Jay Offerdahl
EECS 560
Lab 5 Report

Range Finders & Comparison Counters

Introduction:

After working through the AVL tree lab last week, this lab was a walk in the park. However, it helped that I spent a lot of time making sure Lab 4 worked properly. After implementing the distance function in this program, I added comparison counters to the insert and delete functions in order to effectively track the performance of each operation. Please view the README.txt text file in order to see how to access the distance method.

Testing Structure:

In order to properly test the performance of the insert and delete functions, different sizes of lists had to be used. I have set up my testing to test list sizes from the following:

- N = 10
- N = 20
- N = 50
- N = 100
- N = 500
- N = 1000
- N = 5000
- N = 25000
- N = 50000
- N = 100000

I believe this setup will effectively demonstrate the efficiency of each method (insert, delete by name, and delete by coordinates).

The list names will come from a dictionary file that I implemented for another class (EECS 565), which holds about 167 thousand entries of valid words. This will ensure unique values are being added to the list, and I will append incrementing numbers as the coordinates in order to test the deletion by coordinates method.

Testing Results:

Below is a table of the results when inserting/deleting from the two data structures. For a discussion on their performance, see the below sections.

I tested three different values for name in order to check the front, middle, and end of the value scope. I did the same for deletion by coordinates. This ensured a more balanced average case for my claims. For the name testing, I used the strings “AAAAAAAA”, “MMMMM”, and “ZZZZZZZZZZ”. For the coordinate testing, I used coordinates 0 0, N/2 N/2, and N - 1 N - 1.

String 1: “AAAAAAAA”
String 2: “MMMMM”
String 3: “ZZZZZZZZZZ”

Jay Offerdahl
EECS 560
Lab 5 Report

	# of Comparisons (Unordered Array)									
Testing_Case_Used	10	20	50	100	500	1000	5000	25000	50000	100000
Insert name (1)	0	0	0	0	0	0	0	0	0	0
Insert name (2)	0	0	0	0	0	0	0	0	0	0
Insert name (3)	0	0	0	0	0	0	0	0	0	0
Delete name (1)	1	1	1	1	1	1	1	1	1	1
Delete name (2)	6	11	26	51	251	501	2501	12501	25001	50001
Delete name (3)	10	20	50	100	500	1000	5000	25000	50000	100000
Delete coords. (1)	2	2	2	2	2	2	2	2	2	2
Delete coords. (2)	12	22	52	102	502	1002	5002	25002	50002	100002
Delete coords. (3)	20	40	100	200	1000	2000	10000	50000	100000	200000

	# of Comparisons (AVL Tree)									
Testing_Case_Used	10	20	50	100	500	1000	5000	25000	50000	100000
Insert name (1)	9	11	15	19	21	23	27	29	31	31
Insert name (2)	13	16	19	19	27	25	33	44	47	43
Insert name (3)	14	17	20	23	32	35	44	53	56	55
Delete name (1)	13	15	19	23	25	27	28	30	32	32
Delete name (2)	12	15	18	23	28	26	34	45	48	47
Delete name (3)	15	18	21	24	33	36	45	54	57	57
Delete coords. (1)	20	30	82	150	532	1050	7875	16149	32303	32791
Delete coords. (2)	39	74	141	66	1575	1589	16129	83132	165015	103378
Delete coords. (3)	54	97	220	419	2032	4035	20044	98605	197272	373860

Inserting

Very obviously, inserting into the unordered array takes constant time, as all that needs to be done is you add the element to the index that's the array size plus one. However, this is not the case for the tree. As seen from the table of test values, the tree takes considerably more comparisons to complete this task, however, this amount levels off to an average case of around 40 – 50 comparisons. With this in mind, it's clear that **<u>the unordered array is the better data structure for this operation</u>** because it has O(1) time, whereas the tree has O(lg(n)) time.

