CS 310
Assignment 321

Nischal Shrestha
19 March, 2017

The brute force closest pair algorithm and divide-and-conquer closest pair algorithm calculates the distance between two closest coordinates in a set(vector) of a finite number of coordinates.

The brute force closest pair algorithm computes all possible distances for each pair of distinct points and finds a pair with the smallest distance. Whereas, the divide and conquer closest pair computes the closest distance by brute force algorithm if the number of coordinates in a set are less than four but if the number of coordinates are greater than four, the set of coordinates are split into two roughly equal halves computing the closest distance recursively and combining the results at the end.

The set of co-ordinates inputed are sorted in nondecreasing order of their x-coordinates and y-coordinates seperately. The purpose of this study is to perform an empirical analysis of this two methods.

For both methods, the input size is taken as the number of co-ordinates in the set because it determines the number of basic operations.

In a emperical analysis, the established framework for the analysis of an algorithm's time and space efficiencies are measured as functions of the algorithms input size.

To analyze the brute force closest pair, we chose the following as basic operations :

1. *Squaring the distance* on line 56 in the `distance` function but we put the basic operation count right after the `distance` function is called inside the `bf close pair dist` function because everytime the `distance` function gets called, the basic operation is perfomed. We choose squaring as basic operation because it is technically inside two nested for loops which runs maximum number of time in one run of the method. *Taking the Square root* could also be taken as a basic operation but it is indeed the same thing.

Similarly, to analyze the divide-and-conquer closest pair algorithm we chose the following as basic operations:

1. *Division of the set into two and copying data* on line 120 and 130 because divide (and recombine) occurs in linear time and recurses over the pieces since the increase in number of pieces to divide exactly matches the decrease in the size of the pieces. So, diving the set of data into half is an expensive operation.

2. *Calculation of minimum distance* on line 170 which is computing the minimum distance of the obtained soultions, which philosophically is the main idea of finding the closest distance algorithm.

The empirical data obtained as the result of these two algorithm are recorded in a data file and then presented graphically. The analysis of running two of the algorithm for multiple values of n produces the results shown below. Standard functions $f(n) = n$, $f(n) = nlog(n)$ and $f(n) = n^2$ along with different coefficient in the functions have been added to illustrate the analysis.

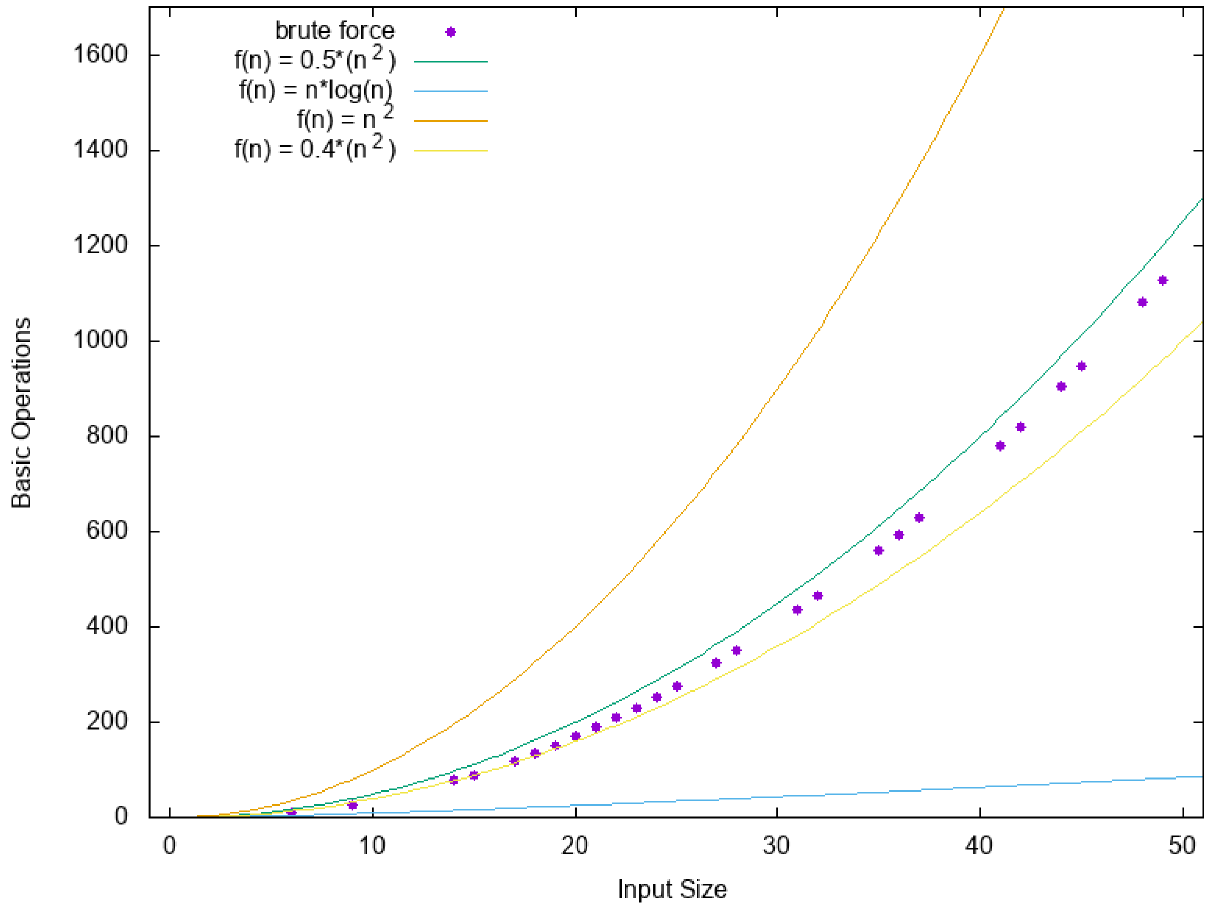

Figure: Brute Force Closest Pair (Basic Operations Vs Input Size)

An examination of the code itself explains the empirical results when we observe that: In the brute force closest pair algorithm, the number of basic operations are directly proportional as a quadratic graph with the input size. We have an asymptotically tight bound with standard quadratic function (n^2). There is no way that the algorithm can end early because it computes all possible distances for each pair of distinct points and finds a pair with the smallest distance. Hence, we have big theta analysis.

