# 

# 

# EC-504

Advanced Data Structures

**Topic: Nearest State/County Finder**

**Team Information**

Hitanshi Jain (U95073268)

Renyu Jiang

Shuyan Zhang

Yuxiang Wan

**Abstract**

The aim of this project is to input the latitude, longitude, and K (number of nearest reference points required) from the user and then output the K nearest reference points as an output. It also finds the state and county for the input referenced point, considering the 5 nearest points. To implement this, we have used two data structures Geohash and Red Black Trees, and compared their results for diverse locations. We have implemented the code in C language and used the US counties dataset to test our results.

**Instructions to Run code**

This is our project’s GitHub link, you can get the code and instructions here:

<https://github.com/hitanshijain/Nearest_State_Finder#readme>

If there is any question about running code, please contact [renyu26@bu.edu](mailto:renyu26@bu.edu) or [woodywan@bu.edu](mailto:woodywan@bu.edu)

**Methods Used**

1. Geohash

Geohash is a public-domain geocode system that encodes a geographic location into a short string of letters and digits. It is a hierarchical spatial data structure that subdivides space into buckets of grid shape, which is one of the many applications of what is known as a Z-order curve, and generally space-filling curves.

**Geohash Process:**

1. Encode latitude and longitude

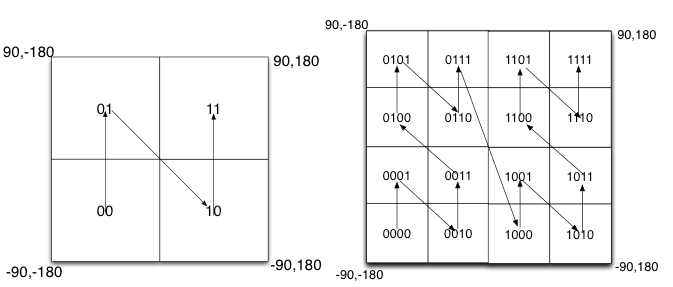
2. Combine encoded latitude and longitude, longitude in even digits, latitude

