# **Homework 2**

**Name:** Adeem Adil Khatri

**ID:** 21801174

**Section:** 3

**System Specs:**

* Windows 10 Pro, x64 bit Operating System, x64 based-processor
* 16.0 GB RAM
* Intel® Core™ i7-5600 CPU @ 2.60GHz (4 CPUs)
* Intel® HD Graphics 5500

***Key for Tables:***

1. The key is close to the beginning
2. The key would be near the middle
3. The key is near the end
4. The key doesn’t exist

**Fig 1.0 (A) - Table for the execution time of Linear Search**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CALCULATED** | | | | |
| ***LINEAR SEARCH O(N)*** | | | | |
|
| Input Size (N) | *Algorithm Time (ms)* | | | |
| 1 | 2 | 3 | 4 |
|
| 10000 | 0.01 | 0.03 | 0.04 | 0.06 |
| 100000 | 0.01 | 0.17 | 0.25 | 0.29 |
| 1000000 | 0.01 | 1.4 | 3.08 | 3.21 |
| 10000000 | 0.01 | 15.27 | 32.62 | 34.75 |
| 100000000 | 0.02 | 172.56 | 336.19 | 343.44 |
|  |  |  |  |  |

**Fig 1.1 (A) - Graph for the execution time of Linear Search**

**Fig 1.0 (B) - Table for the execution time of Binary Search**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CALCULATED** | | | | |
| ***BINARY SEARCH O(logN)*** | | | | |
|
| Input Size (N) | *Algorithm Time (ms)* | | | |
| 1 | 2 | 3 | 4 |
|
| 64 | 0.00028 | 0.00026 | 0.00022 | 0.00034 |
| 1024 | 0.00045 | 0.00121 | 0.00039 | 0.00051 |
| 16384 | 0.00064 | 0.00063 | 0.00056 | 0.00067 |
| 262144 | 0.00087 | 0.00074 | 0.00076 | 0.00123 |
| 4194304 | 0.00138 | 0.00107 | 0.00174 | 0.00154 |

**Fig 1.1 (B) - Graph for the execution time of Binary Search**

## Average, Best and Worst Situations (Case)

For **Linear Search**, we compute upper bound on running time of an algorithm in the worst case. Here, the worst case occurs when the element to be searched for is not present in the array, or is the last element of the array. Cases 4 and 3 show the worst case occurring, as shown in Fig 1.1(A). Therefore the worst case time complexity of linear search would be O*(N).*  
For the average case, we take all possible inputs and get the execution time for all of these inputs. Sum these calculated values, and divide them by (n+1). The average case should occur when the element is found somewhere in the center of the algorithm. The Case 2 line in Fig 1.1(A), shows a sloped-straight line indicating the average case of linear search is also O*(N)*.  
For the best case, we find the lower bound of the execution time of the algorithm. We should know the case that makes least number of tasks to be executed, here Case 1 line in Fig 1.1(A) shows a constant horizontal line which indicates that the best case of linear search is also O*(1)*.

For **Binary Search**, we compute the worst case when the element to be found doesn’t exist in array, or it is not present at either the start or at the end of the array. The average cost of a successful search is almost the same as the worst case, both being roughly equal to log N. The Case 2, 3 and 4 in Fig1.1 (B), show that a graph of log n is observed which indicates that the worst and average case are the same, in big-O notation, is O(*log N*). The Best case in binary search is computed when the element to be found exists in the center of the array, as the element is found in one run. In the Case 1 in Fig 1.1(B), if we are to exclude the anomalies on the graph then it would indicate the best case, with the time complexity O(*1*).

**Fig 2.0 (A) - Table for the run time of (Expected) Linear Search**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EXPECTED** | | | | |
| ***LINEAR SEARCH O(N)*** | | | | |
|
| Input Size (N) | *Algorithm Time (ms)* | | | |
| 1 | 2 | 3 | 4 |
|
| 10000 | 0.000001 | 10000 | 10000 | 10000 |
| 100000 | 0.000001 | 100000 | 100000 | 100000 |
| 1000000 | 0.000001 | 1000000 | 1000000 | 1000000 |
| 10000000 | 0.000001 | 10000000 | 10000000 | 10000000 |
| 100000000 | 0.000001 | 100000000 | 100000000 | 100000000 |

**Fig 2.1 (A) - Graph for the run time of (Expected) Linear Search**

**Fig 2.0 (B) - Table for the run time of Binary Search**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **EXPECTED** | | | | |
| ***BINARY SEARCH O(logN)*** | | | | |
|
| Input Size (N) | *Algorithm Time (ms)* | | | |
| 1 | 2 | 3 | 4 |
|
| 64 | 1.80618 | 0.00005 | 1.80618 | 1.80618 |
| 1024 | 3.01030 | 0.00005 | 3.01030 | 3.01030 |
| 16384 | 4.21442 | 0.00005 | 4.21442 | 4.21442 |
| 262144 | 5.41854 | 0.00005 | 5.41854 | 5.41854 |
| 4194304 | 6.62266 | 0.00005 | 6.62266 | 6.62266 |

**Fig 2.1 (B) - Graph for the run time of Binary Search**

# Comparison and Discussion:

For binary search the best case is given by the Case 2, and a graph of constant horizontal line is obtained in such case. Whereas, the worst cases occur in Fig 2.1(B) when the graph is a log (N) shape and here Case 1, 3 and 4 are overlapping each other so they all indicate worst cases. Moreover, the expected graph shows similarity to the graphs in Fig 1.1(B) for the same size of input N in worst case at, and average case, indicating the time complexity for binary search to be logN.

For Linear search, the best case shows a constant line as seen by Case 1 in Fig 2.1(A). The expected graphs for linear search overlap in average and worst cases as seen by Case 2, 3, and 4 in Fig 2.1(A). There is resemblance in graphs for Fig 2.1(A) and Fig 1.1(A), excluding the anomalous points, hence indicating that the time complexity for linear search is O *(N).*