Mansoura University
Faculty of Computers and Information
Department of Computer Science
First Semester: 2020-2021

[MED121] Bioinformatics: Knuth Morris Pratt

Grade: Third Year (Medical Informatics Program)

Sara El-Metwally, Ph.D.

Faculty of Computers and Information,

Mansoura University,

Egypt.

AGENDA

- Knuth Morris Pratt Algorithm Basic Idea
- LPS
- KMP Algorithm Trace
- Worst Case Running Time

KNUTH MORRIS PRATT (KMP)

A C G A C A A C G T C

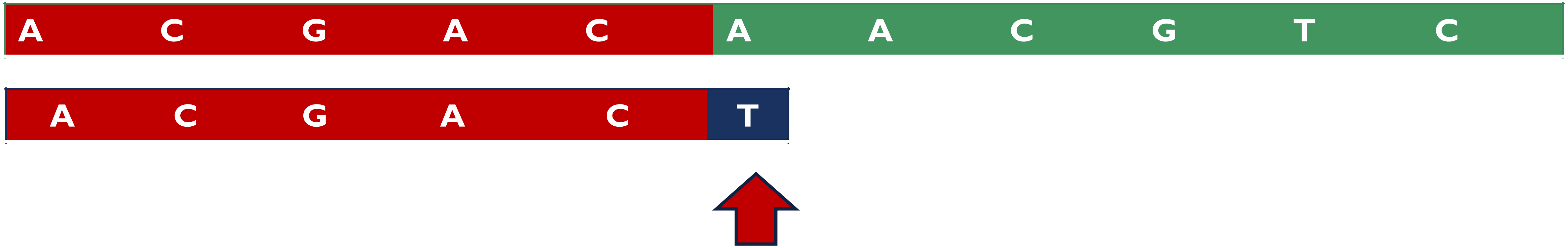
A C G A C T

KNUTH MORRIS PRATT (KMP)

A	C	G	A	C	A	A	C	G	T	C

A	C	G	A	C	T

KNUTH MORRIS PRATT (KMP)

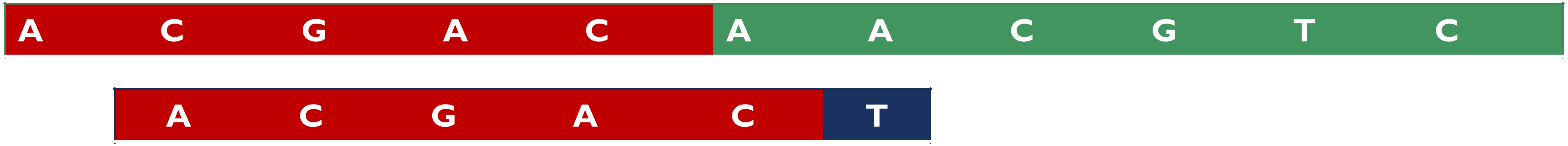


A C G A C A A C G T C

A C G A C T

mismatch position

KNUTH MORRIS PRATT (KMP)



A C G A C A A C G T C

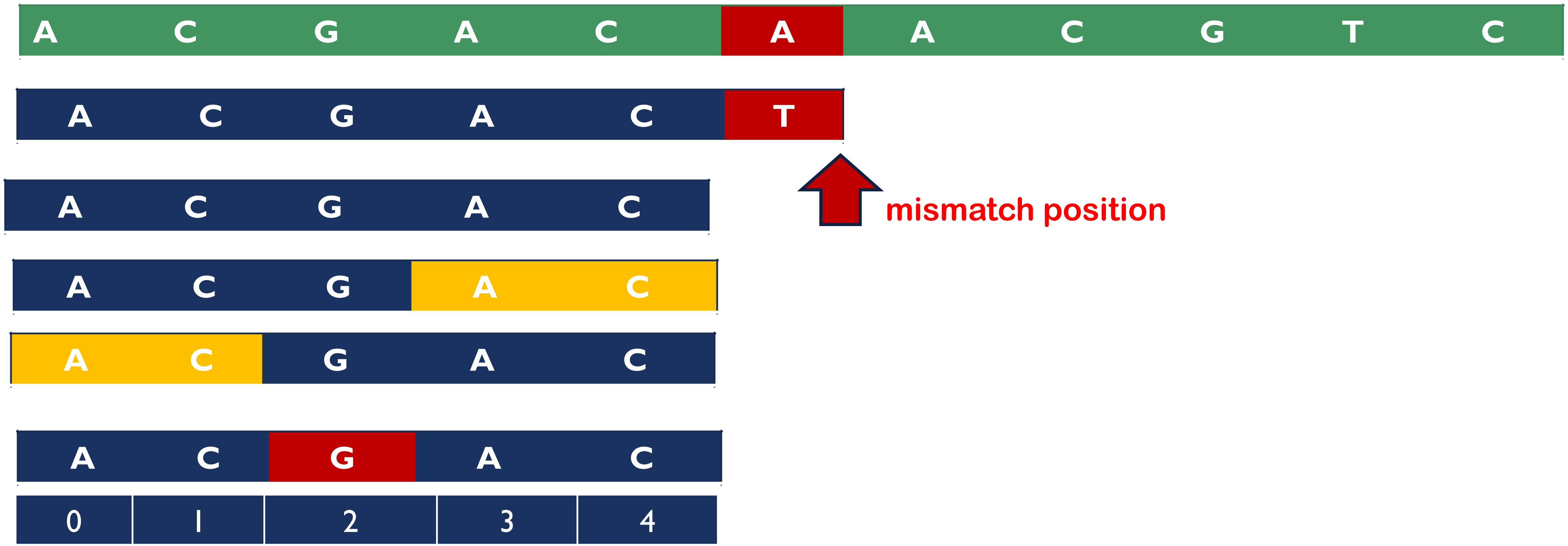
A C G A C T

KNUTH MORRIS PRATT (KMP)

A	C	G	A	C	A	A	C	G	T	C

A	C	G	A	C	T

KNUTH MORRIS PRATT (KMP)