in odd digits, creating a new location binary code

3. Base32 encodes the new location binary code

4. Perform prefix matching in order from long to short to find adjacent

counties



5. Increase the prefix by one bit to re-screen the results of the preliminary

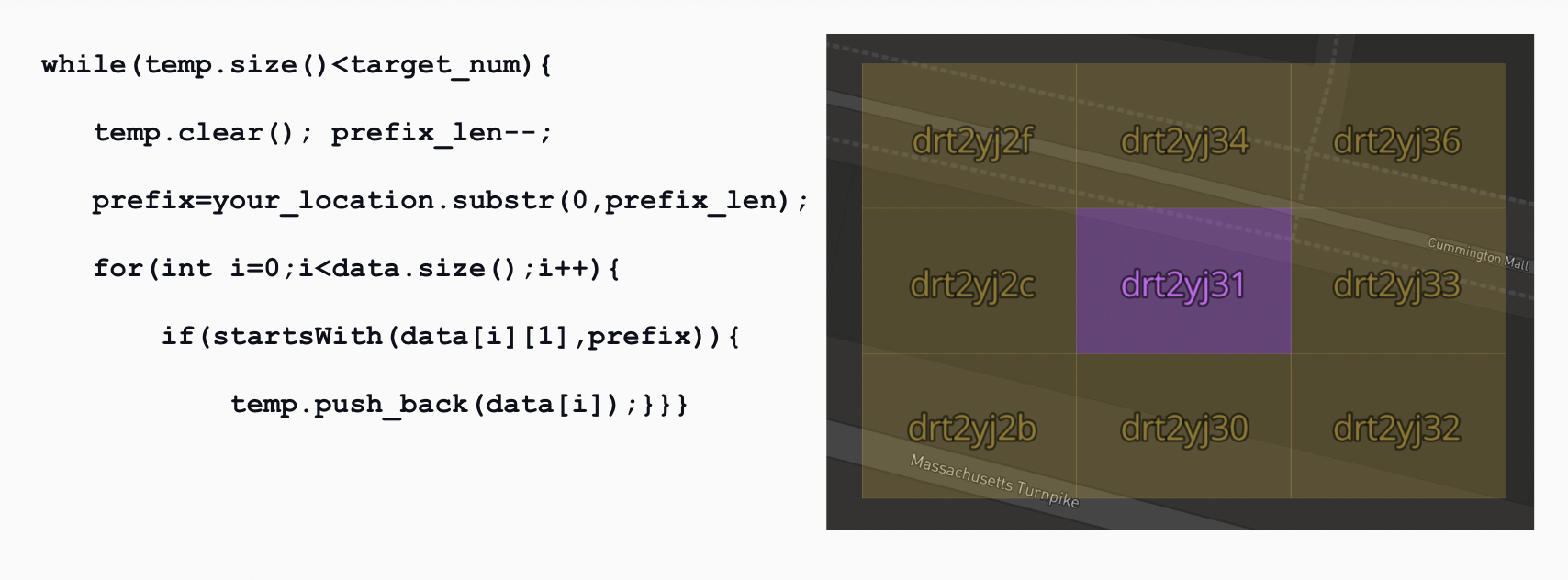
search

6. Calculate the distances of the remaining counties and sort them

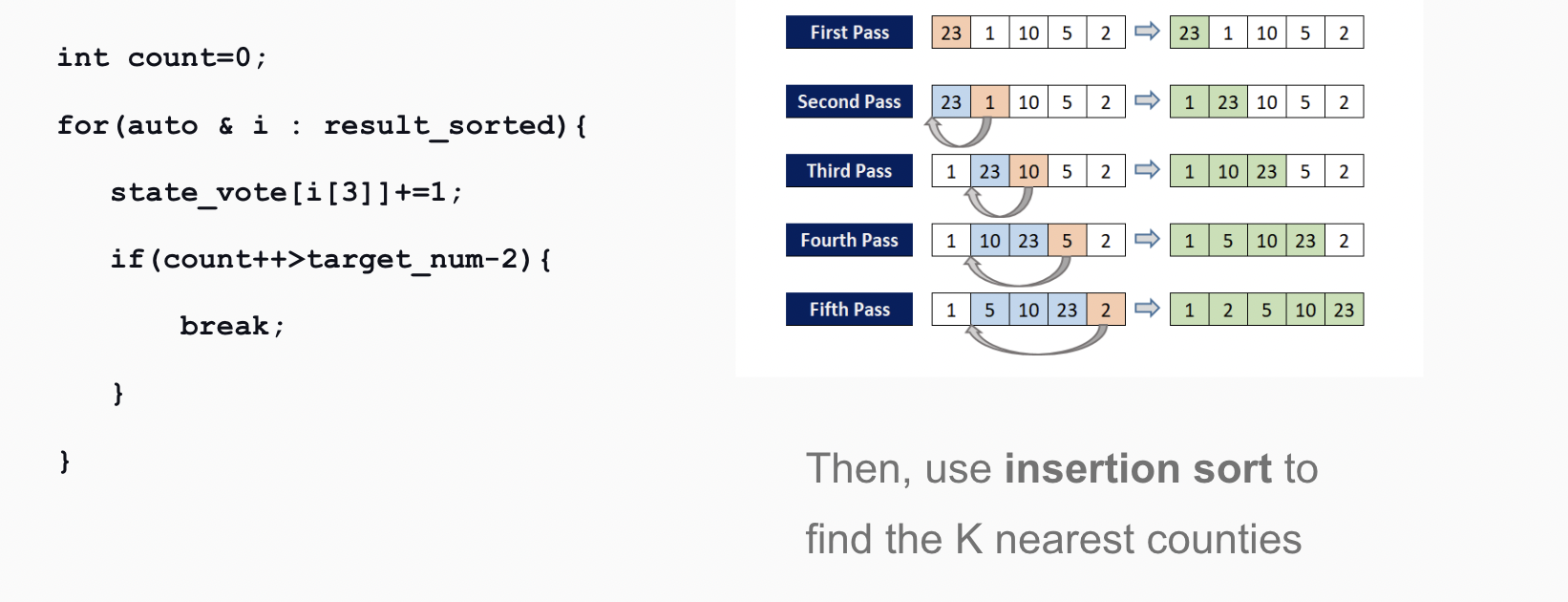
We used Geohash to find the nearest counties of local locations with O(log(log(n)) time complexity.

