The Bucket Sort Algorithm

Presentation by: John Patrick Tobias

The Bucket Sort

The **Bucket Sorting Algorithm**, also known as bin sort, from the name of the algorithm we can derive a general idea on how the algorithm would use or work in order to sort an array of values.

The Bucket Sort

In this sorting algorithm, we are going to use **buckets** in order to **segregate different values into their respective buckets** and sort the values inside the buckets with the use of **insertion sort**.

The Bucket Sort

Lastly, we **return the values into the original array** and they would be sorted, assuming that we start from the first bucket up to the last.

The Bucket Sort

Note: Bucket Sort is only O(n) if the number of buckets is equal to the length of the array being sorted.

The Pseudocode

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

From Wikipedia (*with minor changes*)

Step-by-step

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

Let’s call this array, *A*.

Step 1: Get size of array

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

Array *A* has a **size** (*N*) of 10.

N = 10

Step 2: Get the max value

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

The ***max* value** in the array is 90.

N = 10
max = 90

Step 3: Get amount of buckets

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

Then we get a **right amount of buckets** where we can put and sort the values into.

In this example, we can have 10 buckets (*to satisfy O(n)*).

Step 4: Get the divider

N = 10
max = 90
buckets = 10

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

Then we calculate the **divider** which is required in order to put the values into their respective buckets/bins.

With the formula: divider = ceil((max + 1) / bucket size)

Step 4: Get the divider

N = 10
max = 90
buckets = 10

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

In this case our divider would be:

divider = ceil ((90 + 1) / 10)
divider = ceil (9.2)
divider = 10

Step 5: Assigning to buckets

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

Formula for getting the **assigned bucket** for each value:

assigned_bucket = floor(*A*[i] / *divider*)

Now let's solve everything on the board.

Step 6: Insertion Sort

N = 10
max = 90
buckets = 10
divider = 10

When a new value is inserted into the bucket, **insertion sort** is then performed in order to sort the newly inserted value efficiently.

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	8	9
24	32	5	76	90	12	62	8	18	22

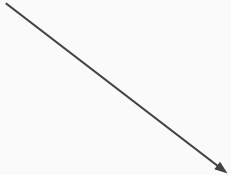
assigned_bucket = floor(24 / 10)
assigned_bucket = **2**

0	1	2	3	4	5	6	7	8	9
/	/	/	/	/	/	/	/	/	/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	8	9
/	32	5	76	90	12	62	8	18	22



assigned_bucket = 2

0	1	2	3	4	5	6	7	8	9
/	/	**24**	/	/	/	/	/	/	/
		/							

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	**1**	2	3	4	5	6	7	8	9
/	**32**	5	76	90	12	62	8	18	22

assigned_bucket = floor (32 / 10) = **3**

0	1	2	3	4	5	6	7	8	9
/	/	24	/	/	/	/	/	/	/
		/							

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	**1**	2	3	4	5	6	7	8	9
/	/	5	76	90	12	62	8	18	22

assigned_bucket = **3**

0	1	2	3	4	5	6	7	8	9
/	/	24	**32**	/	/	/	/	/	/
		/	/						

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	**2**	3	4	5	6	7	8	9
/	/	**5**	76	90	12	62	8	18	22

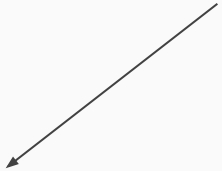
assigned_bucket = floor (5 / 10) = **0**

0	1	2	3	4	5	6	7	8	9
/	/	24	32	/	/	/	/	/	/
		/	/						

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	**2**	3	4	5	6	7	8	9
/	/	/	76	90	12	62	8	18	22



assigned_bucket = **0**

0	1	2	3	4	5	6	7	8	9
5	/	24	32	/	/	/	/	/	/
/		/	/						

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	**3**	4	5	6	7	8	9
/	/	/	**76**	90	12	62	8	18	22

assigned_bucket = floor (76 / 10) = **7**

0	1	2	3	4	5	6	7	8	9
5	/	24	32	/	/	/	/	/	/
/		/	/						

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	**3**	4	5	6	7	8	9
/	/	/	/	90	12	62	8	18	22