A C G A C A A C G T C

A C G A C T

mismatch position

A C G A C

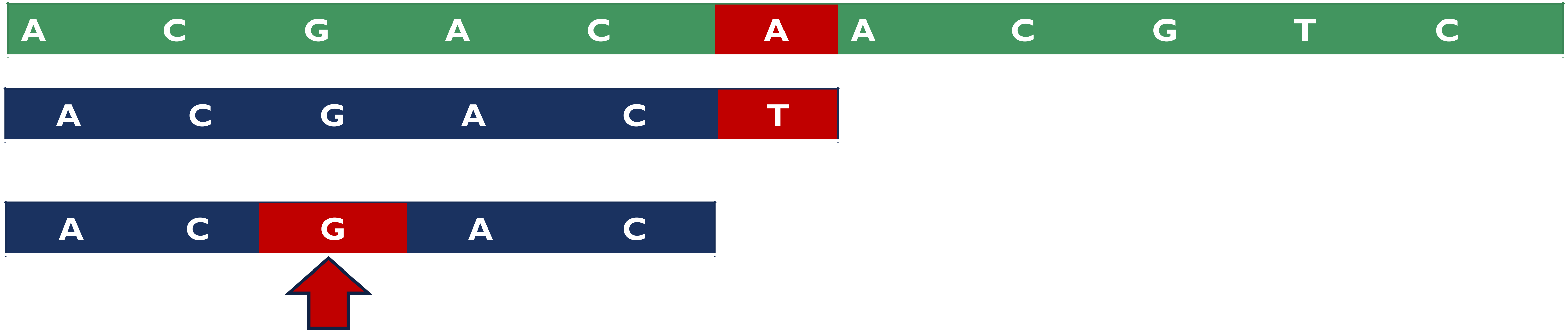
A C G A C

A C G A C

A C G A C

0	1	2	3	4

KNUTH MORRIS PRATT (KMP)



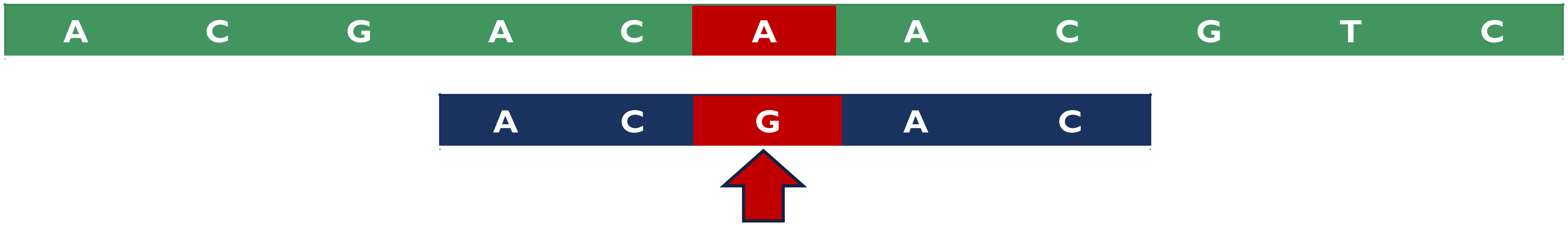
A C G A C A A C G T C

A C G A C T

A C G A C

Updated position

KNUTH MORRIS PRATT (KMP)

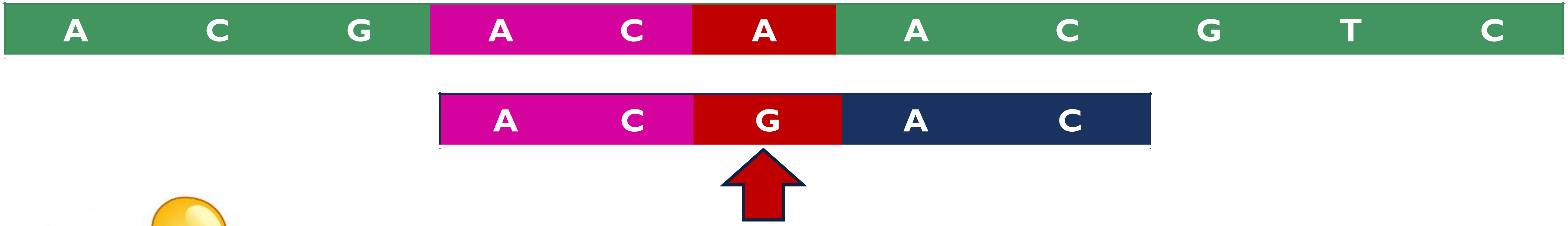


A C G A C A A C G T C

A C G A C

New comparing position

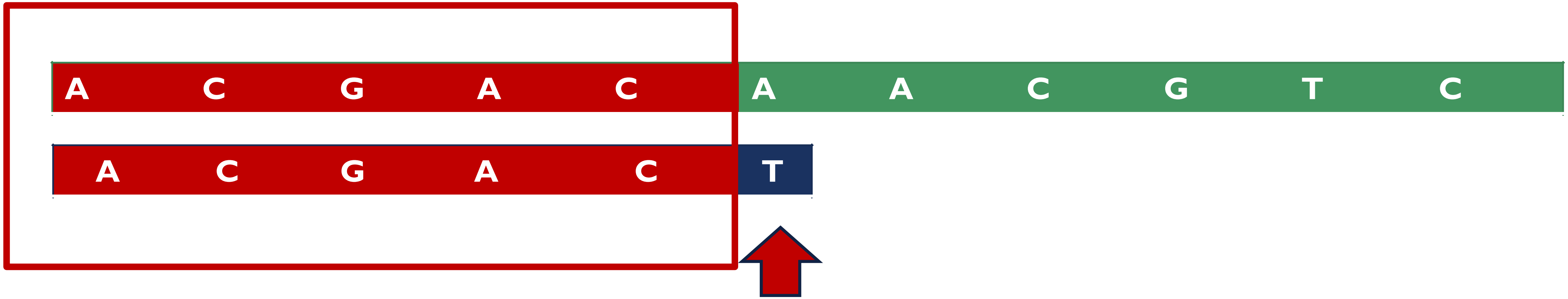
KNUTH MORRIS PRATT (KMP)