Therefore, we conclude that the brute force closest pair's algorithm is described by

$$T(n) \in \Theta(n^2)$$

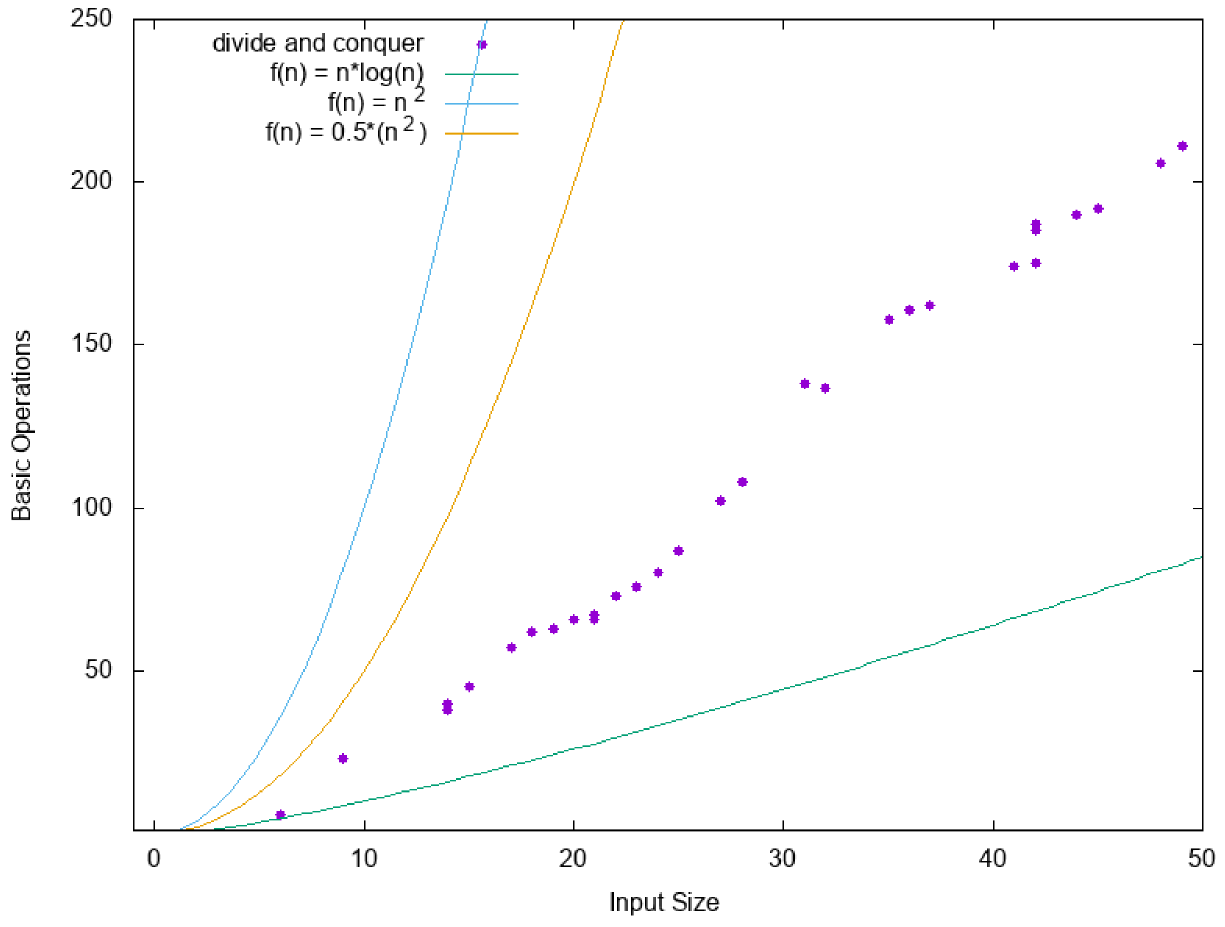

Figure: Divide and Conquer Closest Pair (Basic Operations Vs Input Size)

Similarly, In the divide and conquer closest pair, since the size of the pieces is halved at each time, there are $log(n)$ division stages, so the total running time is $nlog(n)$. As per the author, he has analyzed this algorithm using Master Theorem and described by

$$T(n) \in \Theta(n * log(n))$$

According to my graph, it shows that the standard function $nlog(n)$ asymptotic upper bounds the growth of number of basic operation from above for large enough input sizes. Similarly, the standard quadratic function n^2 asymptotic lower bounds the numbers of basic operations from below for large enough input sizes. The algorthim can also end early if there are less than four co-ordinates in the set. In conclusion, the divide-and-conquer algorithm for finding the closest pair of points in the plane has an analysis of

$$T(n) \in \Omega(nlog(n))$$

which is also the best case for the algorithm and

$$T(n) \in \mathcal{O}(n^2)$$

which is the worst case.