Link-based ranking
Part 2

Goal

• **Intuition:** when Web page points to another Web page, it confers status/authority/popularity to that page
• Find a measure that captures intuition

• Not just web linking
  – Citations in books, articles
  – Doctors referring to other doctors
Review: first measure PageRank

- Given a directed graph with \( n \) nodes
- Assign each node a score that represents its importance in structure
  - Call score PageRank: \( \text{pr}(\text{node}) \)

Conferring importance

Core ideas:
- A node should confer some of its importance to the nodes to which it points
  - If a node is important, the nodes it links to should be important
- A node should not transfer more importance than it has
- Address problems with:
  - Sinks (nodes with no edges out)
  - Cyclic behavior
Random walk model (review)

1. Move from node to linked neighbor with probability 1/outdegree
   Outdegree of a node = number of edges out of a node
2. Randomly jump to any node
   - Break cycles
   - Escape from sinks

Captured with:
\[ pr_{\text{new}}(k) = \frac{\alpha}{n} + (1-\alpha) \sum_{i \text{ with edge from } i \text{ to } k} \left( \frac{pr(i)}{t_i} \right) \]

- \( \alpha \) parameter chosen empirically
- \( t_i \) outdegree of node \( i \)

Steady state probability of being at a node = \( pr(\text{node}) \)

Normalized?

- Would like \( \sum_{1 \leq k \leq n} (pr(k)) = 1 \)
- Consider \( \sum_{1 \leq k \leq n} (pr_{\text{new}}(k)) \)
  \[ = \sum_{1 \leq k \leq n} \left( \frac{\alpha}{n} + (1-\alpha) \sum_{i \text{ with edge from } i \text{ to } k} \left( \frac{pr(i)}{t_i} \right) \right) \]
  \[ = \alpha + (1-\alpha) \sum_{1 \leq k \leq n} \sum_{i \text{ with edge from } i \text{ to } k} \left( \frac{pr(i)}{t_i} \right) \]
  \[ = \alpha + (1-\alpha) \sum_{1 \leq i \leq n} \sum_{k \text{ with edge from } i \text{ to } k} \left( \frac{pr(i)}{t_i} \right) \]

*inner sum \( \sum_{i} \) over incoming edges for one \( k \)
*inner sum \( \sum_{k} \) over outgoing edges for one \( i \)
Problem for desired normalization

• Have
  \[ \sum_{1 \leq k \leq n} (pr_{\text{new}}(k)) = \alpha + (1-\alpha) \sum_{i \text{ with edge from } i} pr(i) \]

• Missing \( pr(i) \) for nodes with no edges from them
  – sinks!

• Solution: add \( n \) edges out of every sink
  – Edge to every node including self
  – Gives \( 1/n \) contribution to every node

Gives desired normalization:
If \( \sum_{1 \leq k \leq n} (pr_{\text{initial}}(k)) = 1 \)
then \( \sum_{1 \leq k \leq n} (pr(k)) = 1 \)

Matrix formulation

• Let \( E \) be the \( n \) by \( n \) adjacency matrix
  \( E(i,k) = 1 \) if there is an edge from node \( i \) to node \( k \)
  \( = 0 \) otherwise

• Define new matrix \( L \):
  For each row \( i \) of \( E \) \((1 \leq i \leq n)\)
  If row \( i \) contains \( t_i \) > 0 ones, \( L(i,k) = (1/ t_i) E(i,k), 1 \leq k \leq n \)
  If row \( i \) contains 0 ones, \( L(i,k) = 1/n, 1 \leq k \leq n \)

• Vector \( pr \) of PageRank values defined by
  \[ pr = (\alpha/n, \alpha/n, \ldots, \alpha/n)^T + (1-\alpha) L^T pr \]

• has a solution representing the steady-state values \( pr(k) \)
Calculation

• Choose $\alpha$
  – No single best value
  – Page and Brin originally used $\alpha = .15$

• Simple iterative calculation
  ▪ Initialize $pr_{\text{initial}}(k) = 1/n$ for each node $k$
    • so $\sum_{1\leq k\leq n} (pr_{\text{initial}}(k)) = 1$
  ▪ $pr_{\text{new}}(k) = \alpha/n + (1-\alpha) \sum_{1\leq i\leq n} L(i,k)pr(i)$

• Converges
  – Has necessary mathematical properties
  – In practice, choose convergence criterion
    • Stops iteration

PageRank Observations

• PageRank can be calculated for any graph
• Google calculates on entire Web graph
• Huge calculation for Web graph
  – precomputed
  – 1998 Google:
    • 52 iterations for 322 million links
    • 45 iterations for 161 million links
• PageRank must be combined with query-based scoring for final ranking
  – Many variations
  – What Google exactly does secret
  – Can make some guesses by results
HITS
Hyperlink Induced Topic Search