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

New comparing position

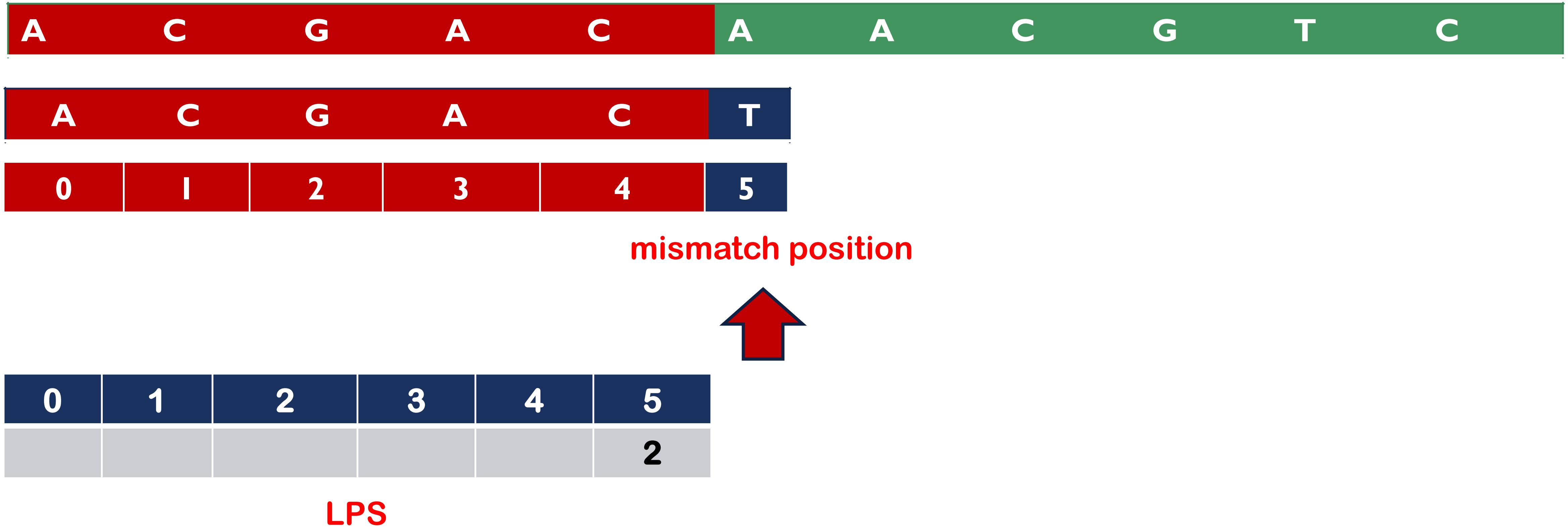
KNUTH MORRIS PRATT (KMP)



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

mismatch position

KNUTH MORRIS PRATT (KMP)



A	C	G	A	C	A	A	C	G	T	C

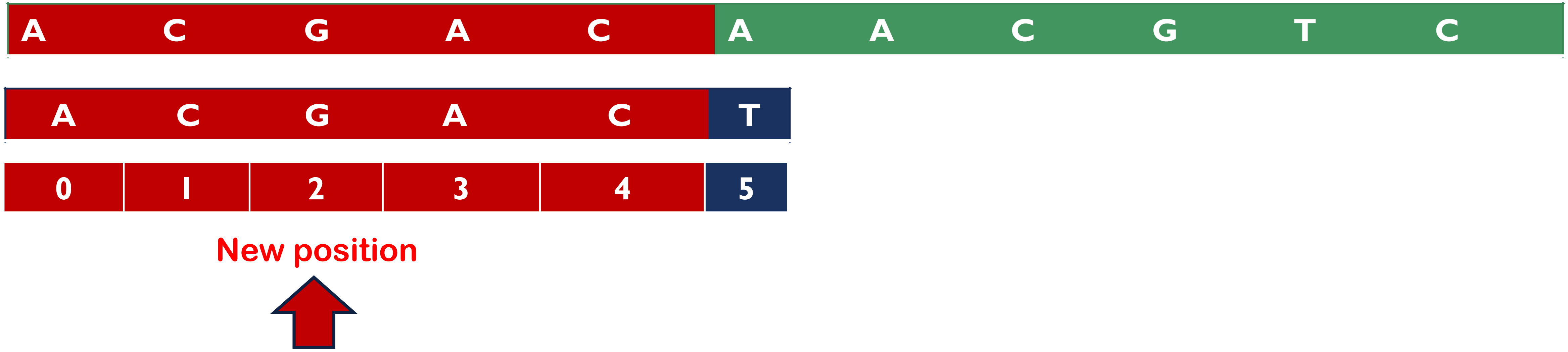
A	C	G	A	C	T
0	1	2	3	4	5

mismatch position

0	1	2	3	4	5
					2

LPS

KNUTH MORRIS PRATT (KMP)



A	C	G	A	C	A	A	C	G	T	C

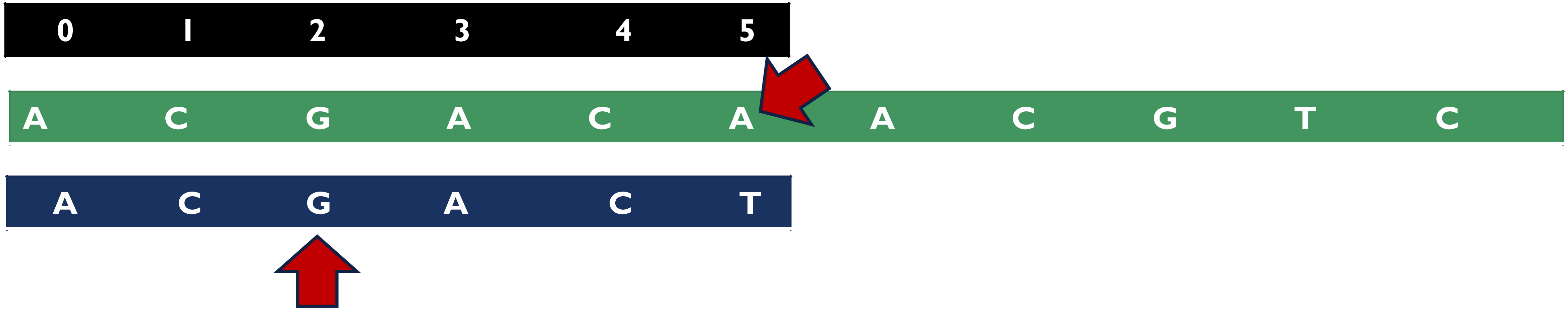
A	C	G	A	C	T
0	1	2	3	4	5

New position

0	1	2	3	4	5
					2

LPS

KNUTH MORRIS PRATT (KMP)



