CS270 Algorithms & Data Structures **Spring 2008**

Lecture 3 Expanders, eigenvalues. : 1.30.08

Lecturer: Satish Rao *Scribe: Rao, for now*

Disclaimer: *These are rough notes, with some exercises.*

The notes have been remembered or inspired by or taken from various sources. Apologies to those who were plagiarized and/or not cited.

We are covering eigenvalues, and their connection to finding good cuts in graphs or proving there are none. The material uses pieces of S. Arora's notes (...), and previous very rough notes (like these) notes of mine (www.cs.berkeley.edu/ satishr/cs273 on eigenvalues). It underlies some of the techniques in the currently popular "Normalized Cut" methods of Shi and Malik (search the internet for info.)

3.1 Expander graphs.

Question: What is a graph without any good cuts?

A complete graph.

Question: How about if the graph has bounded degree?

An *degree* (c, α)*-expander graph* G is defined as a graph where for any set S, where S contains fewer than half the nodes of G there are at least $\alpha|S|$ edges leaving S. That is the cut, $(S, |S|)$ has at least $\alpha|S|$ edges in it.

Question: What might be an application of an expander graph?

Pseudorandomnness!

Question: Oh my goodness. Ok, we can try that. What is an example of a use of randomness?

Primality testing using Fermat's little theorem. For prime numbers N, for all $a \in \{1, \ldots, N-1\}$, $a^{N-1} = 1 \bmod N$, whereas (mostly, i.e., other than Carmichael numbers), if a number N is composite, for at least half the numbers we have $a^{N-1} \neq 1 \bmod N$. Thus, choosing say k random numbers, will fail to catch a composite number with probability at most $1/2^k$.

This uses $k \log N$ bits. We will later show that for an expander graph of degree d, that $k + O(d \log d \log N)$ bits is sufficient. Specifically, we choose a random node (corresponding to some a) in an $N-1$ node expander and do a random walk of length $O(d \log N)$ where each step can be specified by $\log d$ bits. Now, we test an a corresponding to each node that we see in this graph.

We need an explicit notion of expander graph, where one can compute a neighbor of a node, given the name of the node. Search for explicit construction of expanders for more info.

Question: Today. We will give some intuition about one method for proving a graph is an expander. OK?

Sure.

3.2 "Drawing" a graph. An embedding to the line.

Question: How should we draw a graph in the plane?

Perhaps, try to make edges short, but spread out nodes.

Question: Perhaps attempt to minimize the ratio. What ratio?

The ratio of the edge lengths squared to the typical distance squared...or ...

$$\frac{\sum_{e=(u,v)} |x_u - x_v|^2}{\sum_{i,j} |x_i - x_j|^2}. \tag{3.1}$$

The minimum value of this ratio has been called the Raleigh quotient of a graph, which we define as $\rho(G)$.

Question: Dump vertices on the plane, at random in a unit interval. Are they spread out? What is the typical length of an Edge?

Yes, they are spread out, typical distance is $\theta(1)$. It is $\Omega(1)$. What is the ratio $\Theta(1)$.

Question: Are there better drawings? What about a grid?

Well, can make typical edge be $O(1/\sqrt{n})$ length.

Question: So, can we do better than random? At least improve a current embedding nodes to shorten edges. Make a suggestion?

Move each node to center of mass of neighbors. As short as possible given that no one else moves. That is, moving a node to center of mass minimizes the sum of its adjacent edge lengths squared.

Question: Why? Can each dimension be considered separately?

Exercise 1: Prove that one can just consider one dimensional vectors here. We did it in class.

Question: Is this locally a good thing?

Yes, the dimensions separate, so lets just consider one dimension. And for one dimension, $\sum_i (x_i - c)^2$ is minimized when $c = \sum_i x_i/k$. Use calculus to verify this.

Question: Collapses the nodes! How do we fix this?

Rescale so that average distance is still constant. That is, so that $\sum_{i,j} |x_i - x_j|^2 \geq cn^2$.

Question: Can you represent (most of) a step of this algorithm using a matrix?

Sure, use an adjacency matrix for the graph, where each row is divided by d. That way, a node's new value is the average of its neighbors. You can rescale the values, afterward.

You could also modify the matrix, so that the diagonals are not 0, say 1/2, and the off diagonals are $1/2d$ where d is the degree of the node. This has the effect of moving each node toward the center of mass, but not jumping so quickly.

Question: Does this algorithm, produce a one-dimensional embedding that minimizes the ratio in equation 3.1?

For now, it is a gradient descent algorithm for the numerator while ensuring that the denominator is kept constant. It does actually minimize it, but more about that later.

Question: What matrix quantity might this have to do with?

