COMP0005

Algorithms

Team Members

Agnieszka (20090060)

Passawis (20021096)

Oluwaponmile (20015067)

UNIVERSITY COLLEGE LONDON

2020/2021

Introduction

The purpose of this report is to show the difference in performance of various Algorithms trying to compute the convex hull of a set of points. These Algorithms in particular include the Jarvis March and the Graham Scan.

Testing Method

In order to test how well each algorithm operated for different numbers of points we generated 10 sets of points of various sizes (100, 1000, 5000, 10000, 15000 and 20000 points) and timed how long it took to carry out the respective algorithm on these sets of points. The mean time was then calculated to determine how long it takes on average to compute the convex hull. The results shown in this report are presented as points(x-axis) against time in seconds(y-axis).

Jarvis March

Theoretical Complexity

The complexity of Jarvis March is NP, where N is the number of points provided and P is the number of points on the convex hull. The reasoning behind this is because the Jarvis March algorithm works by comparing each point with each of the other N points to see which has the smallest clockwise angle with the previous two points. This is repeated until the first (or left most) point is reached, which occurs after all points on the convex hull, P, have been found. Resulting in NP comparisons.

Worst Case

The worst case for Jarvis March occurs when every point tested is included in the convex hull. Since the number of points on the hull, P, is equal to the number of points given, N, the complexity is $O(N^2)$ in the worst case scenario. Figure 4 shows that the time taken to carry out the algorithm increases exponentially with the number of points provided, in the worst case scenario.