- Second well-known algorithm
- By Jon Kleinberg while at IBM Almaden Research Center
- Same general goal as PageRank
- Distinguishes 2 kinds of nodes
  - Hubs: resource pages
    - Point to many authorities
  - Authorities: good information pages
    - Point to many hubs

Mutual reinforcement

- Authority weight node j: \( a(j) \)
  - Vector of weights \( a \)
- Hub weight node j: \( h(j) \)
  - Vector of weights \( h \)
- Update:
  \[
  a_{\text{new}}(k) = \sum_{i \text{ with edge from } i \text{ to } k} h(i)
  \]
  \[
  h_{\text{new}}(k) = \sum_{j \text{ with edge from } k \text{ to } j} a(j)
  \]
Matrix formulation

Steady state:
\[ a = E^T h \quad a = E^T E a \]
\[ h = E a \quad h = E E^T h \]

Interpretation:
- \( E^T E(i,j) \): number nodes point to both node i and node j
  - “Co-citation”
- \( EE^T(i,j) \): number nodes pointed to by both node i and node j
  - “Bibliographic coupling”

Iterative Calculation

\( a = h = (1, \ldots, 1)^T \)

While (not converged) \{ 
\[ a_{\text{new}} = E^T h \]
\[ h_{\text{new}} = E a \]
\[ a = a_{\text{new}} / \|a_{\text{new}}\| \quad \text{normalize to unit vector} \]
\[ h = h_{\text{new}} / \|h_{\text{new}}\| \quad \text{normalize to unit vector} \]
\}
Convergence

• Linear algebra - eigenvalues
• Kleinberg uses slightly different iteration and slightly different proof than in Intro to IR book
  – Normalization important
    \[ a_0 = h_0 = (1, ... , 1)^T \]
    For k\textsuperscript{th} iteration \{ \n    \[ a_k = \text{normalized} \ (E^T h_{k-1}) \]
    \[ h_k = \text{normalized} \ (E a_k) \quad \text{uses new value of } a \]
\} 
Then \[ a_k = \text{normalized} \ ( (E^T E)^{k-1} E^T a_0 ) \]
Then \[ h_k = \text{normalized} \ ( (EE^T)^k h_0 ) \]

General Theorem:
If \( M \) is a symmetric n by n matrix and \( v \) is a vector not orthogonal to the principal eigenvector \( w_1 \) of \( M \),
then the unit vector in the direction of \( M^k v \) converges to \( w_1 \) as \( k \) goes to infinity.

Application:
Since \( h_0 = (1, 1, ... 1)^T \), \( h_0 \) is not orthogonal to the principal eigenvector of \( EE^T \)
\[ \Rightarrow h_k \text{ converges} \]
\[ a_k \text{ similar but little more work because first vector } E^T a_0 \]
Use of HITS

- Actual use of HITS by IBM people was after find Web pages satisfying query:
  1. Retrieve documents satisfy query and rank by term-based techniques
  2. Keep top $c$ documents: root set of nodes
     - $c$ a chosen constant - tunable
  3. Make base set:
     1. Root set
     2. Plus nodes pointed to by nodes of root set
     3. Plus nodes pointing to nodes of root set
  4. Make base graph: base set plus edges from Web graph between these nodes
  5. Apply HITS to base graph

Results using HITS

- Documents ranked by authority score $a(doc)$ and hub score $h(doc)$
  1. Authority score primary score for search results
- Heuristics:
  - delete all links between pages in same domain
  - Keep only pre-determined number of pages linking into root set (~200)
- Findings (original paper)
  - Number iterations in original tests ~50
  - most authoritative pages do not contain initial query terms
     - Compare LSI "concepts"
Observations

• HITS can be applied to any graph
• Base graph much smaller than Web graph
• Kleinberg identified bad phenomena
  – Topic diffusion: generalizes topic when expand root graph to base graph
    • Want compilers - generalized to programming

HITS and clustering

• Non-principal eigenvectors of $EE^T$ and $E^TE$ have positive and negative component values
  – Denote $a_{e2}, a_{e3}, \ldots$ matching $h_{e2}, h_{e3}, \ldots$
• For a matched pair of eigenvectors $a_{ej}$ and $h_{ej}$
  – Denote $k^{th}$ component of $j^{th}$ eigenvector $a_{ej}(k)$ and $h_{ej}(k)$
  – Make a “community” of size $c$ (a chosen constant):
    • Choose $c$ pages with most positive $h_{ej}(k)$ - hubs
    • Choose $c$ pages with most positive $a_{ej}(k)$ - authorities
  – Make another “community” of size $c$:
    • Choose $c$ pages with most negative $h_{ej}(k)$ - hubs
    • Choose $c$ pages with most negative $a_{ej}(k)$ - authorities
• Compare LSI
Eigenvalues and clustering

General class of techniques for clustering a graph using eigenvectors of adjacency matrix (or similar matrix) called

Spectral clustering

First described in 1973

More later, maybe …