Separators of Delaunay triangulations

Jonathan MacKenzie[1] Evan Thorpe[2]

2019-04-29

[1]Department of Math and Computing Science, Saint Mary's University
[2]Department of Math and Computing Science, Saint Mary's University

Introduction

Our chosen problem is a combination of two problems and their algorithms, namely *Delaunay triangulation* in $\mathbb{R}^2$ and the *Planar Separator theorem* of Lipton and Tarjan [Lipton and Tarjan, 1979].

Introduction

- Delaunay triangulation triangulates (connects into triangles) points in $\mathbb{R}^2$ such that no point is contained in the circumscribed circle of any triangle in the triangulation.[3] Given a set of points in $\mathbb{R}^2$, a Delaunay triangulation can be found in $O(n \log n)$ time.
- Lipton and Tarjan's separator theorem states that for any planar graph, one can always partition the graph into two disjoint connected subgraphs, each of size no more than $2n/3$, by removing at most $2\sqrt{2}\sqrt{n}$ vertices. These vertices to remove can be found in $O(n)$ time [Lipton and Tarjan, 1979].

[3] A delaunay triangulation can also be defined as a triangulation such that for any triangles sharing an edge, their opposing angles add to at least π

The problem

- The question, posed by Stefan Langerman in [ORourke, 2018], observes that Delaunay triangulations are in a sense planar graphs.
- Can such an $O(\sqrt{n})$ separator be found for a Delaunay triangulation in $O(n)$ time given only the set of points in $\mathbb{R}^2$?
- To do so, one would have to avoid computing the full Delaunay triangulation, which is $O(n \log n)$, yet still find an $O(\sqrt{n})$ separator for this triangulation.

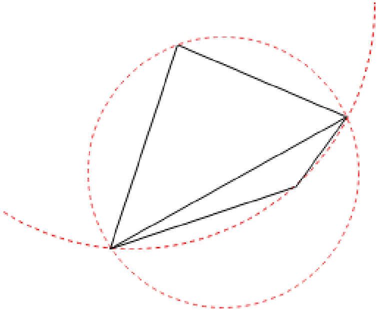
Motivation

- Delaunay triangulation is heavily used in graphics because it avoids creating "sliver triangles" (triangles with one or two very acute angles)
- The ability to split a graph into roughly equal-sized partitions in $O(n)$ time at the cost of $O(\sqrt{n})$ vertices can be very useful for recursive divide-and-conquer algorithms
- A separator for a Delaunay triangulation could be useful for recursive divide-and-conquer algorithms in graphics and computational geometry.

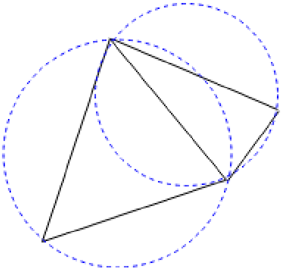
Delaunay triangulation

- Delaunay triangulations are very well-known constructs in computational geometry.
- Two main approaches exist to find them, both of which are $O(n \log n)$ in time.
  1. Incremental approach, as described in [Guibas et al., 1992]
  2. Divide-and-conquer approach, as described in [Guibas and Stolfi, 1983]

Incorrect Delaunay Triangulation



Correct Delaunay Triangulation



The Lipton-Tarjan Separator Theorem

Lipton and Tarjan's separator theorem depends upon three main theorems:

1. The Jordan Curve Theorem [Hall and II, 1955]: Let C be any closed curve in the plane. Removal of C divides the plane into exactly two connected regions, the "inside" and the "outside" of C
2. Any n-vertex planar graph with $n \geq 3$ contains no more than $3n - 6$ edges (a corollary of Euler's Formula $V - E + F = 2$) [Harary, 1969]
3. Kuratowski's Theorem [Kuratowski, 1930]: A graph is planar if and only if it contains neither a complete graph on 5 vertices (K_5) nor a complete bipartite graph on two sets of three vertices ($K_{3,3}$).

