Stat260/CS294: Spectral Graph Methods **Lecture 11 - 02/26/2015**

Lecture: Flow-based Methods for Partitioning Graphs (2 of 2)

Lecturer: Michael Mahoney *Scribe: Michael Mahoney*

Warning: these notes are still very rough. They provide more details on what we discussed in class, but there may still be some errors, incomplete/imprecise statements, etc. in them.

11 Flow-based methods, continued

Recall from last time that we are looking at flow-based graph partitioning algorithm. Last time, we covered the basics of flow-based methods, and we showed how they are very different than spectral methods. This time, we will discuss flow-based graph partitioning from an embedding perspective. We will see that flow-based algorithms implicitly embed the data in a metric space, but one that is very different than the place where spectral-based algorithms embed the data. Thus, not only do they run different steps operationally and get incomparable quality of approximation bounds, but also they implicitly put the data in a very different place—thus "explaining" many of the empirical results that are empirically observed.

(BTW, I have made some comments that spectral methods can scale up to much larger input graphs by using diffusion and random walk ideas, a topic we will get back to later. For the moment, let me just note that the way flow is used is not *immediately* relevant for such "massive" data. For example, the running time of a typical flow-based algorithm will be $O(n^2)$, since it involves a multi-commodity variant of flow; single commodity variants of flow-based algorithms run in $O(n^{3/2})$ time; and more recent work has focused on using Laplacian solver ideas to do even better, i.e., to run in time that is nearly-linear in the size of the input. There is a lot of interest, mostly from within TCS so far, in this area; and these fast solvers hold the promise to make these methods applicable to much larger graphs. I'm hoping to return to some of these Laplacian solver topics at the end of the semester, and I'm also planning on giving at least give a brief introduction to some of the ideas about how to couple spectral methods later.)

11.1 Review some things about ℓ_1 and ℓ_2

Let's review a few things about ℓ_1 and ℓ_2 and related topics.

Definition 1. *The* ℓ_p-norm *on* $\mathbb{R}^k$ *is* $||x||_p = \left(\sum_{i=1}^k |x_i|\right)^{1/p}$. *A finite metric space* (X, ρ) *is* realized *in* ℓ_p^k *if* $\exists f : X \to \mathbb{R}^k$ *s.t.* $\rho(x, y) = ||f(x) - f(y)||_2$, *and it is an* ℓ_p-metric *if it can be realized by* ℓ_p^k *for some* k. *The* metric induced by ℓ_p *is:* $d(x, y) = ||x - y||_p, \quad \forall x, y \in \ell_p$.

For flow-based methods, we will be most interested in ℓ_1, while for spectral-based methods, we are interested in ℓ_2. The ℓ_p norm (except for $p = \infty$, which we won't discuss here) is usually of at most more theoretical interest.

The spaces ℓ_1 and ℓ_2 are very different. (Think ℓ_1 regression, i.e., least absolute deviations, versus ℓ_2 regression, i.e., least-squares regression; or think ℓ_1 regularized ℓ_2 regression, i.e., the lasso, versus ℓ_2 regularized ℓ_2 regression, i.e., ridge regression. These differences are often "explained" in terms of differences between the unit ball in the ℓ_1 norm versus the unit ball in the ℓ_2 norm, with the former being "pointy" and the latter being "smooth." In particular, note that in those situations ℓ_1 often has some sort of connection with sparsity and sparse solutions.)

Here is a comparison between ℓ_1 and ℓ_2 with respect to the spectral/flow algorithms for the graph partitioning problem we have been considering.

- ℓ_2 norm:
  - Good for dimension reduction.
  - Efficient polynomial time algorithm to compute embedding of any finite metric space.
  - Connections to low-dimensional manifolds, diffusions, etc.
- ℓ_1 norm:
  - No good for dimension reduction.
  - NP-hard to compute the optimal embedding.
  - Connections to partitions/cuts, multicommodity flow, etc.

The following is a fundamental result in the area that is also central to understanding why flow-based graph partitioning algorithms work.

