Social Networks and Ranking

1

Generalized Social Networks

- Represent relationship between entities
  - paper cites paper
  - html page links to html page
  - A supervises B

  } directed graph

  - A and B are friends
  - papers share an author
  - A and B are co-workers

  } undirected graph

2

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."

3

How use links to improve information search?

4

How use links to improve information search?

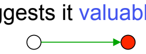
- use structure to compute score for ranking
- include more objects to rank
  - redefines "satisfying" of query?
- add to the content of a document

✧can deal with objects of mixed types
  - images, PDF, …

5

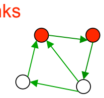
Scoring using structure

- Ideas
  1. link to object suggests it valuable object



  2. distance between objects in graph represents degree of relatedness

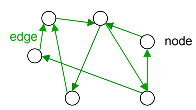
reachable by all in 2 links



6

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

7

Indegree

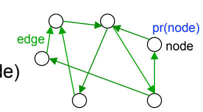
- Indegree = number of links into a node
- Most obvious idea:

  higher indegree => better node
- Doesn't work well
- Need some feedback in system
- Leads us to Page and Brin's PageRank

8

PageRank

- Algorithm that gave Google the leap in quality
  - link structure centerpiece of scoring
- Framework
  - Given a directed graph with *n* nodes
  - Assign each node a score that represents its importance in structure: PageRank: pr(node)



9

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

10

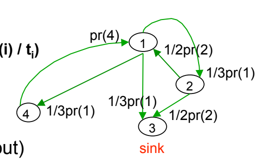
Attempt 1

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

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

Iterate until converges



Problems

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

11

Attempt 2

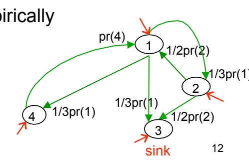
Random walk model

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

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

– α parameter chosen empirically

- Break cycles
- Escape from sinks



12

Normalized?

- Would like $\sum_{1 \le k \le n} (pr(k)) = 1$
- Consider $\sum_{1 \le k \le n} (pr_{new}(k))$

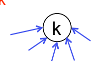
$= \sum_{1 \le k \le n}(\alpha/n + (1-\alpha)\sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i))$
$= \sum_{1 \le k \le n}(\alpha/n)+\sum_{1 \le k \le n}((1-\alpha)\sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i))$ *

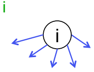
$= \quad \alpha \quad + (1-\alpha)\sum_{1 \le k \le n} \sum_{i \text{ with edge from } i \text{ to } k}(pr(i) / t_i)$
$= \quad \alpha \quad + (1-\alpha)\sum_{1 \le i \le n}\sum_{k \text{ with edge from } i \text{ to } k}(pr(i) / t_i)$ *

$= \quad \alpha \quad + (1-\alpha)\sum_{i \text{ with edge from } i} pr(i))$

*inner sum $\sum_i$ over incoming edges for one k

*inner sum $\sum_k$ over outgoing edges for one i



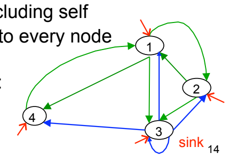


13

Problem for desired normalization

- Have
  $\sum_{1 \le k \le n} (pr_{new}(k)) = \alpha + (1-\alpha)\sum_{i \text{ with edge from } i} pr(i))$
- 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 $\sum_{1 \le k \le n} (pr_{initial}(k)) = 1$
then $\sum_{1 \le k \le n} (pr(k)) = 1$



14

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 ($1 \le i \le n$)
  If row i contains $t_i > 0$ ones, $L(i,k)=(1/ t_i) E(i,k)$, $1 \le k \le n$
  If row i contains 0 ones, $L(i,k) = 1/n$, $1 \le k \le n$
- Vector ***pr*** of PageRank values defined by
  $\boldsymbol{pr} = (\alpha/n, \alpha/n, \ldots \alpha/n)^T + (1-\alpha) L^T \boldsymbol{pr}$
- has a solution representing the steady-state values pr(k)

15

Calculation

- Choose α
  – No single best value
  – Page and Brin originally used α=.15
- Simple iterative calculation
  – Initialize $pr_{initial}(k) = 1/n$ for each node k
  so $\sum_{1 \le k \le n} (pr_{initial}(k)) = 1$
  – $pr_{new}(k) = \alpha/n + (1-\alpha)\sum_{1 \le i \le n} L(i,k)pr(i)$
- Converges
  – Has necessary mathematical properties
  – In practice, choose convergence criterion
    - Stops iteration

16

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

17

HITS

Hyperlink **I**nduced **T**opic **S**earch

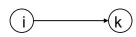
- Second well-known algorithm
- By Jon Kleinberg while at IBM Almaden Research Center
- Same general goal as PageRank
- Distinguishes 2 kinds of nodes
  – Hubs: resource pages
    - **Point to many authorities**
  – Authorities: good information pages
    - **Point to many hubs**

18

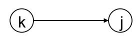
Mutual reinforcement

- Authority weight node j: a(j)
  - Vector of weights ***a***
- Hub weight node j: h(j)
  - Vector of weights ***h***
- Update:

$a_{new}(k) = \sum_{i \text{ with edge from i to k}} (h(i))$



$h_{new}(k) = \sum_{j \text{ with edge from k to j}} (a(j))$

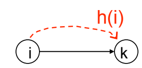


19

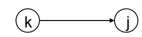
Mutual reinforcement

- Authority weight node j: a(j)
  - Vector of weights ***a***
- Hub weight node j: h(j)
  - Vector of weights ***h***
- Update:

