Mathan Raj.
Arumugam Rajkumar.
1002231625 -

1.

Statement $x=1$, with time operation $O(1)$.

Since there are two nested loops.

Outer loop,

n times from $i=1$ to $i=n$.

inner loop,

n times from $j=1$, to $j=n$.

Statement $x = x+1$, runs n times.

Total no. of executions = $n \times n$.

Total runtime, $T(n) = n \times n$ ($T(n) = n^2$.

$$T(n) = \sum_{i=1}^{n} \sum_{j=1}^{n} 1$$

∴ Total executions is n^2. ∴

The runtime of algorithm = $O(n^2)$.

2. The time taken by f(n) is measured, and tested for various n values, ex: n = 1, 2, 3. . . . The measured time is noted for each plotted values of n on x-axis and time taken on y axis. A quadratic pattern plot is received with consistence $O(n^2)$. So, time complexity is quadratic.

Graph:

(i) The graph shows a quadratic trend where n increases which validates the time complexity $\Theta(n^2)$.

(ii) That curve matches the timing data.

3. Upper bound (Big O): is specified by Light blue dashed line that slightly exceeds the polynomial fitted curve. This indicates that the fn is bounded above $O(n^2)$. This includes time complexity in the worst case as fast as n^2.

Lower bound: Orange dashed line shows the actual curve which is bounded by $\Omega(n^2)$. This shows that n^2 to complete

Tight bound (Big theta): Since the upper and lower bounds are quadratic that follows actual timing data. So, the runtime is $\Theta(n^2)$.

4. Identifying n_0:

The plot quadratic trend n_0 is marked with vertical dashed lines at $n = 4800$. the data shows variability due to overhead or noise in system.

n_0 represents threshold whose the algorithm's performance aligns with expected n^2 complexity.

4. The modified fn will take more time to run because of the added operation $y = i+j$, within inner loop which is consistent with time $O(1)$, since its executed same no of times like other operations. the asymptotic complexity does not change.

5. No, it won't affect the runtime analysis result since the operation is $O(1)$ and therefore the runtime remains $O(n^2)$ original and modified function perform nested loops for n^2 times in a constant time.

Therefore,

Big O: $O(n^2)$

Big omega: $\Omega(n^2)$

Big theta: $\Theta(n^2)$.