Distance Weighted Flows on Geometric Graphs

Timothy Chu
Carnegie Mellon University
tzchu@andrew.cmu.edu

Gary Miller
Carnegie Mellon University
glmiller@cs.cmu.edu

November 12, 2018

Let G be a graph with doubling dimension D. Fix any vetex pair A and B. For each edge p, compute $\overline{w}_p = \min\{d(p,A), d(p,B)\}^{d-1} w_p$. Now consider the flow minimizer:

$$\sum_e \overline{w}_e f_e^2.$$

Technically, any of:

$$\sum_e \overline{w}_e^{-(k-1)} f_e^k$$

or:

$$\sum_e (\overline{w}_e f_e^2)^s$$

could have been chosen. But our choice of flow minimizer allows us to compute the flows via effective resistance computation. These variables were chosen so that if there were r^{d-1} points away from point A, then the total contribution of these points to the flow would be small.

Ah, this measure is rayleigh, since adding an edge in the graph decreases $\min\{d(p,A), d(p,B)\}$. Which decreases w_p. Which decreases the flow minimizer. This assumes d is fixed.

Suppose we wanted to count the number of points approximately r away in Euclidean distance. I think if you're willing to tolerate an ε error, you can do this as quickly as building an ε WSPD. But if we want to calculate the distance in the geodesic, then suddenly things are more complicated....

0.1 Computing using effective resistance flow?

There's no way this will work, but it could be an interesting crappy heuristic. But computing this way is probably not Rayleigh.

0.2 Are balls connected?

It's unclear. I would doubt it. Looks like it's not for the m bond. Or is it? For the 5 step m bond (3 vertices between the two sides of the bond), the measure has value:

$$\frac{1}{m} + \frac{2^{d-1}}{m} + \frac{1}{m}$$

If we found the distance from the middle vertex in the bond to any of the sides, the value would be:

$$1 + 1$$

.

In general, the m bond would have value:

$$\frac{1}{m} + \frac{2^{d-1}}{m} + \ldots + k^{d-1}m + \ldots + \frac{1}{m}$$

Note that the doubling dimension of the m bond is roughly $\log_2 m$.
Meanwhile the path length is $k/2$.
connected.

0.3 Computation Complexity

Pre-compute all pairs shortest distance, for an overhead of $O(n^2)$ with some huge ε dependencies.

Now for each pair, build the local graph. If the graph is size E (probably linear in n for low dimension), then the time it takes to compute a single pair effective resistance is $O(E)$. Then the total runtime is $O(n^2E)$. Which is $O(n^3)$ plus some huge exponential factors in dimension.

THis is pretty bad, even if we assume low dimension! What would be great is if we could get down to quadratic time.

0.4 An alternate measure from geometry

Geometric graphs are very well structured, so we might as well compute some kind of Katz centrality to each vertex. How do we find how close two points are? We can take the geodesic distance, and multiply by the Katz centrality of both points. This takes into account both absolute geodesic distance (which can be ridiculously shortcut) and Katz Centrality.

One great thing that I'd love, but seems hard to come by, is to build a resistance sparsifier for one graph, and show that it actually works great for all graphs. That sounds highly false though, so I'm probably not gonna run it....

Perhaps the best I can hope for in this distance, is to compute a resistance sparsifier for each pair of vertices, which would still be too much space to store. Damn!