Recuperació de la informació

- Modern Information Retrieval (1999)
  Ricardo-Baeza Yates and Berthier Ribeiro-Neto

- Flexible Pattern Matching in Strings (2002)
  Gonzalo Navarro and Mathieu Raffinot

- Algorithms on strings (2001)
  M. Crochemore, C. Hancart and T. Lecroq

- http://www-igm.univ-mlv.fr/~lecroq/string/index.html

String Matching

String matching: definition of the problem (text,pattern)

- **Exact matching: depends on what we have: text or patterns**
  - **The patterns ---> Data structures for the patterns**
    - 1 pattern ---> The algorithm depends on |p| and |Σ|
    - k patterns ---> The algorithm depends on k, |p| and |Σ|
  - **Extensions**
  - **Regular Expressions**
  - **The text ----> Data structure for the text (suffix tree, ...)**
- **Approximate matching:**
  - **Dynamic programming**
  - **Sequence alignment (pairwise and multiple)**
  - **Sequence assembly: hash algorithm**
- **Probabilistic search: Hidden Markov Models**

Extended string matching

There are classes of characters represented by one symbol. For instace the IUPAC code for the DNA alphabet is:

R = {G,A} Y = {T,C} K = {G,T} M = {A,C} S = {G,C} W = {A,T}

B = {G,T,C } D = {G,A,T} H = {A,C,T} V = {G,C,A} N = {A,G,C,T} (any)

1. Classes of characters in the tetx.

There are characters in the text that represent sets of simbols

2. Classes of characters in the pattern.

There are characters in the pattern that represent sets of simbols

Extended alphabets

First part

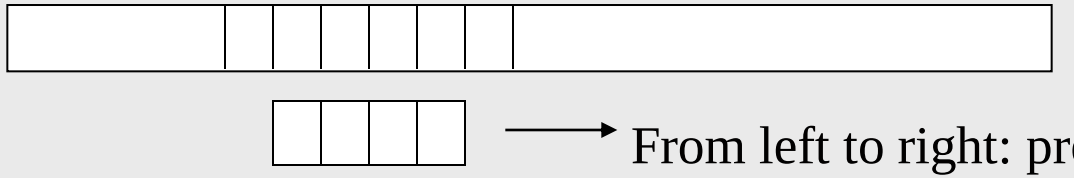
Classes in the text

Classes in the text: Brute force algorithm

- How the comparison is made?

Text : over $2^{|\Sigma|}$

Pattern over Σ



From left to right: prefix

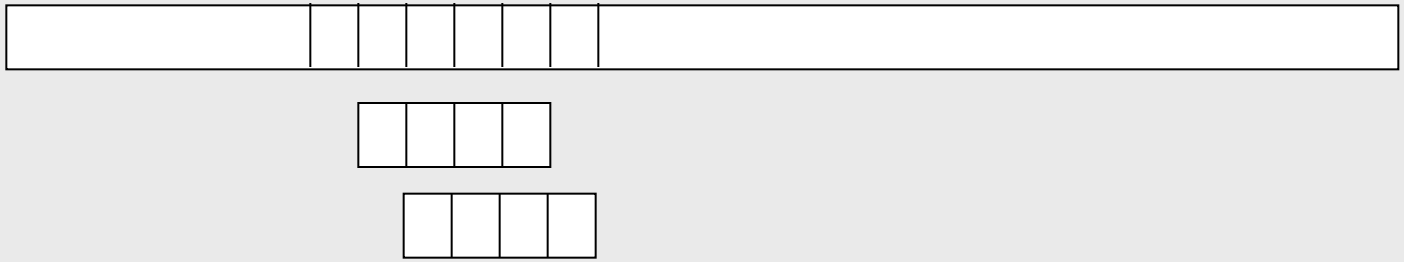
We need the operation: belongs to a set ?

?

- Which is the next position of the window?

Text :

Pattern :



The window is shifted only one cell

Classes in the text: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset of Σ is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet $\Sigma=\{A,C,G,T\}$:

$I(A)=(1,0,0,0)$, $I(C)=(0,1,0,0)$,... $I(R)=I(G,A)=(\ ,\ ,\ ,\ )$

Classes in the text: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset of Σ is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet $\Sigma=\{A,C,G,T\}$:

$I(A)=(1,0,0,0)$, $I(C)=(0,1,0,0)$,... $I(R)=I(G,A)=(1,0,1,0)$...$I(N)=(, , ,)$

Classes in the text: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset of Σ is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet $\Sigma=\{A,C,G,T\}$:

$I(A)=(1,0,0,0)$, $I(C)=(0,1,0,0)$,... $I(R)=I(G,A)=(1,0,1,0)$...$I(N)=(1,1,1,1)$

Then the operation "**A belongs to set X**" is made with ...

Classes in the text: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset of Σ is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet Σ={A,C,G,T} :

I(A)=(1,0,0,0), I(C)=(0,1,0,0),... I(R)=I(G,A)=(1,0,1,0)...I(N)=(1,1,1,1)

Then the operation "**A belongs to set X**" is made with **I(A) and I(X) >0**

G T A R T R N A G G A ...

A T G T A

A T G T A

I(A) & I(T)>0

Classes in the text: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset of Σ is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet Σ={A,C,G,T} :

I(A)=(1,0,0,0), I(C)=(0,1,0,0),... I(R)=I(G,A)=(1,0,1,0)...I(N)=(1,1,1,1)

Then the operation "**A belongs to set X**" is made with **I(A) and I(X) >0**

G T A R T R N A G G A ...

A T G T A

A T G T A

A T G T A

I(A) & I(T)>0

I(A) & I(R)>0 I(T) & I(T)>0 I(G) & I(R)>0 I(T) & I(A)>0

Classes in the text: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset of Σ is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet Σ={A,C,G,T} :

I(A)=(1,0,0,0), I(C)=(0,1,0,0),... I(R)=I(G,A)=(1,0,1,0)...I(N)=(1,1,1,1)

Then the operation "**A belongs to set X**" is made with **I(A) and I(X) >0**

G T A R T R N A G G A ...

A T G T A — I(A) & I(T)>0

A T G T A — I(A) & I(R)>0 I(T) & I(T)>0 I(G) & I(R)>0 I(T) & I(A)>0

A T G T A — I(A) & I(N)>0 I(T) & I(R)>0

...

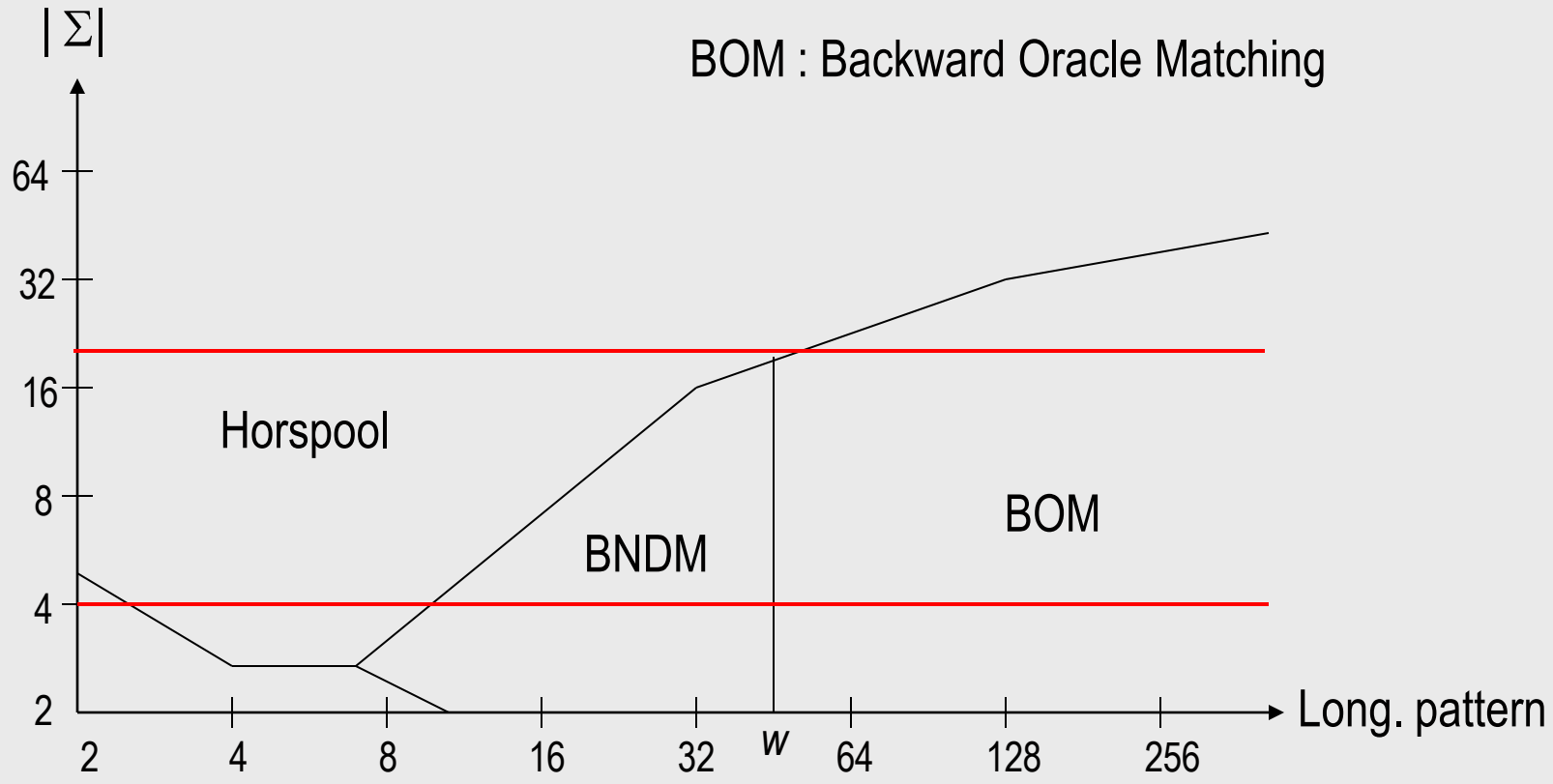
Which is the cost?