Theorem 1 (Bourgain)**.** *Every n-point metric space d admits an α-distortion embedding into ℓ_p, $\forall p$, with $\alpha = O(\log n)$.*

Proof idea. The proof is similar to but more general than the proof for the corresponding embedding claim for ℓ_2. The idea is: to use the so-called Frechet embedding method, where the embedding is given by the distance from points to carefully randomly chosen subsets of nodes. □

Note that we saw the ℓ_2 version of this before. Note also that the original embedding had 2^n dimensions, but LLR proved that it can be done with $O(\log^2 n)$ dimensions.

11.2 Connection between ℓ_1 metrics and cut metrics

First, recall what a metric is.

Definition 2. *A* metric *is a function $d : V \times V \to \mathbb{R}$ s.t.: (1) $d(x, y) = 0$ if $x = 0$ (and sometimes $= 0$ iff $x = y$); (2) $d(x, y) = d(y, x)$; and (3) $d(x, y) + d(x, z) \leq d(x, y)$*

(Recall also that sometimes the word "metric" if one or more of those conditions is/are relaxed.)

Next, recall the definition of the "cut metric," and recall that this is really not a metric but is instead just a semi-metric.

Definition 3. *Given $G = (V, E)$ and a set $S \subset V$, δ_S is the "cut metric" for S if*

$$\delta_S(i, j) = |\chi_S(i) - \chi_S(j)|,$$

where

$$\chi_S(i) = \begin{cases} 0 \text{ if } i \in S \\ 1 \text{ otherwise.} \end{cases}$$

Thus

$$\delta_S(i, j) = \begin{cases} 0 \text{ if } i, j \in S, \text{ or } i, j \in \bar{S} \\ 1 \text{ otherwise.} \end{cases}$$

(That is, if $\delta_s(x, y)$ is the indicator of x and y being on different sides of S.)

There are important connection between ℓ_1 metrics and Cut metrics. In particular:

- There exists a representation of the ℓ_1 metrics as a conical combination of cut metrics.
- Cut metrics are the extreme rays of the ℓ_1 cone.
- For these reasons, instead of minimizing the ratio of linear functions over the convex cone, we can minimize the ratio over the extreme rays of the convex cone. Minimum ratio function over cone $\iff$ minimum over extreme rays.

We'll spend most of today going over these results.

Fact: An n-point metric space can be associated with a vector in $\mathbb{R}^{\binom{n}{2}}$, with each coordinate corresponding to a pair of vertices.

Fact: Given a metric d, we will refer to the corresponding vector as $\bar{d}$. Then, $\alpha\bar{d} + (1 - \alpha)\bar{d'}$ is a metric, $\forall \alpha \in [0, 1]$. In addition, $\forall \alpha \geq 0$, $\forall \bar{d}$, $\alpha\bar{d}$ is a metric. So, the set of all metrics forms a convex cone in $\mathbb{R}^{\binom{n}{2}}$. In somewhat more detail, we have the following result:

Claim 1. *The set of ℓ_1 metrics is a convex cone,* i.e.*, if d_1 and $d_2 \in \ell_1$ metrics, and if $\lambda_1, \lambda_2 \in \mathbb{R}^+$, then $\lambda_1 d_1 + \lambda_2 d_2 \in \ell_1$ metrics.*

Proof. Recall that the line metric is an ℓ_1 metric. Let $d^{(i)}(x, y) = |x_i - y_i|$, for $x, y \in \mathbb{R}^c$. If $d \in \ell_1$ metric, then it is the sum of line metrics. □

Fact. The analogous result does *not* hold for ℓ_2.