Deleting by name

Deleting by name in the unordered array has an average case of N/2, since it's linear and belongs to O(n). This means for a list size of 100000, the average case would take 50000 comparisons, on average. However, looking at the AVLTree, this function also has O(lg(n)) time. Seeing that the unordered array grows faster than the AVLTree, **<u>the AVL tree is the better data structure for this operation.</u>**

Deleting by coordinates

Deleting by coordinates is a nightmare for both the unordered array as well as the tree, but especially so for the tree. I counted the comparisons between the X coordinate and Y coordinate as two item comparisons for this testing. With this in mind, it's clear that the unordered array takes n time on average simply because it's N/2 time on average, times 2 comparisons per loop, making it N time overall. Looking at the AVL Tree, it's evident that since we have to look first search the entire tree to find the element, then delete the element by name. This whole tree search causes the number of comparisons to sky rocket since I implemented a breadth first search, which does four comparisons per loop until the node is found, making the whole process take O(nlg(n)) time. Next, a normal city deletion is carried out, which takes O(lg(n)) time, which is negligible on larger list sizes.

Deleting by coordinates for the unordered array takes O(n) time because it's linear for each list size (2n comparisons for worst case, n comparisons average case).

It's worth mentioning that my tests only took into account deleting cities by coordinates that were mostly located in the leaves of the tree. However, looking at the average case of all the tests, it appears that **<u>the unordered array is the better data structure for this operation.</u>**

****

The Distance Algorithm

Considering the range finder function that we implemented for this lab, the algorithms for the unordered array and the AVL tree are quite similar in terms of performance. First, the AVL tree searches the tree for matching coordinates (O(nlg(n)) complexity). The AVL tree then gets the X and Y coordinates (again, which I'm counting as a comparison), and then checks the left and right children in order to add more nodes to the queue. This means the AVL tree has a complexity of Nlg(N) + 4N.

When looking at the unordered array, a check throughout the entire array is first performed to check for matching coordinates, then each index is checked to see if the city is in range, which takes 2N comparisons (getX and getY for each index). With that in mind, the entire complexity is 3N.

With both of these complexities I can now confidently say that **<u>the unordered array is better for my implementation of the distance algorithm</u>**. Even if I didn't have to search for matching coordinates the first time around, the time would still be 4N for the AVL tree, and 2N for the unordered array. Which reaffirms my answer.

