Algorithms and Data Structures

Arnav Singh

March 2020

7 Sheet 7

7.1 Sorting in Linear Time

(a) Implemented Counting Sort in zip file

(b) Implemented Bucket sort in zip file

(c) Interval-Counter

Algorithm 1 Interval Counter

```
1: function INTERVAL-COUNTER($A, n$)    ▷ Where A - array, n - size of array and a,b are the intervals
2:     Let C[0...k] be the array
3:     for $i = 0$ to k do
4:         C[i]=0    ▷ Initialize array to 0
5:     end for
6:     for $j = 1$ to n do
7:         C[A[j]]=C[A[j]]+1    ▷ Count the number of occurrences
8:     end for
9:     for i=1 to k do
10:        C[i]=C[i]+C[i-1]    ▷ Get the cumulative sum of all occurrences
11:    end for
12: Count← $C[b] - C[a-1]$    $returnCount$
```

(d) Implemented in Words.cpp check zip file

(e) Worst Case of Bucket Sort

Bucket sort is an algorithm which depends on the uniform distribution of elements. The wort case scenario of Bucket sort would be when the elements are not distributed uniformly, which means the elements are all allocated into the same bucket.In this case the algorithm has to sort everything.Hence, if insertion sort is being used worst case will take $\theta(n^2)$ to sort the array. An example of such an array would be :
A:(0.157, 0.123, 0.179, 0.112, 0.139, 0.165, 0.182, 0.101)

(f) Bonus due Wednesday, will be uploaded

7.2 Radix Sort

(a) Implemented in Radix.cpp check zipfile

(b) Prove time and space complexity

The Hollerith's version of Radix Sort starts from the MSB and moves right to the LSB.Here I have implemented a recursive approach the numbers are inserted into the buckets and if a bucket contains more than one element the Bucket-sort function is called again and we move one digit to the right and call the bucket sort. This is repeatedly done untill we have only one element in each bucket. At the end the list is concatenated.

Time Complexity The worst case is when all numbers are the same, they will be in the same bucket and will take $\theta(n * L)$ where L is the length of the list and n is the max number of digits of each element.
Space Complexity The space complexity will be $\theta(n * k)$, the space required by the algorithm is $\theta(n)$ and we have k recursions therefore $\theta(n * k)$

(c) Bonus due Wednesday will be uploaded separately

References- - Geeksforgeeks
- Tutorialspoint
- Youtube
- Quora