Next, we have the following theorem:

Theorem 2. *Let d be a finite ℓ_1 metric. Then, d can be written as*

$$d = \sum_{S \subset [n]} \alpha_S \delta_S,$$

for some constant α_S and cut metrics δ_S. That is,

$$\begin{aligned} CUT_n &= \{d : d = \sum_{S \subseteq V} \alpha_S \delta_S, \alpha \geq 0\} \\ &= \textit{Positive cone generated by cut metrics} \\ &= \textit{All } n\textit{-point subsets of } \mathbb{R}^n\textit{, under the } \ell_1 \textit{ norm.} \end{aligned}$$

Proof. Consider any metric $d \in CUT_n$. Then, $\forall S$ with $\alpha_S > 0$, we have a dimension, and in that dimension, we can put

$$\begin{cases} \alpha_S & \text{if } x \in \bar{S} \\ 0 & \text{if } x \in S. \end{cases}$$

So, $CUT_n \subseteq \ell_1$ Metrics.

For the other direction, consider a set of n points from $\mathbb{R}^n$. Take one dimension d and sort the points in increasing values along that dimension. Say that we get $v_1, \ldots, v_k$ as the set of distinct values; then define $k-1$ cut metrics: $S_i = \{x : x_d < v_{i+1}\}$, and let $\alpha_i = v_{i+1} - v_i$, *i.e.*, $k-1$ coeff. So, along this dimension, we have that

$$|x_d - y_d| = \sum_{i=1}^{k} \alpha_i \delta_{S_i},$$

But, one can construct cut metrics for every dimension. So, we have cut metrics in CUT_n, $\forall$ n-point metrics ℓ_1; thus, $\ell_1 \subseteq CUT$. □

11.3 Relating this to a graph partitioning objective

Why is this result above useful? The usefulness of this characterization is that we are going to want to to optimize functions, and rather than optimize functions over all cut metrics, *i.e.*, over the extreme rays, we will optimize over the full convex cone, *i.e.*, over ℓ_1 metrics.

This leads us to the following lemma:

Lemma 1. *Let $C \subset \mathbb{R}^n$ be a convex cone, and let $f, g : \mathbb{R}^{n,+} \to \mathbb{R}^+$ be linear functions. And assume that $\min_{x \in C} \frac{f(x)}{g(x)}$ exists. Then*

$$\min_{x \in C} \frac{f(x)}{g(x)} = \min_{x \text{ in extreme rays of } C} \frac{f(x)}{g(x)}.$$

Proof. Let x_0 be the optimum. Since $x_0 \in C$, we have that $x_0 = \sum_i a_i y_i$, where $a_i \in \mathbb{R}^+$ and $y_i \in$ extreme rays of C. Thus,

$$\begin{aligned} \frac{f(x_0)}{g(x_0)} = \frac{f(\sum_i a_i y_i)}{g(\sum_i a_i y_i)} &= \frac{\sum_i f(a_i y_i)}{\sum_i g(a_i y_i)} \\ &\geq \frac{f(a_j y_j)}{g(a_j y_j)} \qquad \text{where } j \text{ is the min value} \\ &= \frac{f(y_i)}{g(y_i)}, \end{aligned}$$

where the first and third line follow by linearity, and where the second line follows since

$$\frac{\sum_i \alpha_i}{\sum_i \beta_i} \geq \min_j \frac{\alpha_j}{\beta_j}$$

in general. □

To see the connections of all of this to sparsest cut problem, recall that given a graph $G = (V, E)$ we define the conductance h_G and sparsity ϕ_G as follows:

$$h_G := \min_{S \subseteq V} \frac{E(S, \bar{S})}{\min\{|S|, |\bar{S}|\}}$$
$$\phi_G := \min_{S \subseteq V} \frac{E(S, \bar{S})}{\frac{1}{n}|S||\bar{S}|},$$

and also that:

$$h_G \leq \phi_G \leq 2h_G.$$

(This normalization might be different than what we had a few classes ago.)