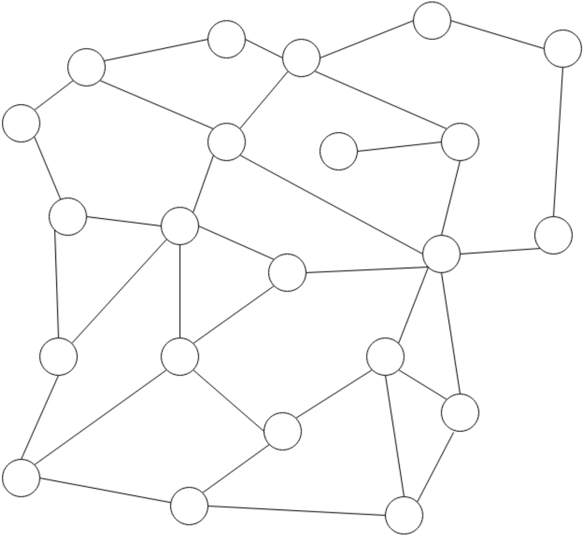
Graph Shrinking

- From Kuratowski's Theorem, it follows that shrinking any edge to a single vertex preserves planarity, as for the shrunken graph to be non-planar it would need to contain either a K_5 or a $K_{3,3}$ and thus so would the original graph, which would then not be planar.
- From this, it follows that any connected subgraph of a planar graph can be shrunken to a vertex without loss of planarity.

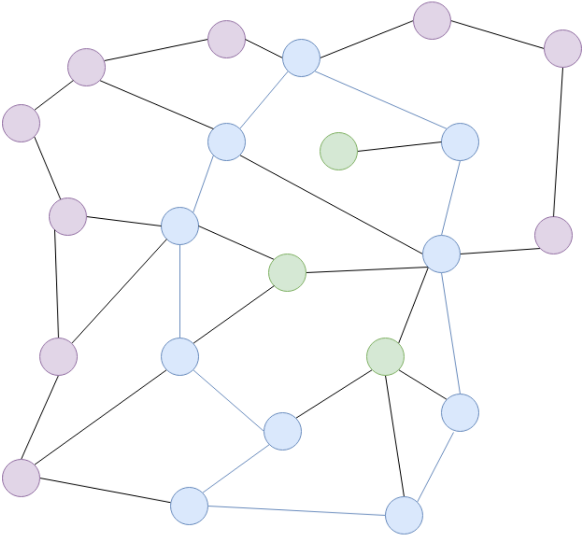
The Lipton-Tarjan Algorithm

- The main idea of the Lipton-Tarjan separator is to apply the Jordan Curve Theorem to the planar embedding of a connected graph.
- A "closed curve" is then a cycle, and, by the theorem, removing this cycle separates the graph into two disjoint subgraphs, assuming there was at least one vertex inside this cycle.

