GeeksforGeeks

A computer science portal for geeks

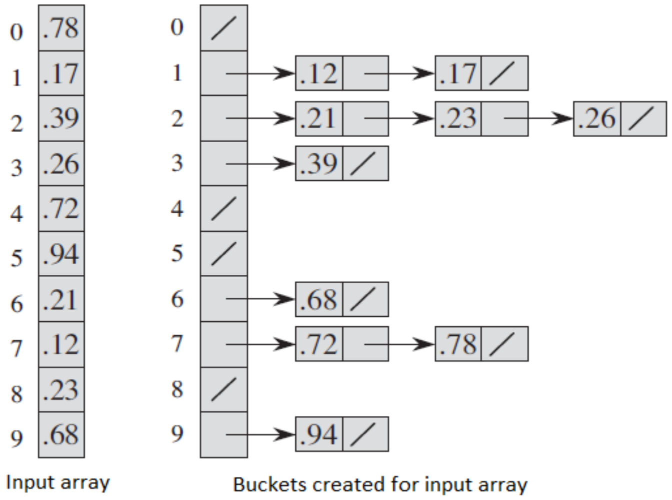
Bucket Sort

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.



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]);
    }

    // 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

Related Topics:

- Find Union and Intersection of two unsorted arrays
- Pythagorean Triplet in an array
- Maximum profit by buying and selling a share at most twice
- Design a data structure that supports insert, delete, search and getRandom in constant time
- Print missing elements that lie in range 0 – 99
- Iterative Merge Sort
- Group multiple occurrence of array elements ordered by first occurrence
- Given a sorted and rotated array, find if there is a pair with a given sum

Tweet 3 g+1 2

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

21 Comments **GeeksforGeeks** 1 **Login**

Recommend 2 Share Sort by Newest

Join the discussion…

mrcloud • 18 days ago

Cormen page : 174, Topic 8.4 (Bucket Sort)

Line 3 of Algorithm : do insert(A[i] into list B[floor (n* A[i])]

The incorporation of floor function is missing in the above code, as most of the time n*a[i] would be returning a decimal value not an integer value, it may give error. I tried code with and without decimal function and it seems to work flawlessly with floor and without floor for some corner values, code seems to give error.

So the right way of representing the code in segment 2 of the code : -

```
int bi = n*arr[i]; // Index in bucket
b[bi].push_back(arr[i]);
```

Should be

```
int bi = floor ( n*arr[i] ) ; // Index in bucket
b[bi].push_back(arr[i]);
```

And don't forget to include <math.h> for the functioning of floor function.

- mrcloud

• Reply • Share ›

Arun Kumar • 4 months ago

running the exact code gives error.

variable length array of non-POD element type 'vector<float>'

```
vector<float> b[n];
```

• Reply • Share ›

Joey • 4 months ago

This is the C implementation of Bucket Sort.
The logic and algo is the same.I have a doubt because when I copy back the contents of all the linked lists to the array,it gives segmentation error in certain inputs but works for others but when i simply print each linked list one by one,it works fine.Can someone please help me ?

• Reply • Share ›

Joey ➦ Joey • 4 months ago

http://ideone.com/qgGK4G

2 ⌃ | ⌄ • Reply • Share ›

Sasha • 4 months ago

There is must be a mistake: int bi = n*arr[i];

⌃ | ⌄ • Reply • Share ›

Praveen Kumar • 8 months ago

Is the average case time complexity O(n) and worst case time complexity O(n^2) for bucket sort ?

1 ⌃ | ⌄ • Reply • Share ›

Mbyta • 9 months ago

How to choose the size of the array of buckets and the range in each bucket ?

⌃ | ⌄ • Reply • Share ›

Jun • 10 months ago

Can somebody tell, how to implement it in C???

4 ⌃ | ⌄ • Reply • Share ›

Kerem Sahin ➦ Jun • 5 months ago

http://ideone.com/DYHfTR

⌃ | ⌄ • Reply • Share ›

Kamlesh ➦ Kerem Sahin • 9 days ago

C++ code hi thik h..:P

⌃ | ⌄ • Reply • Share ›

Nisarg • a year ago

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

Insertion Sort is quadratic.

