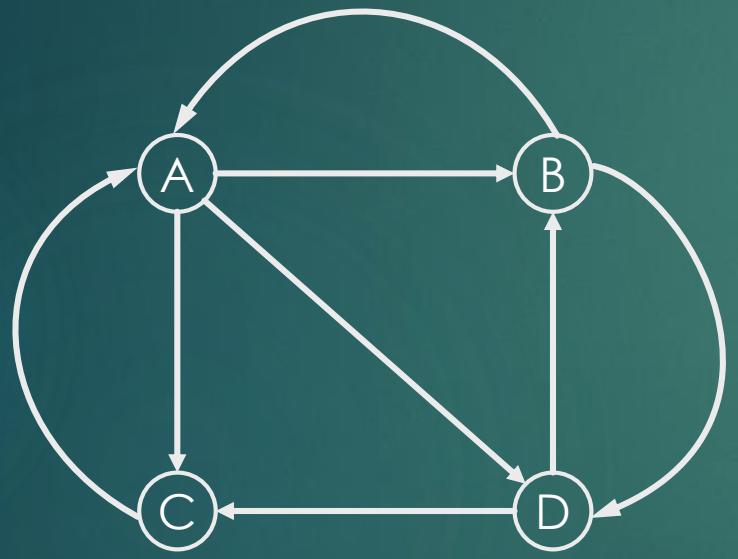
Page Rank

CS 5965/6965 - Big Data Systems - Fall 2014

Webpage quality ranking

- Inverted web indexes help locate matching pages of search words
  - But there are too many matches and humans can't read all
- Both relevance and quality are important in web search
- What is a high-quality web page?
- How to identify a high-quality web page?
  - Hard to spam
- Related to identifying high-quality scientific publications
  - But much bigger dataset

Page Rank



Transition matrix

$$M = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

$\boldsymbol{v}$ → probability distribution for the location of a random surfer

$\boldsymbol{v} \leftarrow \left\{\frac{1}{n}\right\}^n$

Iterate on $\boldsymbol{v} \leftarrow M\boldsymbol{v}$

Page Rank

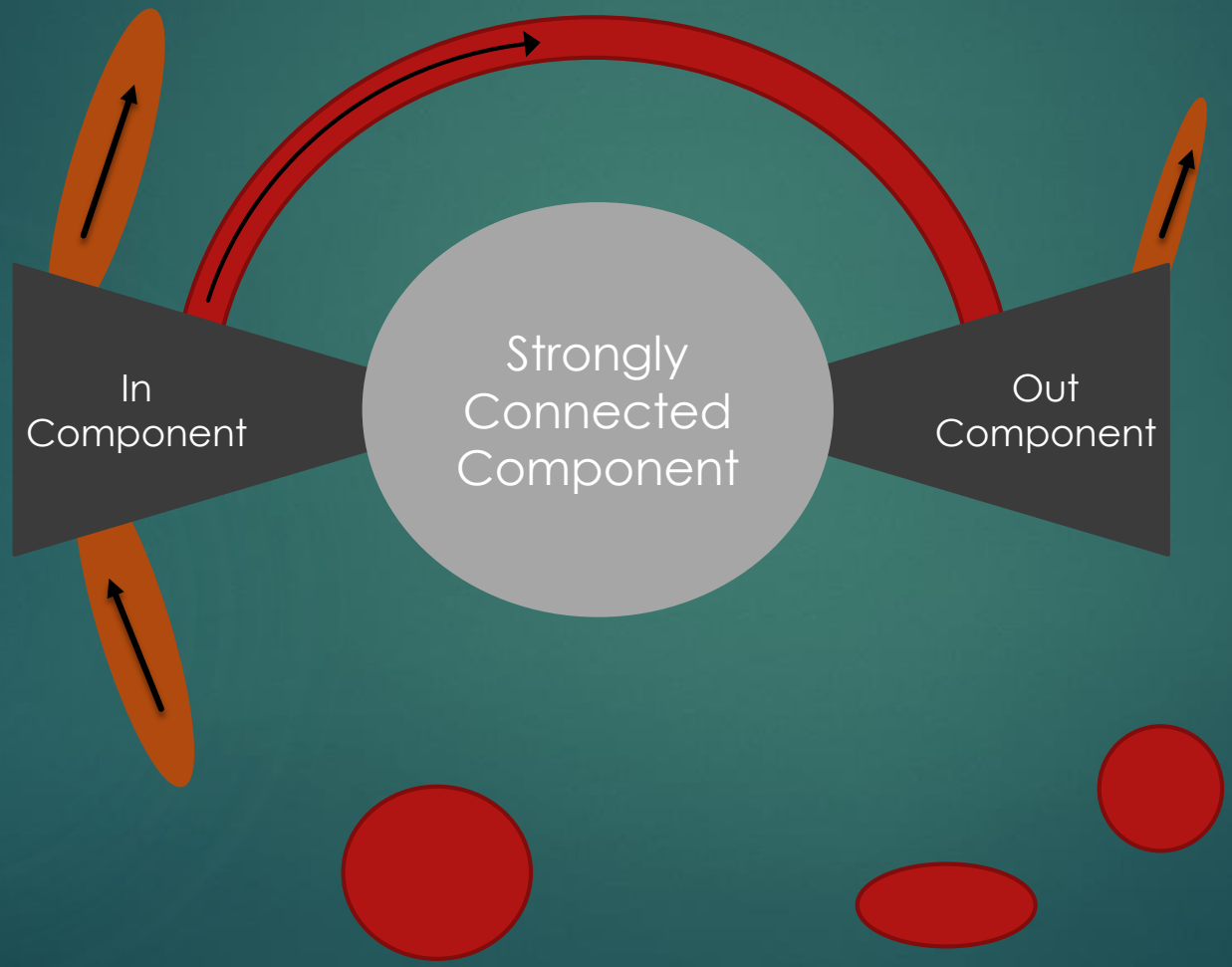
- Markov process
  - Limiting distribution
  - will converge if
    - Strongly connected
    - No dead ends
- Limiting $\boldsymbol{v}$ is an eigenvector of M
  - $\lambda \boldsymbol{v} = M\boldsymbol{v}$
  - v is also the primary eigenvector
- Iterate a few times on $v \leftarrow Mv$ until $\|v_{i+1} - v_i\| < \epsilon$