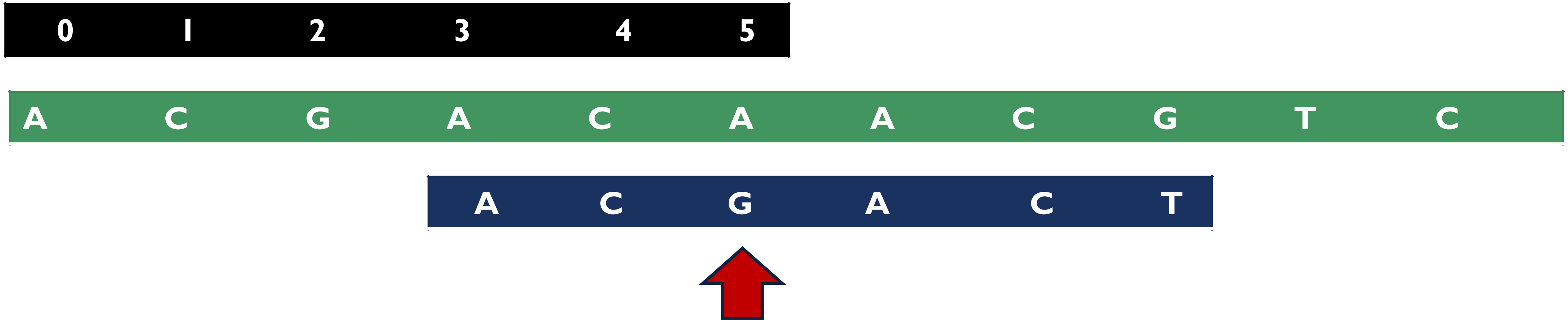
0	1	2	3	4	5					
A	C	G	A	C	A	A	C	G	T	C
A	C	G	A	C	T					

New position

0	1	2	3	4	5
					2

LPS

KNUTH MORRIS PRATT (KMP)



0 1 2 3 4 5

A C G A C A A C G T C

A C G A C T

0	1	2	3	4	5
					2

LPS

KNUTH MORRIS PRATT (KMP)

- KMP algorithm preprocesses pat[] and constructs an auxiliary lps[] of size n (same as size of pattern) which is used to skip characters while matching.

- Name lps indicates longest proper prefix which is also suffix.

- A proper prefix is prefix with whole string not allowed.

KNUTH MORRIS PRATT (KMP)

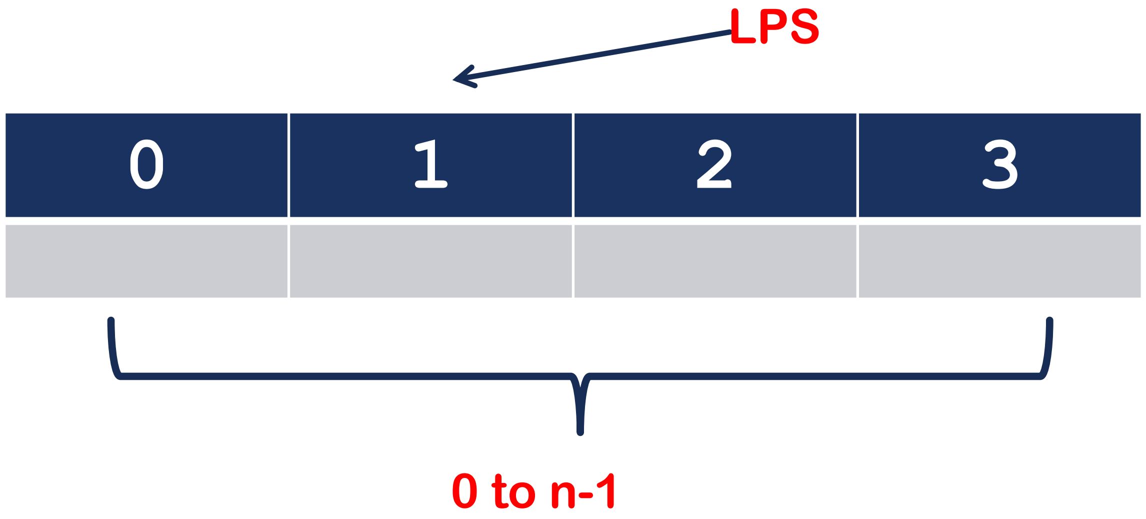
- To preprocess pattern, we focus on sub-strings of patterns that are either **prefix** and **suffix**.

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:
Pattern = AAAA, construct LPS



LPS

0	1	2	3

0 to n-1

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:

Pattern = AAAA, construct LPS

0	1	2	3
A	A	A	A

LPS

0	1	2	3
0			

Pattern = A

Prefixes = A, empty string

Suffixes = A, empty string

Proper prefix which is also a suffix = empty string

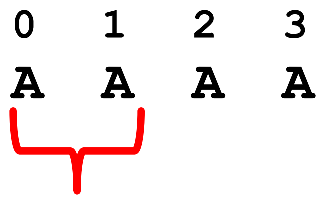
Length of empty string = 0

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:

Pattern = AAAA, construct LPS



LPS

0	1	2	3
0	1		

Pattern = AA

Prefixes = A,AA, empty string

Suffixes = A,AA, empty string

Longest proper prefix which is also a suffix = A

Length of “A” = 1

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:

Pattern = AAAA, construct LPS

LPS

0	1	2	3
0	1	2	

0 1 2 3

A A A A

Pattern = AAA

Prefixes = A,AA,AAA, empty string

Suffixes = A,AA,AAA, empty string

Longest proper prefix which is also a suffix = AA

Length of “AA” = 2

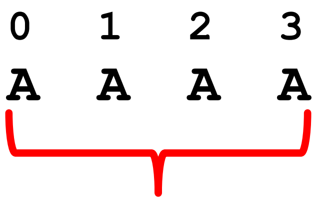
KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:
Pattern = AAAA, construct LPS

LPS

0	1	2	3
0	1	2	3



0 1 2 3
A A A A

Pattern = AAAA

Prefixes = A,AA,AAA,AAAA, empty string
Suffixes = A,AA,AAA,AAAA, empty string

Longest proper prefix which is also a suffix = AAA

Length of “AAA” = 3

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:
Pattern = ABCDE, construct LPS

0	1	2	3	4
0	0	0	0	0

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:
Pattern = AABAACAABAA, construct LPS

0	1	2	3	4	5	6	7	8	9	10
0	1	0	1	2	0	1	2	3	4	5

KNUTH MORRIS PRATT (KMP)

- lps[i] = the length of the longest proper prefix of pat[0..i] which is also a suffix of pat[0..i].

