COMP20003-Algorithms and Data structures

Assignment 2 – 2-D Tree Experimentation

Introduction: Summary of data structures and inputs

In the coding part, our program is designed to read the dataset of Business Establishment trading name data and insert the information into a data structure called k-d trees. In stage 1, user will input a pair of coordinates and our program would traverse the tree and search the information of business shop that are closest to the input coordinate. In stage 2 user will input a pair of coordinates and a radius so that program will return all trading records within that radius.

In this report, we are going to use a set of test key file to obtain average number of key comparisons in three data files. Sortx(sort x coordinate from small to large, obtaining an approximate linked list), random(random coordinates) and median file(let median of x-coordinate as first entry to obtain an approximate balanced tree), and we use those result to compare with that of comparison in stage2 , as well as discussing against theoretical complexity of 2-d trees.

Data (number of key comparisons):

-Stage 1 Number of comparisons

The purpose of Stage 1 is to find the record with the minimal distance against of query point and print that results. For the purpose of finding average number of key comparisons in three different files. We design a search file that mapping from x coordinate from minimum key value (144.90) to maximum value (144.99) with a partition of 50 parts. For y coordinate, the mapping is (-37.8497) to (-37.775) approximately, with a partition of 100.This generates 5000 keys in total. After executing our program, we calculate the sum of the total comparison times for each query point and divided it by 5**000.** The test result is visualized into a bar chart below using three original datasets.

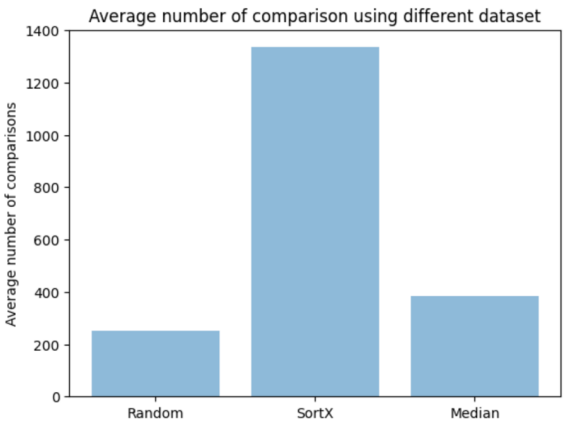

Figure 1 Average number of comparisons using three different datasets with 5000 keys

Analysis

As we can see from above bar chart, y coordinates indicate average number of comparisons using 5000 keys described above. From the chart, it is obvious to observe that sortx file requires more comparisons than other two type of file.

This might be mainly due to tree structure is essentially approximating a linked list in sorted x file. Using this file, root of kd-tree could only have right child, which significantly decrease the efficiency for searching since it only traverses in one direction.

As for the random and median dataset, it shows that number of comparisons for median is larger than that of random, which is also expected. Even though median dataset could possibly generate a balanced tree, unlike binary search tree, it, sometimes still need to traverse two direction to find the nearest neighbor rather than one. Also, it could also be witnessed that both of two datasets break the linearity in sortx file, which drastically improve searching efficiency.

Stage 2 (number of comparison)

-Method Background

In this stage, we are asked to explore the number of comparisons against different radius as well as different type of files. Therefore, using search file in stage 1 and add corresponding radius value, a grouped bar chart that using three original datasets could be used to further illustrate our finding.

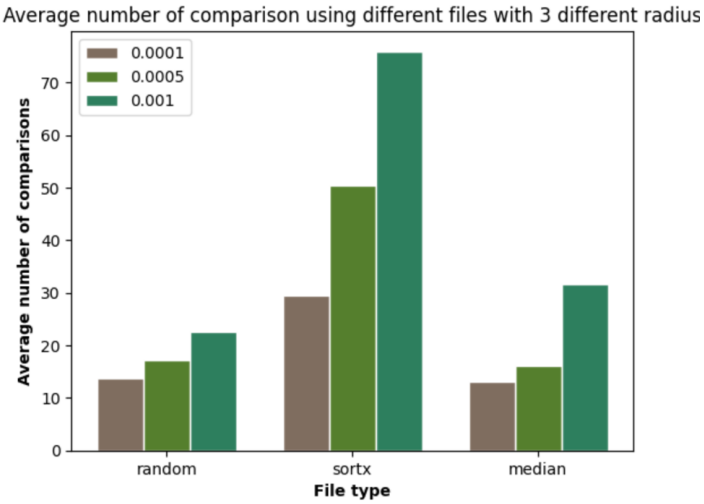

Figure 2 average number of comparisons against different dataset using 3 different radius