$a_{new}(k) = \sum_{i \text{ with edge from i to k}} (h(i))$



$h_{new}(k) = \sum_{j \text{ with edge from k to j}} (a(j))$

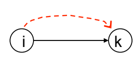


20

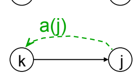
Mutual reinforcement

- Authority weight node j: a(j)
  - Vector of weights ***a***
- Hub weight node j: h(j)
  - Vector of weights ***h***
- Update:

$a_{new}(k) = \sum_{i \text{ with edge from i to k}} (h(i))$



$h_{new}(k) = \sum_{j \text{ with edge from k to j}} (a(j))$



21

Matrix formulation

Steady state:

$\boldsymbol{a} = E^T\boldsymbol{h}$ $\qquad$ $\boldsymbol{a} = E^TE\boldsymbol{a}$

$\boldsymbol{h} = E\boldsymbol{a}$ $\qquad$ $\boldsymbol{h} = EE^T\boldsymbol{h}$

Interpretation?

22

Matrix formulation

Steady state:

$\boldsymbol{a} = E^T\boldsymbol{h}$ $\qquad$ $\boldsymbol{a} = E^TE\boldsymbol{a}$

$\boldsymbol{h} = E\boldsymbol{a}$ $\qquad$ $\boldsymbol{h} = EE^T\boldsymbol{h}$

Interpretation:

- $E^TE(i,j)$: 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"

23

Iterative Calculation

$\boldsymbol{a} = \boldsymbol{h} = (1, \ldots, 1)^T$

While (not converged) {

$\boldsymbol{a}_{new} = E^t\boldsymbol{h}$

$\boldsymbol{h}_{new} = E\boldsymbol{a}$

$\boldsymbol{a} = \boldsymbol{a}_{new} / ||\boldsymbol{a}_{new}||$ normalize to unit vector

$\boldsymbol{h} = \boldsymbol{h}_{new} / ||\boldsymbol{h}_{new}||$ normalize to unit vector

}

Provable convergence by linear algebra

24

Use of HITS

original use **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:
   a) Root set
   b) *Plus* nodes pointed to by nodes of root set
   c) *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

using links to expand matches!

Results using HITS

- Documents ranked by authority score a(doc) and hub score h(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

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
    - example: 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

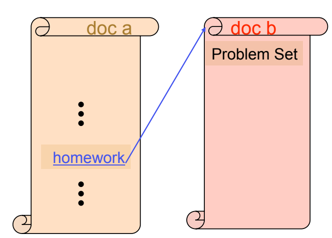
Revisit: How use links in ranking documents?

- use structure to compute score for ranking
  – PageRank, HITS
- include more objects to rank
  – saw in use of HITS

➢ use anchor text (HTML)
  – anchor text labels link
  – include anchor text as text of ***document pointed to***

Using anchor text

"homework" may not occur in *content* of doc b



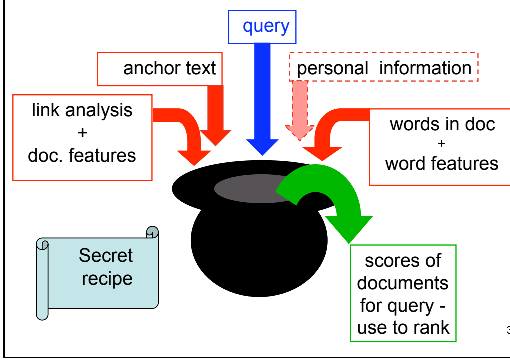
terms in doc b for building index:

⋮
homework: *anchor*
⋮
problem: *title* 1
⋮
set: *title* 2
⋮

Summary

- Link analysis
  - a principal component of ranking by modern Web search engines
  - must be combined with content analysis
- Extend document content with link info
  - anchor text
  - text of URLs
    - e.g. princeton.edu, aardvarksportsshop.com
- Expand set of satisfying docs using links
  - less often used

Ranking documents w.r.t. query



General Framework

- Have set of n features (aka signals) to use in determining ranking score
  - Features depend on query:
    vector $\Psi(d_i,q)$ of feature values f_k for doc d_i, query q
    - eg tf.idf score is feature
  - Features are conditioned to be comparable
- Have parameterized function to combine signals
  - simple: linear $\alpha_0 + \sum_{i=1}^{n} \alpha_i*(f_i)$
    - α_i are adjustable weights
  - how choose α_i ?

Machine Learning

Many possibilities – overview of one

Ordinal Regression Model

- Goal: get comparison of doc.s correct
- capture goal
  - Let $\boldsymbol{\omega}$ represent vector $(\alpha_1, \dots, \alpha_n)$
  - want $\boldsymbol{\omega}^T \bullet \Psi(d_i,q) - \boldsymbol{\omega}^T \bullet \Psi(d_j,q) > 0$ if and only if d_i more relevant than d_j for query q
  - find $\boldsymbol{\omega}$ that works

Finding $\boldsymbol{\omega}$ that work

One learning method

- based on *support vector machine classifiers*
- train on known correct data:
  - humans rank a set of documents for various queries
- for training set solve:

  find $\boldsymbol{\omega}$ and ζ_{ij} for all pairs of doc.s d_i and d_j, so that

  $\frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + c\Sigma_{ij}(\zeta_{ij})$ is minimized

  and

  for all d_i and d_j with d_i more relevant than d_j

  $\boldsymbol{\omega}^T \bullet \Psi(d_i,q) - \boldsymbol{\omega}^T \bullet \Psi(d_j,q) \geq 1 - \zeta_{ij}$