Computational Geometry

Assignment 2

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

January 27, 2021

Q.2(a). δ-*wide Minimum Area Polygonization* (δ-$\mathcal{MAP}$)

Lets denote the problem by δ-*wide Minimum Area Polygonization* (δ-$\mathcal{MAP}$). We consider the following assumptions:

- ▷ The machine tool enters and exits from the left boundary of sheet (any other case can be converted to this case using simple preprocessing)
- ▷ Points have integral, general coordinates.
- ▷ No two discs (defective points) overlap.
- ▷ $\delta \geqslant k$
- ▷ The tool cannot cut infitesimally small arcs (this allows us to ignore any curving and focus on polygonization).

We first define the problem of *Euclidean Steiner Tree* ($\mathcal{EST}$) as follows: "Given n terminal points in the plane, find the minimum length tree such that all n terminals are conected and that any two terminal points may be interconnected by line segments either directly or via other points (called as steiner points) and line segments."

The problem $\mathcal{EST}$ is NP-hard. Intuitively the δ-$\mathcal{MAP}$ can be reduced from $\mathcal{EST}$ by setting $k = 0$ (consider the terminal points of $\mathcal{EST}$ as discs of δ-$\mathcal{MAP}$) and $\delta = \varepsilon$, where $\varepsilon \approx 0$ (such that minimizing the area in δ-$\mathcal{MAP}$ becomes equivalent to minimizing the length of tree connecting the terminal points).

[Algorithm and Complexity] $\mathcal{EST}$ has a PTAS (Polynomial Time Running Scheme) i.e. a $(1+\varepsilon)$-approximation factor polynomial time algorithm [BGRS10]. We use this PTAS as a black-box to obtain a near optimal solution for our problem. We call an edge a *rail-road* if its a rectangular region with the almost the same length as that of edge ($\ell + k$) and a width of δ (its like a bloated edge along its width). Note that we also increase the length by $\frac{k}{2}$ to each side to cover the defective discs completely. Intuitively we find a minimum steiner tree (MSNT) with edges as δ-wide rail-roads. After obtaining the MSNT, we combine the leftmost point of MSNT to the left edge of sheet using a horizontal rail-road. The Algorithm; defined more formally; is given by Algorithm 1.

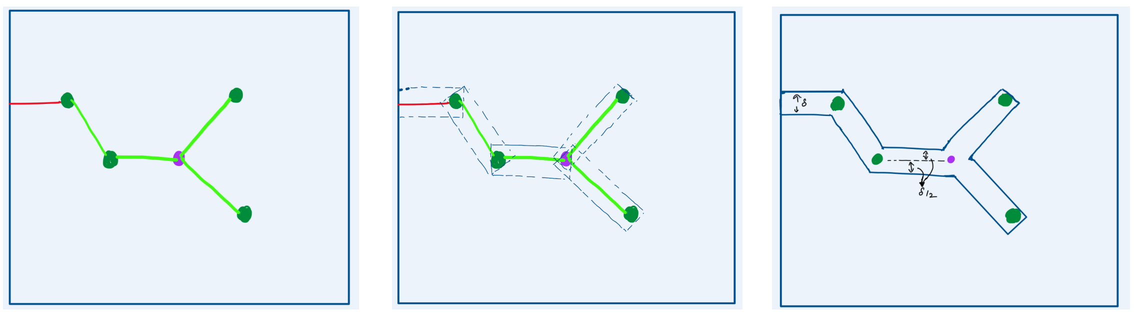
Algorithm 1 PTAS δ-$\mathcal{MAP}$ (Run Shown in Figure 1)

```
1: Run PTAS of $\mathcal{EST}$ with $n$ terminals as centres of $n$-defective spots to obtain MSNT $\mathcal{T}$.
2: Find the leftmost point from the nodes of tree (maybe a non-terminal as well) say $p_l$.
3: Connect a horizontal edge from $p_l$ to left boundary of plane, call the corresponding point on boundary as $p_b$.
4: Add the edge in previous step and $p_b$ to $\mathcal{T}$ (as shown in Figure 1(a)).
5: Replace the edge of $\mathcal{T}$ with rail-roads ($\delta$-wide rectangles of almost the same length as that of edge) such that the original edge
   passes through the centre of respective rail-road (i.e. partitioning it into two rectangles of same length but $\frac{\delta}{2}$-width each) (as
   shown in Figure 1(b)).
6: Consider the $\mathrm{union}^{\text{refer to Techniques\&DataStructures}}$ of region formed by rail-roads, say $\mathcal{T}_{\mathrm{bloated}}$ (clearly this is a
   polygon). Return $\mathcal{T}_{\mathrm{bloated}}$ as near-optimal solution to $\delta$-$\mathcal{MAP}$ (as shown in Figure 1(c)).
```

