Home Algorithms DS GATE Interview Corner Q&A C C++ Java Books Contribute Ask a Q About
Array Bit Magic C/C++ Articles GFacts Linked List MCQ Misc Output String Tree Graph

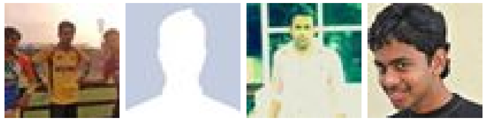
Bucket Sort

Bucket sort is mainly useful when input is uniformly distributed over a range. For example, consider the following problem.
Sort a large set of floating point numbers which are in range from 0.0 to 1.0 and are uniformly distributed across the range. How do we sort the numbers efficiently?

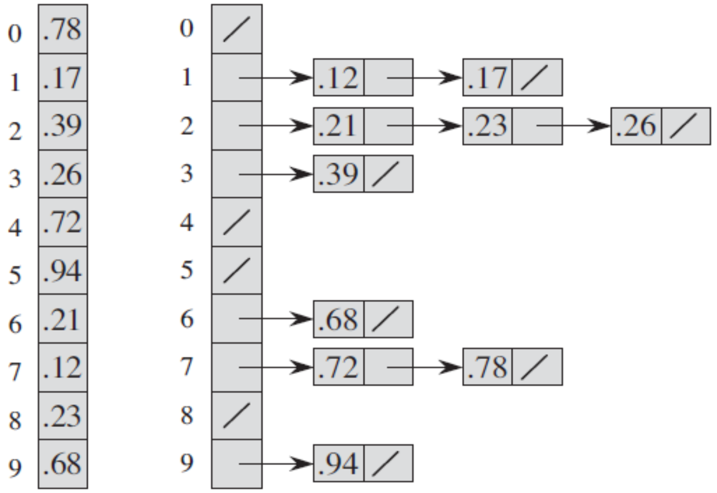
A simple way is to apply a comparison based sorting algorithm. The lower bound for Comparison based sorting algorithm (Merge Sort, Heap Sort, Quick-Sort .. etc) is $\Omega(nLogn)$, i.e., they cannot do better than nLogn.
Can we sort the array in linear time? Counting sort can not be applied here as we use keys as index in counting sort. Here keys are floating point numbers.
The idea is to use bucket sort. Following is bucket algorithm.

```
bucketSort(arr[], n)
1) Create n empty buckets (Or lists).
2) Do following for every array element arr[i].
.......a) Insert arr[i] into bucket[n*array[i]]
3) Sort individual buckets using insertion sort.
4) Concatenate all sorted buckets.
```

Following diagram (taken from CLRS book) demonstrates working of bucket sort.



Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking



Input array Buckets created for input array

Time Complexity: If we assume that insertion in a bucket takes O(1) time then steps 1 and 2 of the above algorithm clearly take O(n) time. The O(1) is easily possible if we use a linked list to represent a bucket (In the following code, C++ vector is used for simplicity). Step 4 also takes O(n) time as there will be n items in all buckets.
The main step to analyze is step 3. This step also takes O(n) time on average if all numbers are uniformly distributed (please refer CLRS book for more details)

Following is C++ implementation of the above algorithm.

```
// C++ program to sort an array using bucket sort
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

// Function to sort arr[] of size n using bucket sort
void bucketSort(float arr[], int n)
{
    // 1) Create n empty buckets
    vector<float> b[n];

    // 2) Put array elements in different buckets
    for (int i=0; i<n; i++)
    {
       int bi = n*arr[i]; // Index in bucket
       b[bi].push_back(arr[i]);
```

Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists

```
    }

    // 3) Sort individual buckets
    for (int i=0; i<n; i++)
       sort(b[i].begin(), b[i].end());

    // 4) Concatenate all buckets into arr[]
    int index = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < b[i].size(); j++)
          arr[index++] = b[i][j];
}

/* Driver program to test above funtion */
int main()
{
    float arr[] = {0.897, 0.565, 0.656, 0.1234, 0.665, 0.3434};
    int n = sizeof(arr)/sizeof(arr[0]);
    bucketSort(arr, n);

    cout << "Sorted array is \n";
    for (int i=0; i<n; i++)
       cout << arr[i] << " ";
    return 0;
}
```

Output:

```
Sorted array is
0.1234 0.3434 0.565 0.656 0.665 0.897
```

References:

Introduction to Algorithms 3rd Edition by Clifford Stein, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest

http://en.wikipedia.org/wiki/Bucket_sort

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST

Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen’s Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3
- Sort n numbers in range from 0 to n^2 – 1 in linear time

19 Tweet 3 1

Writing code in comment? Please use **ideone.com** and share the link here.

6 Comments GeeksforGeeks

Sort by Newest ▾

705 Subscribe

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 2 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 5 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 30 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 32 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 57 minutes ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

Join the discussion…

Nisarg • 14 days ago

The main step to analyze is step 3. This step also takes O(n) time on average
distributed? How?

Insertion Sort is quadratic.

⌃ | ⌄ • Reply • Share ›

GeeksforGeeks Mod ➦ Nisarg • 14 days ago

The main point to note here is there are total n elements and are unifor
divided in n buckets. To get some idea, consider the ideal situation wh
sum of 1<sup>2</sup> for n times is O(n). Please refer CLRS book fo

⌃ | ⌄ • Reply • Share ›

Suman ➦ GeeksforGeeks • 11 days ago

It is not clear to me why we need an insertion sort in step 3. Wh
can make sure we are inserting in right place i.e. we will keep t
So we don't need step 3 at all.

⌃ | ⌄ • Reply • Share ›

GOPI GOPINATH ➦ Suman • 2 days ago

To keep the list sorted u need to do insertion sort...
consider u r having 0.03 and 0.05 in a bucket and a new
the same bucket....u need to check 0.03 but also 0.05 a
insertion sort....so worst case complexity in ur case is a

⌃ | ⌄ • Reply • Share ›

Kartik ➦ Suman • 11 days ago

Your idea looks good. Please note that the array elemer
loop. So it doesn't seem to be an improvement, but ano
complexity.

⌃ | ⌄ • Reply • Share ›

Suman ➦ Kartik • 11 days ago

Insertion sort has worst case complexity O(^2) (http://e
the approach that I mentioned has worst case complexi

⌃ | ⌄ • Reply • Share ›

✉ Subscribe Ⓓ Add Disqus to your site