Classes in the text

Experimental efficiency (Navarro & Raffinot)

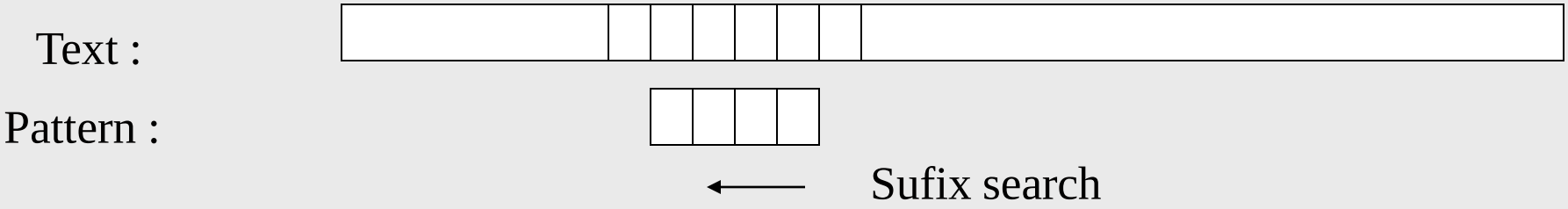
BNDM : Backward Nondeterministic Dawg Matching

BOM : Backward Oracle Matching

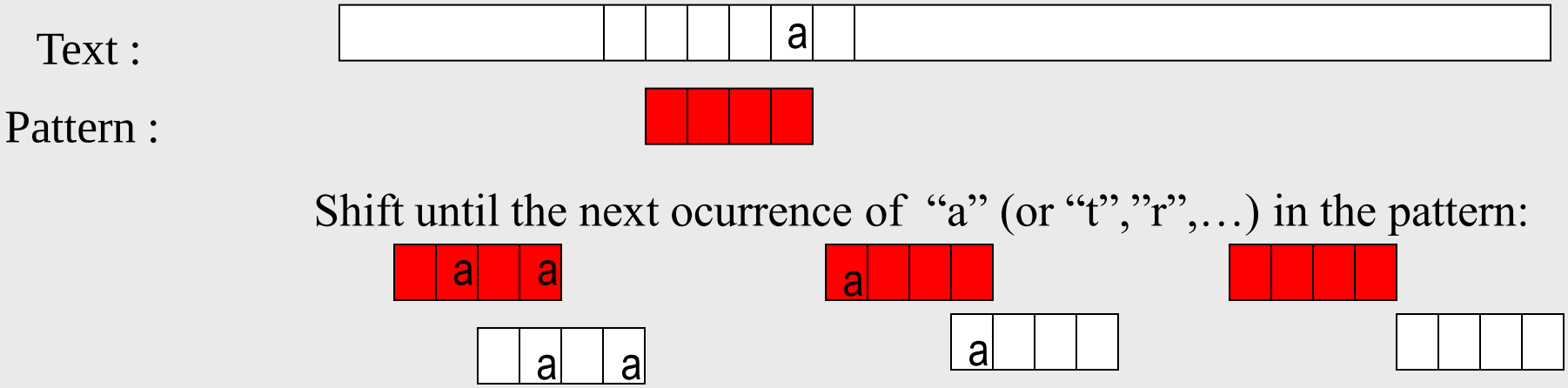


Classes in the text: Horspool algorithm

- **How the comparison is made?**



- **Which is the next position of the window?**



We need a shift table with the extended alphabet.

Classes in the text :Horspool example

Given the pattern ATGTA

- The shift table is:

A	4
C	5
G	2
T	1
R	?
…	
N	?

Classes in the text :Horspool example

Given the pattern ATGTA

- The shift table is:

A	4
C	5
G	2
T	1
R	2
…	
N	?

Classes in the text :Horspool example

Given the pattern ATGTA

- The shift table is:

A	4
C	5
G	2
T	1
R	2
…	
N	1

text : G T A R T R N A A G G A …

A T G T A

A T G T A

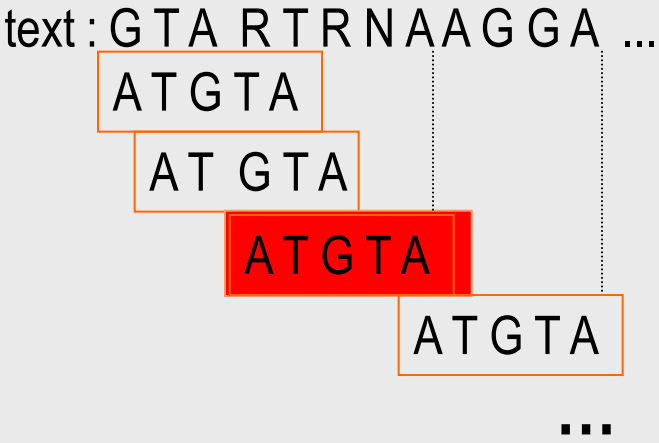
A T G T A

Classes in the text :Horspool example

Given the pattern ATGTA

- The shift table is:

A	4
C	5
G	2
T	1
R	2
…	
N	1

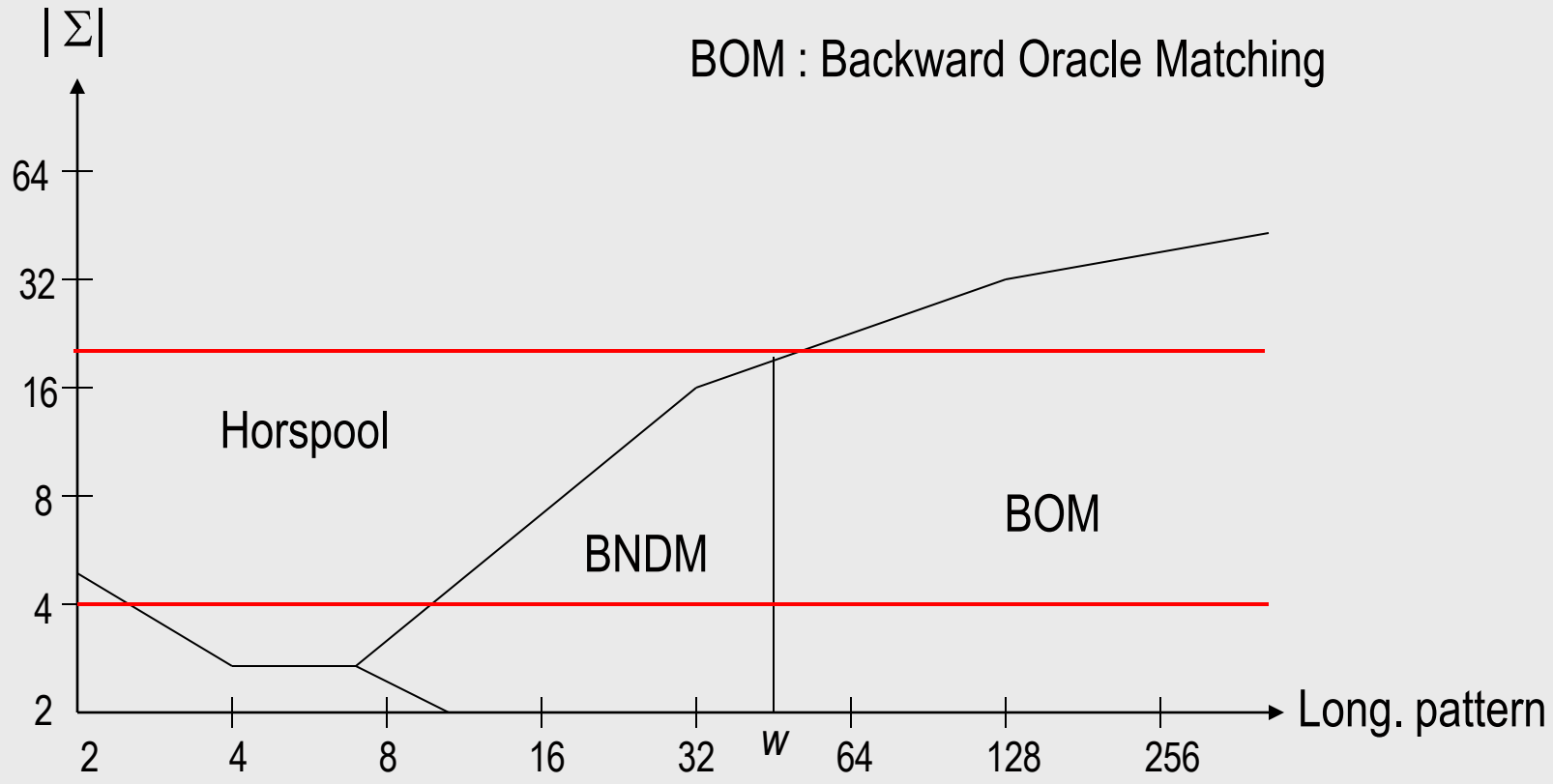


Classes in the text

Experimental efficiency (Navarro & Raffinot)

BNDM : Backward Nondeterministic Dawg Matching

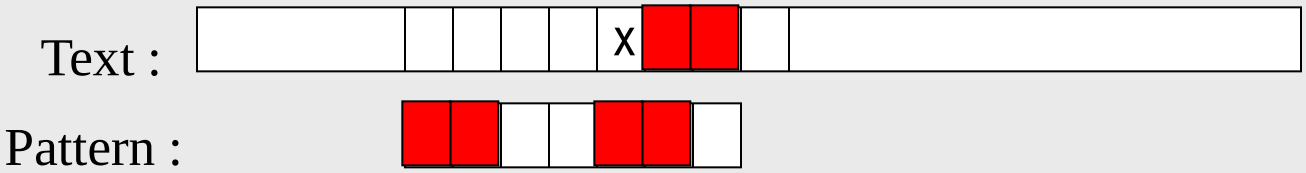
BOM : Backward Oracle Matching



Classes in the text: BNDM algorithm

- **How the comparison is made?**

Search for suffixes of T that are factors of the pattern



…that is denoted as

$D_2 = 1\ 0\ 0\ 0\ 1\ 0\ 0$

Once the next character x is read

$D_3 = D_2 << 1\ \&\ B(x)$

B(x): mask of x in the pattern P.

For instance, if B(x) = (0 0 1 1 0 0 0)

D = (0 0 0 1 0 0 0) & (0 0 1 1 0 0 0) = (0 0 0 1 0 0 0)

- **Which is the next position of the window ?**

Depends on the value of the leftmost bit of D

Classes in the text : BNDM example

Given the pattern ATGTA

- The masks of bits are

B(A) = (1 0 0 0 1) B(R)=( )
B(C) = (0 0 0 0 0) …
B(G) = (0 0 1 0 0) B(N)=( )
B(T) = (0 1 0 1 0)

Classes in the text : BNDM example

Given the pattern ATGTA

- The masks of bits are

B(A) = (1 0 0 0 1)
B(C) = (0 0 0 0 0)
B(G) = (0 0 1 0 0)
B(T) = (0 1 0 1 0)

B(R)=(1 0 1 0 1)
…
B(N)=( )

Classes in the text : BNDM example

Given the pattern ATGTA

- The masks of bits are

B(A) = (1 0 0 0 1)
B(C) = (0 0 0 0 0)
B(G) = (0 0 1 0 0)
B(T) = (0 1 0 1 0)

B(R)=(1 0 1 0 1)
…
B(N)=(1 1 1 1 1)

- text : G T A R T R N A G G A C G ...

A T G T A

A T G T A

D_1 = (0 1 0 1 0)
D_2 = (1 0 1 0 0) & (1 0 1 0 1) = (1 0 1 0 0)
D_2 = (0 1 0 0 0) & (1 0 0 0 1) = (0 0 0 0 0)

Classes in the text : BNDM example

Given the pattern ATGTA

- The masks of bits are

B(A) = (1 0 0 0 1)
B(C) = (0 0 0 0 0)
B(G) = (0 0 1 0 0)
B(T) = (0 1 0 1 0)

B(R)=(1 0 1 0 1)
…
B(N)=(1 1 1 1 1)

- text : G T A R T R N A G G A C G ...