[Complexity] Algorithm 1 gives a PTAS i.e. $(1+\varepsilon)$-approximation poly time algorithm for δ-$\mathcal{MAP}$. Step 5 and 6 of algorithm take constant time (see Techniques & Data Structures). This is the best we can perform for a NP-hard problem with a polynomial time algorithm.

[Correctness] Trivially, an optimal solution to δ-$\mathcal{MAP}$ must be a polygon with width no more (no less) than δ (if the width is greater than δ at some point, from our initial assumptions we can shrink the polygon at that place to get a smaller-area polygon). Since the width can be considered uniform (i.e. $= \delta$) throughout, the problem reduces to finding a minimum length tree than connects all the defective points (probably by using other points in the plane). Step 3 is justified as it appends a minimum area polygon/rail-road (minimum length edge) to the resulting tree. Again, since $\delta \geqslant k$ we can safely assume that the discs lie completely inside the returned polygon $\mathcal{T}_{\mathrm{bloated}}$.

[Techniques & Data Structures] Apart from the PTAS-blackbox of $\mathcal{EST}$, a major challenge while obtaining $\mathcal{T}_{\mathrm{bloated}}$ from $\mathcal{T}$ is to compute union of convex polygons (i.e. rail-roads) obtained. We can do this by performing union of polygons considering two at a time. This can be done in constant time by doing triangulations of two polygons and then computing the Minkowski sum of each triangle obtained by a plane sweep algorithm which merges the boundaries of two polygons tracing out the new boundary (in general case it is $O(m+n)$, where m, n are number of vertices of two polygons which is 4 each in our case).



(a) (b) (c)

Figure 1: A Run of Algorithm 1. (a) Step 1-4 : Defective Points (colored darkgreen), Steiner Point (colored purple), MSNT(Shown in light green) and an additional edge to leftmost point on tree(colored red). (b) Step 5: Replacing edges by δ-wide rail-roads. (c) Step 6 : Merging the δ-wide rail-roads to obtain $\mathcal{T}_{\mathrm{bloated}}$.

Q.2b. *Polygonization over a hole* ($\mathcal{PH}$)

Lets denote the problem as *Polygonization over a hole* ($\mathcal{PH}$). We label a point as *Red* or *Blue* based on whether it belongs to the set to be polygonized or to the hole (lets denote cardinality of these sets by r and b respectively and $n = r + b$). We consider the following assumptions:

- ▷ All points have integral, general coordinates.
- ▷ No red point lies inside $\mathcal{H}$.
- ▷ Every pair of blue points has a finite distance between them (used to find a point infinitesimaly away from each blue point).

A trivial necessary (not sufficient) condition for the instance to be a Yes Instance is that the convex hull of red points completely contains $\mathcal{H}$ inside it. In case a such a convex hull doesn't exist, we can always deal it by adding 3- extra red points in the form of a triangle which contains all the red and blue points (imagine a large such triangle). W.l.o.g from now on we will implicitly assume that such a convex hull exists. Although it should be noted that this is not a sufficient condition as there may be points trapped inside $\mathcal{H}$ (if it is spiral) not "visible" to the convex hull.

We now realize the problem from a graph theoretic view. We compute a (simple, undirected) visibility graph $\mathcal{G}$ as follows: Vertices are Red Points. There is an edge between two nodes iff they are visible to each other (i.e. the line segment joining the two given points has no intersection with the hole). This takes $O(n^2)$ time and space. It is evident that there is a planar hamiltonian cycle in $\mathcal{G}$ if there is a polygonization of the points (we need it to planar to ensure that the polygon is simple and no-edges are crossing each other). Again, this condition alone is not sufficient as the ham-cycle may output a polygon with the $\mathcal{H}$ completely outside it (as shown in Figure 2(a).

