CSCI 115 Lab

Week 6- Counting and Radix Sort

- Professor: Dr. Matin Pirouz
  Email: mpirouz@csufresno.edu

- TA: Shreeja Miyyar
  Email:shreejarao12@mail.fresnostate.edu

Table of Contents

Counting Sort

- It is an algorithm for sorting the numbers according to the keys that are small integers.
- It uses an auxiliary array to store the count of the distinct input numbers. The size of the auxiliary array should be equal to the maximum element in the input array plus one.
- It is not a comparison-based algorithm.

- Time Complexity:
    - O(n+k), if k=O(n). This means that counting sort is not performant for very large values of k
- Space complexity:
    - O(n+k)

Counting Sort algorithm

- Get the maximum element from the input array.
- Define an auxiliary array with array size of (maximum element)+1
- Set all the elements in the auxiliary array to 0.
- Loop through the input array and store the count of each element in the auxiliary array such that, the element is the index of the auxiliary array and the value is the count of that element.
- Loop through the auxiliary array and cumulatively add the values of the array.

  For e.g. -> Array[i] = Array[i] + Array[i - 1]
- Define an output array having the same size as input array.
- Loop from the end of the input array
  - Get the value at that position of the input array.
  - Using that value as the index to the Auxiliary array, get the value at that position of the auxiliary array and decrement it by 1.
  - Now using this value as index to the output array insert the value at that position.

Example

Input Array = [2, 0, 1, 0]

Expected Output = [0, 0, 1, 2]

Initial Auxiliary array = [0, 0, 0]

Auxiliary array after adding the count of each element = [2, 1, 1]

Auxiliary array after cumulation = [2, 3, 4]

Building output array:

Iteration 1:

Output array = [x, 0, x, x]

Auxiliary array = [1, 3, 4]

Iteration 2:

Output array = [x, 0, 1, x]

Auxiliary array = [1, 2, 4]

Iteration 3:

Output array = [0, 0, 1, x]

Auxiliary array = [0, 2, 4]

Iteration 4:

Output array = [0, 0, 1, 2]

Auxiliary array = [0, 2, 3]

Radix Sort

- It is an integer sorting algorithm that sorts the data based on the individual digits of the integers.
- It uses Counting sort subroutine to perform the sorting.
- It uses an auxiliary array similar to counting sort which stores the number of occurrences of the digits of the input integers.
- It is not a comparison-based algorithm.

- Time Complexity:
  - O(d*(n+k))

    where d is the number of digits of the maximum input number
    N is the length of the input array
    K is the range of digits -> (0 - 9)

Radix Sort algorithm

- Get the maximum element from the input array.
- Get the number of digits d of the maximum element.
- For loop from i = 1 to d
  - Define an auxiliary array with array size of 10
  - Set all the elements in the auxiliary array to 0.
  - Loop through the input array and store the count of that digit of each element in the auxiliary array such that, the element is the index of the auxiliary array and the value is the count.
  - Loop through the auxiliary array and cumulatively add the values of the array.

    For e.g. -> Array[i] = Array[i] + Array[i - 1]
  - Define an output array having the same size as input array.
  - Loop from the end of the input array
    - Get the value at that position of the input array.
    - Using that value as the index to the Auxiliary array, get the value at that position of the auxiliary array and decrement it by 1.
    - Now using this value as index to the output array insert the value at that position.
  - Return the array

Example

Input Array = [11, 5, 240]

Expected Output = [5, 11 240]

Initial Auxiliary array = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Auxiliary array after adding the count of each element = [2, 1, 1]

Building output array:

Iteration 1:

Input Array = [11, 5, 240]

Auxiliary array after adding the count of the units digits = [1, 1, 0, 0, 0, 1, 0, 0, 0, 0]

Auxiliary array after cumulation = [1,2,2,2,2,3,3,3,3,3]

Output array = [240, 11, 5]

Iteration 2:

Input Array = [240, 11, 5]

Auxiliary array after adding the count of the tens digits = [1, 1, 0, 0, 1, 0, 0, 0, 0, 0]

Auxiliary array after cumulation = [1,2,2,2,3,3,3,3,3,3]

Output array = [5, 11, 240]

Iteration 3:

Input Array = [5, 11, 240]

Auxiliary array after adding the count of the hundreds digits = [2, 0, 1, 0, 0, 0, 0, 0, 0, 0]

Auxiliary array after cumulation = [2,2,3,3,3,3,3,3,3,3]

Output array = [5, 11, 240]

Lab Assignment

1. Write a program that takes a list and sorts it using Counting Sort.

   Hint: Use the counting sort algorithm mentioned in the slides to write the program.

2. Write a program that takes a list and sorts it using Radix sort.

   Hint: Use the Radix sort algorithm mentioned in the slides to write the program. You can use a part of counting sort function to do the sorting based on the digits.

3. Compare the time complexity:

Hint: You can use clock() function to record execution time.

4. Fill out the report sheet.

   Hint: Write a detailed report as per the template and provide an elaborate explanation on how the execution time is different for each algorithm.

Coding guidelines

- Counting Sort:
  - In the main function provide the input array to be sorted.
  - Create a function which takes the input array as an argument and performs the necessary operations to sort the array using counting sort.
  - Use the clock method in the main function to find the execution time for different input arrays.
- Radix Sort:
  - In the main function provide the input array to be sorted.
  - You can create a function which takes the input array as an argument and performs the necessary operations to sort the array using counting sort.
  - You can try reusing the counting sort method to perform sorting.
  - Use the clock method in the main function to find the execution time for different input arrays.

Questions?