Cmpe-300

Ahmet Hacıoğlu,2020400132

November 12, 2023

1 Introduction

Q6

Algorithm 1

```
procedure (arr)
    for i = 1 to length(A) - 1 do
        key ← A[i]
        left ← 0
        right ← i − 1
        while left ≤ right do   ▷ Binary search to find the correct position
            mid ← ⌊(left + right)/2⌋
            if A[mid] > key then
                right ← mid − 1
            else
                left ← mid + 1
            end if
        end while
        for j = i - 1 down to left do ▷ Shift elements for space for key
            A[j + 1] ← A[j]
        end for
        A[left] ← key                 ▷ Insert the key at correct position
    end for
end procedure
```

The modified insertion sort algorithm reduce the total number of comparisons with a binary search to efficiently find the correct position for the current element in the sorted part of the array.

Drawbacks:

While the number of comparisons is reduced, the number of other operations, such as element shifts, may increase. The binary search introduces additional overhead.

The binary insertion sort is more complex than the traditional insertion sort.

The binary search loop adds complexity, making the code harder to understand and maintain.

In summary, the modified insertion sort reduces the total number of comparisons but introduces increased other operations and complexity. The choice between the modified and traditional insertion sort depends on the specific characteristics of the data being sorted and the performance requirements of the application.

Complexity

In the worst case, the binary insertion sort performs a binary search for each element to find its correct position in the sorted part of the array. The binary search takes $O(\log n)$ comparisons. Since this process is repeated for each element in the array, the overall time complexity is $O(n \log n)$.
The time complexity is $O(n^2)$ for traditional insertion sort and $O(n \log n)$ for binary insertion sort.