However combining the above two conditions satisfies the sufficiency. This can be intuitively observed from the fact that a case like Figure 2(a) cannot exist if the convex hull of red points encloses the hole. This gives us the necessary and sufficient conditions for a the given $\mathcal{PH}$ instance to be a Yes Instance:

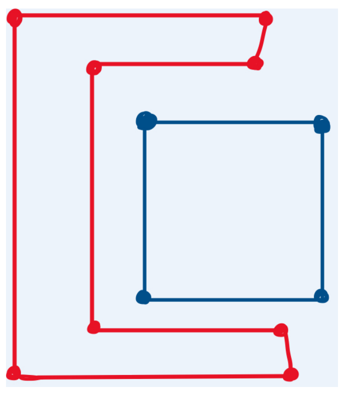
1. $\exists$ a planar hamiltonian cycle in visibility graph $\mathcal{G}$ of input instance.
2. The convex hull of red points must be enclosing $\mathcal{H}$

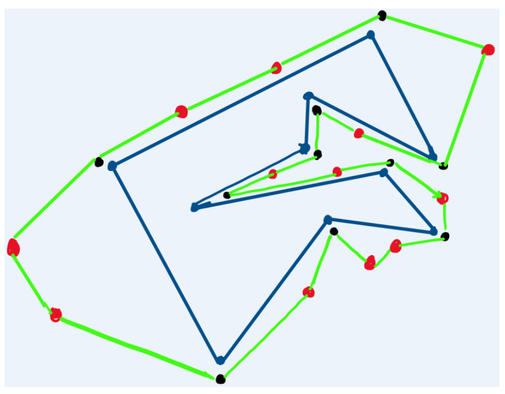
However we realise that finding a Ham-Cycle which is planar (note that it is not same as Planar-HamCycle Problem) is NP-hard, implying that we need exponential time to solve the problem (enumerate all possible vertex orderings and check if they form a planar HamCycle : $O^*(n^n)$).

Now assuming that we have the fraternity to add linear number of additional points to the red set. For every blue vertex we add a point infinitesimally close to it and outside of $\mathcal{H}$ (this can be done by considering exterior angle bisector and placing point on it at a some ε distance away). This adds almost b extra points. Lets label this new set of points as gray. Intuitively, we do this to handle the case when there are points not connected to (visible to) points of convex hull. Adding the gray points ensures that every point can be connected (using a disjoint subset of gray points) to the convex hull (consider it as an envelop over the hole which acts as an escape route for the hidden points).

[Enveloping Algorithm] More formally, to find $\mathcal{P}$, we start with choosing two gray points corresponding to an edge of $\mathcal{H}$). For these two points, find the maximal simple-polygonal chain of red points; disjoint from the area of the hole; starting and ending at these two points. Then we consider the next pair of gray points (corresponding to the adjoining edge of previously considered edge) and form a similar simple-polygonal chain. We halt at the point when all edges have been considered. (its analogous to a walk around the hole $\mathcal{H}$, iteratively building the chain to find $\mathcal{P}$). This completes the description of algorithm.

[Complexity] We first do a simple pre-processing in time $O(n^2)$ as done in visibility graph. Then at each step of iterative algorithm, we have to check all other points and their visibilities (searching the adjacency matrix of $\mathcal{G}$) which will again take $O(n^2)$ time. Summing up over all iterations, time complexity comes out to be $O(b.n^2)$. (Please note that adding b gray vertices, changes n from $(r + b)$ to $(r + 2b)$, and thus doesn't change n (asymptotically)).


(a) A Counter Example where visibility graph has a planar ham-cycle but $\mathcal{PH}$ is a NO instance.


(b) A run of Enveloping algorithm. Green color decribes the envelop formed by algorithm. Other vertices are colored as per convention.

References

[BGRS10] Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanita. An improved lp-based approximation for steiner tree. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 583–592, 2010.