A T G T A

A T G T A

D_1 = (0 1 0 1 0)
D_2 = (1 0 1 0 0) & (1 0 1 0 1) = (1 0 1 0 0)
D_2 = (0 1 0 0 0) & (1 0 0 0 1) = (0 0 0 0 0)

D_1 = (1 0 0 0 1)
D_2 = (0 0 0 1 0) & (1 1 1 1 1) = (0 0 0 1 0)
D_3 = (0 0 1 0 0) & (1 0 1 0 1) = (0 0 1 0 0)
D_4 = (0 1 0 0 0) & (0 1 0 1 0) = (0 1 0 0 0)
D_5 = (1 0 0 0 0) & (1 0 1 0 1) = (1 0 0 0 0)

Classes in the text : BNDM example

Given the pattern ATGTA

- The masks of bits are

B(A) = (1 0 0 0 1)
B(C) = (0 0 0 0 0)
B(G) = (0 0 1 0 0)
B(T) = (0 1 0 1 0)

B(R)=(1 0 1 0 1)
…
B(N)=(1 1 1 1 1)

- text : G T A R T R N A G G A C G ...

A T G T A

D_1 = (0 1 0 1 0)
D_2 = (1 0 1 0 0) & (1 0 1 0 1) = (1 0 1 0 0)
D_2 = (0 1 0 0 0) & (1 0 0 0 1) = (0 0 0 0 0)

D_1 = (1 0 0 0 1)
D_2 = (0 0 0 1 0) & (1 1 1 1 1) = (0 0 0 1 0)
D_3 = (0 0 1 0 0) & (1 0 1 0 1) = (0 0 1 0 0)
D_4 = (0 1 0 0 0) & (0 1 0 1 0) = (0 1 0 0 0)
D_5 = (1 0 0 0 0) & (1 0 1 0 1) = (1 0 0 0 0)

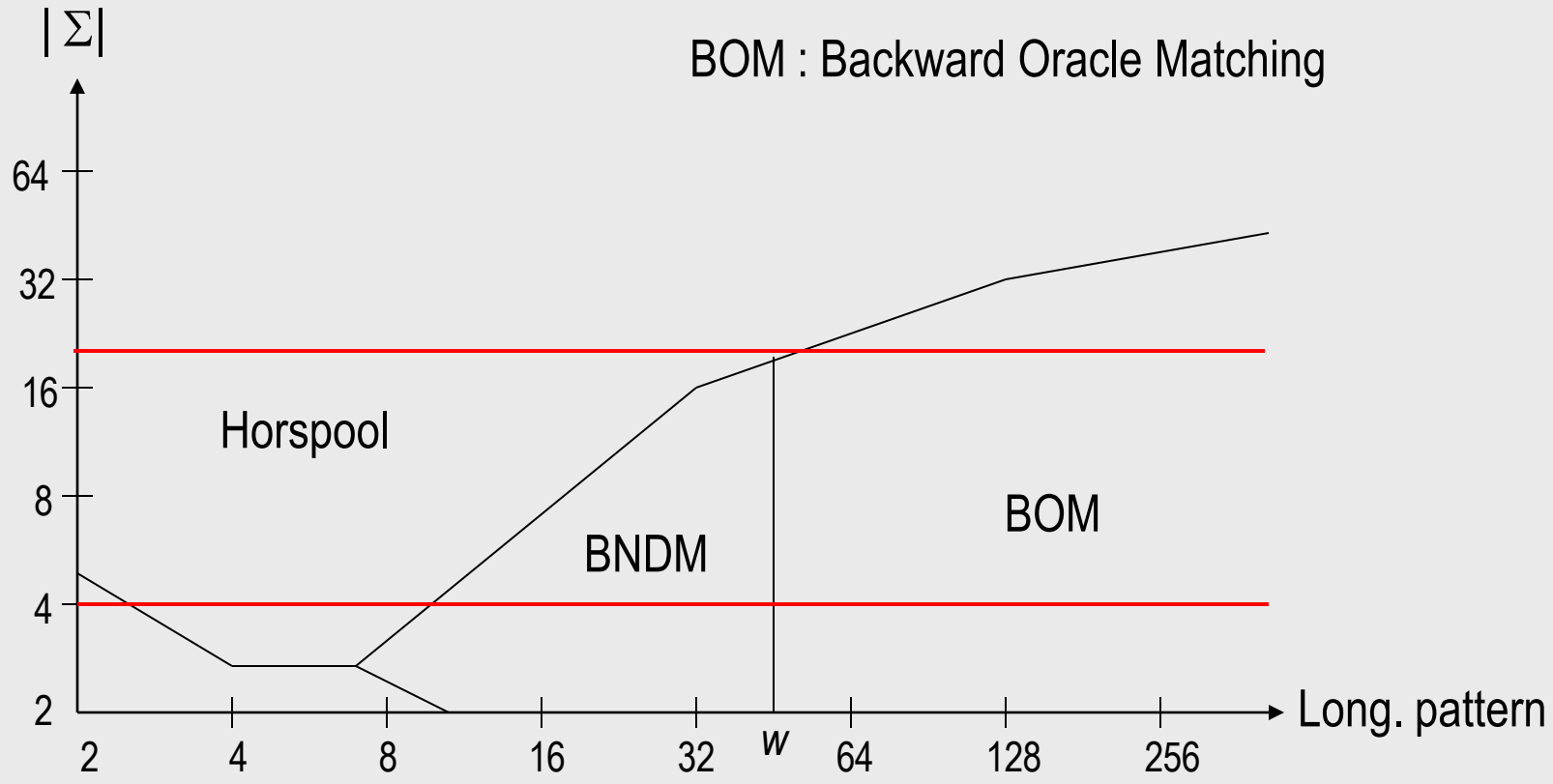
A T G T A …

Classes in the text

Experimental efficiency (Navarro & Raffinot)

BNDM : Backward Nondeterministic Dawg Matching

BOM : Backward Oracle Matching



BOM algorithm (Backward Oracle Matching)

- **How the comparison is made?**

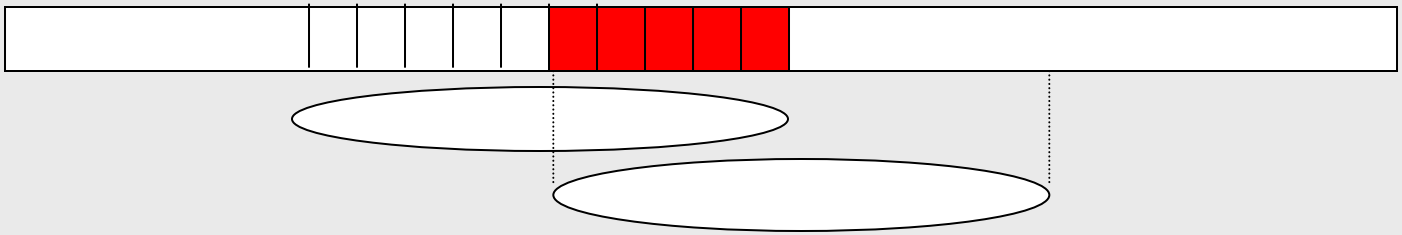
Text :

Pattern :

Automata: Factor Oracle

Check if the suffix is a factor

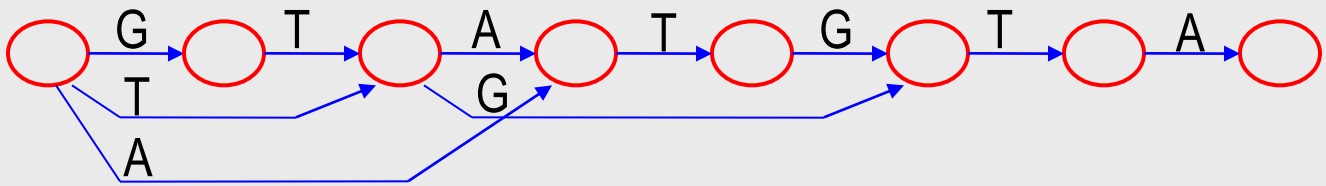
- **Which is the next position of the window?**



The position determined by the last character of the text
with a transition in the automata

Classes in the text: BOM example

The we build the AFO of the inverse pattern of ATGTATG

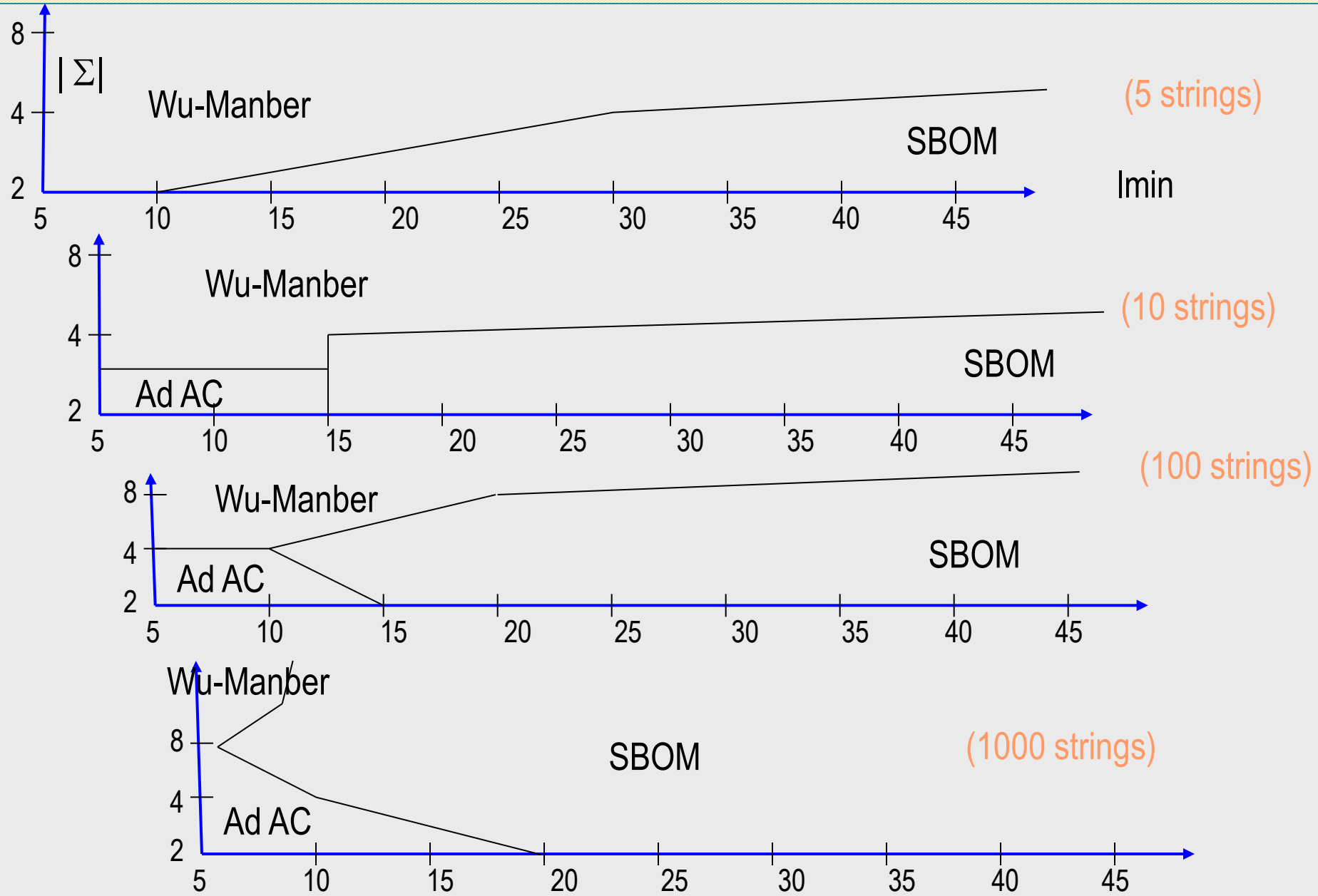


… and we try to find… : G T A R T R N A A T G…

A T G T A T G

It’s not possible any improvement!