**Geohash Code Screenshots:**

County Search:

****

Majority State Finder:



1. Red-Black Trees

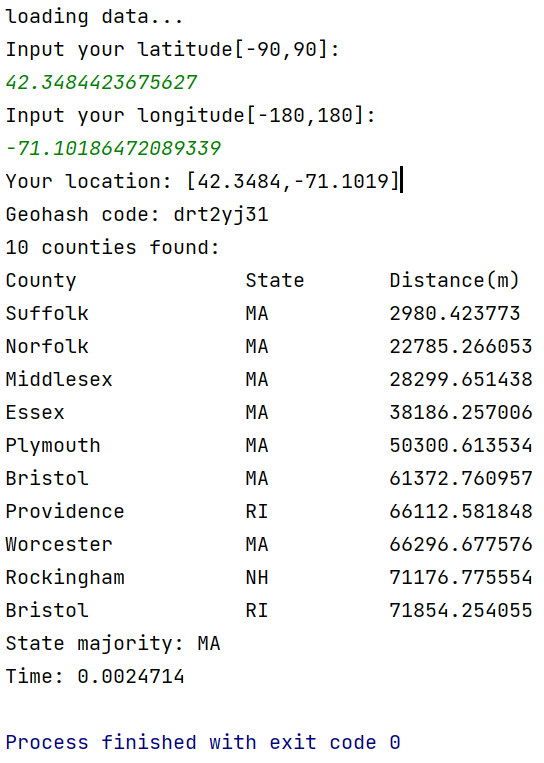
In Red-Black Tree the insert/delete operations are O(log(N)) and re-balancing rotation is an O(1) operation, making the Red-Black tree more efficient in this aspect of the re-balancing stage. The R-B tree method is for oversea locations including locations far from North America like Hawaii and Guam. The average time complexity of the finding process will be O((N)log(N) + N)) in the case of Red-black trees.

**Results**

Our software can find the nearest counties at a very fast speed (Avg 2.4 ms) no matter what location you input. For overseas locations, the speed would be a little slow, but it is still under 15 ms, which is significantly faster than other algorithms.

* Demo 1 (Domestic location):