0	1	2	3	4	5	6	7	8	9
5	/	24	32	/	/	/	**76**	/	/
/		/	/				/		

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	**4**	5	6	7	8	9
/	/	/	/	**90**	12	62	8	18	22

assigned_bucket = floor (90 / 10) = **9**

0	1	2	3	4	5	6	7	8	9
5	/	24	32	/	/	/	76	/	/
/		/	/				/		

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	**4**	5	6	7	8	9
/	/	/	/	/	12	62	8	18	22

assigned_bucket = **9**

0	1	2	3	4	5	6	7	8	9
5	/	24	32	/	/	/	76	/	**90**
/		/	/				/		/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	**5**	6	7	8	9
/	/	/	/	/	**12**	62	8	18	22

assigned_bucket = floor (12 / 10) = **1**

0	1	2	3	4	5	6	7	8	9
5	/	24	32	/	/	/	76	/	90
/		/	/				/		/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	**5**	6	7	8	9
/	/	/	/	/	/	62	8	18	22

0	1	2	3	4	5	6	7	8	9
5	**12**	24	32	/	/	/	76	/	90
/	/	/	/				/		/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	**6**	7	8	9
/	/	/	/	/	/	**62**	8	18	22

assigned_bucket = floor (62 / 10) = **6**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	/	76	/	90
/	/	/	/				/		/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	**6**	7	8	9
/	/	/	/	/	/	/	8	18	22

assigned_bucket = **6**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	**62**	76	/	90
/	/	/	/			/	/		/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	**7**	8	9
/	/	/	/	/	/	/	**8**	18	22

assigned_bucket = floor (8 / 10) = **0**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	62	76	/	90
/	/	/	/			/	/		/

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	**7**	8	9
/	/	/	/	/	/	/	/	18	22

assigned_bucket = **0**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	/	76	/	90
8	/	/	/				/		/
/									

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	**8**	9
/	/	/	/	/	/	/	/	**18**	22

assigned_bucket = floor (18 / 10) = **1**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	/	76	/	90
8	/	/	/				/		/
/									

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	**8**	9
/	/	/	/	/	/	/	/	/	22

assigned_bucket = **1**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	/	76	/	90
8	**18**	/	/				/		/
/	/								

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	8	**9**
/	/	/	/	/	/	/	/	/	**22**

assigned_bucket = floor (22 / 10) = **2**

0	1	2	3	4	5	6	7	8	9
5	12	24	32	/	/	/	76	/	90
8	18	/	/				/		/
/	/								

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

0	1	2	3	4	5	6	7	8	**9**
/	/	/	/	/	/	/	/	/	/

assigned_bucket = **2**

0	1	2	3	4	5	6	7	8	9
5	12	**22**	32	/	/	/	76	/	90
8	18	24	/				/		/
/	/	/							

Bucket Sort Algorithm

N = 10
max = 90
buckets = 10
divider = 10

The Final Bucket

0	1	2	3	4	5	6	7	8	9
5	12	22	32	/	/	/	76	/	90
8	18	24	/				/		/
/	/	/							

Final Step: Put back EVERYTHING

When all values have been put into their respective buckets, we then **put them back** into array *A* starting from the first bucket up to the last bucket.

DONE !

0	1	2	3	4	5	6	7	8	9
5	12	22	32	/	/	/	76	/	90
8	18	24	/				/		/
/	/	/							

0	1	2	3	4	5	6	7	8	9
5	8	12	18	22	24	32	62	76	90

The values are then sorted!

Assumptions and Limitations

Assumes that all values belong to any range of a bucket.

O(n) is only achieved when the number of buckets is equal to the length of the array currently being sorted. If not, it can achieve $O(n^2)$ which would be like insertion sort, which can make bucket sort not necessary.