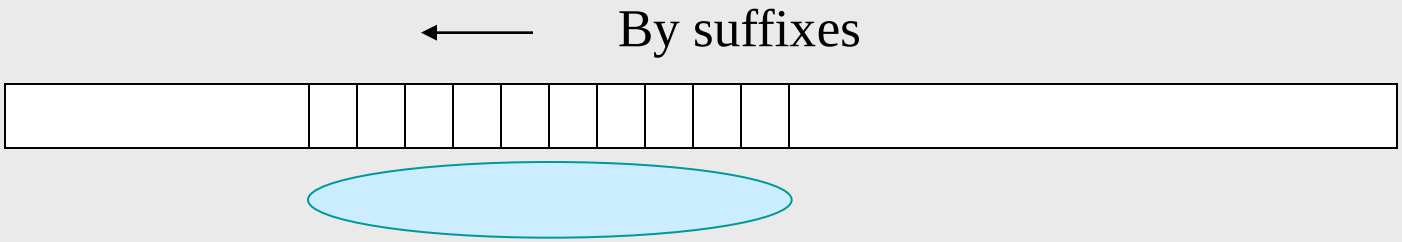
Multiple string matching



Classes in the text: Set Horspool algorithm

- **How the comparison is made?**

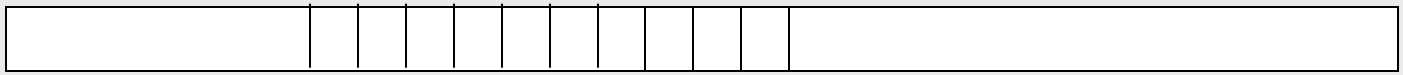
Text :



Patterns:

Trie of all inverse patterns

- **Which is the next position of the window?**

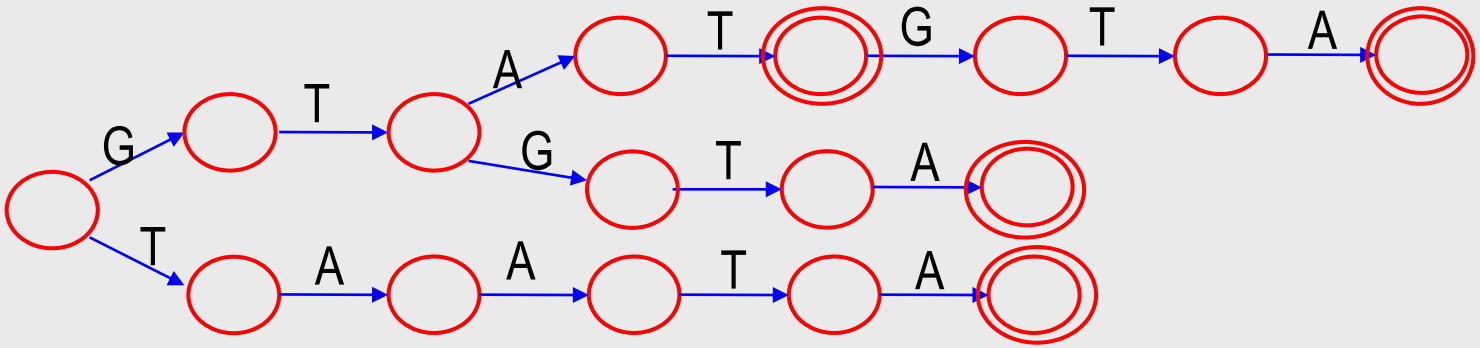


?

Set Horspool algorithm

Search for ATGTATG,TATG,ATAAT,ATGTG

1. Construct the trie of GTATGTA, GTAT, TAATA i GTGTA



2. Determine *lmin=4*

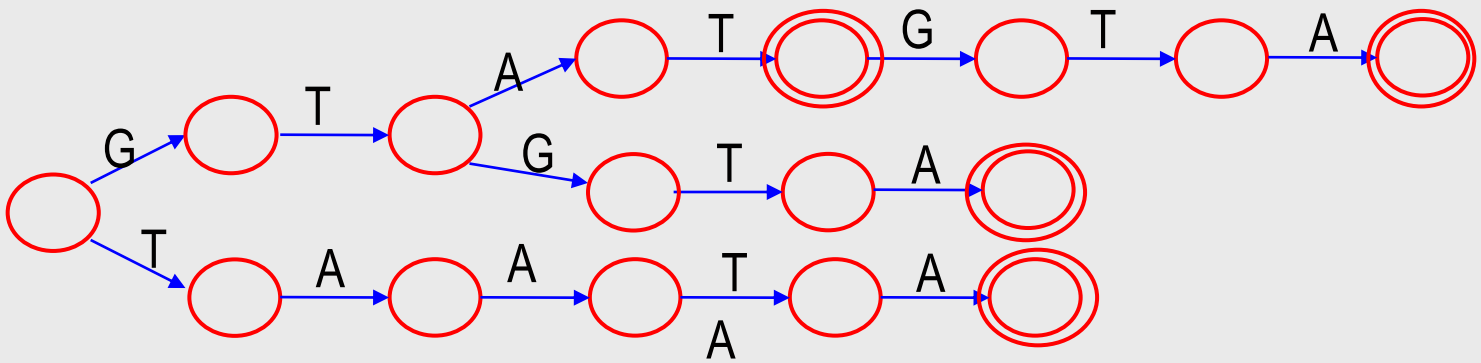
3. Determine the shift table

A	1
C	4 (lmin)
G	2
T	1

4. Find the patterns

Classes in the text: Set Horspool

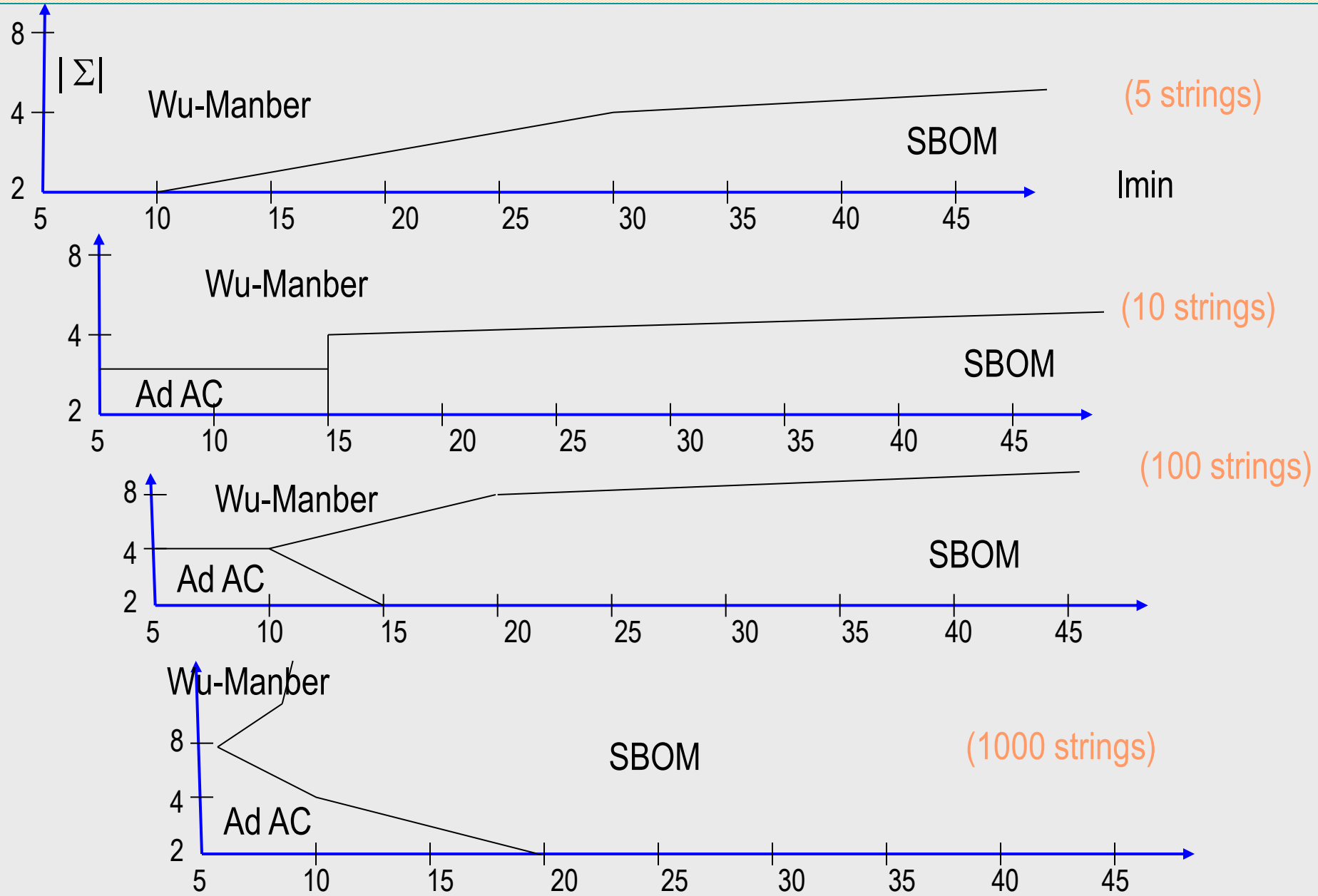
Search for the patterns
ATGTATG,TATG,ATAAT,ATGTG



text: ARTGNCTATGTGACA…

It’s not possible any improvement!