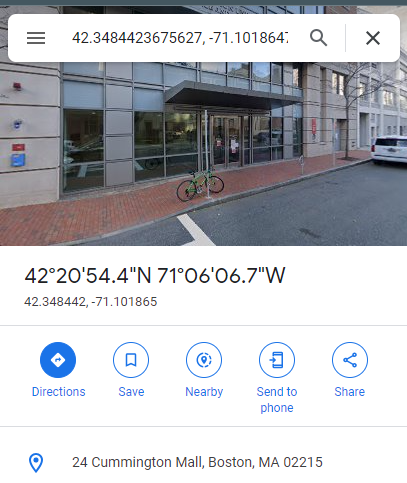
**Geohash Result**



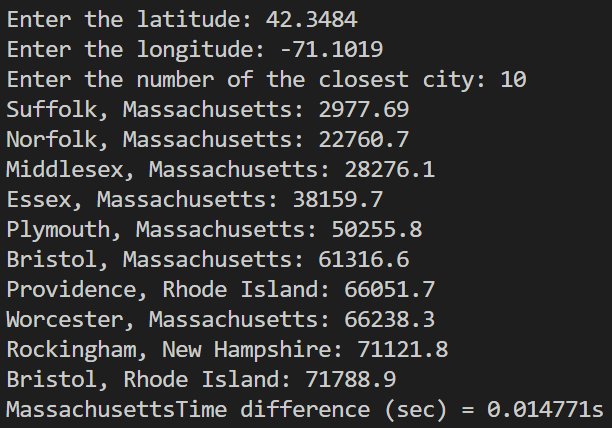
Test Location: BU LSE in Massachusetts

Latitude: 42. 3484423675627

Longitude: -71.10186472089339

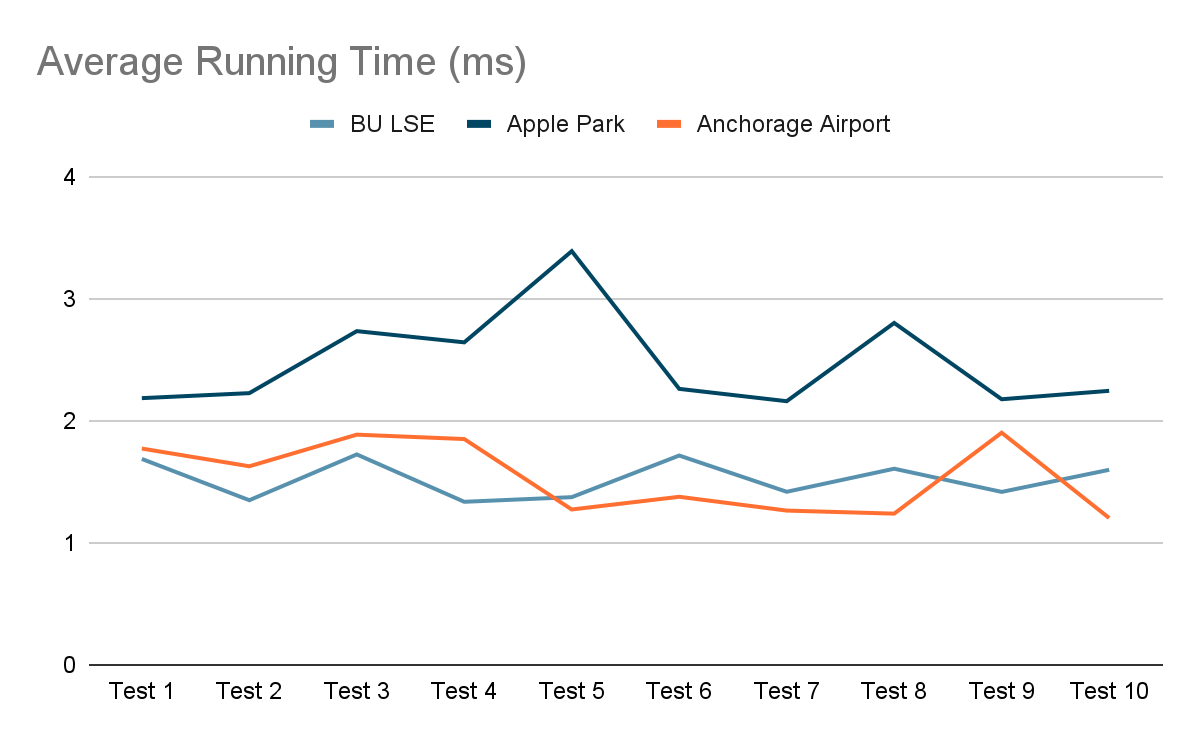
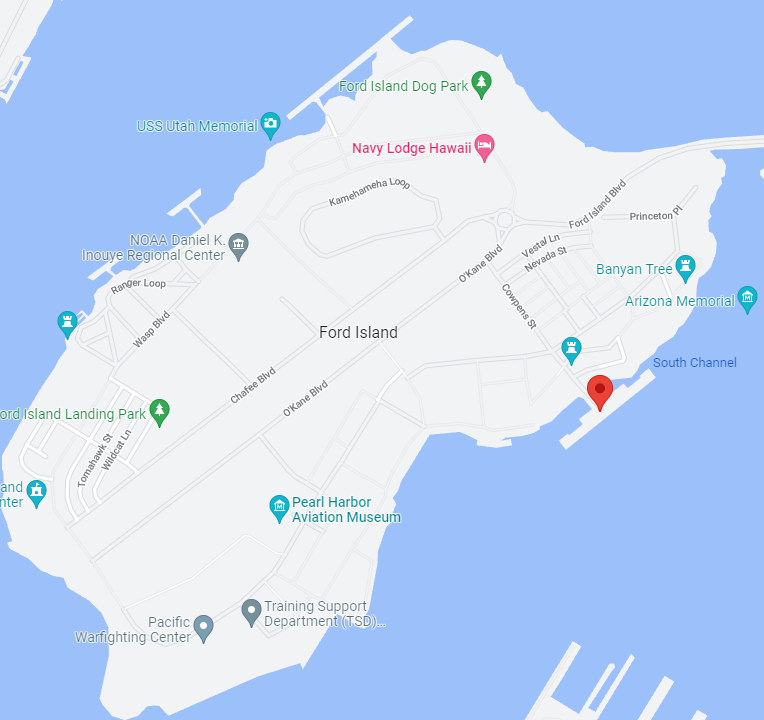


