Counting Stars and Suddenly Counting Sort

Your Name

December 21, 2023

1 Full Counting Sort Code

Here is the complete implementation of the Counting Sort algorithm in Python:

```python
def counting_sort(arr):
    max_value = max(arr)
    frequency_counter = [0] * (max_value + 1)

    for element in arr:
        frequency_counter[element] += 1

    sorted_array = []
    for value, frequency in enumerate(frequency_counter):
        for _ in range(frequency):
            sorted_array.append(value)

    return sorted_array
```

2 Analysis of Complexity

Consider an example array $A = [2, 2, 2, 2, 2, 2, 2, 2]$. For this array, the maximum value (k) is 2, and the number of elements (n) is 8. The frequency counter for this array will be $[0, 0, 8]$.

- The outer loop iterates over each index in the frequency counter, which has a length of $k + 1$. In our example, it will iterate 3 times (for values 0, 1, and 2).
- The inner loop iterates a total number of times equal to the sum of all frequencies. In this case, the inner loop will execute 8 times in total, all for the value 2.
- The crucial observation is that the total number of iterations of the inner loop across all iterations of the outer loop is equal to n. Thus, the complexity of these nested loops is not quadratic ($O(n^2)$), but linear with respect to the number of elements ($O(n)$).

Therefore, even with the nested loops, the time complexity of this portion of the Counting Sort algorithm remains $O(n + k)$, and not $O(n^2)$.