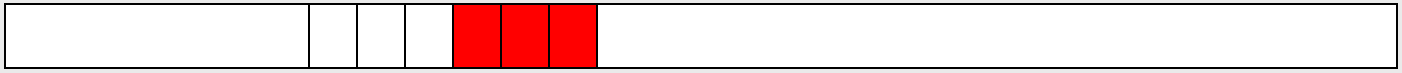
Multiple string matching



Classes in the text: SBOM algorithm

- **How the comparison is made?**

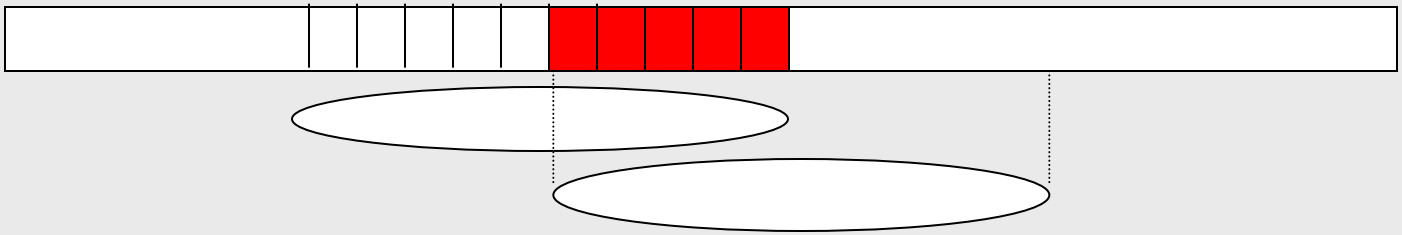
Text :



Pattern : Automata: Factor Oracle (Inverse patterns of length *lmin)*

Check if the suffix is a factor of any pattern

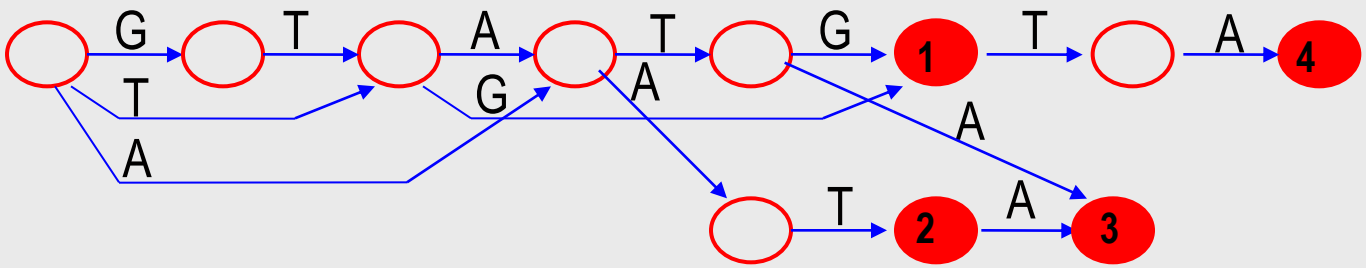
- **Which is the next position of the window?**



The position determined by the last character of the text
with a transition in the automata

Classes in the text: SBOM example

Search for the patterns ATGTATG, TAATG,TAATAAT i AATGTG



text: ACATN C TAGC TA TA ATAATGTATG

It’s not possible any improvement!

Extended alphabets

Classes in the:

	text	pattern
Horspool	✓	
BNDM	✓	
BOM	✗	
Set-Horspool	✗	
SBOM	✗	

Extended search

Second part

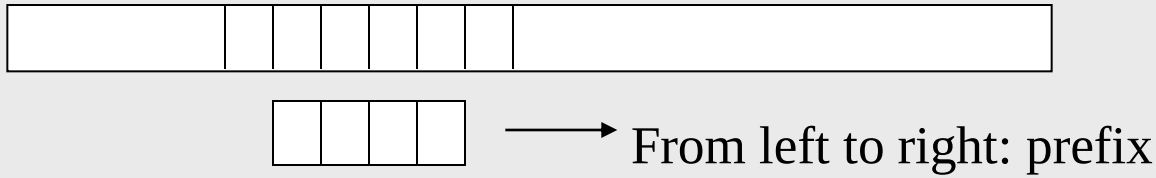
Classes in the pattern

Classes in the pattern: Brute force algorithm

- How the comparison is made?

Text : over Σ

Pattern : over $2^{|\Sigma|}$



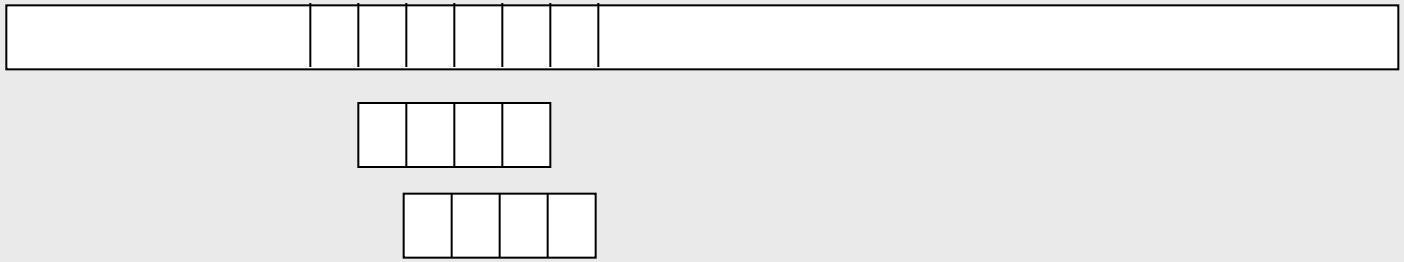
We need the operation: belongs to a set ?

?

- Which is the next position of the window?

Text :

Pattern :



The window is shifted only one cell

Classes in the pattern: Brute force algorithm

When $|\Sigma|$ < computer word

Every subset is represented by a string of bits of length $|\Sigma|$.

For instance, given the DNA alphabet Σ={A,C,G,T} :

I(A)=(1,0,0,0), I(C)=(0,1,0,0),... I(R)=(1,0,1,0,),...., I(N)=(1,1,1,1)

Then the operation "**A belongs to set X**" is made with **I(A) and I(X) >0**

G T A C T A G A G G A C G T A T G T A C T G ...

I(T) and I(R) >0 — A T N T R

I(A) and I(R) >0 — A T N T R

I(T) and I(T) >0

I(C) and I(N) >0

I(A) and I(T) >0

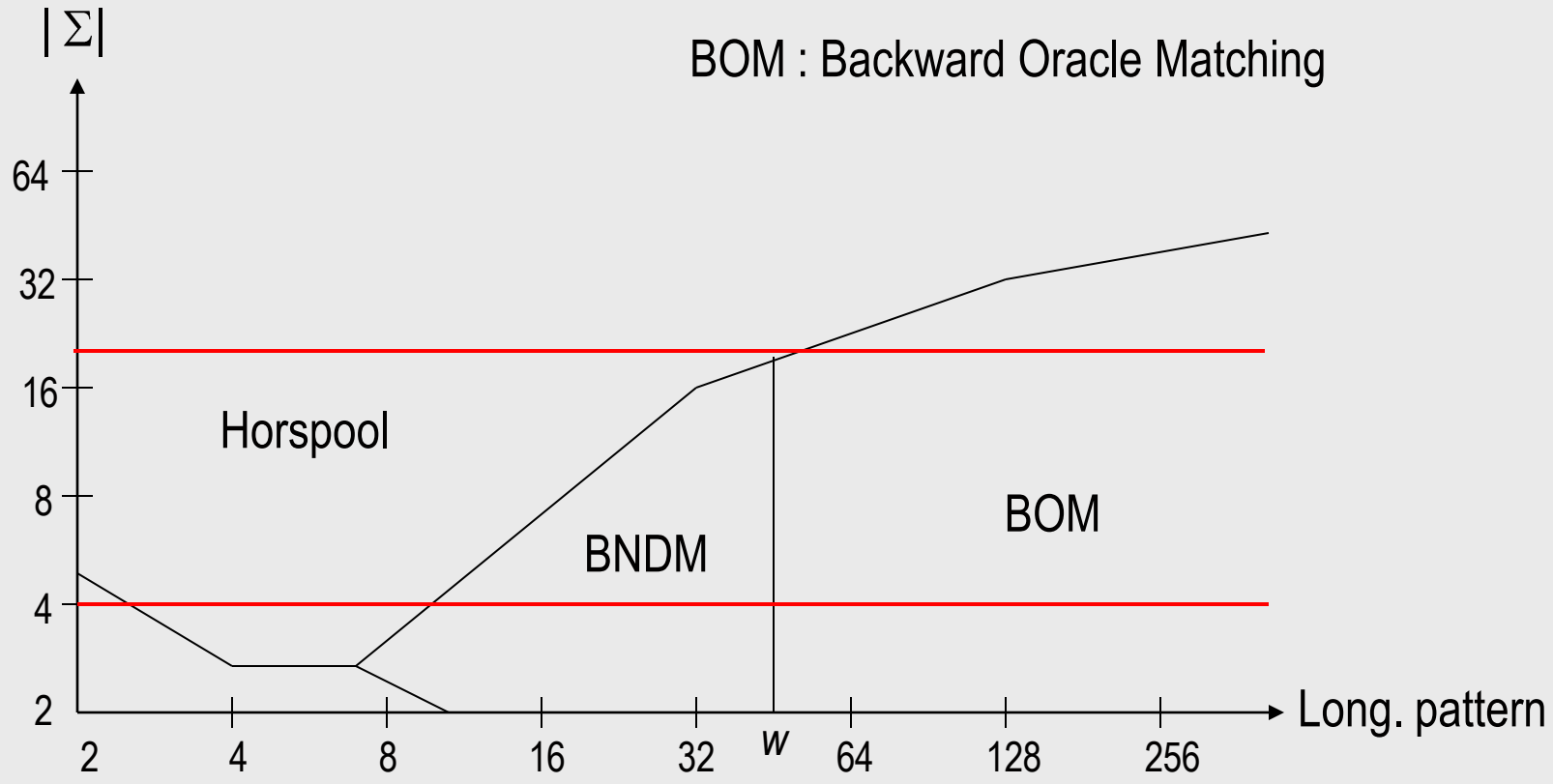
...

Classes in the text

Experimental efficiency (Navarro & Raffinot)

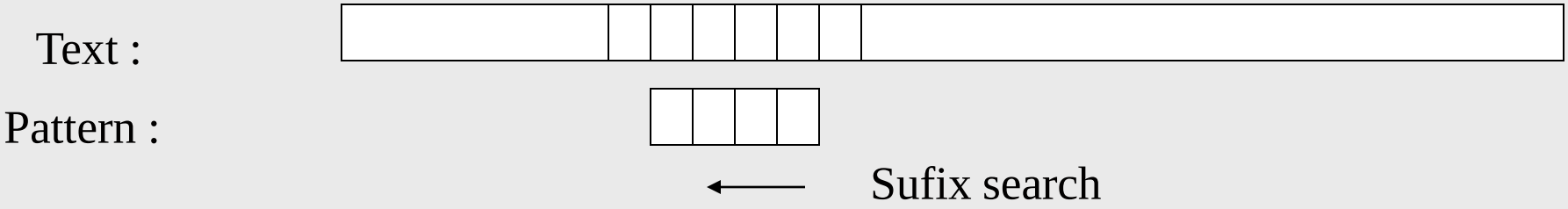
BNDM : Backward Nondeterministic Dawg Matching

BOM : Backward Oracle Matching

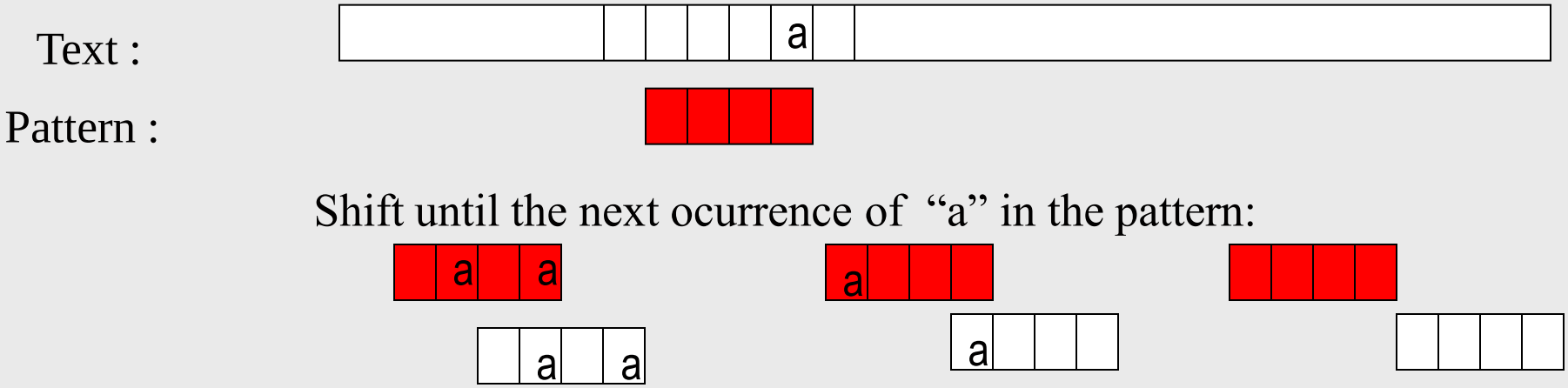


Classes in the pattern: Horspool algorithm

- **How the comparison is made?**



- **Which is the next position of the window?**



We need a preprocessing phase to construct the shift table.

