SURYA Gold
Date Page

Brute Force Pattern Searching or Naive Pattern Searching

→ Simplest method to solve the string matching problem.

→ Problem :- Given a 'text' string of length n and a 'pattern' string of length 'm', find all the occurrences of the pattern within the text.

→ Brute Force algo compares the pattern to the text, one character ~~at~~ at a time, until unmatching characters found.

→ This method is effective for short texts & patterns

How It works

We try to match the first character of the pattern with the first character of our main text and if we succeed try to match the 2nd character & so on.

→ If we hit a failure point, ~~if~~ shift the pattern over one character and try again.

→ When we find a match, return its starting location.

SURYA Gold

Date _____ Page _____

Algorithm BruteForceStringMatch (TEXT[0--n-1], PATTERN[0--m-1])

// Implement brute-Force String matching
// Input : An array TEXT[0---n-1] of n characters representing a text
// and an array PATTERN[0--m-1] of m characters representing a pattern.
// Output : Returns the position of first occurence of pattern or -1 if the pattern is not found.

Step 1 :- Declare variable i, j and initialize its value to 0.

Step 2 : for i=0 to (n-m) do
for j=0 to m do
if (TEXT[i+j] ≠ PATTERN[j]) then
break the loop
end for

if j==patLen then
The pattern found at the position i+1
return (i+1)
end if
end for

Step 3 : return -1

Example:

Text T : A B A B A B C C A
Pattern P : A B A B C

n → no. of characters in Text
m → no. of characters in Pattern.

n-m : Ensure that every position in the text where the pattern could possibly fit is checked.

	0	1	2	3	4	5	6	7	8
T :	A	B	A	B	A	B	C	C	A

	0	1	2	3	4
P :	A	B	A	B	C

→ n = 9 m = 5 , n-m = 9-5 = 4

Pos = 1

i = 0 to 4
j = 0 to 4
if T[0+0] ≠ P[0]
A ≠ A ×

j = 1
T[0+1] ≠ P[1] ×
B B

j = 2
T[0+2] ≠ P[2]
A A ×

j = 3
T[0+3] ≠ P[3]
B ≠ B ×

j = 4
T[0+4] ≠ P[4]
A ≠ C —
end j loop

if j == pattlen
4 == 5
end for

Date Page

Pass=2

i = 1 to 4

j = 0 to 4

If (T(i+j)) ≠ pat(j)

T(1+0) ≠ Pat(0)

B ≠ A ✓

end for

If j == Patlen

0 ≠ 5

Pass=3

i = 2 to 4

j = 0 to 4

If T(2+0) ≠ P(0)

A = A X

j = 1

If T[2+1] ≠ P[1]

B ≠ B X

j = 2

If T[2+2] ≠ P[2]

A ≠ A X

j = 3

If T[2+3] ≠ P[3]

B ≠ B X

j = 4

if T[2+4] ≠ P[4]

C ≠ C X

j = 5

5 < 4 X

If j == Patlen

5 == 5 ✓

Pattern found at i+1 position

i.e, 2+1 = 3

SURYA Gold

Date _____ Page _____

Found.

A B A B A B C C A

A B A B C

Time Complexity

Best Case :- When the pattern matches with the first set of characters of the text.

So the complexity is $O(m)$ where m is the length of pattern

Worst Case : When all characters of the text and pattern are same or only the last character is different or when pattern not found.

→ For each text position, the algorithm runs through the entire pattern before deciding a match or mismatch., leading to the complexity of $O(m*n)$.

Average Case : $O(m*n)$ – for every position of the text, a comparison is made with the pattern.

→ The no. of comparisons in the average case can be less than in the worst case but it is still proportional to mn.