Computational Geometry

Assignment 4

submitted by
Divyang Mittal (17CS10012)
Amatya Sharma (17CS30042)

March 4, 2021

1 Problem 1

2 Problem 2

We describe the pseudocode for the problem in Algorithm 1. Intuitively we iteratively apply Jarvis March algorithm and remove the current convex hull layer from the current set of points. Once this is done we binary search for query point q.

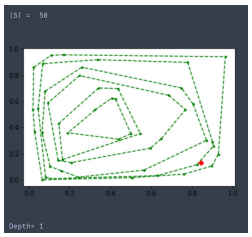
Algorithm 1 Convex Layering

1: $S \leftarrow$ Set of randomly allocated points
2: $ConvLayers \leftarrow \phi$ (Dynamically alloted 2D array to store layers).
3: $l = 0$
4: **while** S is not empty **do**
5: Run Jarvis March on S. Let the convex hull computed be $\mathcal{H}^S$
6: $ConvLayers[k] \leftarrow \mathcal{H}^S$
7: $S \leftarrow S \setminus \mathcal{H}^S$, $k++$
8: **end while**
9: Return $ConvLayers$ representing the k convex layers.
10: Run a binary search on $ConvLayers$ with query point q and output the depth accordingly.

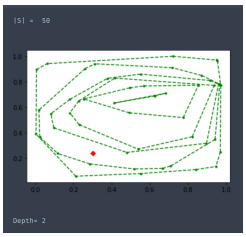
2.1 Time and Space Complexity

Time taken to compute the convex hull layers is clearly $O(\sum_{i\in[k]}(|ConvLayers[i]|.n))$ which is $O(n^2)$ in the worst case. Since $ConvLayers$ is dynammically alloted, it takes $O(n)$ space. The data structure $ConvLayers$ stores all layers with vertices in each layer sorted in clockwise manner.

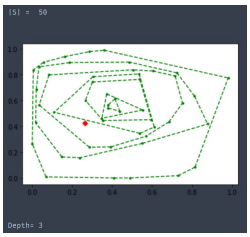
For computing the depth of query point q, we use simple binary search which takes $O(logk)$ time. To find whether a layer $ConvLayers[j]$ contains q or not we require a $O(log|ConvLayers[j]|)$ time. This gives us a worst case runtime of $O((logn)^2)$, since $k \leqslant n$ and $|ConvLayers[j]| \leqslant n \ \forall j \in [k]$.



(a) $|S| = 50,\ depth = 1$

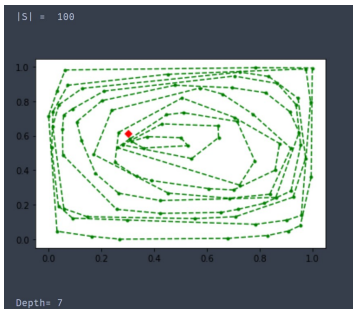


(b) $|S| = 50,\ depth = 2$

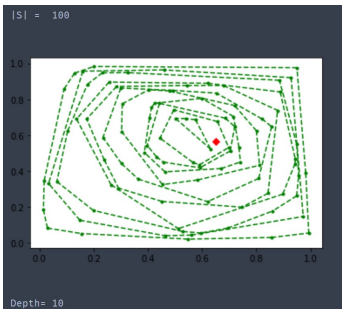


(c) $|S| = 50,\ depth = 3$

(d) $|S| = 100,\ depth = 3$



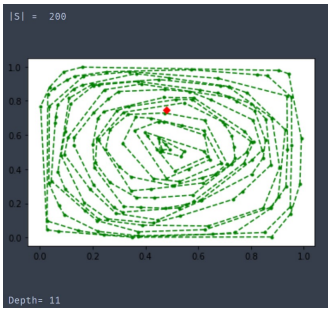
(e) $|S| = 100,\ depth = 7$



(f) $|S| = 100,\ depth = 10$

(g) $|S| = 200,\ depth = 0$

(h) $|S| = 200,\ depth = 4$



(i) $|S| = 200,\ depth = 11$

Figure 1: Runs of Algorithm 1 with $|S| \in \{50,\ 100,\ 200\}$. Red Point denotes the query point.