Classes in the pattern: Horspool example

Given the pattern ATNTR

- The shift table is:

A	
C	
G	
T	

Classes in the pattern: Horspool example

Given the pattern ATNTR

- The shift table is:

A	2
C	
G	
T	

Classes in the pattern: Horspool example

Given the pattern ATNTR

- The shift table is:

A	2
C	2
G	
T	

Classes in the pattern: Horspool example

Given the pattern ATNTR

- The shift table is:

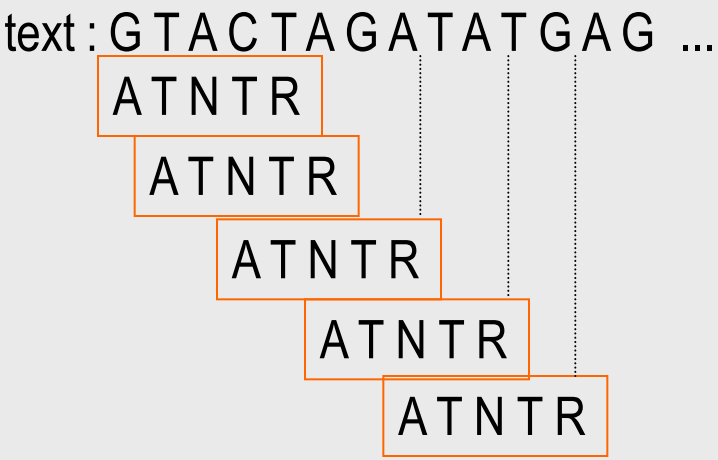
A	2
C	2
G	2
T	

Classes in the pattern: Horspool example

Given the pattern ATNTR

- The shift table is:

A	2
C	2
G	2
T	1

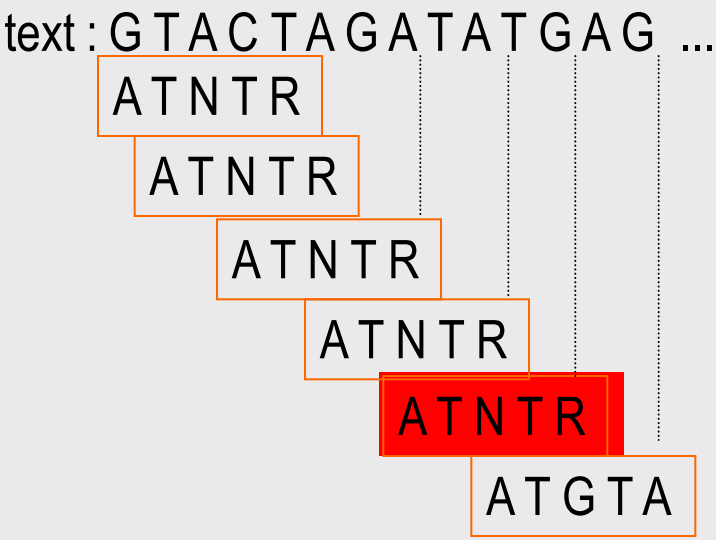


Classes in the pattern: Horspool example

Given the pattern ATNTR

- The shift table is:

A	2
C	2
G	2
T	1



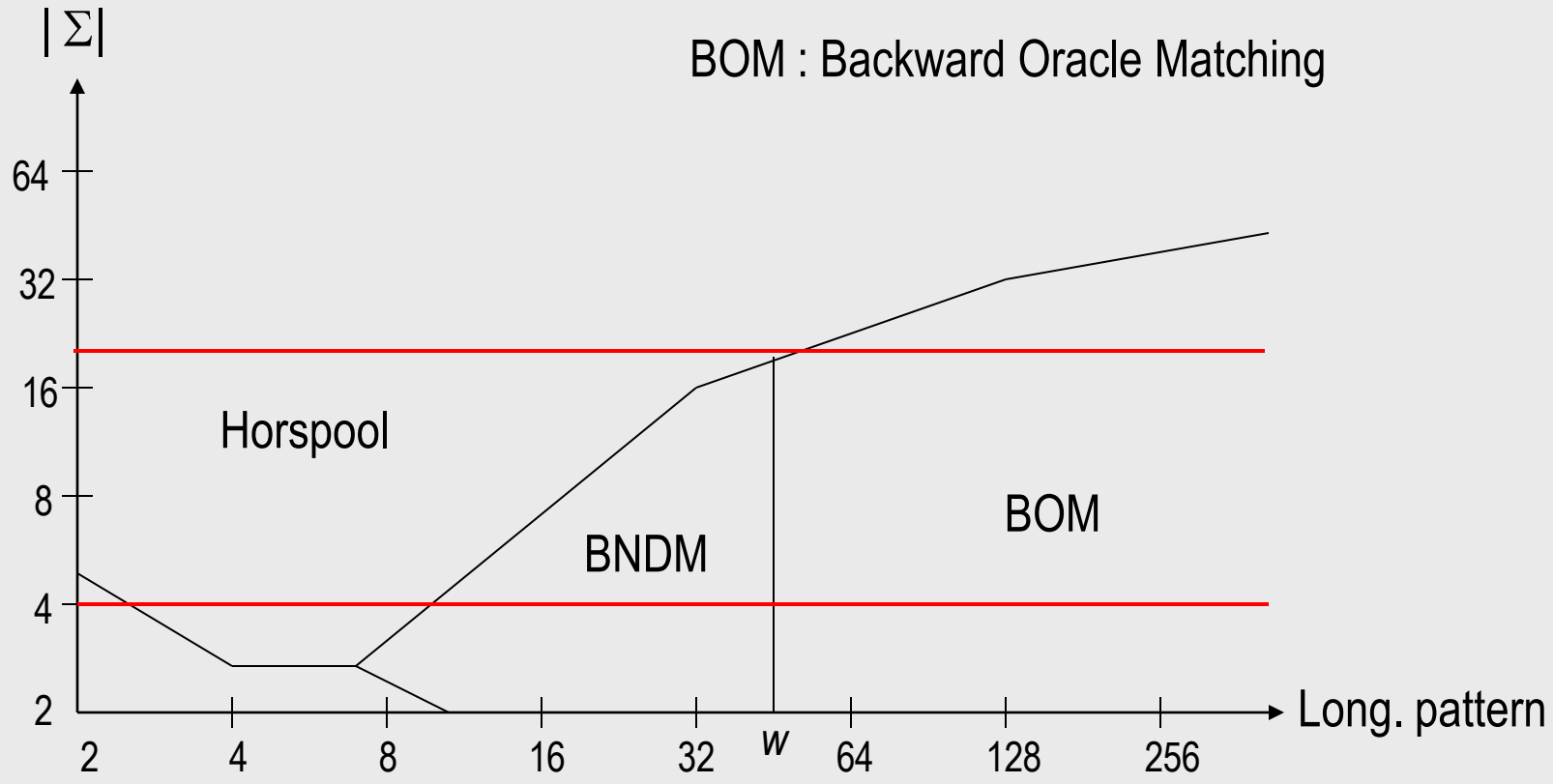
Shorter shifts!

Classes in the text

Experimental efficiency (Navarro & Raffinot)

BNDM : Backward Nondeterministic Dawg Matching

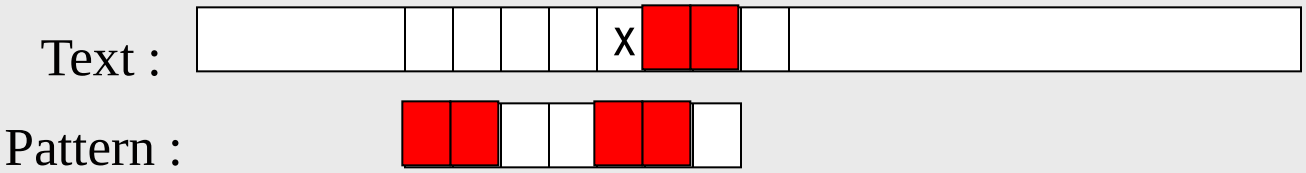
BOM : Backward Oracle Matching



Classes in the text: BNDM algorithm

- **How the comparison is made?**

Search for suffixes of T that are factors of the pattern



…that is denoted as

$D_2 = 1\ 0\ 0\ 0\ 1\ 0\ 0$

Once the next character x is read

$D_3 = D_2 << 1\ \&\ B(x)$

B(x): mask of x in the pattern P.

For instance, if B(x) = (0 0 1 1 0 0 0)

D = (0 0 0 1 0 0 0) & (0 0 1 1 0 0 0) = (0 0 0 1 0 0 0)

- **Which is the next position of the window ?**

Depends on the value of the leftmost bit of D

Classes in the pattern : BNDM example

Given the pattern ATNTR

- The masks of bits of symbols are

B(A) = ( )
B(C) = ( )
B(G) = ( )
B(T) = ( )

Classes in the pattern : BNDM example

Given the pattern ATNTR

- The masks of bits of symbols are

B(A) = (1 0 1 0 1)
B(C) = ( )
B(G) = ( )
B(T) = ( )

Classes in the pattern : BNDM example

Given the pattern ATNTR

- The masks of bits of symbols are

B(A) = (1 0 1 0 1)
B(C) = (0 0 1 0 0)
B(G) = ( )
B(T) = ( )

Classes in the pattern : BNDM example

Given the pattern ATNTR

- The masks of bits of symbols are

B(A) = (1 0 1 0 1)
B(C) = (0 0 1 0 0)
B(G) = (0 0 1 0 1)
B(T) = ( )

Classes in the pattern : BNDM example

Given the pattern ATNTR

- The masks of bits of symbols are

B(A) = (1 0 1 0 1)
B(C) = (0 0 1 0 0)
B(G) = (0 0 1 0 1)
B(T) = (0 1 1 1 0)

- text : G T A C T A G A G G A C G T A T G T A C T G ...