Example:
Pattern = AAACAAAAAC, construct LPS

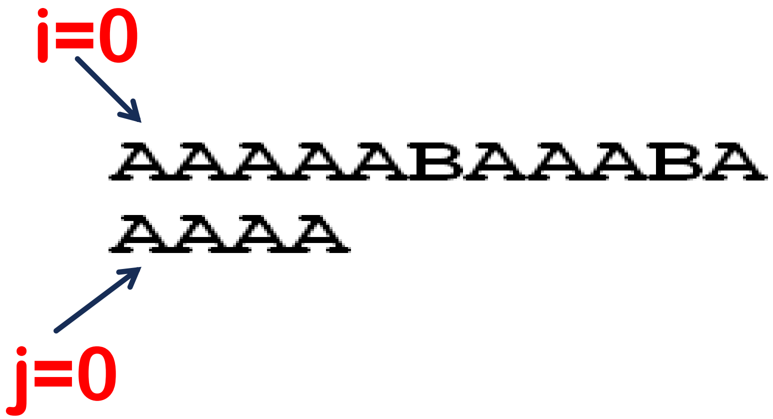
0	1	2	3	4	5	6	7	8	9
0	1	2	0	1	2	3	3	3	4

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP

0	1	2	3
0	1	2	3



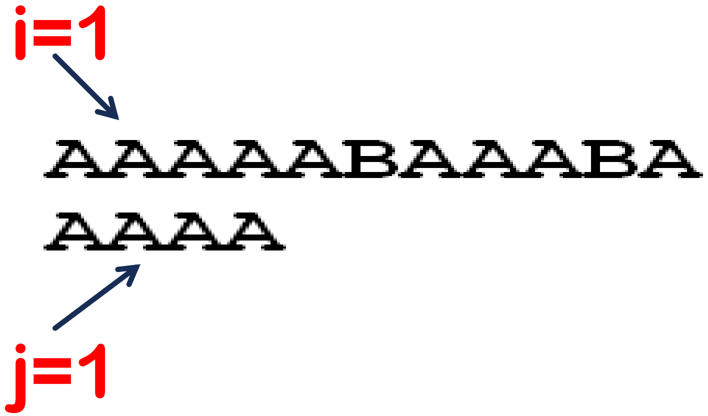
T[i]==P[j] , do i++, j++

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=1

AAAAABAAABA

AAAA

j=1

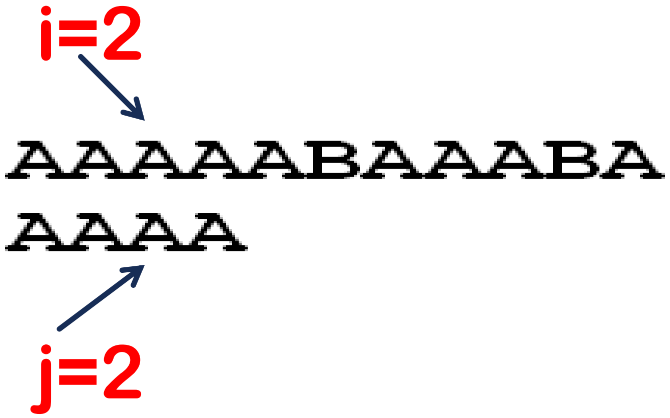
T[i]==P[j] , do i++, j++

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=2

AAAAABAAABA

AAAA

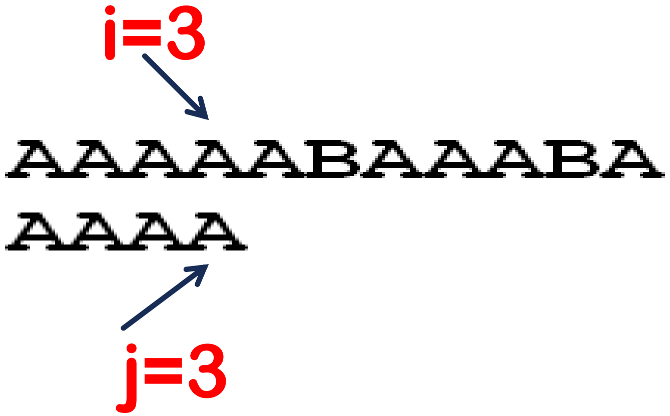
j=2

T[i]==P[j] , do i++, j++

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=3

AAAAABAAABA

AAAA

j=3

0	1	2	3
0	1	2	3

T[i]==P[j] , do i++, j++

i= 4, j=4

j=4, j==m, j = pattern length

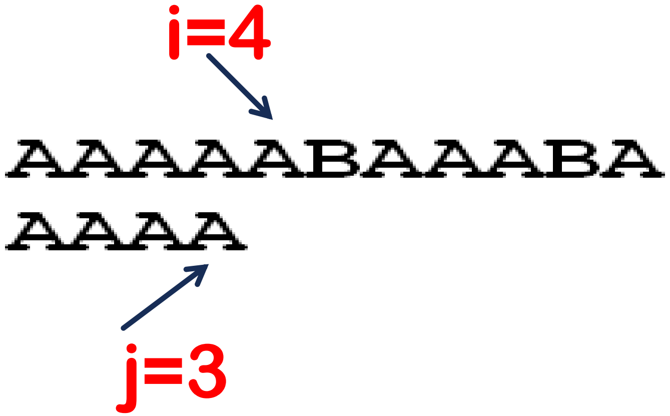
Pattern found

Reset j   j =LPS[j-1]=LPS[3]=3

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=4

AAAAABAAABA

AAAA

j=3

0	1	2	3
0	1	2	3

T[i]==P[j] , do i++, j++

i= 5, j=4

j=4, j==m, j = pattern length

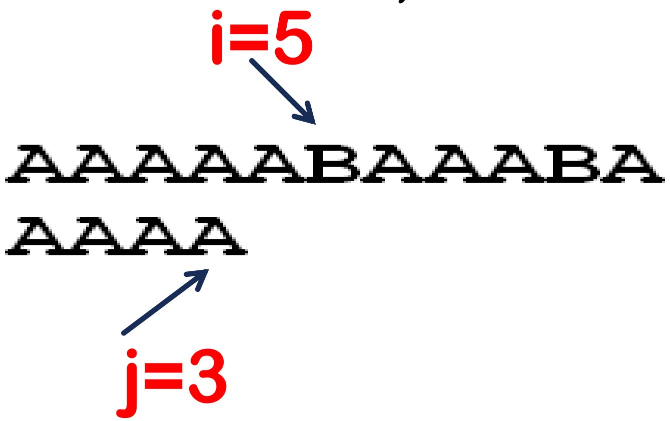
Pattern found

Reset j j =LPS[j-1]=LPS[3]=3