**R-B Tree Result**



The screenshots above show that Geohash is significantly faster as compared to R-B Trees for a domestic US location.

**Performance Analysis for Domestic Locations**



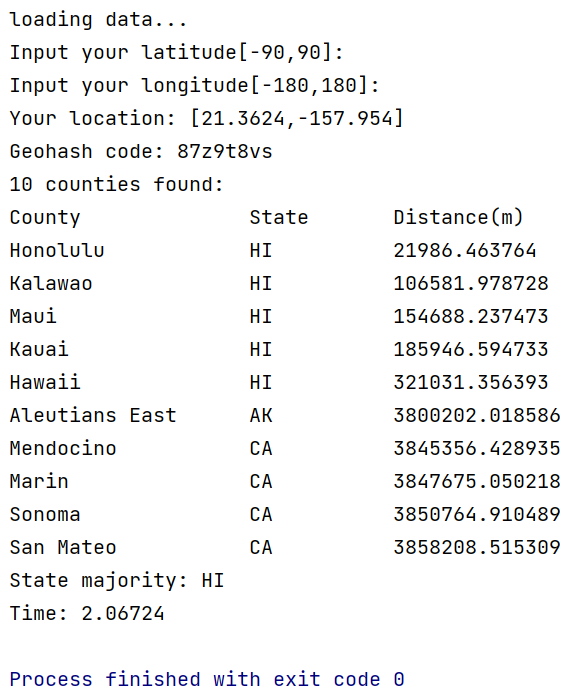
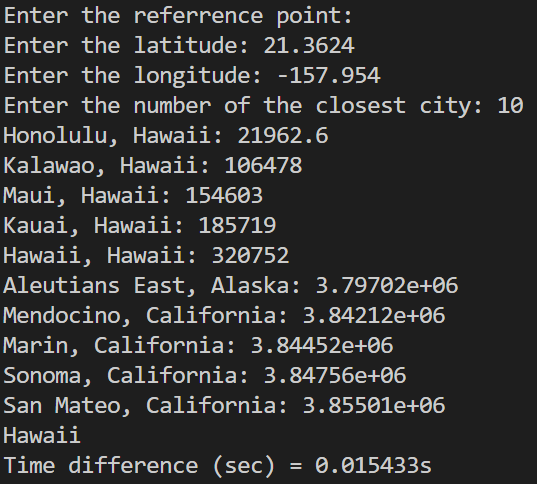
* Demo 2 (Overseas location):

Test Location: Pearl Harbor in Hawaii

Latitude: 21.362374050539042

Longitude: -157.95365360883602

**Geohash Result** **R-B Tree Result**



The screenshots above show that R-B Trees is significantly faster as compared to Geohash for an overseas location.

**References**

1. <https://en.wikipedia.org/wiki/Geohash>
2. <https://blog.csdn.net/weixin_41519463/article/details/88999339>