3/13 CS485 Notes (Jae-Ho Lee – JL592)

Review:

Eigenvalues of specific graphs. Connected, regular, degree d with an odd cycle.

$$d = \lambda_1 > \lambda_2 > \ldots > -d$$

If G has no odd cycle, then $\lambda_n = -d$

Lemma
If G has exactly k components then

$$d = \lambda_1 = \lambda_2 = \ldots = \lambda_k > \lambda_{k+1} \ldots \lambda_n = -d$$

Proof
Adjacency matrix is block diagonal with exactly k blocks.

Eigenvalues of big matrix are union of eigenvalues of blocks

If $B1x = \lambda x$

$$\begin{pmatrix} B1 & 0 & 0 & \cdots & 0 \\ 0 & B2 & & & \\ 0 & & . & & \\ \vdots & & & . & \\ 0 & & & & Bk \end{pmatrix} \begin{pmatrix} x \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} x1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

D appears exactly k times as eigenvalues

All others less than d. (Proof only holds for regular d degree graph)

Corollary : A regular degree d graph is connected iff $\lambda_1 > \lambda_2$

Relax regular degree d condicion, connectivity, etc

Arbitrary graphs.

Adding edges to a graph does not decrease max eigenvalue.

Lemma
The eigenvector associated with λ_1 has all non-negative components.

Proof
Let A_1 be adjacency matrix of G_1 and $\lambda_1(A_1)$ be the max eigenvalue.

$$\lambda_1(A_1) = \gamma^T A_1 \gamma \leq \gamma^T A_2 \gamma$$

But

$$\begin{aligned} \lambda_1(A_2) = \max_{|x|=1} x^T A x &\geq \gamma^T A_2 \gamma \\ &\geq \gamma^T A_1 \gamma = \lambda_1(A_1) \end{aligned}$$

$$\therefore \lambda_1(A_2) \geq \lambda_1(A_1)$$

Note

Definition

$$\lambda_1 = \max_{|x|=1} |Ax|_2$$

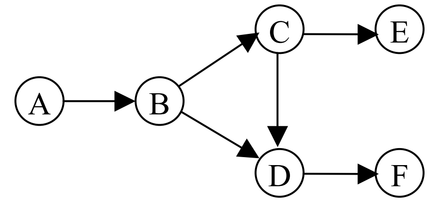
Theorem

$$\lambda_1 = \max_x \frac{x^T A x}{x^T x}$$

$$= \max_{|x=1|} x^T A x$$

CS 485 Lecture 22 – 13 March 2006 – second half
Jeff Wong (jmw92)

Random Walk on a Directed Graph



- In the case of a connected undirected graph, iterating the probability vector by multiplying $\mathbf{p} \rightarrow \mathbf{A}\mathbf{D}^{-1}\mathbf{p}$, we get a steady-state probability proportional to the degree of the graph.
- For a directed graph, we can get the iterated probability by multiplying $\mathbf{p} \rightarrow \mathbf{A}\mathbf{D}_R^{-1}\mathbf{p}$, where $\mathbf{D}_R^{-1}$ is a diagonal matrix where each element is the inverse of the sum of the rows of **A**, equivalent to the outgoing degree of each node.
- There exists a problem: what happens at dead-end nodes, like E? The probability that we were at E just "disappears," and also, multiplying by $\mathbf{D}_R^{-1}$ causes us to multiply by 1/0 for a dead end.
- One solution is to add a self-loop to every node, so that there are no dead ends. But this results in a probability that saturates at dead end nodes and is dependent on where you start.
- Google came up with a solution when looking at the web as a directed graph: at each step, with probability ε, reset your location and go to a random node.
  - This is equivalent to adding an edge from each node to each other node; at each step we take one of these new edges with probability ε/n. But this adds n^2 new edges, which becomes unwieldy.
  - A better way: add a new node, with incoming edges from every node and outgoing edges to every node. Then, from each node, go to the new node with probability ε; then, at the new node, go to all other nodes with uniform probability.

How long do you have to iterate this process until you get a steady state? It depends on the graph, but it turns out that for a random graph, you approach the steady state exponentially fast.

Random Algorithms are algorithms that use the outcome of random variables (e.g. coin flips)

Example: Is n prime?

If n is prime, then $a^{n-1} = 1 \bmod n$ for all $1 < a \leq n-1$

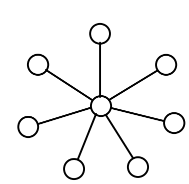
If n is not prime, then $a^{n-1} \neq 1 \bmod n$ for at least half of all $1 < a \leq n-1$

The algorithm: choose 100 random a's and calculate $a^{n-1} \bmod n$. If we get any results not equal to 1, then we know that n is composite. If we get all 1's, then we know that the probability of n being composite is less than $1/2^{100}$.

But we need 100 random numbers (100 log n bits) to do this. We would like to reduce this number.

Spectrum of a Star

The spectrum of a graph is the set of eigenvalues.
A star has an adjacency matrix



$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & & & & \\ 1 & & 0 & & \\ \vdots & & & & \\ 1 & & & & \end{pmatrix} \qquad \text{This has rank=2.}$$

We can see the following easily:

$$\begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & & & & \\ 1 & & 0 & & \\ \vdots & & & & \\ 1 & & & & \end{pmatrix} \begin{pmatrix} \sqrt{n-1} \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \begin{pmatrix} n-1 \\ \sqrt{n-1} \\ \sqrt{n-1} \\ \vdots \\ \sqrt{n-1} \end{pmatrix} \text{ so one eigenvalue is } \sqrt{n-1}$$

$$\begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & & & & \\ 1 & & 0 & & \\ \vdots & & & & \\ 1 & & & & \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ \vdots \\ -(n-1) \\ 1 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \text{ so (n-2) eigenvalues are 0}$$

$$\begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & & & & \\ 1 & & 0 & & \\ \vdots & & & & \\ 1 & & & & \end{pmatrix} \begin{pmatrix} -\sqrt{n-1} \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \begin{pmatrix} n-1 \\ -\sqrt{n-1} \\ -\sqrt{n-1} \\ \vdots \\ -\sqrt{n-1} \end{pmatrix} \text{ so one eigenvalue is } -\sqrt{n-1}$$