Social Networks and Ranking

Social Networks

• Represent relationship between entities
  – paper cites paper
  – html page links to html page
  – A supervises B
  – A and B are friends
  – papers share an author
  – A and B are co-workers

Hypertext

• document or part of document links to other parts or other documents
  – construct documents of interrelated pieces
  – relate documents to each other

• pre-dates Web
• Web “killer app.”

How use links to improve information search?

• use structure to compute score
• include more objects to score

• can deal with objects of mixed types
  – images, PDF, …

Scoring using structure

• Ideas
  1. link to object suggests it valuable object
  2. distance between objects in graph represents degree of relatedness reachable by all in 2 links

Pursuing linking and value

• 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
**Indegree**

- Indegree = number of links into a node
- Most obvious idea: higher indegree => better node
- Doesn’t work well
- Need some feedback in system
- Leads us to Page and Brin’s PageRank

**PageRank**

- Algorithm that gave Google the leap in quality
- Used link structure between pages in fundamental way to score pages
  - link structure centerpiece of scoring
- published
  

**PageRank framework**

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

**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

**Attempt 1**

Refer to nodes by numbers 1, ..., $n$ (arbitrary numbering)

Let $t_i$ denote the number of edges out of node $i$ (outdegree)

Node $i$ transfers $1/t_i$ of its importance on each edge out of it

Define

$$pr_{new}(k) = \sum_{i \text{ with edge from } i \text{ to } k} \frac{pr(i)}{t_i}$$

Iterate until converges

Problems

- Sinks (nodes with no edges out)
- Cyclic behavior

**Attempt 2**

Random walk model

- Attempt 1 gives movement from node to linked neighbor with probability $1/$outdegree
- Add random jump to any node

$$pr_{new}(k) = \frac{\alpha}{n} + (1-\alpha) \sum_{i \text{ with edge from } i \text{ to } k} \frac{pr(i)}{t_i}$$

- $\alpha$ parameter chosen empirically

- Break cycles
- Escape from sinks
Normalized?

- Would like \( \sum_{i=1}^{k} (pr(k)) = 1 \)
- Consider \( \sum_{i=1}^{k} (pr_{\text{new}}(k)) \)
  \[ = \sum_{i=1}^{k} \left( \frac{\alpha}{n} + (1-\alpha) \sum_{i=1}^{k} (pr(i) / t_i) \right) \]
  \[ = \frac{\alpha}{n} \sum_{i=1}^{k} (1-\alpha) \sum_{i=1}^{k} (pr(i) / t_i) \]
  \[ = \frac{\alpha}{n} + (1-\alpha) \sum_{i=1}^{k} (pr(i) / t_i) \]

Problem for desired normalization

- Have \( \sum_{i=1}^{k} (pr_{\text{new}}(k)) = \alpha + (1-\alpha) \sum_{i=1}^{k} (pr(i) / t_i) \)

- Missing \( pr(i) \) for nodes with no edges from them
  - sink
- 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 \( \sum_{i=1}^{k} (pr_{\text{initial}}(k)) = 1 \) then \( \sum_{i=1}^{k} (pr(k)) = 1 \)

Matrix formulation

- Let \( E \) be the \( n \) by \( n \) adjacency matrix
  \[ E(i,k) = 1 \text{ if there is an edge from node } i \text{ to node } k \]
  \[ = 0 \text{ otherwise} \]
- 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
  \[ pr = \left( \frac{\alpha}{n}, \frac{\alpha}{n}, \ldots, \frac{\alpha}{n} \right) \text{ and } \frac{1}{1-\alpha} L^T pr \]
  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 \( k \)
  \[ \sum_{i=1}^{k} (pr_{\text{initial}}(k)) = 1 \]
  \[ pr_{\text{new}}(k) = \frac{\alpha}{n} + (1-\alpha) \sum_{i=1}^{k} L(i,k) pr(i) \]
- Converges
  - Has necessary mathematical properties
  - In practice, choose convergence criterion
  - Stops iteration

PageRank Observations

- Can be calculated for any directed graph
- Google calculates on entire Web graph
  \- query independent scoring
- Huge calculation for Web graph
  \- precomputed
  \- 1998 Google published:
    \- 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:

\[
\begin{align*}
a_{\text{new}}(k) &= \sum \text{i with edge from i to k} \ (h(i)) \\
h_{\text{new}}(k) &= \sum \text{j with edge from k to j} \ (a(j))
\end{align*}
\]

Matrix formulation

Steady state:

\[
\begin{align*}
a &= E^T h \\
h &= Ea
\end{align*}
\]

Interpretation:

- $E^T h(i)$: number nodes point to both node $i$ and node $j$
  - “Co-citation”
- $E a(j)$: number nodes pointed to by both node $i$ and node $j$
  - “Bibliographic coupling”

Iterative Calculation

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

While (not converged) {

\[
\begin{align*}
a_{\text{new}} &= E^T h \\
h_{\text{new}} &= E a \\
a &= a_{\text{new}} / ||a_{\text{new}}|| \quad \text{normalize to unit vector} \\
h &= h_{\text{new}} / ||h_{\text{new}}|| \quad \text{normalize to unit vector}
\end{align*}
\]

Provable convergence by linear algebra

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
    - $c$ 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 $a(\text{doc})$ and hub score $h(\text{doc})$
  - 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 directed 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
PageRank and HITS

• designed independently around 1997
• indicates time was ripe for this kind of analysis
• lots of embellishments by others