Eigenvectors. For example, an eigenvector (up to scaling) the matrix is a fixed point of a matrix. That is, in the above setting each node would already be at its scaled center of mass. Don't worry for now, we will come back to this.

Question: Can you come up with another analogy for this process?

We can think of it is the evolution of a random walk. That is, given a distribution on the graph (e.g., probability 1 of being at u and 0 everywhere else) one can think of a matrix where the effect of one step of a random walk on the probability distribution. Indeed, we will see that $1/(n\rho(p))$ is essentially the number of steps one needs to get "close" to the uniform distribution on the edges. For example, for the line graph it takes $\Theta(n^2) = 1/n(n^3)$ steps to get approximately uniform.

Exercise 2: Show that the distribution of a random walk (with stationary probability of $1/2$) on an undirected graph converges to a distribution where the probability of each node is appearing at step n is proportional to its degree. Well, at least show that it is a fixed point of the walk?

Question: As an aside, how might you embed into two dimensions?

Perhaps, make second "orthogonal" to first. Something to do, perhaps, with eigenvalues.

Question: A "dumbbell" graph is two complete graphs connected by an edge. What is a good Raleigh embedding of the graph?

All the points on one side mapped to 0, the others mapped to 1. Rayleigh quotient is around $1/n^2$.

Question: How about for a path of n nodes?

Well, exactly as a path might be good (though, it is not quite optimal.) Rayleigh quotient is around $1/n^3$.

Question: How about for a $k \times k$ grid?

Map each column to a point on the line. Equally spaced might be good. Rayleigh quotient is around $1/n^{3/2}$.

Question: How about a binary tree?

Map one subtree to 0 and the other to 1. The Raleigh quotient is around $1/n^2$.

Question: What is the worst Raleigh quotient for any degree d graph?

The random embedding gives a quotient of around d/n where d is the degree bound.

3.3 The embedding and graph cuts.

Question: Recall the definition of separator?

Sure, the minimum number of edges whose removal leaves connected components of maximum size 2/3.

Question: Why is it interesting?

Recursive algorithms.

Question: Today, we will talk about edge cuts, and an amortized notion of quality. For

example, how about minimizing the ratio of the number of edges used to the number of nodes of the graph that we cut off. Why?

Cost/benefit notion, i.e., cut off largest section of soldiers for cheapest. Find bottlenecks in networks. Recursive algorithms.

Question: What are some possibilities?

Find a cut $(S, \overline{S})$ that minimizes *edge expansion*

$$\frac{c(S, \overline{S})}{\min |S|, |overlineS|},$$

or minimizes *Cheeger value*

$$\frac{c(S, \overline{S})}{\min vol(S), vol(\overline{S})},$$

where $vol(S)$ is the sum of the degrees of nodes in S. Or we could minimize the *sparsity*

$$\frac{c(S, \overline{S})}{|S||overlineS|},$$

or even the *normalized cut value*

$$\frac{c(S, \overline{S})}{vol(S)vol(overlineS)},$$

(though it is defined differently, but equivalently up to a scale factor, by its authors.) We remark that these definitions can also be modified appropriately for node or edge weights.

Exercise 3: Show that any cut which finds the minimum value for edge expansion α has S that is within a factor of two of the minimum possible S.

Question: Let's make it easy. How?

Use bounded degree graphs. Makes cheeger same as edge expansion (up to scaling.) Makes S same as normalized cut.

Let's choose

$$\frac{c(S, \overline{S})}{|S||overlineS|},$$

Question: What is the S of our example graphs above?

The dumbbell has S $1/n^2$, as does the binary tree, as does the line graph. The S of the grid is $1/n^{3/2}$.

Question: It is NP-complete to compute the S (or any of the above) for a graph. What to do?

Question: Is there a relationship between S and the Rayleigh quotent? What if we restrict the x_i to be $0, 1$ in the definition of Raleigh quotient?

The $0, 1$ values of the vertices define a cut which has S equal to the value of the Rayleigh quotient! That is,

$$\rho(G) \leq \text{ S}(G).$$

Question: For what graphs is this tight?

The complete graph looks tight. The binary tree is good. For the grid and the line graph these are far apart. Indeed, off by a factor of $\sqrt{n}$ for the grid, and a factor of n for the line.

Question: How about expander graphs? Can we use the Raleigh quotient to prove that a graph is an expander?

Well...leads to the following question.

Question: Can we get any upper bound on S from the Rayleigh quotient?

Well, next time, we will prove the following...

$$\frac{(n\mathrm{S}(G))^2}{d} \leq \rho(G) \leq \ \mathrm{S}(G). \tag{3.2}$$

Question: For our example graphs; binary trees, dumbbell, grids, and lines, which inequality is tight (within constant factors)?

For binary tree, dumbbell, the right inequality is tight. Whereas for grids and dumbbell graphs the left inequality is tight.