Results of Testing the Distance Algorithm

```
*********************************************************************

                        Lab 5 - Jay Offerdahl

*********************************************************************

*********************************************************************
                            Unordered Array
*********************************************************************

Cities in a 50 mile range of (0, 0):
origin is located at (0, 0)!
MysticCaverns
   Distance: 13.9284
Smithton
   Distance: 17.6918
Jonestown
```

```
   Distance: 24.1868
Hooterville
   Distance: 23.3238
JamesTown
   Distance: 40.6079
Maryville
   Distance: 28.8617
Lexington
   Distance: 26.9258
GhostTown
   Distance: 25
BaldwinCity
   Distance: 17
Pultneyville
   Distance: 25.4951
Monroeville
   Distance: 20
HighCityHeights
   Distance: 15
HendersonFlats
   Distance: 15.8114
PrairieMeadows
   Distance: 27.5136
Russellville
   Distance: 28.1603
AncientCity
   Distance: 32.6497
Anywhereville
   Distance: 30
SnowySummit
   Distance: 37.6431
Monterey
   Distance: 37.4833
BridgerRange
   Distance: 32.0156
EasyLanding
   Distance: 33.6006
Williamstown
   Distance: 7.61577
FortHays
   Distance: 29.6142
RadioCity
   Distance: 27.2029
NewYork
   Distance: 1.41421
LosAngeles
   Distance: 2.23607
Chicago
   Distance: 3.16228
Houston
   Distance: 4.12311
Philadelphia
   Distance: 5.09902
Phoenix
```

```
   Distance: 6.08276
SanAntonio
   Distance: 7.07107
SanDiego
   Distance: 8.06226
Dallas
   Distance: 9.05539
SanJose
   Distance: 10.0499
Austin
   Distance: 2.23607
Jacksonville
   Distance: 2.82843
SanFrancisco
   Distance: 3.60555
Indianapolis
   Distance: 4.47214
Columbus
   Distance: 5.38516
FortWorth
   Distance: 6.32456
Charlotte
   Distance: 7.28011
Detroit
   Distance: 8.24621
ElPaso
   Distance: 9.21954
Seattle
   Distance: 10.198
Denver
   Distance: 3.16228
Washington
   Distance: 3.60555
Memphis
   Distance: 4.24264
Boston
   Distance: 5
Nashville
   Distance: 5.83095
Baltimore
   Distance: 6.7082
OklahomaCity
   Distance: 7.61577
Portland
   Distance: 8.544
LasVegas
   Distance: 9.48683
Louisville
   Distance: 10.4403
Milwaukee
   Distance: 4.12311
Albuquerque
   Distance: 4.47214
Tucson
```

```
   Distance: 5
Fresno
   Distance: 5.65685
Sacramento
   Distance: 6.40312
LongBeach
   Distance: 7.2111
KansasCity
   Distance: 8.06226
Mesa
   Distance: 8.94427
Atlanta
   Distance: 9.84886

*********************************************************************
                               AVL Tree
*********************************************************************

Cities in a 50 mile range of (0, 0):
origin is located at (0, 0)!
Jonestown
   Distance: 24.1868
EasyLanding
   Distance: 33.6006
Pultneyville
   Distance: 25.4951
Chicago
   Distance: 3.16228
GhostTown
   Distance: 25
MysticCaverns
   Distance: 13.9284
Smithton
   Distance: 17.6918
BaldwinCity
   Distance: 17
Dallas
   Distance: 9.05539
FortHays
   Distance: 29.6142
Houston
   Distance: 4.12311
Maryville
   Distance: 28.8617
Philadelphia
   Distance: 5.09902
SanDiego
   Distance: 8.06226
Williamstown
   Distance: 7.61577
Anywhereville
   Distance: 30
BridgerRange
   Distance: 32.0156
```

```
Columbus
   Distance: 5.38516
Detroit
   Distance: 8.24621
ElPaso
   Distance: 9.21954
FortWorth
   Distance: 6.32456
HighCityHeights
   Distance: 15
Jacksonville
   Distance: 2.82843
Lexington
   Distance: 26.9258
Monroeville
   Distance: 20
NewYork
   Distance: 1.41421
Portland
   Distance: 8.544
Russellville
   Distance: 28.1603
SanJose
   Distance: 10.0499
Tucson
   Distance: 5
AncientCity
   Distance: 32.6497
Austin
   Distance: 2.23607
Boston
   Distance: 5
Charlotte
   Distance: 7.28011
Denver
   Distance: 3.16228
Fresno
   Distance: 5.65685
HendersonFlats
   Distance: 15.8114
Hooterville
   Distance: 23.3238
Indianapolis
   Distance: 4.47214
JamesTown
   Distance: 40.6079
LasVegas
   Distance: 9.48683
LosAngeles
   Distance: 2.23607
Mesa
   Distance: 8.94427
Monterey
   Distance: 37.4833
```

```
Nashville
   Distance: 5.83095
OklahomaCity
   Distance: 7.61577
Phoenix
   Distance: 6.08276
PrairieMeadows
   Distance: 27.5136
RadioCity
   Distance: 27.2029
SanAntonio
   Distance: 7.07107
SanFrancisco
   Distance: 3.60555
Seattle
   Distance: 10.198
SnowySummit
   Distance: 37.6431
Washington
   Distance: 3.60555
Albuquerque
   Distance: 4.47214
Atlanta
   Distance: 9.84886
Baltimore
   Distance: 6.7082
KansasCity
   Distance: 8.06226
LongBeach
   Distance: 7.2111
Louisville
   Distance: 10.4403
Memphis
   Distance: 4.24264
Milwaukee
   Distance: 4.12311
Sacramento
   Distance: 6.40312

********************************************************************
                           Unordered Array
********************************************************************

Cities in a 9 mile range of (12, 13):
Smithton is located at (12, 13)!
MysticCaverns
   Distance: 8.06226
Hooterville
   Distance: 8.06226
GhostTown
   Distance: 7.61577
BaldwinCity
   Distance: 5.83095
HendersonFlats
```

```
   Distance: 7.28011
Atlanta
   Distance: 8.94427

*********************************************************************
                               AVL Tree
*********************************************************************

Cities in a 9 mile range of (12, 13):
Smithton is located at (12, 13)!
GhostTown
   Distance: 7.61577
MysticCaverns
   Distance: 8.06226
BaldwinCity
   Distance: 5.83095
HendersonFlats
   Distance: 7.28011
Hooterville
   Distance: 8.06226
Atlanta
   Distance: 8.94427

*********************************************************************
                            Unordered Array
