Introduction to Bucket Sort

Bishwajit Saha

Dept. of CSE, BUET

5th International Workshop on Algorithms

Outline

Table of Contents

Introduction

Introduction

- What is meant by bucket??.

Introduction

- What is meant by bucket??.

Introduction

What is bucket sort??

Introduction

What is bucket sort??

- Distributes all the elements of an array into a number of buckets

Introduction

What is bucket sort??

- Distributes all the elements of an array into a number of buckets
- Then each bucket is sorted individually

Introduction

What is bucket sort??

- Distributes all the elements of an array into a number of buckets
- Then each bucket is sorted individually
    - Either using a different sorting algorithm like insertion sort

Introduction

What is bucket sort??

- Distributes all the elements of an array into a number of buckets
- Then each bucket is sorted individually
    - Either using a different sorting algorithm like insertion sort
    - Or by recursivly applying the bucket sort algorithm

Introduction

What is bucket sort??

- Distributes all the elements of an array into a number of buckets
- Then each bucket is sorted individually
  - Either using a different sorting algorithm like insertion sort
  - Or by recursivly applying the bucket sort algorithm
- Also called Bin sort

Introduction

What is bucket sort??

- Distributes all the elements of an array into a number of buckets
- Then each bucket is sorted individually
  - Either using a different sorting algorithm like insertion sort
  - Or by recursivly applying the bucket sort algorithm
- Also called Bin sort
- Cousine of radix sort

Table of Contents

Working Strategy

Pseudocode :

```
function bucketSort(array, n) is
buckets ← new array of n empty lists
for i = 0 to (length(array)-1) do
insert array[i] into buckets[msbits(array[i], k)]
for i = 0 to n - 1 do
nextSort(buckets[i]);
return the concatenation of buckets[0], ...., buckets[n-1]
```

Working Strategy

- Set up an array of initially empty ”buckets”

Working Strategy

- Set up an array of initially empty "buckets"
- **Scatter:** Go over the original array, putting each object in its bucket.

Working Strategy

- Set up an array of initially empty "buckets"
- **Scatter:** Go over the original array, putting each object in its bucket.
- Sort each non-empty bucket.

Working Strategy

- Set up an array of initially empty "buckets"
- **Scatter:** Go over the original array, putting each object in its bucket.
- Sort each non-empty bucket.
- **Gather:** Visit the buckets in order and put all elements back into the original array.

Working Strategy

- Set up an array of initially empty "buckets"
- **Scatter:** Go over the original array, putting each object in its bucket.
- Sort each non-empty bucket.
- **Gather:** Visit the buckets in order and put all elements back into the original array.

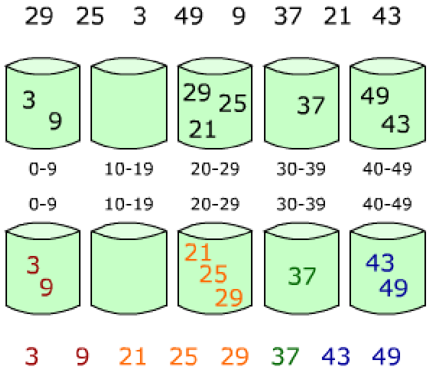

Table of Contents

Conclusion

- **Generic bucket sort**
- **ProxmapSort**
- **Histogram sort**
- **Postman's sort**
- **Shuffle sort**

Table of Contents

- A common optimization is :

- A common optimization is :
  - put the unsorted elements of the buckets back in the original array first

- A common optimization is :
  - put the unsorted elements of the buckets back in the original array first
  - then run insertion sort over the complete array

- A common optimization is :
  - put the unsorted elements of the buckets back in the original array first
  - then run insertion sort over the complete array
- insertion sort's runtime is based on how far each element is from its final position

- A common optimization is :
    - put the unsorted elements of the buckets back in the original array first
    - then run insertion sort over the complete array
- insertion sort's runtime is based on how far each element is from its final position
- the number of comparisons remains relatively small

Why we use this bucket sort??

Why we use this bucket sort??

- Think about an array of 100000 elements!!

Why we use this bucket sort??

- Think about an array of 100000 elements!!
- If we use insertion sort algorithm to sort this array then we have to comapare almost 10000000000 times

Why we use this bucket sort??

- Think about an array of 100000 elements!!
- If we use insertion sort algorithm to sort this array then we have to comapare almost 10000000000 times
- But if we use only 100 buckets to sort it then we have to compare only 100*(10000)=10000000 times

Why we use this bucket sort??

- Think about an array of 100000 elements!!
- If we use insertion sort algorithm to sort this array then we have to comapare almost 10000000000 times
- But if we use only 100 buckets to sort it then we have to compare only 100*(10000)=10000000 times
- It's here what this buckets give us advantages!!