Sean Richardson

Cs141 s17

4-28-17

Assignment #1: Nearest Neighbor Algorithm

This assignment covers two implementations of the nearest neighbor algorithm, one in brute force, and the other in divide and conquer. The naïve implementation uses nested loops to search for the distance where the theoretical runtime would be O(n^2). The divide and conquer method uses a strategy which has a theoretical runtime O(nlogn).

Based on observations about the theoretical runtime the divide and conquer method is superior in terms of speed. This is confirmed through the experimental results. For small input sizes, the brute force and the divide and conquer algorithms both run relatively quickly. However once input size starts to grow large, the brute force algorithm becomes very slow. Running the brute force on the largest set of points took almost 87 minutes compared to about 3 seconds for the divide and conquer.

When inspecting the structures of the algorithms the divide and conquer method contains a nested loop, like the brute force, however in the divide and conquer the total loop time is still O(n) because the inner loop will have to do at most 8 iterations, which is constant, multiplied by n points from the outer loop. Then the base case of the recursive function calls the brute force when size of the array of points is less than or equal to 3. Then the brute force will have at most 9 computations, which can be constant time as well.

The theoretical analysis abstracts away much of the low-level things and focuses of the algorithm at a high level. There is extra overhead and computations which are not accounted for in the theoretical analysis that the actual computer does that would affect the runtime. The divide and conquer implementation is much faster because it continuously breaks the problem into smaller sub-problems which are solved, while the brute force iteratively works on the whole problem which is less efficient.

Input file	Brute force	Divide && conquer
Input_10.txt	.000125 seconds	.000383 seconds
Input_100.txt	.003393 seconds	.000921 seconds
Input_10e5.txt	26.99573 seconds	.227331 seconds
Input_10e6.txt	5234.350 seconds	2.75291 seconds