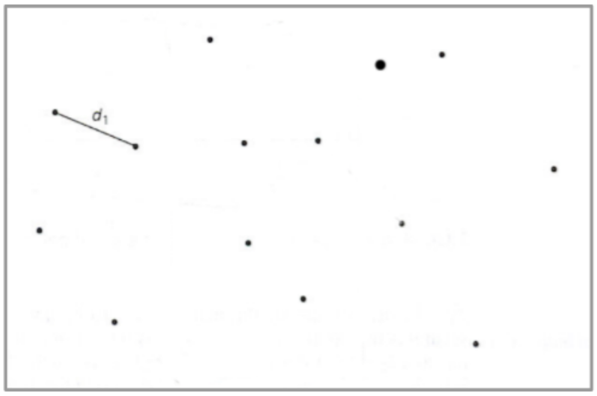
An Analysis of the Nearest Neighbor Problem

Jose Paredes Hernandez, 861234967

May 1, 2018

1 Introduction

For the nearest neighbor you have to find the smallest distance between two points on a plane. There are two methods for solving this problem: using a brute force algorithm and a divide and conquer algorithm.



2 Algorithms

2.1 Brute Force Algorithm

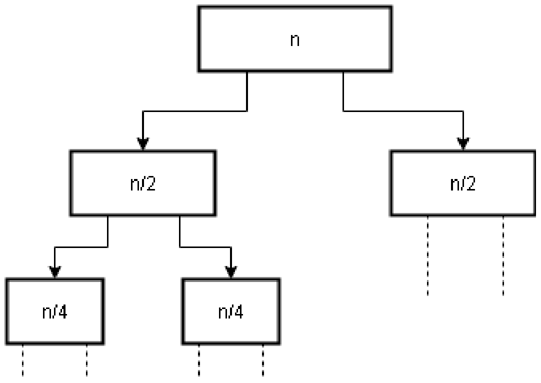
With the brute force algorithm you simply calculate the distance between every two points in the dataset and find the smallest distance.

Run-time

For this type of algorithm, a nested for loop is employed. Each loop iterates approximately n times. For this reason the theoretical runtime is $O(n^2)$

2.2 Divide and Conquer Algorithm

The second method for solving this problem is with a divide and conquer algorithm. With this method you divide the the dataset into its left and right pieces (essentially dividing it in half). From here you solve each side recursively to get the smallest distance. You then use that distance to set a border a new search space down the middle of the plane. This is done to check the distance of the points that lie in the middle of the plane where the plane was cut in half in case one of those pairs is the actual shortest distance.



Run-time

The theoretical runtime is as follows: $T(n) = 2T(\frac{n}{2}) + n^2$. There are two recursive calls each of which breaks the problem into subproblems half the size of the original and there is an extra n^2 work done since the brute force algorithms is still used (however at a smaller scale). Therefore, the theoretical runtime for this algorithm is $O(n^2)$.

3 Results

Both of the algorithms had a theoretical runtime of $O(n^2)$. In the data below you can see that they both take around the same time to find the nearest neighbor. However, as the input size gets bigger, the divide and conquer algorithm seems to take longer than the brute force. This can be because of constant factor that has been omitted during the big-O evaluation. Changing the base case of my divide and conquer algorithm probably won't be very helpful. However, a reason for changing the base case would be if there were circumstance where the brute force algorithm or some other algorithm works faster at the smaller input size than the divide and conquer algorithm.

Table 1: Algorithm Runtime

Input Size	**Brute Force Time (s)**	**D & C Time (s)**
10	4.727×10^{-4}	1.115×10^{-4}
100	5.321×10^{-3}	8.069×10^{-3}
1000	0.463	0.742
10000	52.553	86.915
100000	N/A	N/A