IV.4 Topic-Specific & Personalized PageRank

- PageRank produces **"one-size-fits-all" ranking** determined assuming uniform following of links and random jumps

- How can we obtain **topic-specific** (e.g., for *Sports*) or **personalized** (e.g., based on my bookmarks) rankings?
  - bias **random jump probabilities** (i.e., modify the vector **j**)
  - bias **link-following probabilities** (i.e., modify the matrix **T**)

- What if we do not have hyperlinks between documents?
  - construct **implicit-link graph** from user behavior or document contents

Topic-Specific PageRank

- Input: Set of **topics** C (e.g., *Sports*, *Politics*, *Food*, …)
  Set of **web pages** S_c **for each topic** c (e.g., from dmoz.org)

- Idea: Compute a **topic-specific ranking** for c by **biasing the random jump** in PageRank toward web pages S_c of that topic

$$\mathbf{P}_c = (1-\epsilon)\,\mathbf{T} + \epsilon\,\left[1\,\ldots\,1\right]^T\,\mathbf{j}_c \text{ with } \mathbf{j}_{ci} = \begin{cases} 1/|S_c| & : i \in S_c \\ 0 & : i \notin S_c \end{cases}$$

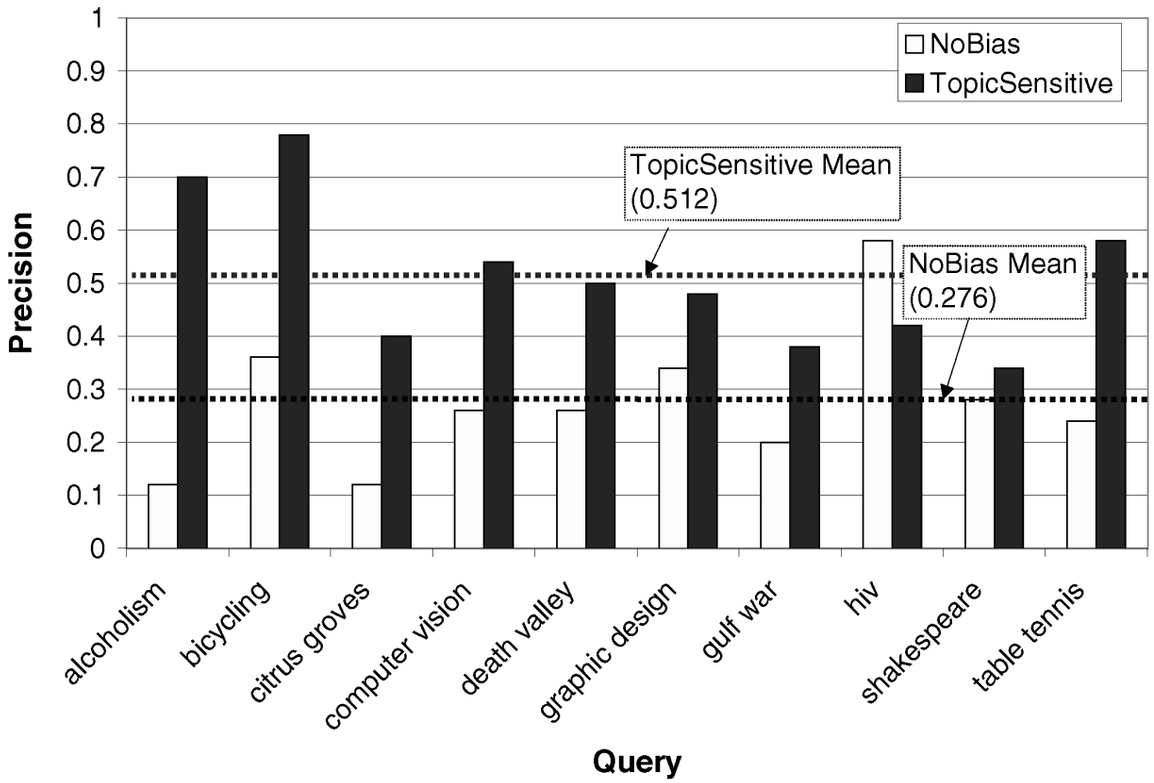
- Method:
  - Precompute **topic-specific PageRank** vectors π_c
  - Classify user query q to obtain **topic probabilities** $P[c|q]$
  - Final importance score obtained as **linear combination**

$$\pi = \sum_{c \in C} P[c|q]\,\pi_c$$

Topic-Specific PageRank (cont'd)