Given this, we can write sparsest cut as the following optimization problem:

Lemma 2. *Solving*

$$\phi_G = \min_{S \subseteq V} \frac{E(S, \bar{S})}{\frac{1}{n}|S||\bar{S}|}$$

is equivalent to solving:

$$\begin{aligned} \min & \sum_{(ij) \in E} d_{ij} \\ s.t. & \sum_{ij \in V} d_{ij} = 1 \\ & d \in \ell_1 \, metric \end{aligned}$$

Proof. Let $\delta_S =$ the cut metric for S. Then,

$$\frac{|E(S, \bar{S})|}{|S| \cdot |\bar{S}|} = \frac{\sum_{ij \in E} \delta_S(i, j)}{\sum_{\forall ij} \delta_S(i, j)}$$

So,

$$\phi_G = \min_S \frac{\sum_{ij \in E} \delta_S(i, j)}{\sum_{\forall ij} \delta_S(i, j)}$$

Since ℓ_1-metrics are linear combinations of cut metrics, and cut metrics are extreme rays of ℓ_1 from the above lemma, this ratio is minimized at one of the extreme rays of the cone. So,

$$\phi_G = \min_{d \in \ell_1} \frac{\sum_{ij \in E} d_S(ij)}{\sum_{\forall ij} d_S(ij)}.$$

Since this is invariant to scaling, WLOG we can assume $\sum_{\forall ij} d_{ij} = 1$; and we get the lemma. □

11.4 Turning this into an algorithm

It is important to note that the above formulation is still intractable—we have just changed the notation/characterization. But, the new notation/characterization suggests that we might be able to *relax* (optimize the same objective function over a larger set) the optimization problem—as we did with spectral, if you recall.

So, the relaxation we will consider is the following: relax the requirement that $d \in \ell_1$ Metric to $d \in$ Any Metric. We can do this by adding $3\binom{n}{3}$ triangle inequalities to get the following LP:

$$\begin{aligned} \lambda^* = \min & \sum_{ij \in E} d_{ij} \\ \text{s.t.} & \sum_{\forall ij \in V} d_{ij} = 1 \\ & d_{ij} \geq 0 \\ & d_{ij} = d_{ji} \\ & d_{ij} \leq d_{ik} + d_{jk} \quad \forall i, j, k \quad \text{triples} \end{aligned}$$

(Obviously, since there are a lot of constraints, a naive solution won't be good for big data, but we will see that we can be a bit smarter.) Clearly,

$$\lambda^* \leq \phi^* = \text{ Solution with } d \in \ell_1 \quad \text{Metric constraint}$$

(basically since we are minimizing over a larger set). So, our goal is to show that we don't loose too much, *i.e.*, that:

$$\phi^* \leq O(\log n)\lambda^*.$$

Here is the ALGORITHM. Given as input a graph G, do the following:

- Solve the above LP to get a metric/distance $d : V \times V \to \mathbb{R}^+$.
- Use the (constructive) Bourgain embedding result to embed d into an ℓ_1 metric (with, of course an associated $O(\log n)$ distortion).
- Round the ℓ_1 metric (the solution) to get a cut.
  - For each dimension/direction, covert the ℓ_1 embedding/metric along that to a cut metric representation.
  - Choose the best.

Of course, this is what is going on under the hood—if you were actually going to do it on systems of any size you would use something more specialized, like specialized flow or push-relabel code.

Here are several things to note.

- If we have ℓ_1 embedding with distortion factor ξ then can approximate the cut up to ξ.
- Everything above is polynomial time, as we will show in the next theorem.
- In practice, we can solve this with specialized code to solve the dual of corresponding multi-commodity flow.
- Recall that one can "localize" spectral by running random walks from a seed node. Flow is hard to localize, but recall the IMPROVE algorithm, but which is still $\tilde{O}(n^{3/2})$.
- We can combine spectral and flow, as we will discuss, in various ways.

Theorem 3. *The algorithm above is a polynomial time algorithm to provide an $O(\log n)$ approximation to the sparsest cut problem.*