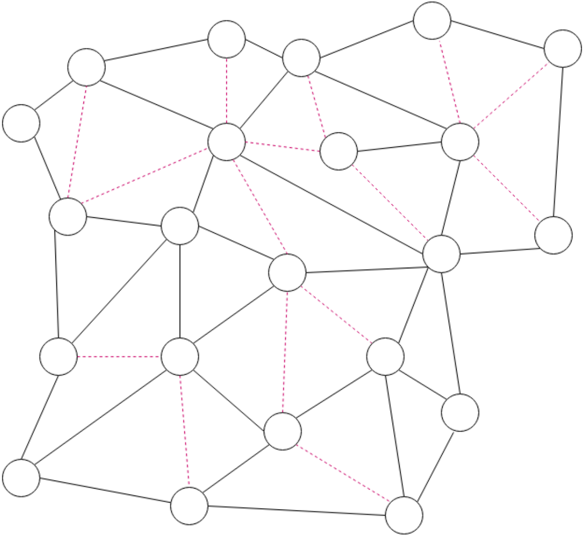
Lipton-Tarjan Illustration



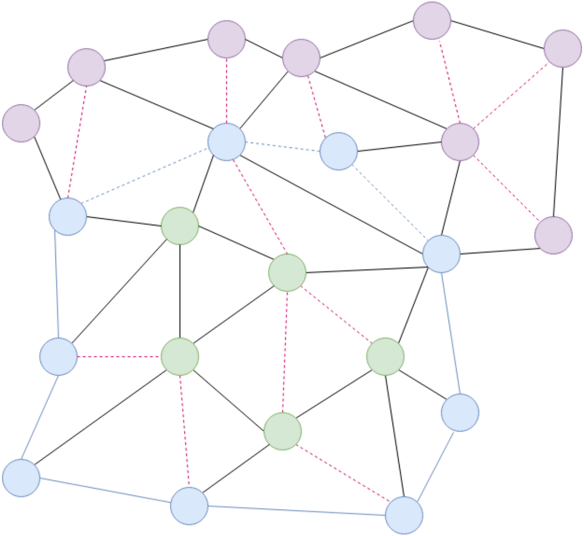
Lipton-Tarjan Illustration



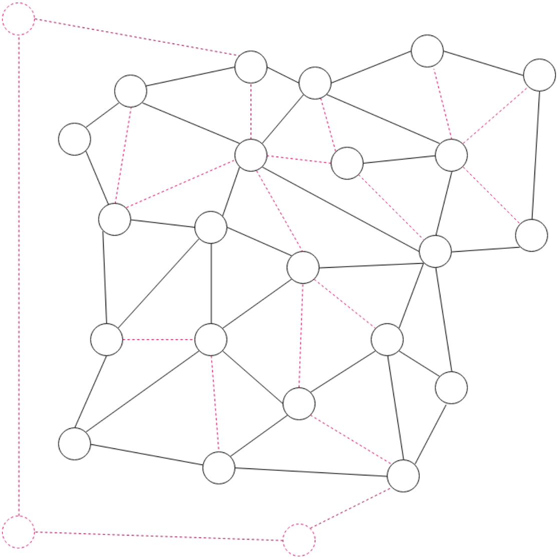
Lipton-Tarjan Illustration



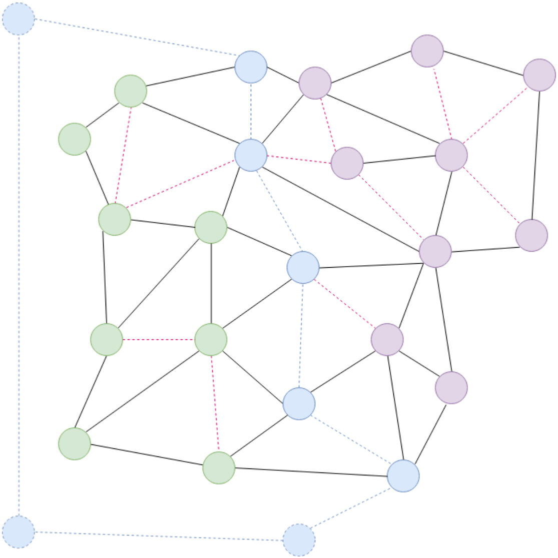
Lipton-Tarjan Illustration



Lipton-Tarjan Illustration



Lipton-Tarjan Illustration



The Lipton-Tarjan Algorithm

- To be a good separator, such a cycle would need to be on $O(\sqrt{n})$ vertices, and the "inside" and "outside" its removal partitions the graph into must each contain no more than $2n/3$ vertices.
- The actual separator algorithm is more subtle than this.

The Lipton-Tarjan Algorithm

1. Find a planar embedding of the graph[4] and store each edge with a pointer to each of its vertices and the immediate clockwise and counterclockwise neighbours of those vertices.
2. Compute a breadth-first spanning tree of the graph and record each vertex's level in the tree.
3. Delete vertices in the higher levels. Apply "shrinking" to the lower levels, reducing them so a weighted vertex x.
4. Modify the original spanning tree so it is rooted at x. Add non-tree edges to make all faces of the graph triangles.

(continued next slide)

[4] In our case, we are given the locations of the "vertices" in the plane, but not the edges. In the graph theory case, the Lipton-Tarjan algorithm is given an abstract representation of a graph and must position the vertices in the plane.

The Lipton-Tarjan Algorithm

5. Choose any non-tree vertex and follow the parent pointers of the vertices this edge connects to form a cycle on the spanning tree.
6. Compute the cost of the vertices on the "inside" and "outside" of this cycle.
7. If this cycle does not partition the graph according to the $2n/3$ constraint, locate a better cycle.[5]

We have left out many details in this description, but in essence the cycle found by this algorithm contains the $O(\sqrt{n})$ vertices that partition the graph into disjoint subgraphs of size no more than $2n/3$.

[5] Once again, the details are omitted from this presentation. Lipton and Tarjan prove that the initially-chosen cycle can be modified into a satisfactory cycle in $O(n)$ time. [Lipton and Tarjan, 1979]

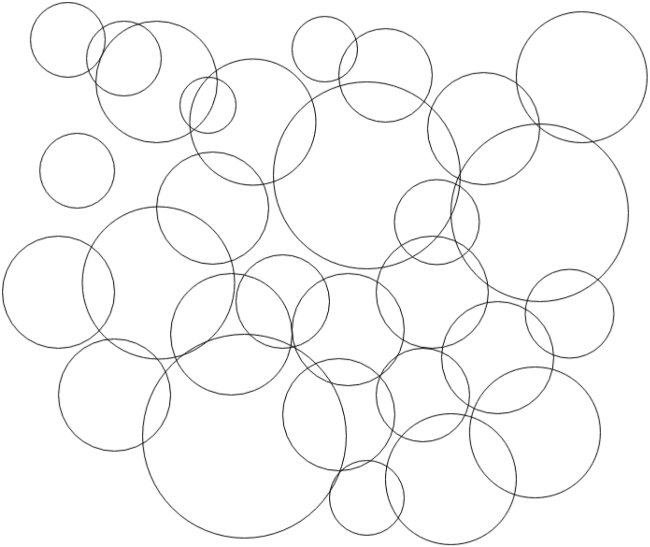
Sphere-Packing Separator Theorem

Another paper we examined was [Miller et al., 1997], which examines sphere packings and gives an $O(n)$ separator theorem. Their result generalises to higher dimensions, but in the two-dimensional case their result is as follows:

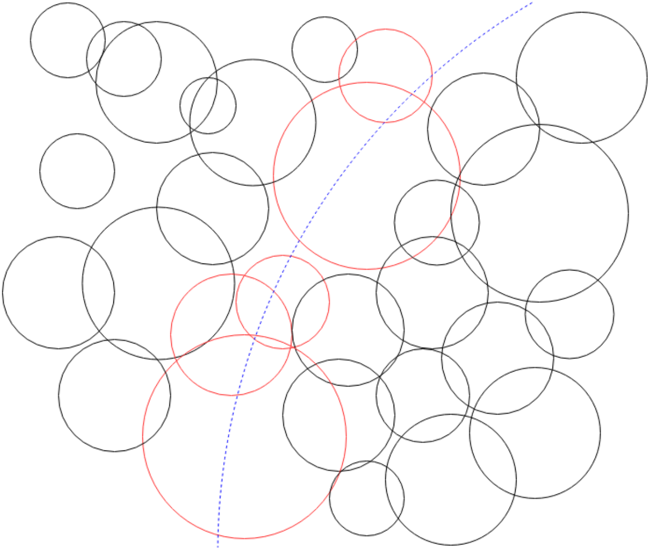
- For every k-ply system,[6] there exists a circle that intersects at most $O(\sqrt{kn})$ circles and divides the rest of the circles into two disjoint parts, the largest containing at most $3n/4$ circles. [Miller et al., 1997]

[6] A collection of circles embedded in a plane forms a k-ply system if no point in the plane is within more than k of those circles.

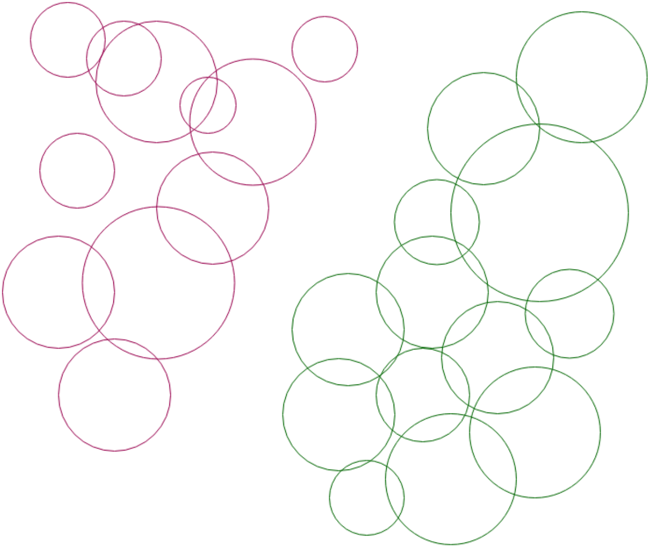
Sphere-Packing Illustration



Sphere-Packing Illustration



Sphere-Packing Illustration



Sphere-Packing Separator Theorem

- This result is a proof[7] of the Lipton-Tarjan theorem through the circle packing theorem [Koebe, 1936], which states that for every planar graph G, there is a circle packing (a collection of non-overlapping circles) in the plane whose intersection (tangency) graph is (isomorphic to) G.
- Much like the original Lipton-Tarjan algorithm, this method divides the circles into the “inside” and “outside” of a Jordan curve analogue, in this case a circle.
- The circle is found by projecting the centres of the circles up onto a three-dimensional unit sphere, finding a great circle corresponding to the centrepoint of these centres, and then projecting this great circle back down onto the two-dimensional plane.

[7] Their result is not only a proof but a generalisation of the Lipton-Tarjan theorem into higher dimensions.

Algorithm/ Theoretical Analysis

- Our most promising lead was the sphere-packing separator theorem described in [Miller et al., 1997].
- At first there appears to be a connection: the circumcircles considered in a Delaunay triangulation form a k-ply system[8] and one might expect that a separator of such circles would translate nicely to a separator of the Delaunay triangulation of a set of points.
- But there is a problem...

