Algorithm Comparison

COT6405: Analysis of Algorithms

Dr. Mihaela Cardei

Michael Cuomo

# Problem

For this assignment we are comparing two different algorithms for finding the closest pair of points in a set of points. The algorithm is presented with many points and must find which set of two points are the closest.

For this test we are using large sets of points to stress test the algorithms. We are also will do multiple runs of the algorithms with growing sets to capture how these algorithms handle problems of growing complexity. Each run with a specific number of points will be run 10 times, and the average time length will be used in this report.

Each algorithm will be run on the same set of points to have a fair time comparison. The first set of points will have 10,000 instances. Each increase will be from 10,000 instances to a total of 100,000 instances in the final run. The outcome from these runs will be displayed in both tables and graphs.

This can be like real world problems of finding closest pairs. One such example is the algorithms used in this exploration can be modified slightly to run k-nearest neighbors for uses of clustering or classification of samples that were recorded another time. If there are many samples, it would be helpful to find an algorithm that is able to process the samples in the most efficient way possible.

# Algorithms and RT Analysis

For this exploration we are comparing two algorithms, one of which is a more intuitive approach, and another which can reach a more efficient running time considering the problem. The more intuitive approach is a brute force algorithm, and iteratively compares each sample to each other sample. It is represented in the following pseudocode:

BruteForceClosest(P)

n = P.length

dmin = infinity

for i = 1 to n-1

for j = i+1 to n

d = ((xi - xj)^2 + (yi-yj)^2)

if d < dmin

dmin = d;

index1 = i;

index2 = j;

return index1, index2;