Solving Linear Systems

- $\boldsymbol{Mx = y \;\Rightarrow\; x = M^{-1}y}$

- Gaussian Elimination → $\mathcal{O}(n^3)$
- Iterative approaches → $\mathcal{O}(kn^2)$
  - For sparse systems → $\mathcal{O}(kn)$
  - Use optimal solvers → k independent of n

Structure of the Web



Dead Ends

Spider Traps

Dead Ends

- Remove dead ends from the graph
  - And incoming links
- Compute page-rank on strongly connected component
- Restore graph, retaining page ranks
- Use existing page ranks to compute ranks for dead-end nodes

Spider traps & Taxation

- modify the calculation of PageRank by allowing each random surfer a small probability of teleporting to a random page

$$\boldsymbol{v}' = \beta M \boldsymbol{v} + \frac{(1-\beta)\boldsymbol{e}}{n}$$

- β is a constant that represent the probability that the surfer follows a link on the page
- Approach will still be biased towards spider traps

Efficient Computation of PageRank

- Transition Matrix M is very sparse
- Store locations of non-zero entries
- In general for sparse matrices
  - (i, j, M_{ij}) → 4+4+8 bytes
- Further compression possible for transition matrix
  - Store degree of column plus indices
  - Number of links on a page plus the indices of those pages

Topic Sensitive PageRank

- Weight certain pages more because of their topic
- Allows personalization of results to users
  - Ideally a separate page rank vector for each user
  - Not scalable
- Create one vector for each of a small set of topics
  - Basis vectors
  - Determine weights for each individual user
    - size → number of basis vectors

Biased Random Walks

- Identify certain pages that represent a given topic
- (re) introduce random surfers to only topic specific pages
- Let S be the set of integers consisting of the indices of topic-specific pages, and $\boldsymbol{e}_S$ be a vector that is 1 in S and 0 elsewhere
- Topic sensitive PageRank