A T N T R

D_1 = (0 1 1 1 0)
D_2 = (1 1 1 0 0) & (0 0 1 0 0) = (0 0 1 0 0)
D_3 = (0 1 0 0 0) & (1 0 1 0 1) = (0 0 0 0 0)

A T N T R

D_1 = (0 0 1 0 1)
D_2 = (0 1 0 1 0) & (0 0 1 0 1) = (0 0 0 0 0)

A T N T R

D_1 = (1 0 1 0 1)
D_2 = (0 1 0 1 0) & (0 1 1 1 0) = (0 1 0 1 0)
D_3 = (1 0 1 0 0) & (0 0 1 0 1) = (0 0 1 0 0)
D_4 = (0 1 0 0 0) & (0 0 1 0 0) = (0 0 0 0 0)

A T N T R

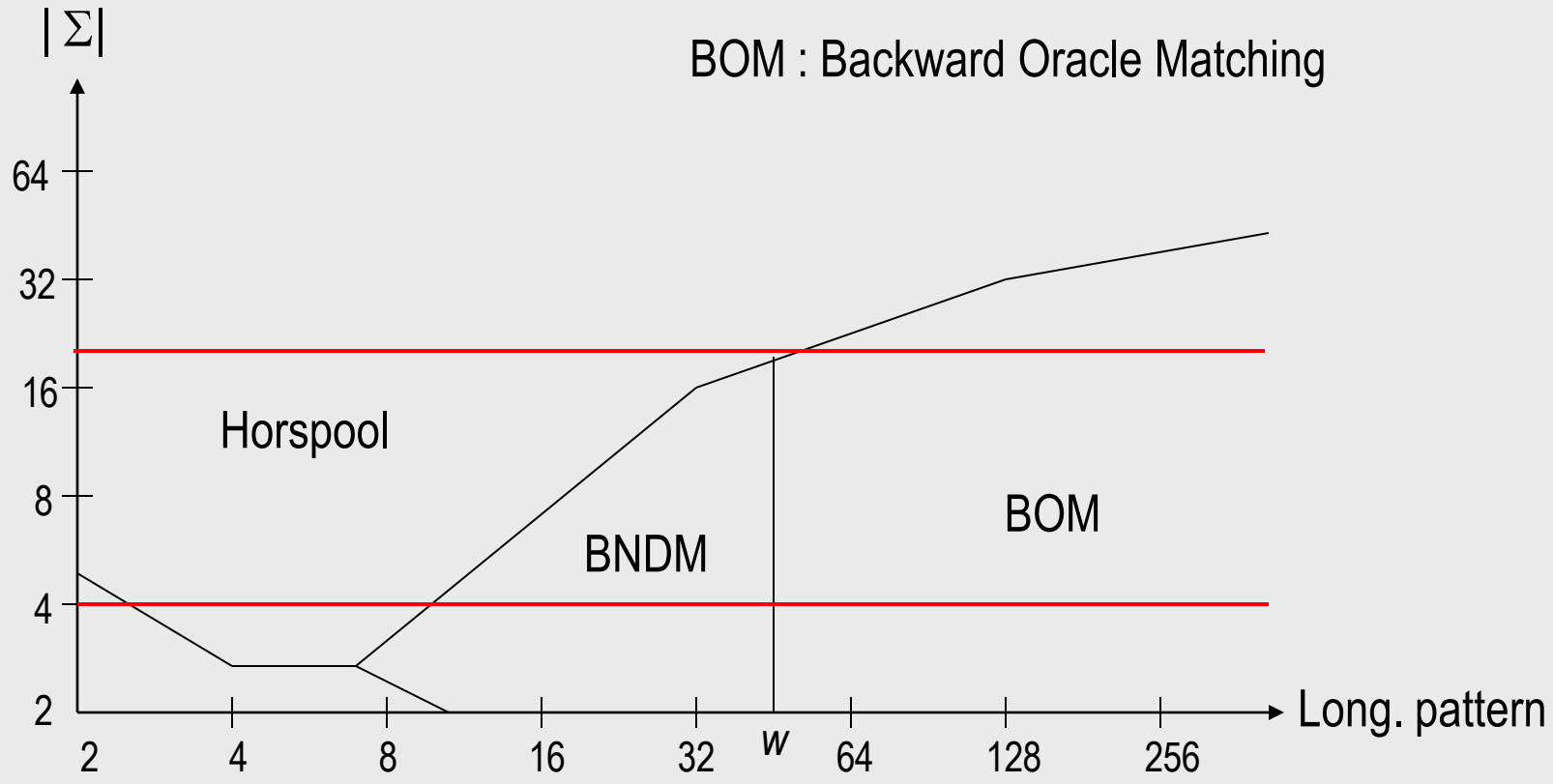
...

Classes in the text

Experimental efficiency (Navarro & Raffinot)

BNDM : Backward Nondeterministic Dawg Matching

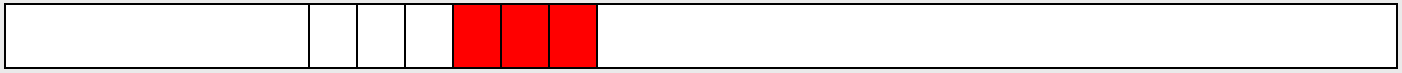
BOM : Backward Oracle Matching



BOM algorithm (Backward Oracle Matching)

- **How the comparison is made?**

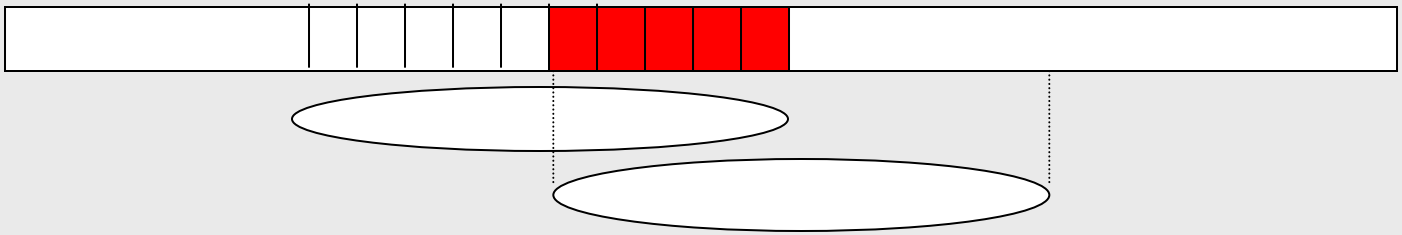
Text :



Pattern : Automata: Factor Oracle

Check if the suffix is a factor

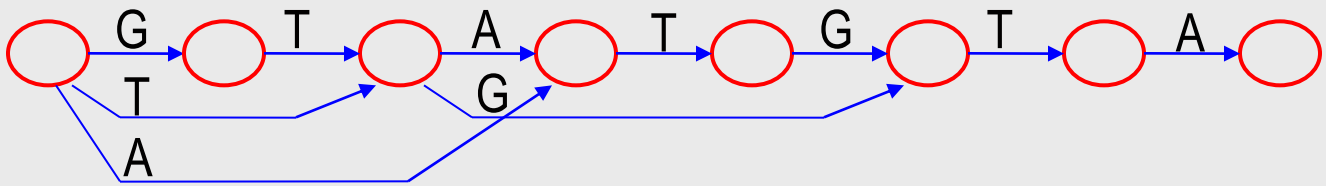
- **Which is the next position of the window?**



The position determined by the last character of the text
with a transition in the automata

Classes in the pattern: BOM example

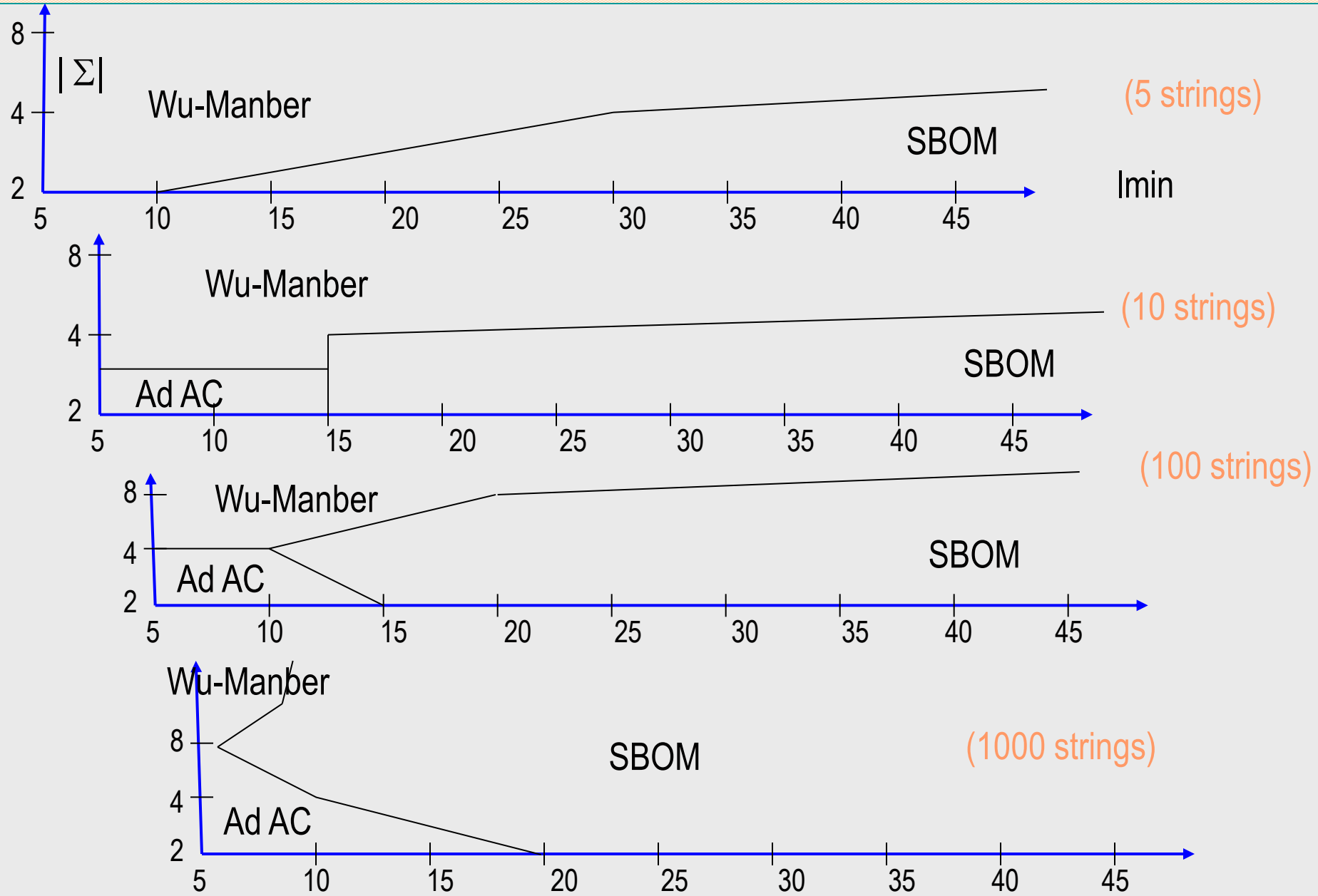
- Given the pattern ATGTATG, the AFO is



but for the patter ATNTRTG?