Proof. First, note that solving the LP is a polynomial time computation to get a metric d^*. Then, note that the Bourgain embedding lemma is constructive. Finding an embedding of d^* to $d \in \ell_1^{O(\log^2 n)}$ with distortion $O(\log n)$. So, we can write d as a linear combination of $O(n \log^2 n)$ cut metrics $d = \sum_{S \in \mathcal{S}} \alpha_S \delta_S$, where $|\mathcal{S}| = O(n \log^2 n)$. Note:

$$\begin{aligned} \min_{S \in \mathcal{S}} \frac{\sum_{ij \in E} \delta_S(ij)}{\sum_{\forall ij} \delta_S(ij)} &\leq \frac{\sum_{ij \in E} d_{ij}}{\sum_{\forall ij} d_{ij}} \\ &\leq O(\log n) \frac{\sum_{ij \in E} d^*_{ij}}{\sum_{\forall ij} d^*_{ij}}, \end{aligned}$$

where the first inequality follows since d is in the cone of cut metrics, and where the second inequality follows from Bourgain's theorem. But,

$$\frac{\sum_{ij \in E} d^*_{ij}}{\sum_{\forall ij} d^*_{ij}} = \min_{d' \text{ is metric}} \frac{\sum_{ij \in E} d'_{ij}}{\sum_{\forall ij} d'_{ij}} \leq \min_{\forall S} \frac{\sum_{ij \in E} d_S(ij)}{\sum_{\forall ij} d_S(ij)},$$

where the equality follows from the LP solution and the inequality follows since LP is a relaxation of a cut metric. Thus,

$$\min_{S \in \mathcal{S}} \frac{\sum_{ij \in E} \delta_S(ij)}{\sum_{\forall ij} \delta_S(ij)} \leq O(\log n) \min_{\forall S} \frac{\sum_{ij \in E} d_S(ij)}{\sum_{\forall ij} d_S(ij)}.$$

This establishes the theorem. □

So, we can also approximate the value of the objective—how do we actually find a cut from this? (Note that sometimes in the theory of approximation algorithms you *don't* get anything more than an approximation to the optimal number, but that is somewhat dissatisfying if you want you use the output of the approximation algorithm for some downstream data application.)

To see this:

- Any ℓ_1 metric can be written as a conic combination of cut metrics—in our case, with $O(n \log^n)$ nonzeros—$d^\sigma = \sum_S \alpha_S \delta_S$.
- So, pick the best cut from among the ones with nonzero α in the cut decomposition of d^σ.

11.5 Summary of where we are

Above we showed that

$$\begin{aligned} \phi_G &= \min_{S \subset V} \frac{E(S, \bar{S})}{|S||\bar{S}|} \\ &= \min \sum_{ij \in E} d_{ij} \\ &\quad \text{s.t.} \sum_{ij \in V} d_{ij} = 1 \\ &\quad d \in \ell_1 \text{ metric} \end{aligned}$$

can be approximated by relaxing it to

$$\begin{aligned} \min & \sum_{ij \in E} d_{ij} \\ \text{s.t.} & \sum_{ij \in V} d_{ij} = 1 \\ & d \in \text{Metric} \end{aligned}$$

This relaxation is different than the relaxation associated with spectral, where we showed that

$$\phi = \min_{x \in \{0,1\}^V} \frac{A_{ij}|x_i - x_j|^2}{\frac{1}{n}\sum_{ij}|x_i - x_j|^2}$$

can be relaxed to

$$d - \lambda_2 = \min_{x \perp \vec{1}} \frac{A_{ij}(x_i - x_j)^2}{\frac{1}{n}\sum_{ij}(x_i - x_j)^2}$$

which can be solved with the second eigenvector of the Laplacian.

Note that these two relaxations are very different and incomparable, in the sense that one is not uniformly better than the other. This is related to them succeeding and failing in different places, and it is related to them parametrizing problems differently, and it can be used to diagnose the properties of how each class of algorithms performs on real data. Later, we will show how to generalize this previous flow-based result and combine it with spectral.

Here are several questions that the above discussion raises.

- What else can you relax to?
- In particular, can we relax to something else and improve the $O(\log n)$ factor?
- Can we combine these two incomparable ideas to get better bounds in worst-case and/or in practice?
- Can we combine these ideas to develop algorithms that smooth or regularize better in applications for different classes of graphs?
- Can we use these ideas to do better learning, e.g., semi-supervised learning on graphs?

We will address some of these questions later in the term, as there is a lot of interest in these and related questions.