NoBias	Arts
"RailRiders Adventure Clothing" www.RailRiders.com www.Waypoint.org/default.html www.Gorp.com/ www.FloridaCycling.com/	"Photo Contest & Gallery (Bicycling)" www.bikescape.com/photogallery/ www.trygve.com/ www.greenway.org/ www.jsc.nasa.gov/Bios/htmlbios/young.html
Business	**Computers**
"Recumbent Bikes and Kit Aircraft" www.rans.com www.BreakawayBooks.com java.oreilly.com/bite-size/ www.carbboom.com	"GPS Pilot" www.gpspilot.com www.wireless.gr/wireless-links.htm www.linkstosales.com www.LiftExperts.com/lifts.html
Games	**Kids and Teens**
"Definition Through Hobbies" www.flick.com/~gretchen/hobbies.html www.BellaOnline.com/sports/ www.npr.org/programs/wesun/puzzle/will.html www.trygve.com/	"Camp Shohola For Boys" www.shohola.com www.EarthForce.org www.WeissmanTours.com www.GrownupCamps.com/homepage.html
Recreation	**Science**
"Adventure travel" www.gorp.com/ www.GrownupCamps.com/homepage.html www.gorp.com/gorp/activity/main.htm www.outdoor-pursuits.org/	"Coast to Coast by Recumbent Bicycle" hypertextbook.com/bent/ www.SiestaSoftware.com/ www.BenWiens.com/benwiens.html www.SusanJeffers.com/jeffbio.htm
Shopping	**Sports**
"Cycling Clothing & Accessories for Women" www.TeamEstrogen.com/ www.ShopOutdoors.com/ www.jub.com.au/books/ www.bike.com/	"Swim, Bike, Run, & Multisport" www.multisports.com/ www.BikeRacing.com/ www.CycleCanada.com/ www.bikescape.com/photogallery/

Query: *bicycling*



- Full details: [Haveliwala '03]

Personalized PageRank

- Idea: Provide every user with a **personalized ranking** based on her **favorite web pages** F (e.g., from bookmarks or likes)

$$\mathbf{P}_F = (1-\epsilon)\,\mathbf{T} + \epsilon\,\left[1\,\ldots\,1\right]^T\,\mathbf{j}_F \text{ with } \mathbf{j}_{F_i} = \begin{cases} 1/|F| & : i \in F \\ 0 & : i \notin F \end{cases}$$

- Problem: **Computing and storing** a personalized PageRank vector for every single user is **too expensive**

- Theorem [*Linearity of PageRank*]: Let $\mathbf{j}_F$ and $\mathbf{j}_{F'}$ be personalized random jump vectors and let $\boldsymbol{\pi}$ and $\boldsymbol{\pi}$**'** denote the corresponding personalized PageRank vectors. Then for all w, $w' \geq 0$ with $w + w' = 1$ the following holds:

$$(w\,\boldsymbol{\pi} + w'\,\boldsymbol{\pi}') = (w\,\boldsymbol{\pi} + w'\,\boldsymbol{\pi}')\,(w\,\mathbf{P}_F + w'\,\mathbf{P}_{F'})$$

Personalized PageRank (cont'd)

