Search Test Lab Report

Names: Jian Han

**1. Linear Search**

We know from class that the theoretical time complexity of linear search over *unordered lists* is:

|  |  |  |
| --- | --- | --- |
| **Best Case** | **Worst Case** | **Average Case** |
| *1* | *N* | *N/2* |

**Q1:** Increasing the number of trials and the value of N

1. Run experiments with an increasing value of N (from 1000 to 10,000). Does increasing N affect how many trials you have to run to get accurate results? Explain.

When performing experiments with an increasing value of N for linear search over unordered lists, increasing N can affect the number of trials required to obtain accurate results. In general, as N increases, the variability of the results tends to decrease, but the amount of time required to perform each trial also increases. Therefore, it may be necessary to run more trials to obtain accurate results when N is large.

1. Write down the number of trials that seem to have worked well for N=10,000.

|  |
| --- |
| **Number of Trials** |
| 10,000 |

**Q2:** Linear Search Time Complexity Plot (Unordered List)

|  |
| --- |
| *Linear SearchInsert plot here* |

**Q3:** Does the order of the data in the list affect the number of comparisons? In the table below, guess the time complexity of Linear Search on an *Ordered List.*

|  |  |  |
| --- | --- | --- |
| **Best Case** | **Worst Case** | **Average Case** |
| *1* | *N* | *N/2* |

Linear Search Time Complexity Plot (Ordered List)

|  |
| --- |
| Linear Search |

**Conclusion:**

Regarding the order of the data in the list, the number of comparisons performed during linear search will not be affected. The best case time complexity of linear search on an ordered list is O(1), the worst-case time complexity is O(N), and the average-case time complexity is O(N/2).

**2. Binary Search**

We know from class that the theoretical time complexity of binary search over *ordered lists* are:

|  |  |  |
| --- | --- | --- |
| **Best Case** | **Worst Case** | **Average Case** |
| *1* | *log\_2(N)* | *log\_2(N)* |

**Q4:** Binary Search Time Complexity Plot

|  |
| --- |
| Binary Search  *Insert plot here* |

**Conclusion:** What do your results tell you about the average-case complexity of Binary Search?

It shows that the average case time complexity of binary search over ordered lists can be calculated as O(log2(N)), the same as the worst case.

This is because in the average case, binary search will need to make roughly log2(N) comparisons to find the target element. This is because with each comparison, the search space is cut in half, so it takes log2(N) comparisons to reduce the search space to a single element.

**3.** **Median**

Q5: We hypothesize that the time complexity of find\_median is:

|  |  |  |
| --- | --- | --- |
| **Best Case** | **Worst Case** | **Average Case** |
| O(N) | O(N^2) | O(N^2/2) |

**Justification:**

1. Best case scenario:

*Happens when... the list has odd length and the median is the first element.*

1. Worst case scenario:

*Happens when...the list has even length and the median is the last element or the median is not in the list.*

1. Average case scenario:

O(N^2/2).

Find\_median Time Complexity Plot

|  |
| --- |
| Median Search  *Insert plot here* |

**Conclusion:** Did your results support your hypothesis? If not, why not, and how does it change your original hypothesis?

Yes the results support my hypothesis.