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=5

AAAAABAAABA

AAAA

j=3

0	1	2	3
0	1	2	3

T[i]!=P[j] , j >0, change only j

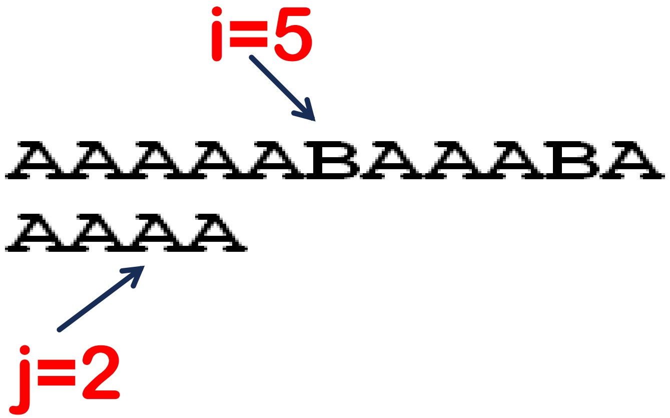
i= 5

Reset j j =LPS[j-1]=LPS[2]=2

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=5

AAAAABAAABA

AAAA

j=2

0	1	2	3
0	1	2	3

T[i]!=P[j] , j >0, change only j

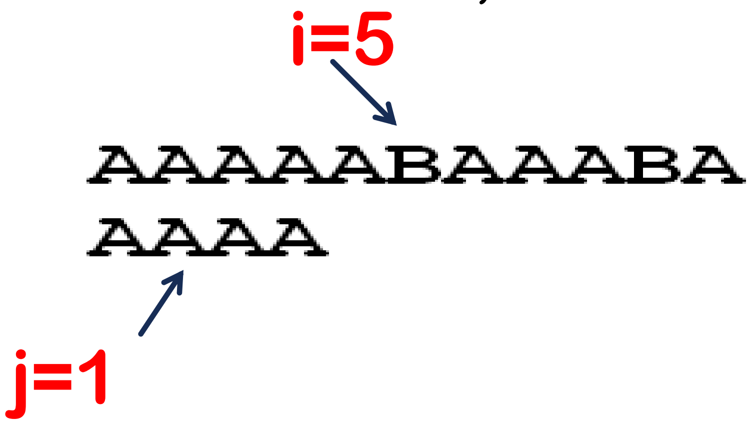
i= 5

Reset j   j =LPS[j-1]=LPS[1]=1

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



0	1	2	3
0	1	2	3

T[i]!=P[j] , j >0, change only j

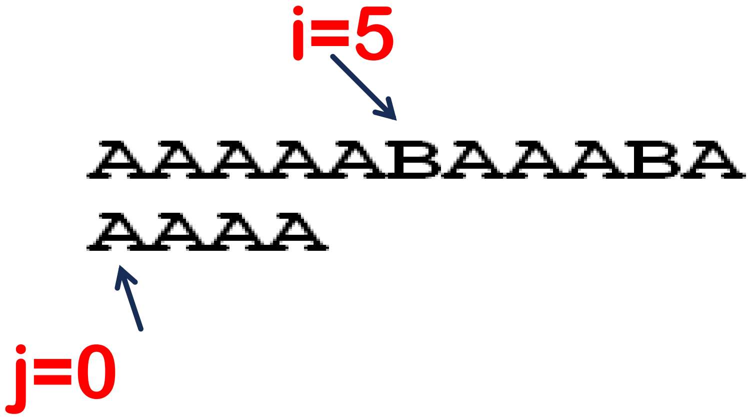
i= 5

Reset j j =LPS[j-1]=LPS[0]=0

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=5

AAAAABAAABA

AAAA

j=0

0	1	2	3
0	1	2	3

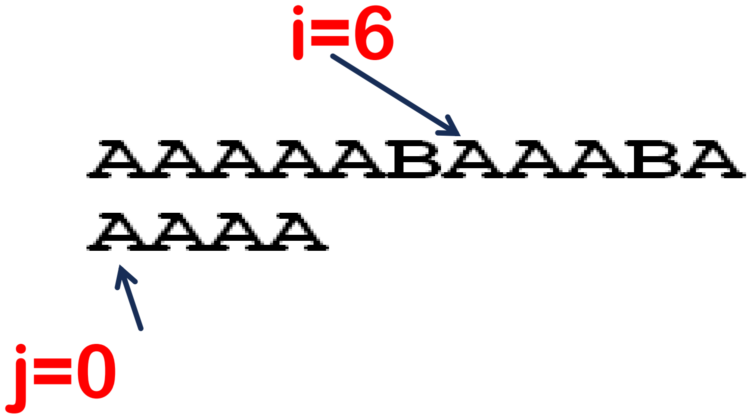
T[i]!=P[j] , j ==0, i++

i= 6, j=0

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=6

AAAAABAAABA

AAAA

j=0

0	1	2	3
0	1	2	3

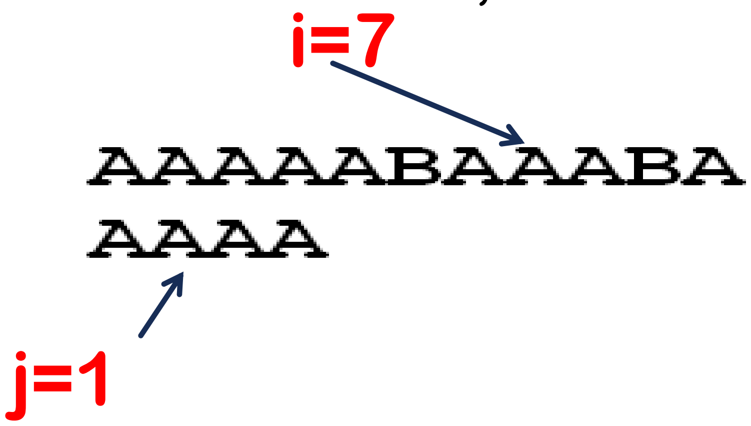
T[i]==P[j] , i++, j++

i= 7, j=1

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



0	1	2	3
0	1	2	3

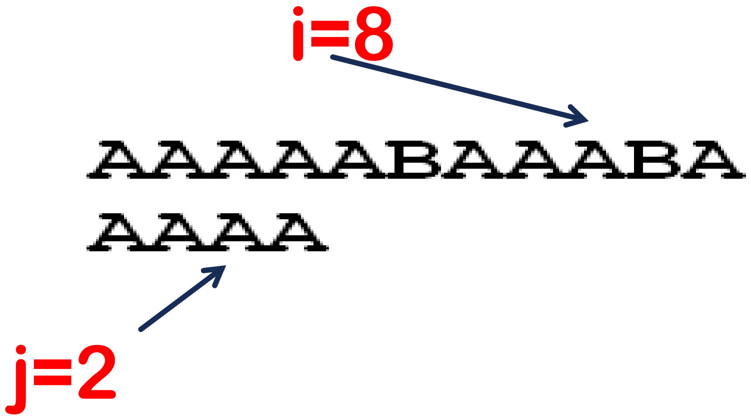
T[i]==P[j] , i++, j++

i= 8, j=2

KNUTH MORRIS PRATT (KMP)

