A Distributed Greedy Heuristic for Computing Voronoi Tessellations With Applications Towards Peer-to-Peer Networks

Brendan Benshoof Andrew Rosen
Anu G. Bourgeois Robert W. Harrison
Department of Computer Science, Georgia State University

May 28, 2015

Outline

Motivation

- By "distributed system" we mean DHT
-

Voronoi Tessellation and Delaunay Triangulation

- Hopefully this slide is unnecessary.
- Pictures are coming if they are needed

Voronoi Example

- flipping back and forth for these two slides is recommended.
- the points are in the same locations
- this is intended to visualize the primal-dual nature of these two problems.

Delaunay Example

- Each circle has 3 points on it
- Each circle only contains the three points that define it
- Waving your hand at the images seems to help people understand them
- or at least it prompts them to smile and nod.

Distributed Hash Tables

- DHTs are strongly related to P2P-overlay network.
- (some people use the terms interchangeably)
- Fault Tolerance/Robustness is a core goal of a DHT

How are DHTs and Vonroi Tesselation/Delunay Trianguation related?

- This is one of those generalizations that people generally never consider, but consider trivial in retrospect.
- This makes it both important to write papers on and then difficult to get those papers published.

Applications of DHTs

- DHTs are commonly used as a place to "meet in the middle" and find other peers for a specific task
- Bittorent and MainlineDHT (bittorent's DHT) are the largets DHT network in use (aprox 20,000,000 nodes)
- The DHT stores a list of peers serving a given file at the hash of that file.
- In general P2P file sharing is the BIGGEST use case for DHTs

Why do we need a distributed Voronoi heuristic?

- designing algorithms to solve voronoi/delaunay in weird metric spaces and higher dimensions is hard. I want to test if it is useful before I invest that effort.
- This approximation (and the gossip protocol), when you come up with a creative metric space, approximates the behavior of many DHTs

Distributed Greedy Voronoi Heuristic

- This is meant to be on par of sophistication with "just pick the 6 closest nodes" and "all the nodes within 100ft"
- But it ensures the result is fully connected/reachable.
- It is a subset of the Delaunay Triangulation

DGVH Intuition

- This slide is boring. Move on quickly.

DGVH Algorithim

This algorithm is "egocentric". It is meant to be run by a single node in a distributed network and is actively seeking to find it's deluany peers.

1. 'n' is the "myself" node, and the location we are seeking to find the peers of.
2. peers is a set that will build the peerlist in
3. We sort the candidates from closest to farthest.
4. The closest candidate is always guaranteed to be a peer.
5. Iterate through the sorted list of candidates and either add them to the peers set or discard them.
6. We calculate the midpoint between the candidate and the center 'n'.
7. If this midpoint is closer to a peer than 'n', then it does not fall on the interface between the location's voronoi regions.
8. in this case discard it
9. otherwise add it the the current peerlist

Theoretically, this is worst case $O(n^2)$
However in practice, this is $O(nlog(n)(sorting) + kn)$ where k is the number of delunay peers.
We are well aware that 2d-euclidean algorithms exist in O(nlog(n)) time. While we use that use case to communicate the algorithim, it is intended to be used in more exotic spaces.
realistically k is the function of the metric space and is O(1) for euclidean spaces.

DGVH Example

- Note the two edges missing compared to the correct delaunay triangulation.
- This configuration was specifically chosen to demonstrate this failure.

Realistic Candidate set Size

- practically we only need to keep radius 2 hops worth of peers as candidates
- since the number of peers is O(1) in most cases, in the distributed use case this is not the time $O(n^2)$ it could be.
- it is possible for nodes to have a peer count as high as n-1 in contrived cases. Solution: don't do that.
- Realistically worst case is $\Theta(\frac{\log(n)}{\log(\log(n))})$ which is expected maximum degree in a triangulation of random points (regardless of metric or dimensions)

Error Mitigation

- The error rate is essentially the rate at which node occlusions happen.
- It is important to note, that even if nodes are occluded, there is always a multi-hop path between them. (thus fully connected)

Experiment 1

- we compare our heuristic with ground truth in 2D euclidean with random points on a 1.0x1.0 square.
- we calculate both the ground truth delaunay triangulation and results of DGVH.
- We only did in 2D because of time and money. (3D is practically possible but more complex.)
- Higher dimensions and other metric spaces do not have efficient algorithms we could implement with our feeble minds.

Results

This slide is a lot of fanfare for the fact:
very clearly a relation of 1 error per node

Experiment 2

- Essentially, through a combination of DGVH and peer-gossiping (effectively I know 2-hop peers) we build a routable network
- To Gossip: each cycle I exchange 1-hop peers with one of my peers selected at random. Then I recalculate my peer-list using the new information.

Results

- All the networks converge to 100
- Nice sigmoid curves
- Higher dimensionality slows convergence
- we could do higher dimensions here becuase we avoid calculating a ground truth graph
- rather we sample the graph and determine the ground truth for each sample.
- Despite our 1-error per node, routing is still succeeding. it is "Good Enough".

Other Applications

Essentially Wireless Sensor Networks are another field that uses the fast and greedy method of voronoi/delaunay approximation (pick 5 closest nodes or all nodes in 100ft). So our solution should work for them too.

Conclusions

- It is an improvement over bad approximations
- It caps out at $O(n^2)$ complexity, no matter how many dimensions or complexities of the metric space (unless calculating distance or midpoint is worse than $O(1)$)
- for example This means you can use in it an 100-dimensional euclidean space with $O(n^2)$ rather than $O(n^{50})$ (maybe we should have opened with this...)

Bogdan Carbunar, Ananth Grama, and Jan Vitek.
Distributed and dynamic voronoi overlays for coverage detection and distributed hash tables in ad-hoc networks.
In *Parallel and Distributed Systems, 2004. ICPADS 2004. Proceedings. Tenth International Conference on*, pages 549–556. IEEE, 2004.

Xinyu Chen, Michael R Lyu, and Ping Guo.
Voronoi-based sleeping configuration in wireless sensor networks with location error.
In *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pages 1459–1464. IEEE, 2008.