[8] We do not know k, but as long as it is less than n a $O(\sqrt{kn})$ separator of these circles will still be $O(\sqrt{n})$.

Algorithm/ Theoretical Analysis

- Problem: the sphere separator algorithm works by finding the centrepoint of the centres of all these circles, thus requiring the full Delaunay triangulation to begin with.
- While we may be able to compute a small subset of the final circumcircles and use them as a heuristic, we would not be able to rigourously prove a separator algorithm that does not first compute the full Delaunay triangulation, defeating our hope for a linear-time algorithm.

Existing software — Lipton-Tarjan

- We found existing packages in both Python and C++.
- Both packages contain useful tools for testing.
- The python package has a random planar graph generator, as well as a way to test if the algorithm worked correctly.
- The C++ package has multiple pre-defined graphs that we can test on.
- We were unable to find any libraries for the sphere-packing separator algorithm described in [Miller et al., 1997].

Existing software — Delaunay Triangulation

- We have found two different implementations, both of them using python with numpy. The two differ in the fact that they implement different algorithms.
- The implementation in [Likhosherstov, 2018] uses the Bowyer-Watson algorithm, which has an average time complexity of $O(n \log n)$, however, its worse case running time is $O(n^2)$ [Arens, 2002]. As the author of the code notes, they did not write it to be efficient, but rather wrote it for readability.
- The other implementation, seen in [Pletzer, 2015], uses the s-hull method. This method also has a time complexity of $O(n \log n)$. The author of this code admits that it could be improved with support for holes, and also by having it automatically add Steiner points.

Methodology

The initial goal was of course a linear-time separator algorithm for the Delaunay triangulation of a set of points in $\mathbb{R}^2$. However, we soon realised that this was infeasible given the limited time and resources, and developed the following plan:

- Begin from a simple concatenation of Delaunay triangulation and graph separation and find areas where this algorithm can be optimised.
- Make these optimisations and run some empirical tests, comparing it to the simple concatenation algorithm and examining how it reacts to various inputs.
- The hope would be for an algorithm that, while still $O(n \log n)$, is empirically faster than computing the full Delaunay triangulation and separating it.

Methodology

- Unfortunately, due to time constraints and unforeseen complexity, we were unable to develop such an optimisation.
- We instead decided our time would be best spent investigating the problem and presenting it in a manner that can lay the groundwork for further research.
- Rather than trying to pull together a program from code we did not understand to implement algorithms we don't understand.

Future Work

- Dissect and degeneralise both algorithms, putting them in a common framework for analysis.
- Discover common assumptions and requirements between the algorithms and optimise accordingly.
- Develop new representations of the problems space and algorithms involved to find common ground.

Conclusion

We were unable to develop a linear-time algorithm to separate the Delaunay triangulation of a set of points in $\mathbb{R}^2$ but...

- Laid the groundwork for future attempts
- Showed possible bijections to other methods
- Ruled out using un-modified sphere-packing separator algorithm but ideas from this theorem could still be applicable

Questions?

References I

Arens, C. (2002).
The bowyer-watson algorithm; an efficient implementation in a database environment.
Case study report TU Delft, July 2002, 52 p.

Guibas, L. J., Knuth, D. E., and Sharir, M. (1992).
Randomized incremental construction of Delaunay and Voronoi diagrams.
Algorithmica, 7.

Guibas, L. J. and Stolfi, J. (1983).
Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams.
Proceedings of the fifteenth annual ACM symposium on Theory of computing.

Hall, D. W. and II, G. L. S. (1955).
Elementary Topology.
John Wiley & Sons.

Harary, F. (1969).
Graph Theory.
Westview Press.

Koebe, P. (1936).
Kontaktprobleme der konformen Abbildung.
Hirzel.

Kuratowski, C. (1930).
Sur le problème des courbes gauches en topologie.
Fundamenta Mathematicae, 15(1):271–283.

References II

Likhosherstov, V. (2018).
lipton_tarjan.
https://github.com/ValeryTyumen/lipton_tarjan.

Lipton, R. J. and Tarjan, R. E. (1979).
A separator theorem for planar graphs.
SIAM Journal on Applied Mathematics, 36.

Miller, G. L., Teng, S.-H., Thurston, W., and Vavasis, S. A. (1997).
Separators for sphere-packings and nearest neighbor graphs.
Journal of the ACM, 44.

ORourke, J. (2018).
Open problems from CCCG 2017.
Canadian Conference on Computational Geometry.

Pletzer, A. (2015).
Delaunay triangulation (python recipe).
https://code.activestate.com/recipes/579021-delaunay-triangulation/.