Analysis

As can be seen from this chart, it clearly shows that larger radius leads to larger comparison times, which is expected. Since larger radius imply larger possibilities to visit both branches in tree rather than one. That might be the reason to obtain this result.

Moreover, it also shows a similar pattern that using different file structure could cause different number of comparisons as stated above. From the chart, it also demonstrates a similar pattern, with random file obtaining the lowest number of comparisons and sortx files shares largest number of comparisons.

- **Comparison of the two stages**

-Method / Background

In this part, our purpose is to compare comparison in nearest neighbor search against comparison in radius search. Key file for Stage 1 is same as above while key file for Stage 2 add a value of radius of 0.0005 as well as a same search file with radius of 5. Below is the experiment visualization

Analysis

As we can see from the grouped bar chart, comparison in stage 1 is significantly larger than that of radius search with radius of 0.0005. The reason of that might be attributed to the use of small radius.

Apart from this, as expected, the radius of 5 creates the worst case each of them shares 4181 comparisons, which is also dramatically larger than that of Stage 1.

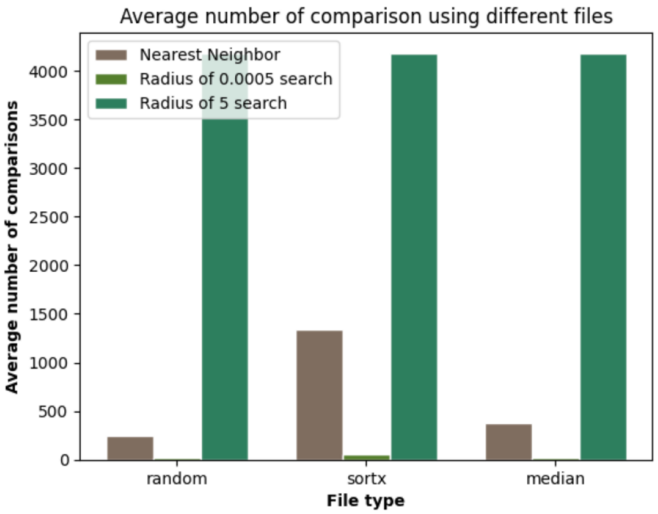

figure 3 Average number of comparisons using different file to compare stage 1 and 2

In this case, the entire tree would be searched, traversing each node along the tree, leading to worst case result.

- **Comparison with theory**

For stage 1

In order to compare the complexity with theoretical complexity, we partition three original datasets from 1000 to 19000 records, with an increase of 1000 each time. Using each of partition to obtain number of comparisons for each sample and record every average number of comparisons each time.

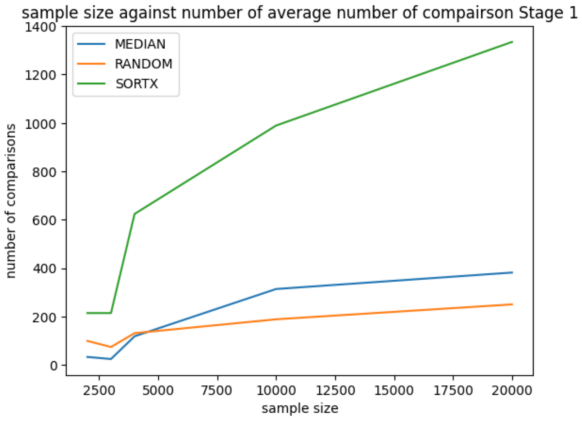

Figure 4 increase of the sample size against number of comparisons

From the figure 4, we can observe that as the increase of sample size, number of comparisons increase correspondingly for three different datasets. For sortx, it has the fastest growing complexity which approximately approach a linear function, so it may be said that sortx data structure search requires O(n) time, which matches theoretical complexity for worst case. As for random and Median, both of them shows a smaller growing rate relatively that may leads to a O(log n) complexity since both of this k-d trees in both files break out the linearity and leads to faster search.