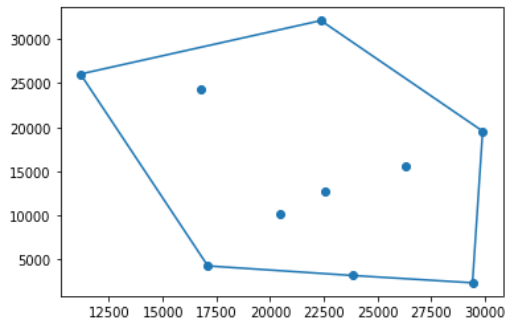


Figure 1: Average case for Jarvis March

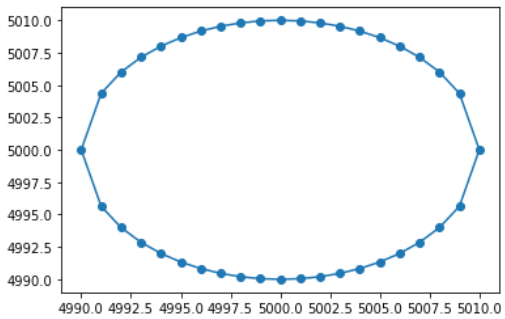


Figure 2: Worst case for Jarvis March

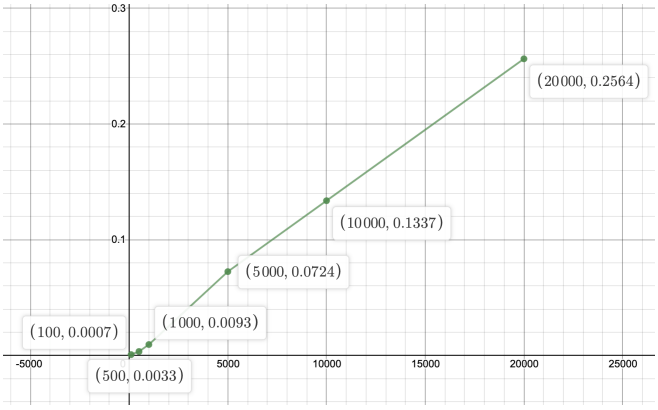


Figure 3: Jarvis March Average Case

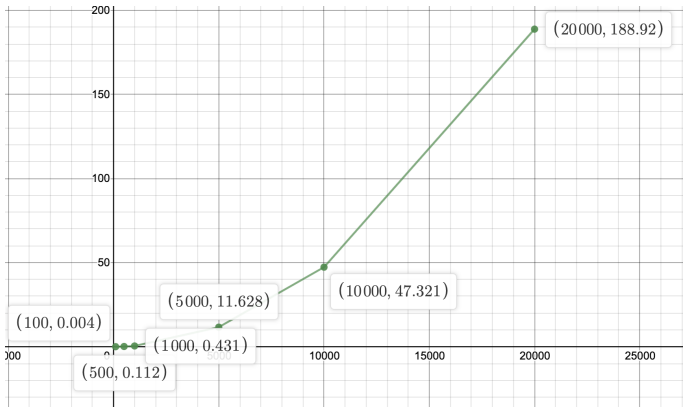


Figure 4: Jarvis March Worst Case

Graham Scan

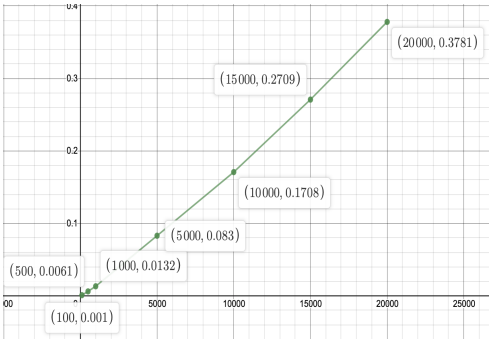
Theoretical Complexity

Graham Scan is another, more efficient algorithm for finding the convex hull of a finite set of points. In best, average and worse cases the time complexity of this algorithm is O(NlogN).

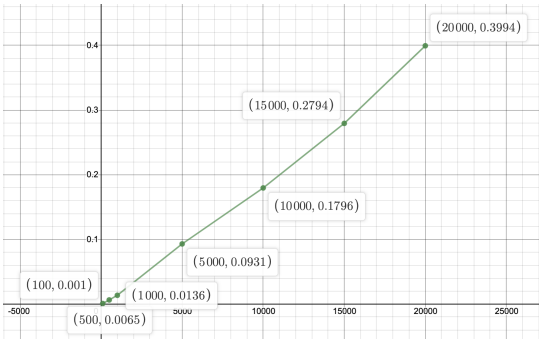
The first step - finding the most-bottom point is linear, therefore it takes O(N) time. The second step is sorting, which can be done in time O(NlogN) (complexity guaranteed by Merge Sort algorithm). The third step is operating on a stack, each point is pushed and popped at most once, thus time complexity is O(N). Then iterating through an array of points one by one takes O(N) time. Therefore the total complexity (O(1) neglected) is O(N) + O(NlogN) + O(N) +O(N) → O(NlogN).

Worst Case

Since the complexity of the Graham Scan Algorithm is O(NlogN), the computational complexity is based on the sorting algorithm used, in our case merge sort. To create the worst case scenario for merge sort, we sorted the set of points, and then unsorted them, by splitting and swapping adjacent points. By doing this to resort the unsorted points, the merge sort algorithm would need to carry out the maximum possible merges. Though this does slightly slow down operation of the overall algorithm, as seen in figures 5 and 6, the overall complexity does not change, so the worst case scenario for Graham Scan still has complexity O(NlogN)



Figures 5 and 6 Graham Scan average(left) and worst case(right)



Extended Graham Scan

Theoretical Complexity

To further improve Graham Scan algorithm we implemented Akl-Toussaint Heuristic to reduce the number of points which would have to be processed by Graham Scan. The Akl-Toussaint heuristic generates a polygon made out of four furthest points (smallest and greatest x and y points) found in the given set (time complexity O(N)). Then we search for all the points that lay outside the said polygon (O(N)). We only consider those points as anything within the polygon would require a right turn to reach, and therefore cannot be part of the convex hull. This was done by computing the cross product between each side of the polygon and each point. If a required turn to reach the point is outside (or on the edge) of the polygon then the point can be considered in the main Graham Scan algorithm. This significantly reduces the number of points, N, required to be sorted before carrying out the algorithm.

Worst Case

The worst case results differ slightly from the average case, this is because we used Toussaint heuristic to remove the majority of the points that lie inside the polygon. This results in our merge sort having to do less merges overall.