⌃ | ⌄ • Reply • Share ›

GeeksforGeeks Mod ➦ Nisarg • a year ago

The main point to note here is there are total n elements and are uniformly distributed. These elements are divided in n buckets. To get some idea, consider the ideal situation when there is 1 element in every bucket, sum of 1<sup>2</sup> for n times is O(n). Please refer CLRS book for complete proof.

⌃ | ⌄ • Reply • Share ›

Suman ➦ GeeksforGeeks • a year ago

It is not clear to me why we need an insertion sort in step 3. While inserting an element in bucket if we can make sure we are inserting in right place i.e. we will keep the list sorted. This can be done in o(n). So we don't need step 3 at all.

• Reply • Share ›

GOPI GOPINATH ➦ Suman • a year 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 element say 0.04 must be placed in the same bucket....u need to check 0.03 but also 0.05 and only decide...so its nothing but insertion sort....so worst case complexity in ur case is again O(n^2)

2 • Reply • Share ›

Suman ➦ GOPI GOPINATH • a year ago

I am not sure how it is O(n^2). Following is the pseudo code. Am I missing something.

```
for each element e in linked list
if ( key > e.data && key < e.next.data)
insert new node between e and e.next.
endif;
end for
```

• Reply • Share ›

GOPI GOPINATH ➦ Suman • a year ago

so whats the number of comparisons u need to do to insert a number in list?? u need to check all the nodes in worst case...and in worst case if all the elements come under one bucket ? Hope u get it.

• Reply • Share ›

Suman ➦ GOPI GOPINATH • a year ago

Does the number of comparison really matter here. If there are n elements in a bucked we will have (2*n) comparisons. Even if all the elements(say k) are in the same bucket we will have (2*k) comparisons. How is complexity O(n^2). Correct me if I am missing something.

• Reply • Share ›

Niket ➦ Suman • a year ago

the complexity is O(n^2) as in the worst case , all the n elements have to be inserted in the same bucket and for inserting each of them , you have to compare with each of the elements already present in the list(if it is greatest of all).....Hope this helps

1 • Reply • Share ›

Kartik ➦ Suman • a year ago

Your idea looks good. Please note that the array elements are inserted in buckets inside a for loop. So it doesn't seem to be an improvement, but another way to achieve the same time complexity.

⌃ | ⌄ • Reply • Share ›

Suman ➦ Kartik • a year ago

Insertion sort has worst case complexity O(^2) (http://en.wikipedia.org/wiki/I...). But the approach that I mentioned has worst case complexity O(n).

1 ⌃ | ⌄ • Reply • Share ›

Ryan ➦ Suman • 7 months ago

Suman, what you have proposed is the same thing as insertion sort. In this case however it is "online", meaning that you are sorting the items as you are receiving them. Even if you are sorting the bucketed items as they are being inserted into the bucket it will still add up to O(n^2) complexity. You proposed using a linked list, which is fine. You will be able to insert the new item in place and not have to copy items into new positions, etc. However, you still need to walk the linked list from the beginning, and this will take O(n) time, where n is the current size of the linked list. In the worst case scenario, each newly inserted item would be larger than the other, so you would always have to walk the entire list each time, comparing each value as you go, before finally adding your new item at the end of the list. The list grows as you add each element, so on each insertion you will have to access 1 item, then 2 items, then 3 items for each addition. This sum of 1 + 2 + 3 + ... + n is equal to n(n+1)/2 which is a function of O(n^2) complexity.

4 ⌃ | ⌄ • Reply • Share ›

✉ Subscribe   Add Disqus to your site   Privacy

-
-
-
-
    - Interview Experiences
    - Advanced Data Structures
    - Dynamic Programming
    - Greedy Algorithms
    - Backtracking
    - 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
    - Lowest Common Ancestor in a BST.
    - Check if a binary tree is BST or not
    - Sorted Linked List to Balanced BST
- Follow @GeeksforGeeks

- ## Recent Comments

    - lt_k

      i need help for coding this function in java...

      Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago
- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago
- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago
- lucy

  @GeeksforGeeks i don't n know what is this long...

  Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 2 hours ago
- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 2 hours ago