Catalog

EDGE

Best, worst and Expected Case

Sometimes we get lucky in life. Exams cancelled when you were not prepared, surprise test when you were prepared etc. ⟹ Best case

Some times we get unlucky. Questions you never prepared asked in exams, rain during sports period etc. ⇒ Worst case

But overall the life remains balance with the mixture of lucky and unlucky times. ⇒ Expected case.

Analysis of a search algorithm

Consider an array which is sorted in increasing order

1	7	18	28	50	180

We have to search a given number in this array and report whether its present in the array or not.

Algo 1 → Start from first element until an element greater than or equal to the number to be searched is found.

Algo 2 → Check whether the first or the last element is equal to the number. If not find the number between these two elements (center of the array). If the center element is greater than the number to be searched, repeat the process for first half else repeat for second haly until the number is found.

Analyzing Algo 1

If we really get lucky, the first element of the array might turn out to be the element we are searching for. Hence we made just one comparison.

Best case Complexity = $O(1)$

If we are really unlucky, the element we are searching for might be the last one.

Worst case Complexity = $O(n)$

For calculating Average case time, we sum the list of all the possible case's runtime and divide it with the total number of cases.

⇓

Sometimes calculation of average case time gets very complicated

Analyzing Algo 2

If we get really lucky, the first element will be the only one which gets compared

Best Case Complexity = $O(1)$

If we get unlucky we will have to keep dividing the array into halves until we get a single element (the array gets finished)

EDGE

Worst case Complexity = $O(\log n)$

What $\log(n)$? What is that

$\log(n)$ → Number of times you need to half the array of size n before it gets exhausted

$\log 8 = 3 \Rightarrow \frac{8}{2} \rightarrow \frac{4}{2} \rightarrow \frac{2}{2}$ → Can't break anymore.

$1 + 1 + 1$

$\log 4 = 2 \Rightarrow \frac{4}{2} \rightarrow \frac{2}{2}$ → Can't break anymore.

$1 + 1$

Log n simply means how many time I need to divide n units such that we cannot divide them (into halves) anymore.

Space Complexity

Time is not the only thing we worry about while analyzing algorithms. Space is equally important.

Creating an array of size n → $O(n)$ space
↳ Size of input

If a function calls itself recursively n times its space complexity is $O(n)$

URBAN EDGE

Quick Quiz → Calculate Space Complexity of a function which calculates factorial of a given number n.

Why cant we calculate Complexity in seconds?

→ Not everyone's Computer is equally powerful

→ Asymptotic analysis is the measure of how time (runtime) grows with input

Average Case