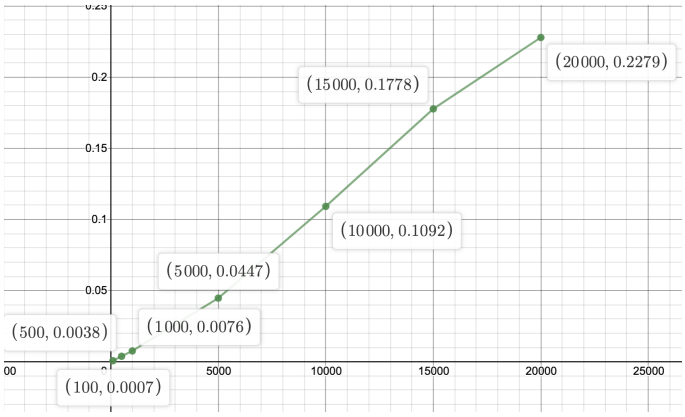


Figure 7: Extended Average Case

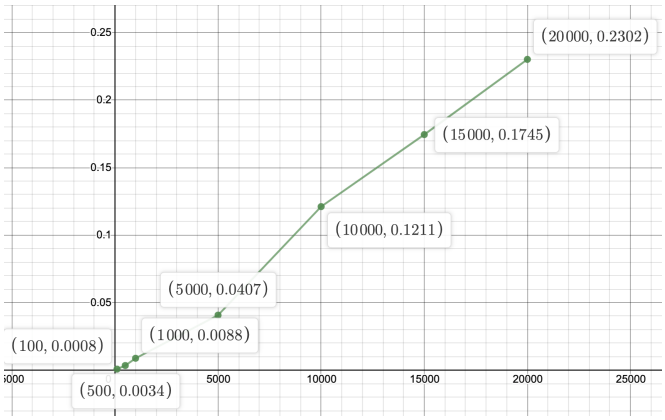


Figure 8: Extended Worst Case

Conclusion

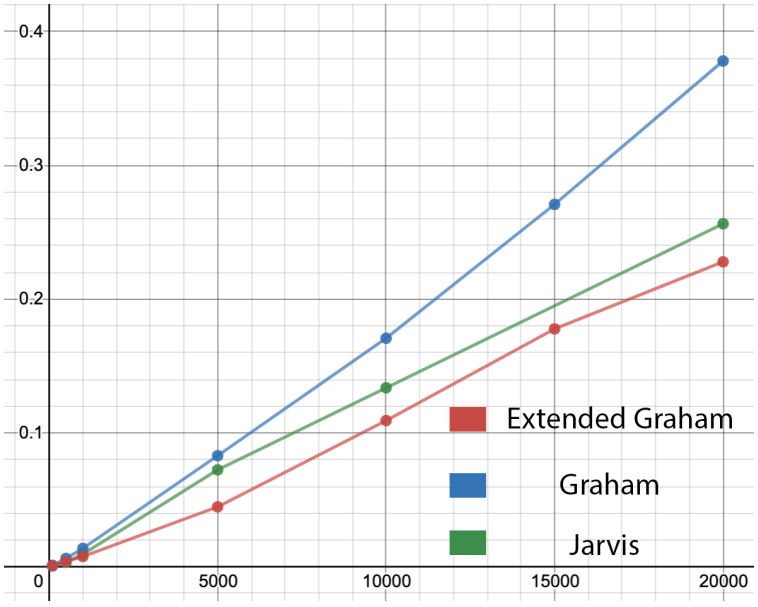


Figure 9: Comparison on Average Case

As shown in Figure 9, the Jarvis March algorithm performs better than Graham Scan on most average dataset. Graham Scan's time complexity grows with the amount of the data set, but Jarvis's growth, in addition to the number of inputs, is also dependent on the number of points on the convex hull. Since the majority of the convex hull does not have a lot of points on the hull Jarvis would be better fit in most applications than Graham Scan.

The extended Graham Scan performs better than both of the other algorithms as the number of points being considered is drastically reduced thanks to the Akl-Toussaint Heuristic.