Arrays : Prefix-Sum

Mar 30, 2022

AGENDA :

* 3-4 interesting problems using the concept of Prefix Sum

Q. Given arr $= [-3, 6, 2, 4, 5, 2, 8, -9, 3, 1]$ (indices 0 1 2 3 4 5 6 7 8 9)

and Q queries,
For each query [L,R], find sum of all elements from index [L,R].

* $[0, 2] \rightarrow -3+6+2 = 5$
* $[L, R] \rightarrow O(N)$

$[0 \quad n-1]$

$[2, 4] \rightarrow 2+4+5 = 11$
$[0, 3] \rightarrow -3+6+2+4 = 9$

Input:
Q = 3 queries.

[0, 2] → ✓
[1, 4] → ✓
[0, 3] → ✓

Range Sum Query.

Brute force approach.

```
# arr as input
for i in range(0, Q):
    # L, R as input
    s=0
    for j in range(L, R+1)
        s= s+ arr[j]
    print(s)
```

→ $O(N)$

TC : $O(N * Q)$ // ≈ Quadratic n ^ Q

$O((N^2))$

$\begin{cases} N = 10^{10} \\ Q = 10^{10} \end{cases}$

* Q is not a constant.
  Q is an input.
* Q >>> N
  Q <<< N

$O(Q * N) \neq O(N^2)$
$\neq O(Q^2)$

✓ $O(N * Q)$

[2, 3, 5]

N=3

Q = [0, 1]
[0, 2]
[0, 1]
[0, 2]
[1, 2]

10^{10} times.

$Q >>> N.$

$Q <<<< N$

** IPL

	11th over	12	13	14	15	16	17	18	19	20.
Score board:	100	120	123	130	150	175	180	200	210	220

How many runs were scored in last over → ?

$220 - 210 = 10$

436 439

3 runs

Cricket 50-50.

pre

	41	42	43	44	45	46	47	48	49	50
	288	312	330	349	360	383	394	406	436	439

L R

✓ Score in 49th over $[49, 49]$ $= 436 - 406$
$= 30$.

$O(1)$

✓ Score in last 5 overs $[46, 50]$ $= 439 - 360$
$= 79$

$O(1)$

✓ Score in $(42-45)$th over $[42, 45]$ $= 360 - 288$.

42, 43, 44, 45th

Runs at the

Runs at

[L - R]

over.

end of 45th over

end of 41st over

$pre[R] - pre[L-1]$ = 72 O(1)

Score board in cricket match.

= Prefix Sum Array / Cumulative Sum Array.

41 42 43

a → b → c

a a+b a+b+c

Back to Array.

arr = [-3, 6, 2, 4, 5, 2, 8, -9, 3, 1]

Can we do some pre-processing?

[3, 2, 6, 1, 0] XX

Cumulative sum array = [3 5 11 12 12]

[0, 3] = 12

```
for i in range (Q):
```

Optimised approach.

\# arr as input.

\# Build up cumulative array / prefix sum

```
# arr = [3, -1, 0, 2, 5]
# prefix = [3, 2, 0, 0, 0]
```

2 2+0 = 2 2+2 = 4 4+5 = 9

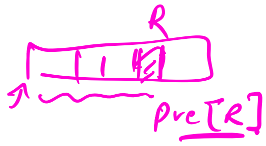
```
n = len(arr)
pre = [0] * n
pre[0] = arr[0]        // sum of elements from 0 to i = pre[i]

for i in range(1, n):
    pre[i] = pre[i-1] + arr[i].
```

\# Prefix arr is built.

```
for i in range (Q):
    # L, R as input
    if L == 0:
        print(pre[R])
    else: print(pre[R] - pre[L-1])
```

L = 0 ?



× pre[R] → Sum of elements from zero 0 to R.

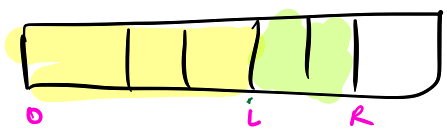
pre[L-1] - pre[R] → Negative value. ×

pre[R+1] - pre[L-1] → ×

{ Sum of elements from 0 to R ✓

= Sum of elements from 0 to L-1 ✓

+ Sum of elements from L to R.



⇒ pre[R] = pre[L-1] + Sum of elements from L to R

⇒ Sum of elements from L to R

= pre[R] - pre[L-1].

prefix[i] = Sum of elements in arr from 0 to i.

