Space complexity

23 September 2024 21:16

It is the total space taken by the algorithm w.r.t the input size

Represent - Asymptotic Notations

Calculating Space Complexity

1. Fixed Part - independent of input size
   Ex:- Constant variables, Fixed size arrays / DS
2. Variable Part - Dependent on input size
   Ex: Dynamic arrays, Recursion call stack

Space Complexity = Fixed Part + Variable Part

O(1) - Constant

(1)

```
fun(int n) {
    for (i=1 to n) {
        print("Hi")
    }
}
```

(2)

```
fun(int n) {
    int arr[10];
    for (i=1 to 10) {
        arr[i] = n
    }
}
```

n = 10
n = 100
n = 10000

O(n) - Linear

```
findMax (arr[], n) {          O(n)  O(1)
    max = arr[0];             O(1)
    for (i=1 to n) {          O(1)
        if (arr[i] > max) max = arr[i];
    }
    return max;
}
```

= O(n) + O(1)
= O(n)

O(N²) - Quadratic

Matrix Operations

Auxiliary Space

→ Extra space taken by an algo temporarily w.r.t input size.

```
findMax (arr[], n) {
    max = arr[0];
    for (i=1 to n) {
        if (arr[i] > max) max = arr[i];
    }
    return max;
}
```

S.P : O(n)
A.S : O(1)

Comparing Sorting Algo's

	A.S	S.C
Insertion Sort, Bubble Sort, Heap Sort	O(1)	O(n)
Merge	O(n)	O(n)

To reduce Space Complexity - try using in-place algos

2D Matrix O(n²) → 1D array O(n) → Constant Variables O(1)

Widely used in D.P