*********************************************************************

Cities in a 10 mile range of (30, 0):
Pultneyville
   Distance: 7.07107
Monroeville
   Distance: 10

*********************************************************************
                               AVL Tree
*********************************************************************

Cities in a 10 mile range of (30, 0):
NOT_A_CITY is located at (30, 0)!
Pultneyville
   Distance: 7.07107
Monroeville
   Distance: 10

*********************************************************************
                            Unordered Array
*********************************************************************

Cities in a 10 mile range of (25, 5):
Pultneyville is located at (25, 5)!
Jonestown
   Distance: 8.06226
Hooterville
   Distance: 8.60233
```

```
Monroeville
   Distance: 7.07107

********************************************************************
                              AVL Tree
********************************************************************

Cities in a 10 mile range of (25, 5):
Pultneyville is located at (25, 5)!
Jonestown
   Distance: 8.06226
Monroeville
   Distance: 7.07107
Hooterville
   Distance: 8.60233

********************************************************************
                           Unordered Array
********************************************************************

Cities in a 11 mile range of (25, 5):
Pultneyville is located at (25, 5)!
Jonestown
   Distance: 8.06226
Hooterville
   Distance: 8.60233
BaldwinCity
   Distance: 10.4403
Monroeville
   Distance: 7.07107

********************************************************************
                              AVL Tree
********************************************************************

Cities in a 11 mile range of (25, 5):
Pultneyville is located at (25, 5)!
Jonestown
   Distance: 8.06226
BaldwinCity
   Distance: 10.4403
Monroeville
   Distance: 7.07107
Hooterville
   Distance: 8.60233

********************************************************************
                           Unordered Array
********************************************************************

Cities in a 5 mile range of (25, 20):
BridgerRange is located at (25, 20)!

********************************************************************
```

```
                              AVL Tree
*******************************************************************

Cities in a 5 mile range of (25, 20):
BridgerRange is located at (25, 20)!

*******************************************************************
                           Unordered Array
*******************************************************************

Cities in a 6 mile range of (24, 29):
SnowySummit is located at (24, 29)!
JamesTown
   Distance: 3.16228
Monterey
   Distance: 4.24264
EasyLanding
   Distance: 4.47214

*******************************************************************
                              AVL Tree
*******************************************************************

Cities in a 6 mile range of (24, 29):
SnowySummit is located at (24, 29)!
EasyLanding
   Distance: 4.47214
JamesTown
   Distance: 3.16228
Monterey
   Distance: 4.24264

*******************************************************************
                           Unordered Array
*******************************************************************

Cities in a 5 mile range of (10, 25):
Lexington is located at (10, 25)!
Maryville
   Distance: 4.24264
PrairieMeadows
   Distance: 1.41421
Russellville
   Distance: 2.82843

*******************************************************************
                              AVL Tree
*******************************************************************

Cities in a 5 mile range of (10, 25):
Lexington is located at (10, 25)!
Maryville
   Distance: 4.24264
Russellville
```

```
   Distance: 2.82843
PrairieMeadows
   Distance: 1.41421

********************************************************************
                            Unordered Array
********************************************************************

Cities in a 5 mile range of (0, 15):
HighCityHeights is located at (0, 15)!
HendersonFlats
   Distance: 5

********************************************************************
                               AVL Tree
********************************************************************

Cities in a 5 mile range of (0, 15):
HighCityHeights is located at (0, 15)!
HendersonFlats
   Distance: 5

********************************************************************
                            Unordered Array
********************************************************************

Cities in a 3 mile range of (15, 20):
GhostTown is located at (15, 20)!
RadioCity
   Distance: 2.23607

********************************************************************
                               AVL Tree
********************************************************************

Cities in a 3 mile range of (15, 20):
GhostTown is located at (15, 20)!
RadioCity
   Distance: 2.23607
```

Conclusion:

In conclusion, it's clear that the unordered array is better at inserting, deleting by coordinates, as well as the distance function. The AVL tree is better at deleting by name (rightfully so), and only loses to the unordered array on inserting because the complexity of inserting into an array is constant. Lastly, I support the claim that the unordered array is better at the distance algorithm because it can perform a linear search on the array rather than an Nlg(N) search, and each index only takes 2N comparisons as compared to 4N comparisons for the AVL tree.

As always, if you have any questions or concerns, please email me at jayofferdahl@ku.edu.