Limitations

Partition interval of this dataset might not be small enough to observe all changes. Smaller interval might be used. More experiment should be used to observe patterns for theoretical time complexity.

For stage 2

For stage2, figure5 illustrates the relationship between sample size and average comparisons. And For convention, we have set the radius size as 0.0005.

A visualization against this result are below

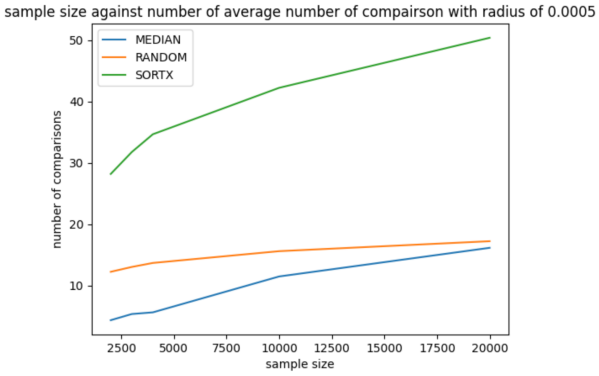

Figure 5 sample size against number of comparisons with the radius of 0.0005

As can be seen from this graph, this figure shows a similar pattern for time complexity in stage 1, that the sortx file shares the worst case, growing more rapidly than other two. Also, best case of those three, are median and random file, with only 10 comparisons in average. As for theoretical complexity discussed in lectures, best case is O(log n) and worst case is O(n).

Sorted file demonstrates the worst case with O(n). As for best case, both median and random shows a slow growing rate with approximating O(log n) time complexity.

Limitation:

This graph only considers the value of radius 0.0005 for easy interpretation, however, larger radius implies faster growing rate of complexity and results in worst case regardless of types of file. More radius values are needed to be considered.

- **Discussion**

In conclusion, we have tested a specific search file that cover every point in our dataset, with a size of 5000. In the first two part, we have used three original datasets as our input data.

In the stage 1 part of data comparisons, we observe that sortx file share the worst time complexity by finding.

In the stage two part of data comparisons, we examine the number of comparisons by different radius and different data files. Limitation of this part exists. To further demonstrate, choice of radius value is limited to 4 (0.0005,0.0001,0.001,5). There may not enough evidence to further explain different dataset may imply larger number of comparisons. To further improve that, we could use smaller sample dataset and using more testing radius value.

In the second part, we examine the number of comparisons between stage1 and stage 2 using different datafiles, our limited test

result shows that comparison of radius search might be small compared with neighbor search if radius is small. If radius is large, it shows worst case for all dataset

Finally, for the last part, we increase the sample size each time and obtain number of comparisons to visualize the changes in three files. Drawing a corresponding visualization to compare the graph of time complexity of our finding against theoretical.

It was found that theoretical complexity may match theoretical time complexity.

However, limitation still exists, especially for the median dataset. For example, first 1000 record of median dataset may be biased, since root data is the median for the whole dataset, for improvement. We could use median of partitioning dataset instead of original.

Conclusion/ Improvement

To conclude, it could be said that 2-d tree experiment mostly matches our expectations. However, larger search key files and narrower size of input data should be made to generate a visualization in order to obtain a more accurate result for complexity research of 2-d trees.

Also, more random type of data should be created and be experimented to further conclude the relationship of time complexity between randomness chosen files.

Finally, experiment can be further implemented based on key-result in datafile and not in datafile. So that more discrepancy could be found.