Time Complexity - II

Today's Content

- Comparing two Algos
  - Using Execution time
  - Using iterations & graphs
- Why Big O needed ?
  - Why lower orders terms are ignored ?
  - Why const. coefficients terms are neglected
  - Issues in Big O
  - Worst Case
- Space Complexity
- TLE
  - Why TLE occurs ?
  - How to approach any given problem ?
  - Importance of constraints

Comparing Algos using Execution time

Given N elements, sort them in increasing order.

$N = 10^4$ (input size)

Algo1 {Tripathi}

Exec. time

15 sec
↓
(Windows XP)
↓
(Macbook M_2)
↓
7 sec
↓
(C++)
↓
7 sec
(Top of Hot volcano)
↓
Mt. Everest
↓
5 sec

Algo2 {Aradhana}

10 sec
↓
(Macbook M_2)
↓
10 sec
↓
(Python)
↓
(C++)
5 sec
Mt. Everest
↓
5 sec

Execution time : It depends on so many external factors, hence we generally don't compare 2 algos using exec. time.

Comparing using Iterations & Graphs

Algo1 {Sagar}

iterations : $100 \log_2(N)$

Algo2 {Shashi}

$N/10$

for N=10

$100 \log_2(10)$	→	~300
$10/10$	→	1

for $N <= 3900$ Algo2 is better

$N > 3900$ Algo1 is better

Google results : 1M+

Baby Shark : 10.8B views

Asymptotic Analysis of Algorithms

↳ Performance analysis of Algos for very large inputs

Use Big O notation

1. Calculate iterations
2. Take higher-order terms
3. Ignore const. coefficients

Why neglect lower-order terms ?

$N^2 + 10N$

input size	Total iterations	% of lower order terms
$N = 10$	200	$\frac{10N}{N^2+10N} = \frac{100}{200} = 50\%$
$N = 100$	$10^4 + 10^3$	$\frac{10^3}{10^4+10^3} = \frac{1}{11} \sim 9\%$
$N = 10^3$	$10^6 + 10^4$	$\frac{10^4}{10^6+10^4} = \frac{1}{101} < 1\%$
$N = 10^6$		Insignificant

Why ignore const. coefficients

$10N$

$N = 10^5$

$N \to 10^5$

$10N \to 10^6$

Claim 1: for all inputs, we can decide which Algo is better. ✗

Claim 2: for all inputs $>= x$, we can decide which Algo is better. ✓

final Claim: When we compare 2 Algos using Big O Algo1 will always be better than Algo2 for all input values above a certain threshold point.

→ After threshold, Big O holds.
→ Please don't worry about threshold.

Issues in Big O

$2N^2 + 4N$

$O(N^2)$

$3N^2$

$O(N^2)$

$2N^2 + 4N - 2N^2$

$4N$ is better than N^2

$3N^2 - 2N^2$

→ If we have same Big O for 2 Algos, then Big O will fail.

Worst Case

Ques: search of an element = K

```
bool search ( a[], K) {
    for(i=0; i< a.size(); ++i) {
        if(a[i] == K)
            return true
    }
    return false
}
```

total iterations = N

O(N)

best-case scenario iteration = 1

worst-case scenario iteration = N

Manger → {Task}

5 days → Best Case

30 days → Worst case

BREAK : 10:00 - 10:10 PM

Space Complexity

Code → Time Complexity → no. of iteration → Big O notation

Code → Space Complexity → total space → Big O notation

```
def func(N) {
    int x = 0
    int y = N
    long p = 5
}
```

int → 4 Bytes

long → 8 Bytes

total = 16 Bytes O(1)

```
def func (int a[]) {
    int m = a[0];
    for (i=1; i<a.size(); ++i) {
        m = max(m, a[i]);
    }
    return m;
}
```

m → 4 B

i → 4 B

a[] → 4 × n B (n → no. of elements in a)

total space = 8 + 4n B

= 8 B

O(1)

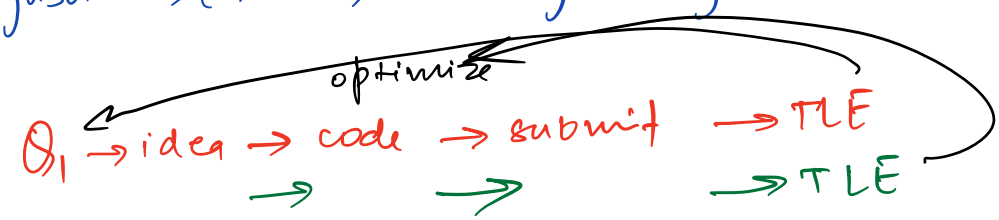
```
void func (int a[], int n) {
    int pf[n];          ----> 4n
    pf[0] = a[0];
    for (i=1; i<n; ++i) {   var i -> 4
        pf[i] = a[i] + pf[i-1]
    }
}
```

total space = 4n+4

O(N)

Time Limit Exceeded - TLE

Ayush -> (Amazon) -> Hiring Challenge -> 3Q (1.5 hrs)



Online Editors -> 1 GHz -> 10^9 instructions / sec

```
S=0   +1
for (int i=0;  i<n;  ++i) {
     +1      +1xn    +1xn
    S=S+i;
      +1xn
}
```

iteration = N

total instructions = 2+3N

Approx :

1 iteration $\longrightarrow$ 10 instructions

Approx 2 :

1 iteration $\longrightarrow$ 100 instructions

1 sec $\rightarrow$ 10^9 instructions

1 sec $\rightarrow$ $[10^7 - 10^8]$ iterations

Importance of Constraints

$1 <= N <= 10^6$

int a[N][N]

10^{12} ✗

Algo $\rightarrow$ $O(N^2)$

iteration $= (10^6)^2 = 10^{12}$ iterations — TLE

Algo 2 $\rightarrow$ $O(N \log N)$

$10^3 \approx 2^{10}$

iteration $= 10^6 \times \log_2 10^6$

$= 10^6 \log_2 2^{20} = 10^6 \times 20$

$\approx 2 \times 10^7$ iterations ✓

$1 <= N <= 100$

$O(N^3)$

iteration $= (100)^3 = 10^6$ ✓

↳ no need of $O(N)$, $O(N^2)$