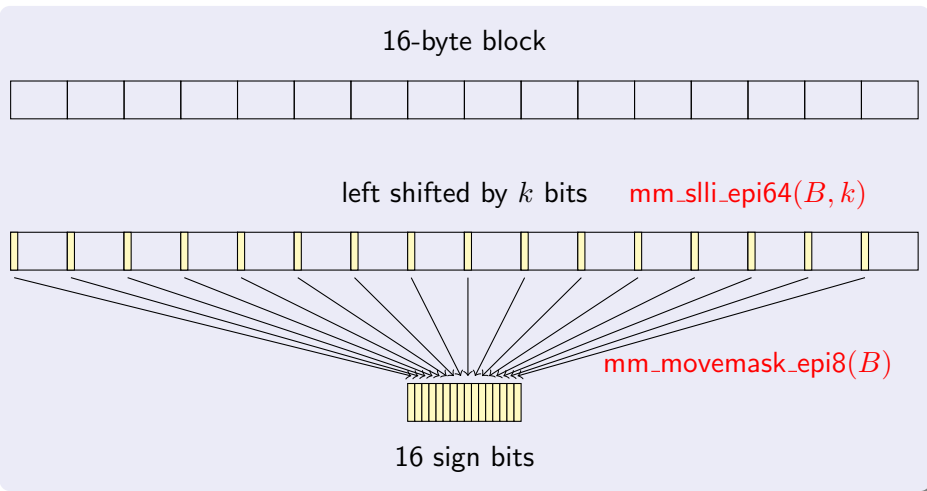


left shifted by  $k$  bits `mm_slli_epi64(B, k)`



16 sign bits

# SSEF (Külekci, 2009)



## SSEF (Külekci, 2009)

- uses Streaming SIMD Extension instructions
- $O(mn)$  worst case time complexity
- $O(n/m + mn/2^{16})$  average case time complexity
- needs to store at least  $2^{16} = 65536$  integers

# Outline

- 1 Introduction
- 2 Classical solutions
- 3 New efficient solutions
- 4 Experimental framework**

# SMART: String Matching Algorithm Research Tool

<http://www.dmi.unict.it/~faro/smart/>

The screenshot shows a web browser window with the URL <http://www.dmi.unict.it/~faro/smart/>. The browser's address bar shows the URL and a search icon. The website has a teal background and a navigation bar with links for SMART, DOWNLOAD, DOCUMENTATION, and POINTERS. The main content area features the title "smart string matching research tool by simone faro and thierry lecroq" and four key statistics: 1 TOOL, 40 YEARS, 85 ALGOS, and 12 TEXTS. A sidebar on the right lists various resources like smart documentation, implemented algorithms, data resources, experimental results, string matching authors, string matching bibliography, fast download, Linux binaries, Mac OS X binaries, source codes, and smart corpus.

SMART. String Matching Algorithms Research Tool

SMART DOWNLOAD DOCUMENTATION POINTERS

## smart string matching research tool

by simone faro and thierry lecroq

**1** TOOL

**smart (string matching algorithms research tool)** is a tool which provides a standard framework for researchers in string matching. It helps users to test, design, evaluate and understand existing solutions for the exact string matching problem. Moreover it provides the implementation of (almost) all string matching algorithms and a wide corpus of text buffers.

In the last **40 years of research** in computer science string matching was one of the most extensively studied problem, mainly due to its direct applications to such diverse areas as text, image and signal processing, speech analysis and recognition, data compression, information retrieval, computational biology and chemistry. Moreover String matching algorithms are also basic components used in implementations of practical softwares existing under most operating systems.

Since 1970 **more than 80 string matching algorithms** have been proposed, and more than 50% of them in the last ten years. The smart tool provides a comprehensive collection of all string matching algorithms, implemented in C programming language, and helps researcher to perform experimental results and compare them from a practical point of view. Smart provides a practical and standard platform for testing string matching algorithms and sharing results with the community.

The smart tool provides also a **corpus of 12 texts** on which the string matching algorithms can be tested. Texts in the corpus are of different types, including natural language texts, genome sequences, protein sequences, and random texts with a uniform distribution of characters.

the smart tool

- smart documentation
- implemented algorithms
- data resources
- experimental results
- string matching authors
- string matching bibliography

fast download

- Linux binaries
- Mac OS X binaries
- Source codes
- Smart corpus

# SMART: String Matching Algorithm Research Tool

- more than 80 string matching algorithms
- a corpus of 12 texts
- select/deselect string matching algorithms
- output experimental results in  $\text{\LaTeX}$ , xml, html and txt formats
- easy to plug new algorithms



# References



S. Faro and T. Lecroq

The Exact Online String Matching Problem: a Review of the Most Recent Results

Accepted in *ACM Computing Surveys*

# What's next

- next session

- sampling



F. Claude, G. Navarro, H. Peltola, L. Salmela and J. Tarhio  
String Matching with Alphabet Sampling

To appear in *Journal of Discrete Algorithms*

- constant space



D. Breslauer, R. Grossi and F. Mignosi

Simple Real-Time Constant-Space String Matching

CPM'11

# Perspectives

- multicore architecture