$$v' = \beta M v + \frac{(1-\beta)e_S}{|S|}$$

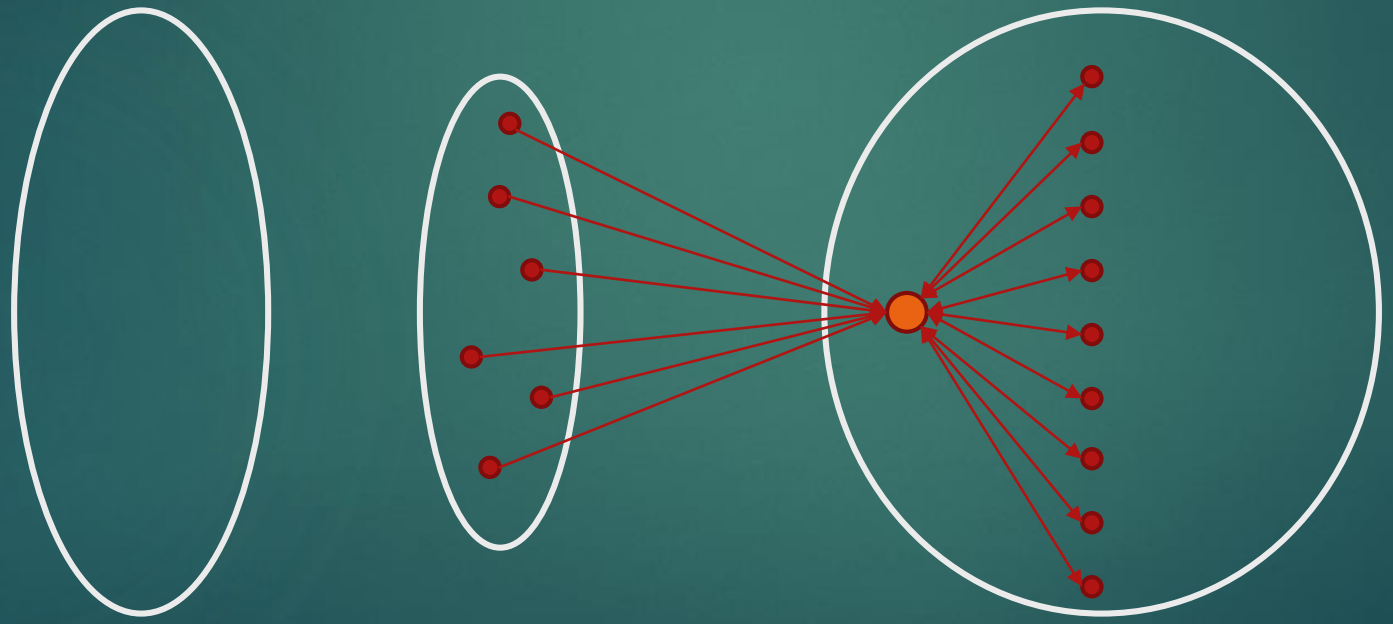
Using topic-sensitive PageRank

- Decide on the topics for which we shall create specialized PageRank vectors
  - Manually
  - From Data
- Pick the set S for each of these topics, and use that set to compute the topic-sensitive PageRank vector for that topic
- Determine which topics are of most interest to a particular user/query
- Use the PageRank vectors for those topics in ordering the results

How to cheat?

Link Spam

- Techniques for artificially increasing the PageRank of a page
- Spam Farm



Analysis of a Spam farm

- β → taxation parameter
- n → total number of webpages
- Target t with m supporting pages
- Let x be the amount of PageRank contributed by accessible pages
- Let us compute y, the PageRank of t

Analysis of a Spam farm

- PageRank of each supporting page

$$\frac{\beta y}{m} + \frac{1-\beta}{n}$$

- PageRank of target

$$y = x + \beta m \left(\frac{\beta y}{m} + \frac{1-\beta}{n} \right) + \frac{1-\beta}{n}$$

Analysis of a Spam farm

- PageRank of each supporting page

$$\frac{\beta y}{m} + \frac{1-\beta}{n}$$

- PageRank of target

$$y = x + \beta^2 y + \beta(1-\beta)\frac{m}{n}$$

Analysis of a Spam farm

- PageRank of each supporting page

$$\frac{\beta y}{m} + \frac{1-\beta}{n}$$

- PageRank of target

$$y = \frac{x}{1-\beta^2} + \frac{\beta}{1+\beta}\frac{m}{n}$$

- If $\beta = 0.85$

Analysis of a Spam farm

- PageRank of each supporting page

$$\frac{\beta y}{m} + \frac{1-\beta}{n}$$

- PageRank of target

$$y = 3.6\,x + 0.46\,\frac{m}{n}$$

Combating Link Spam

- *TrustRank*: variation of topic-sensitive PageRank
- *Spam mass*: calculation that identifies spam farms

Trust Rank

- topic-sensitive PageRank, where the *topic* is a set of pages believed to be trustworthy
- Manually select trustworthy pages
- Avoid trustworthy sites where anyone can create links
  - Many websites prevent users from entering URLs in comments
- Domains where membership is controlled
  - .edu .gov etc …

Spam Mass

- Measure the fraction of the pagerank that comes from spam
- Compute the ordinary pagerank (r) and trustrank (t) of a page
  - Spam mass $= \frac{r-t}{r}$
- Negative or small positive spam mass → not spam
- Closer to 1 → spam