ECE374 Fall2020

Lab5: Counting Sort

Name: Zhang Yichi 3180111309

Nov. 2^{nd} 2020

1 Introduction

In this lab, we will implement counting sort that accepts negative numbers. Counting sort is a sorting algorithm with time complexity O(k+n).

2 Python Code for Counting Sort

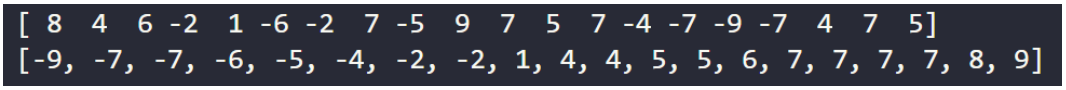
```python
def counting_sort(A):
    minx = min(A)
    maxx = max(A)
    k = maxx-minx+1
    C = []
    B = []
    for i in range(k):
        C.append(0)
    for i in A:
        C[i-minx] += 1
    for i in range(k):
        while C[i] > 0:
            B.append(i+minx)
            C[i] -= 1
    return B
```

minx is the minimum number in A.
maxx is the maximum number in A.
k is the number of buckets needed. I use maxx-minx+1 to normalize these numbers and minimize the buckets needed.

First, create k empty buckets, C. Then, go through A, put the numbers in A into buckets in C and record the numbers of the numbers in each buckets. Finally, take the numbers out in order and append them to B. B is the sorted array.

3 Test Example for Counting Sort

```
A = np.random.randint(-10,10,20)
print(A)
A = counting_sort(A)
print(A)
```



```
[ 8  4  6 -2  1 -6 -2  7 -5  9  7  5  7 -4 -7 -9 -7  4  7  5]
[-9, -7, -7, -6, -5, -4, -2, -2, 1, 4, 4, 5, 5, 6, 7, 7, 7, 7, 8, 9]
```

Figure 1: counting sort test result

Here I generate 20 random numbers from -10 to 10, sort them with the counting_sort function I developed and output them to see the results.

4 Time Complexity Analysis

The first loop is O(k). The second loop is O(n). The third loop consists of two loops. The outer loop is O(k). The inner loop iterates C[i]. C contains the number of numbers from A so the sum of C[i] is n. Thus, the inner loop iterates n times in total. Therefore, the third loop is O(k+n). In all, the time complexity of this counting sort is O(k+n).