

Arrays, Strings & Linked Lists Lecture 5

Saturday, 27 July 2024 3:04 PM

String Matching

pat in txt
 $|pat| = m$ $|txt| = n$

Naive: $O(m \cdot n)$

Rabin-Karp: $O(m+n)$

KMP algorithm

Knuth-Morris-Pratt

Txt :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
a	b	a	b	c	a	b	c	a	b	a	b	a	b	d
								✓	✓	✓	✓	✓	x	

$|Txt| = n = 15$

Pat :

0	1	2	3	4
a	b	a	b	d
✓	✓	✓	✓	x
		↑	↑	

$|Pat| = m = 5$

LPS
pi

0 0 1 2 0

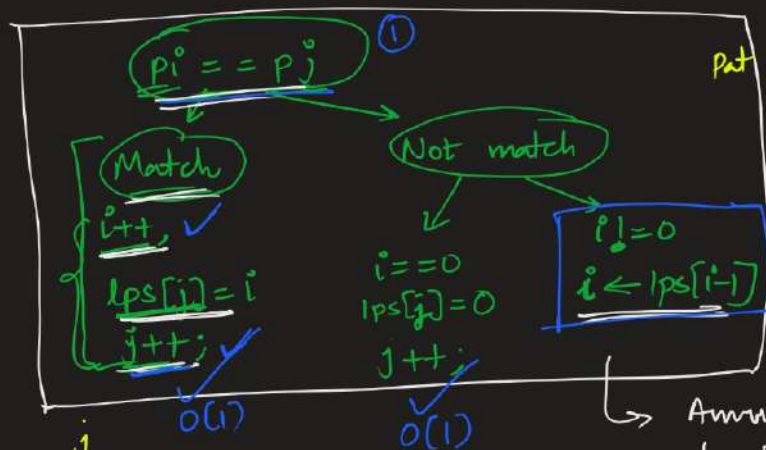
a b a b →
a b a b d

Find pre and suffix such that they are not equal to the string.

Pat: a a a a a
LPS: 0 1 2 3 4

Pat:-
 a b a b d
 j j j j j
LPS 0 0 1 2

i i-1 i j
 a b a b c a b a d
 0 0 1 2 0 1 2 3 0
 $i \leftarrow lps[i-1]$

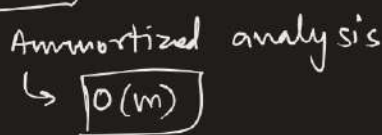


pat
 0 1 2 3 4 5
 a b a c a b
 0 0 1 0 1 2

Amortized analysis
 $\hookrightarrow O(m)$

$i \leftarrow lps[i-1]$

pat



i
 \downarrow \downarrow \downarrow
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
 a b a c a b a b d a b a b c a b
 \uparrow \uparrow \uparrow \uparrow \uparrow \uparrow
 0 0 1 0 1 2 3 2 0 1 2 3 2 0 1 2

$i \leftarrow \text{lps}[i-1]$
 i
 \downarrow
 $a \quad b \quad a \quad a \quad b \quad a \quad a \quad a$
 LPS 0 1 0 1 2 3 4 5 2
 $i \leftarrow \text{lps}[i-1]$

<https://www.geeksforgeeks.org/problems/search-pattern0205/1>

eg
 Txt : 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
 a b a b c a b c a b a b a b d
 $| \text{Txt} | = n = 15$
 Pat : 0 1 2 3 4
 a b a b d
 $| \text{Pat} | = m = 5$
 LPS 0 0 1 2 0
 j inc at most n times
 j dec
 $j \leftarrow \text{lps}[j-1]$
 $O(n)$
 $t[i] == p[j]$
 Match
 $i++$
 $j++$
 No match
 $j = 0$
 $i++$
 $j \neq 0$
 $j \leftarrow \text{lps}[j-1]$
 $j == m$
 $\text{print}(i - m)$
 $j \leftarrow \text{lps}[j-1]$
 Txt 0 1 2 3 4 5 6 7 8 9
 a b c a b c a b c a

Txt 0 1 2 3 4 5 6 7 8 9
 | | | | | | | | | |
 a b c a b c a b c a
 ↑ ↑ ↑ ↑ ↑ ↑ ↑

Pattern ↓ ↓ ↓
 a b c a

LPS 0 0 0 1

$i = 4$
 $j = 4$

T.C :- $O(n+m)$

```

def calc_lps(p):
    m = len(p)
    lps = [0]
    i = 0
    j = 1
    while j < m:
        if p[i] == p[j]:
            i += 1
            lps.append(i)
            j += 1
        else:
            if i == 0:
                lps.append(0)
                j += 1
            else:
                i = lps[i-1]
    return lps

```

$TC :- O(n+m)$
 $SC :- O(m)$

```

class Solution:
    def search(self, pat, txt):
        lps = calc_lps(pat)
        # print(lps)
        ans = []
        i = 0
        j = 0
        n = len(txt)
        m = len(pat)
        while i < n:
            if txt[i] == pat[j]:
                i += 1
                j += 1
                if j == m:
                    ans.append(i-m+1)
                    j = lps[j-1]
            else:
                if j == 0:
                    i += 1
                else:
                    j = lps[j-1]
        return ans

```