CS 325 - Analysis of Algorithms
January 20, 2017
Group members: Eric Stevens, Jong Park, Zhenggan Zheng

Programming Assignment 1 - Write up

Pseudo-code (details on the code, please see README.txt

Case 1. Brute Force method: O(n^2)

```
1. Loop through every point in the list
2. For every point, compare it to every other point
3. Check their distance from each other and return point/points with smallest distance
```

Case 2. Divide and Conquer method:O($n\log^2 n$)
Input: list of points sorted by their x values.

```
1. If len(points)<=3:
    a. Return bruteforce(points)
2. Sort the list from low to high on x axis.
3. Recursively find midpoint of x and split them into L and R until 3 points remain
    a. Return the best distance (and pairs)
```

Case 3. Enhanced Divide and Conquer method: O(nlogn)
Input 1: a set of points X described by: X_x (points in X sorted based on x)
Input 2: a set of points X described by: X_y (points in X sorted based on y)

```
1. If len(points) <= 3:
    a. Return bruteforce(points)
2. Sort the list
    a. From low to high on x axis
    b. From low to high on y axis
3. Find Midpoints to split
    a. Recursively find midpoint of x and split them into L and R
    b. Recursively find midpoint of y and split them into Q and R
4. Return the best distance D = min(d1,d2)
    a. D1 = best distance found with ($Q_x$, $Q_y$)
    b. D2 = best distance found with ($R_x$, $R_y$)
```

Asymptotic Analysis of run time

Brute Force:
This algorithm contains a nested since it has to compare every point to every other point. Since the brute force that we implemented avoids comparing the same points multiple times, our runtime is $O((n^2-n)/2)$ which simplifies to $O(n^2)$.

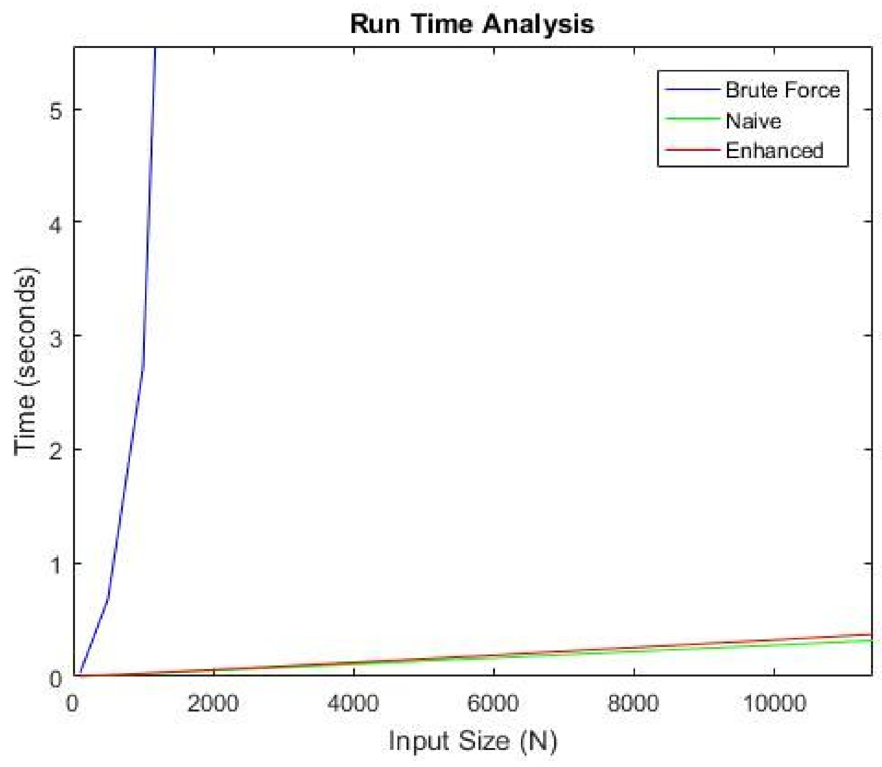
Naive Divide and Conquer:

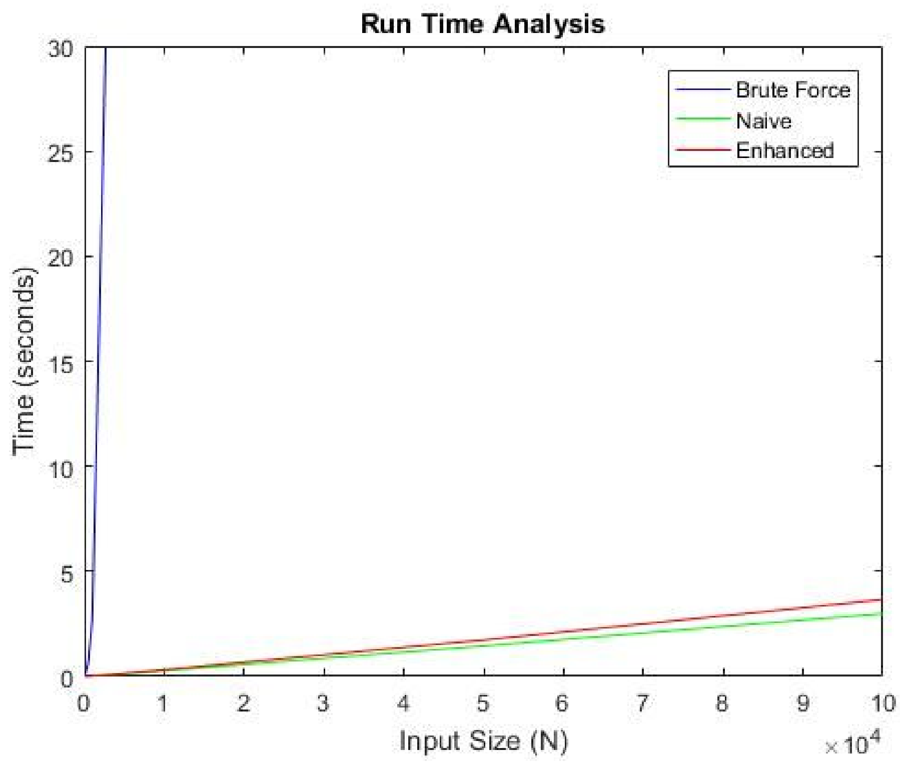
1. $T(n) = 2T(n/2) + O(nLogn)$
2. $T(n) = T(n* Logn * Logn)$
3. $O(nLog^2n)$

Enhanced Divide and Conquer:

1. $T(n) = 2T(n/2)+O(n)$
2. Use master theorem: $aT(n/b)+cn^d$
3. Since $a = 2$, and $b = 2$, and $d = 1$, $a = b^d$
4. Therefore, $T(n) = O(n^d logn) = O(nlogn)$

Plot of the Runtime





Interpretation and discussion

For the most part, our runtime plots do match our expectations. This is because brute force grew exponentially while the two divide and conquer algorithms did not grow as fast. However, the one twist that we experienced was that the enhanced divide and conquer was actually slower than than the naive divide and conquer. We think that this is because the enhanced divide and conquer has to sort both the x and the y axis while the naive divide and conquer only has to sort the x axis.