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=8

AAAAABAAABA

AAAA

j=2

0	1	2	3
0	1	2	3

T[i]==P[j] , i++, j++

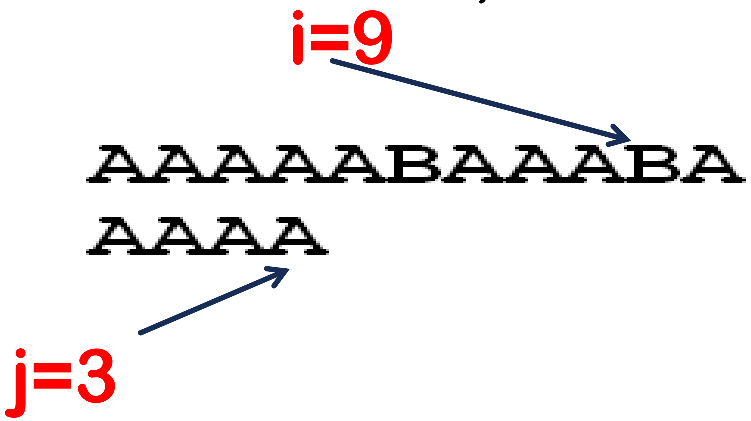
i= 9, j=3

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



T[i]!=P[j] , j>0, change only j

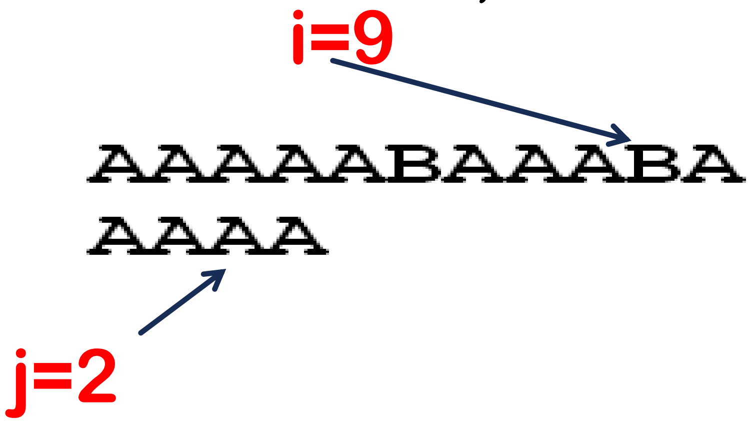
i= 9, j=lps[3-1]=lps[2]=2

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=9

AAAAABAAABA

AAAA

j=2

T[i]!=P[j] , j>0, change only j

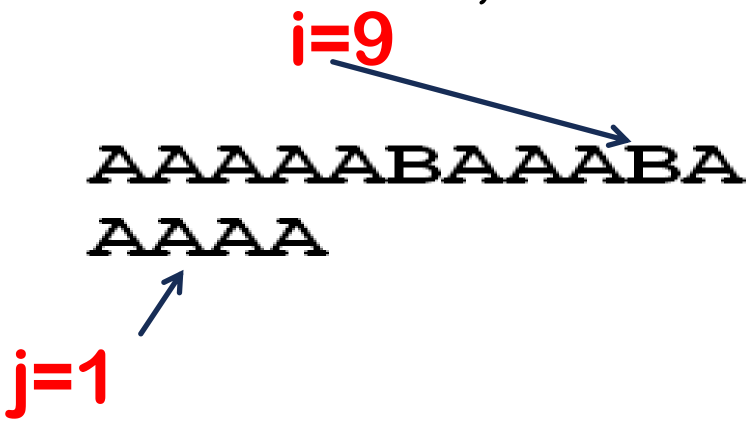
i= 9, j=lps[2-1]=lps[1]=1

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=9

AAAAABAAABA

AAAA

j=1

T[i]!=P[j] , j>0, change only j

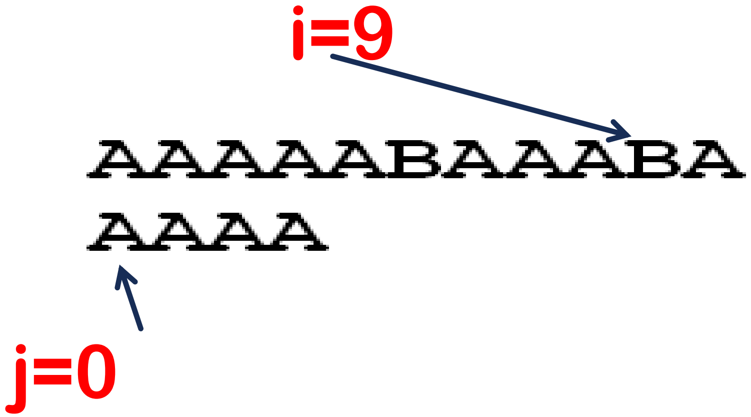
i= 9, j=lps[1-1]=lps[0]=0

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=9

AAAAABAAABA

AAAA

j=0

T[i]!=P[j] , j==0, i++

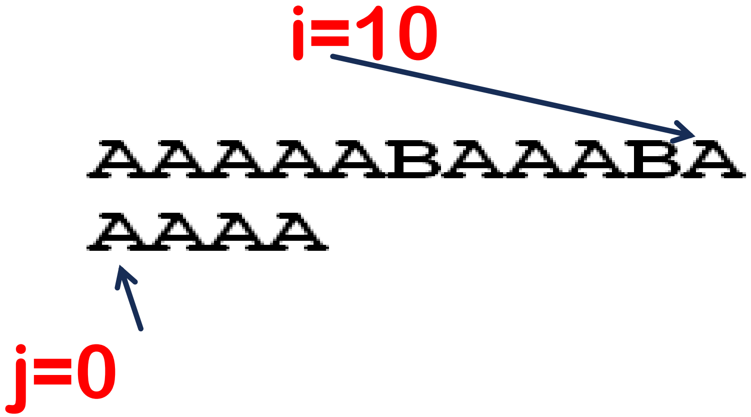
i= 10, j=0

KNUTH MORRIS PRATT (KMP)