Time Complexity:

1. Building up prefix array. → $O(N)$

2. Answering Q Queries. $O(Q)$

$\underline{N} + \underline{Q}$

$\boxed{TC : O(N+Q)}$

TC : $O(N)$

$Q >>>> N$.

$O(NQ) \neq O(N^2)$
$\neq O(Q^2)$

✓ $O(N+Q)$: There is no lower order term.

Both N & Q are inputs.

Q is not a constant.
$Q \to \infty$.
$N \to \infty$.

~~$O(1)$~~

$\boxed{\text{Break Till 10:10}}$ ✓

**. Do not modify input array unless asked to do so.

Q. Equilibrium index

Given N array elements, return True if there exists a equilibrium index.

An index i is said to be equilibrium index if sum of all elements before ith index = sum of all elements after ith index

sum → integer × null
→ 0

$[0, i-1] = [i+1, N-1]$

e.g. [-7, 1, 5, 2, -4, 3, 0]   O(N)

x   N-x-1

O(N)

$N * N = N^2$

Ans = 3.

Brute force.

① For every i, ← N iterations.
check if that i is equilibrium

{Get sum of element from 0 to i-1}
== {Get sum of elements from i+1 to N-1}
→ O(N) O(1)
n.
True.

$O(N^2)$: Brute force approach.

Optimise!   → Pre[i-1]   → $\boxed{pre[n-1] - pre[i]}$ pre[-1]

0	1	2	3	4	5	6 → n-1
-7	1	5	2	-4	3	0

i

For i = 0: 0 == sum of element [1, 6] ✓

For i = 1: Sum of element [0,0] == Sum of element [2, 6]

pre[0] | pre[6] - pre[1]

For i = 3: Sum [0, 2] == [4, 6]

pre[2] | pre[6] - pre[3]

** Build up prefix array.

Pseudo-code.

```
# prefix array is built

if i==0:
    if pre[n-1] - pre[0] == 0 ?
        return True.
for i in range(1, n):
    # check whether i is equilibrium
           2* pre[i]                  i=n-1
    if pre[i-1] == pre[n-1] - pre[i]:
        return True.
```

3 | 2 -2

[3 2 -2]

45th over: 100

46th over: 200.

Q. Given an array of size N and Q queries,

Queries of format :-

→ [Start, end, 'o'/'e']

: o ⇒ sum of all odd indexed elements from [start, end]

: e ⇒ sum of all even indexed elements from [start, end]

Array → 2 3 1 -1 0 8 5 4

0 1 2 3 4 5 6 7

$-1 + 8 = 7$ ✓

[3, 6, 'o'] → Sum of elements present at odd indices in range [3, 6].

Prefix array :

0 1 2 3 4 5 6 7

→ 2 5 6 5 5 13 18 22

12

preO[] → 0 5 5 10 = 0

[0, 2] [0, 3] → 10

Build of 2 arrays :

odd_indexed. → preO[i] → Sum of element from 0 to i, with only odd indices.

i = 0 to N-1

even indexed.

preE[i] → sum of elements from 0 to i with only even indices.

[3, 6, 'o'] : preO[6] − preO[2]

~~preO[5]~~

[2, 7, 'e'] : preE[7] − preE[1]

0	1	2	3
3	6	2	−1

preO : 0 6 6 5 ← 1

Sum of elements from 0 to 0, considering only odd elements
Sum " from 0 to 1, "
" from 0 to 2, "
" from 0 to 3, "

*	0	1	2	3	4
	2	4	3	1	5
preO :	0	4	4	5	5

N/2 ×

*	0	1	2	3	4	5	6
	4	1	0	−2	3	3	5
preE :	4	4	4	4	7	7	12

arr[0]

N/2 ×

[3, 6, 'e'] : 12 − 4 = 8

preE[6] − preE[2].

L, R, 'e':   preE[R] − preE[L−1].

L, R, 'o'   preO[R] − preO[L−1]

How to create: preE and preO?

Pseudo-code :

```
preE = [0]*n
preE[0] = arr[0]
for i in range(1,n):
    if i%2 != 0:
        preE[i] = preE[i-1]
    else:
        preE[i] = preE[i-1] + arr[i]


preO = [0]*n
preO[0] = 0

for i in range(1,n):
    if i%2 == 0:
        preO[i] = preO[i-1]
    else:
        preO[i] = preO[i-1] + arr[i]
```

N elements
Q queries.

TC: $O(N+Q)$ ✓