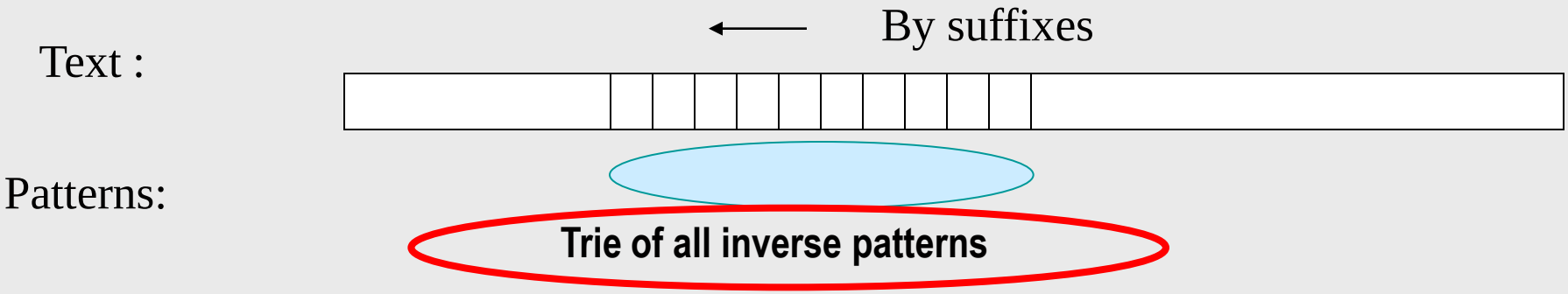
We should apply the SBOM algorithm!

Multiple string matching

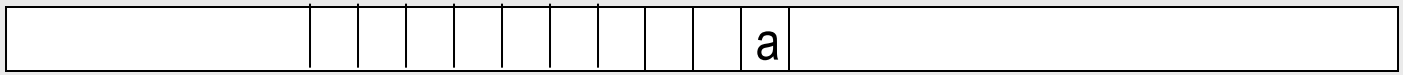


Set Horspool algorithm

- **How the comparison is made?**



- **Which is the next position of the window?**



We shift until a is aligned with the first a in the trie not longer than *lmin*, or *lmin*

Set Horspool algorithm

Search for **ATNTARG,RTGR,NTTNAR,ATRTG**

1. Construct the trie of the **46 possible inverse patterns**

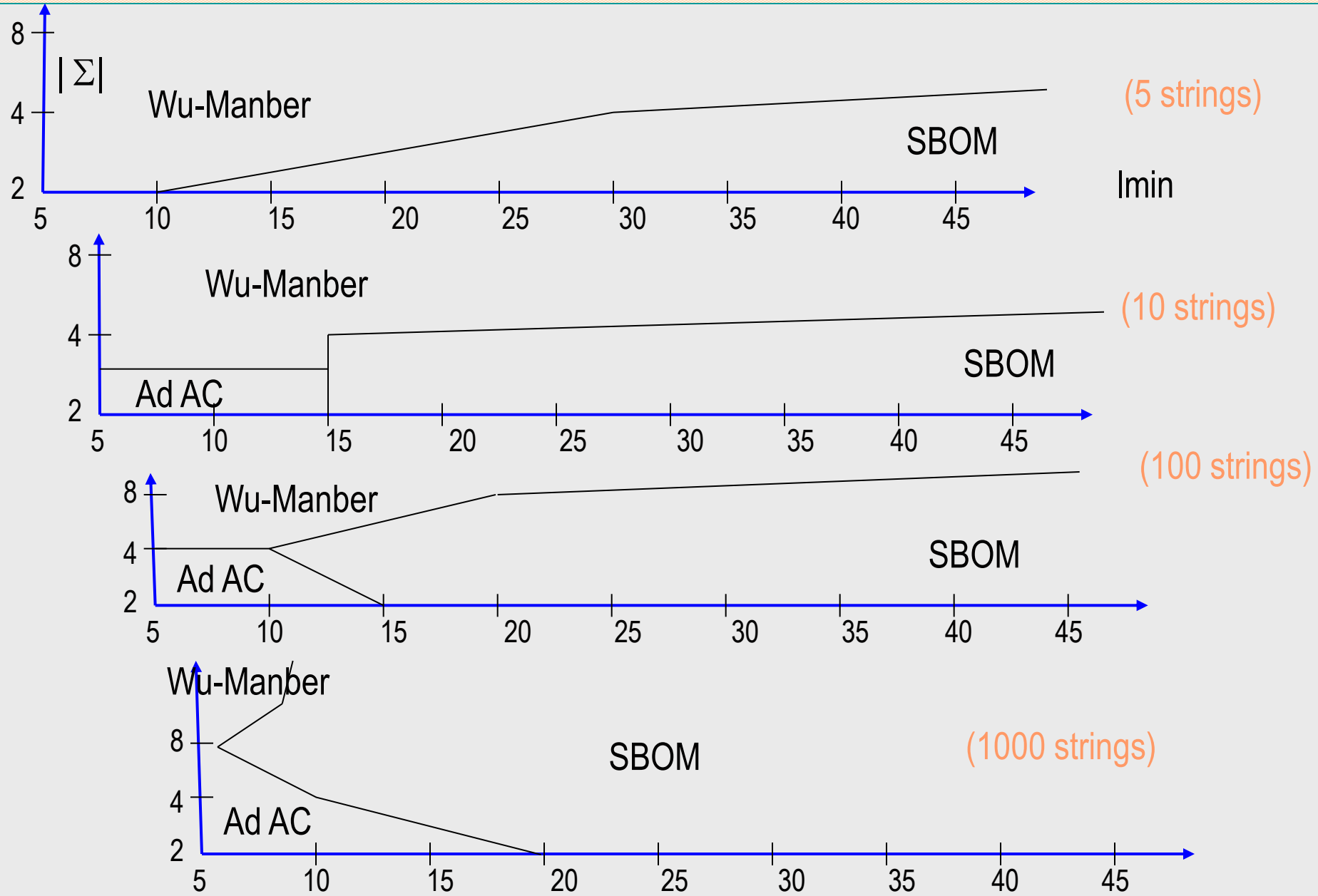
2. Determine *lmin=4*

3. Determine the shift table

A	1
C	2
G	1
T	1

4. Find the patterns

Multiple string matching



SBOM algorithm

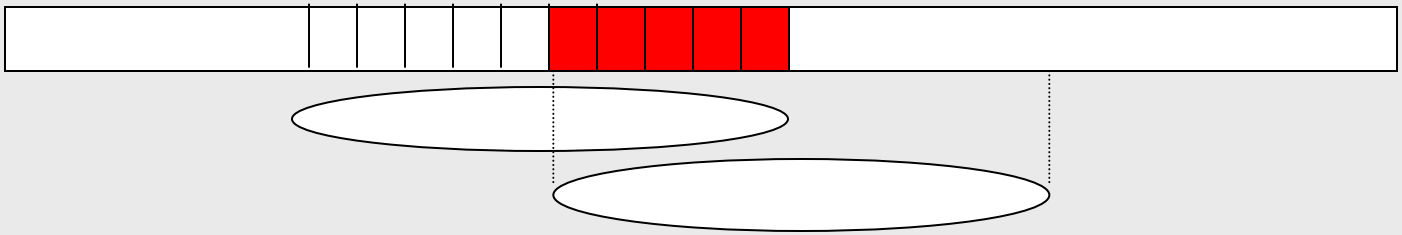
- **How the comparison is made?**

Text :

Pattern : Automata: Factor Oracle (Inverse patterns of length *lmin)*

Check if the suffix is a factor of any pattern

- **Which is the next position of the window?**



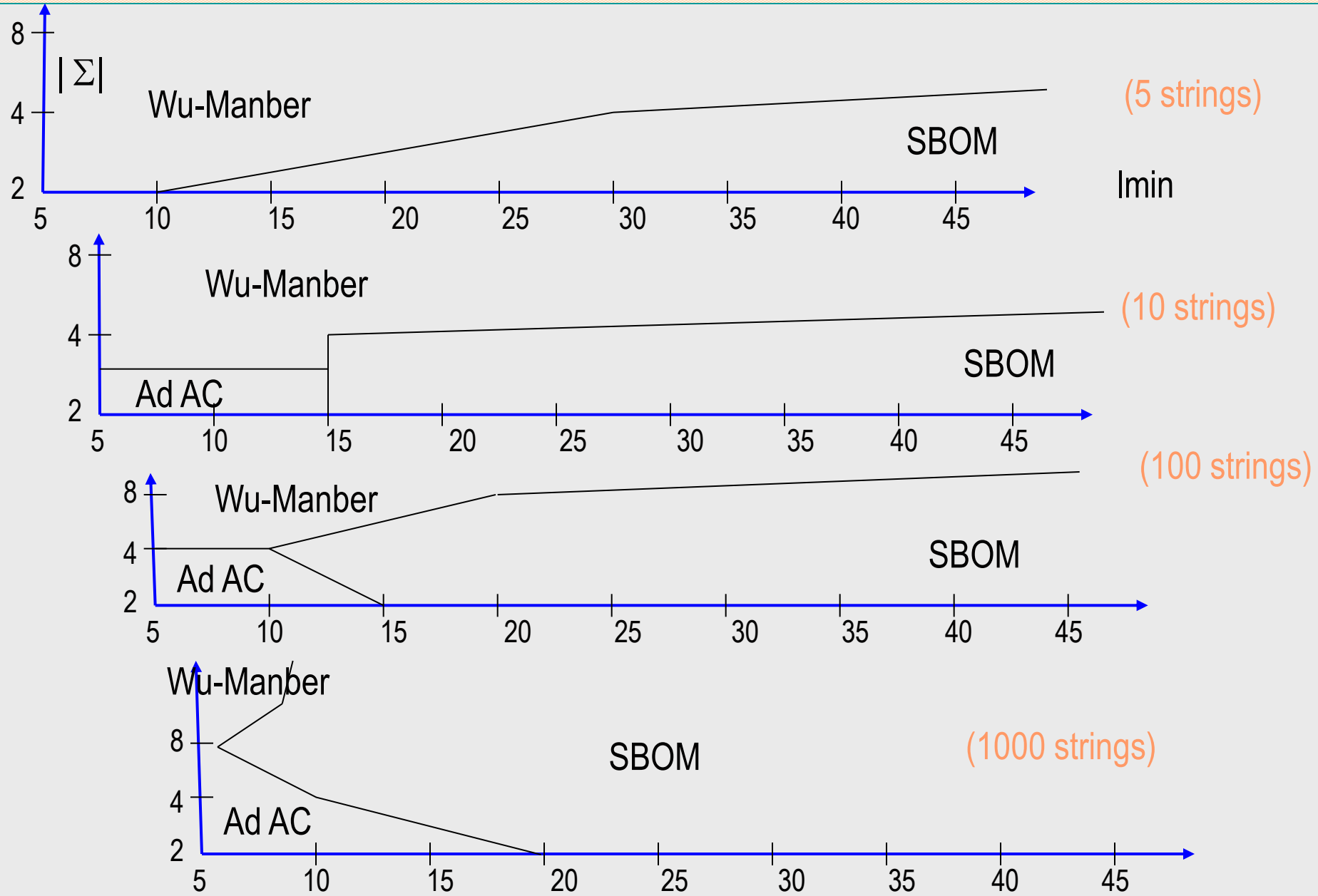
The position determined by the last character of the text
with a transition in the automata

Classes in the patterns: SBOM example

Given the patterns
ATGNARG, TRATR,TAATAAT i ANTNTGR

the Automata Factor Oracle of all 21 possible patterns is built …

Multiple string matching



Extended alphabets

Classes in the:

	text	pattern
Horspool	✓	✓
BNDM	✓	✓
BOM	✗	≈
Set-Horspool	✗	≈
SBOM	✗	≈