0	1	2	3
0	1	2	3

Example:

Pattern = AAAA, Text= AAAAABAAABA, apply KMP



i=10

AAAAABAAABA

AAAA

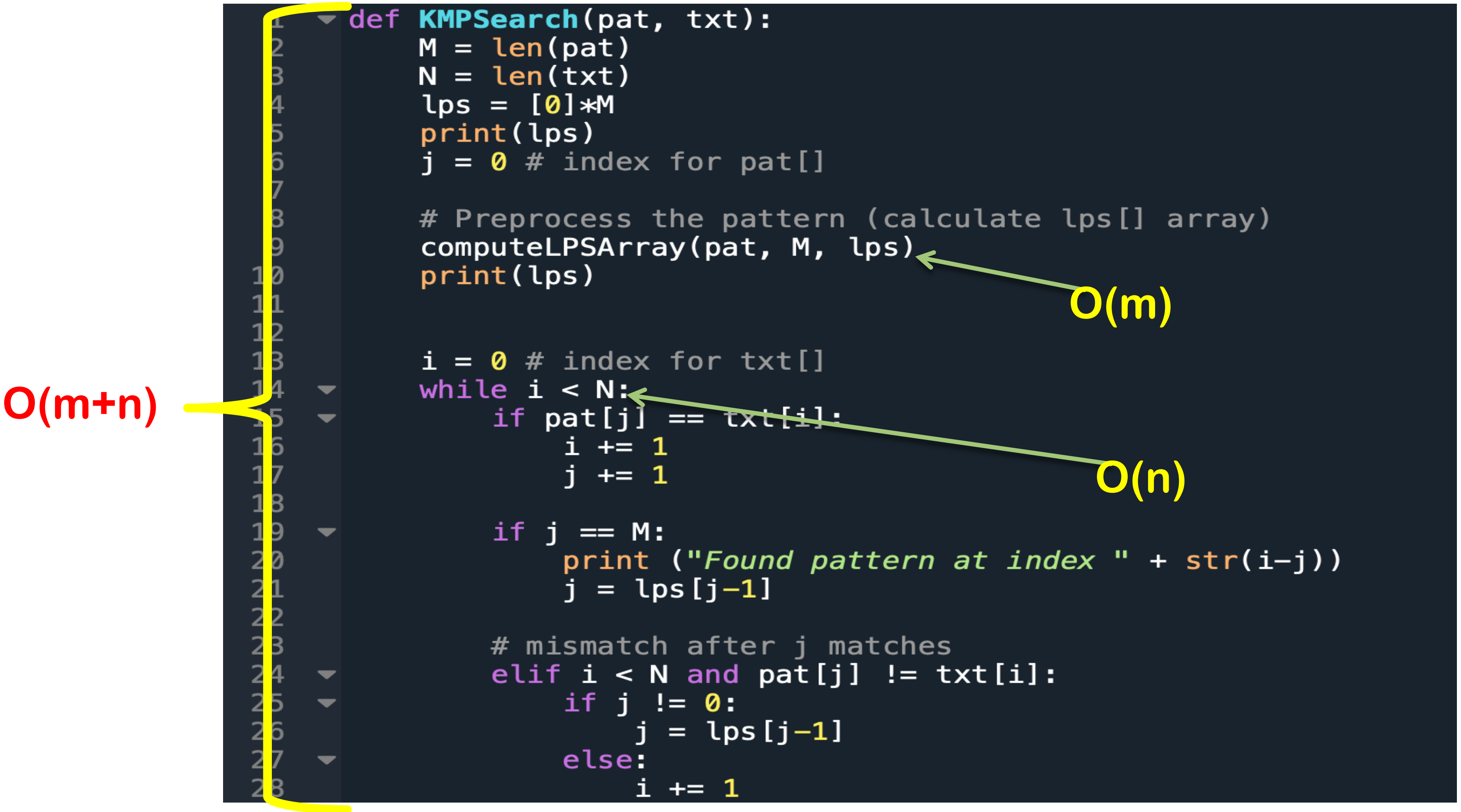
j=0

T[i]==P[j] , i++, j++

i= 11, j=1

i= 11 == length of text = n = stop algorithm

KNUTH MORRIS PRATT (KMP)



```
def KMPSearch(pat, txt):
    M = len(pat)
    N = len(txt)
    lps = [0]*M
    print(lps)
    j = 0 # index for pat[]

    # Preprocess the pattern (calculate lps[] array)
    computeLPSArray(pat, M, lps)
    print(lps)


    i = 0 # index for txt[]
    while i < N:
        if pat[j] == txt[i]:
            i += 1
            j += 1

        if j == M:
            print ("Found pattern at index " + str(i-j))
            j = lps[j-1]

        # mismatch after j matches
        elif i < N and pat[j] != txt[i]:
            if j != 0:
                j = lps[j-1]
            else:
                i += 1
```

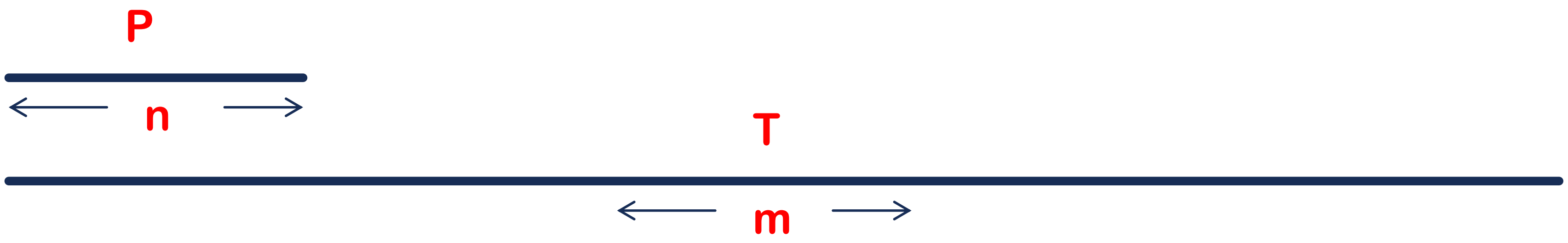
O(m+n)

O(m)

O(n)

Q

- What is the worst case scenario?



Thank you!