- Corollary: For a random jump vector $\mathbf{j}_F$ and basis vectors $\mathbf{e}_k$

$$\mathbf{e}_{k_i} = \begin{cases} 1 : i = k \\ 0 : i \neq k \end{cases}$$ with corresponding PageRank vectors $\boldsymbol{\pi}_k$

we obtain the personalized PageRank vector $\boldsymbol{\pi}_F$ as

$$\mathbf{j}_F = \sum_k w_k \, \mathbf{e}_k \qquad \boldsymbol{\pi}_F = \sum_k w_k \, \boldsymbol{\pi}_k$$

- Full details: [Jeh and Widom '03]

Link Analysis based on Users' Browsing Sessions

- Simple data mining on **browsing sessions** of many users, where each session i is a sequence (p_{i1}, p_{i2}, …) of **visited web pages**:
  - consider all pairs (p_{ij}, p_{ij+1}) of **successively visited web pages**
  - determine for each pair of web pages (i, j) its **frequency** $f(i, j)$
  - select pairs with $f(i, j)$ above **minimum support threshold**
- Construct **implicit-link graph** with the selected page pairs as edges and their normalized total frequencies as edge weights
- Apply **edge-weighted PageRank** to this implicit-link graph
- Approach has been extended to factor in how much time users spend on web pages and whether they tend to go there directly
- Full details: [Xue et al. '03] [Liu et al. '08]

PageRank without Hyperlinks

- Objective: Re-rank documents in an initial query result to bring up **representative documents** similar to many other documents

- Consider implicit-link graph derived from **contents of documents**
  - weighted edge (i, j) present if document d_j is among the k documents having the **highest likelihood** $P[d_i|d_j]$ of **generating document** d_i (estimated using unigram language model with Dirichlet smoothing)

- Apply **edge-weighted PageRank** to this implicit-link graph

$$\mathbf{T}_{ij} = \begin{cases} \frac{w(i,j)}{\sum_{(i,k)\in E} w(i,k)} & : (i,j) \in E \\ 0 & : (i,j) \notin E \end{cases}$$

- Full details: [Kurland and Lee '10]

Summary of IV.4

- **Topic-Specific PageRank**
  biases random jump **j** toward web pages known to belong to a specific topic (e.g., *Sports*) to favor web pages in their vicinity

- **Personalized PageRank**
  biases random jump **j** toward user's favorite web pages
  linearity of PageRank allows for more efficient computation

- **PageRank on Implicit-Link Graphs**
  can be derived from user behavior or documents' contents
  biases link-following probabilities **T**

Additional Literature for IV.4

- **D. Fogaras, B. Racz, K. Csolgany, and T. Sarlos**: *Towards Fully Scaling Personalized PageRank: Algorithms, Lower Bounds, and Experiments*, Internet Mathematics 2(3): 333-358, 2005

- **D. Gleich, P. Constantine, A. Flaxman, A. Gunawardana**: *Tracking the Random Surfer: Empirically Measured Teleportation Parameters in PageRank*, WWW 2010

- **T. H. Haveliwala**: *Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search*, TKDE 15(4):784-796, 2003

- **G. Jeh and J. Widom**: *Scaling Personalized Web Search*, KDD 2003

- **O. Kurland and L. Lee**: *PageRank without Hyperlinks: Structural Reranking using Links Induced by Language Models*, ACM TOIS 28(4), 2010

- **Y. Liu, B. Gao, T.-Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li**: *BrowseRank: Letting Web Users Vote for Page Importance*, *SIGIR 2008*

- **G.-R. Xue, H.-J. Zeng, Z. Chen, W.-Y. Ma, H.-J. Zhang, C.-J. Lu**: *Implicit Link Analysis for Small Web Search*, SIGIR 2003

IV.5 Online Link Analysis

- PageRank and HITS operate on a (partial) **snapshot of the Web**
- Web **changes all the time**!
- Search engines **continuously crawl** the Web to keep up with it

- How can we compute a PageRank-style measure of importance online, i.e., as new/modified pages & hyperlinks are discovered?

OPIC

- Ideas:
  - **integrate** computation of page importance **into the crawl process**
  - compute small fraction of importance as crawler proceeds **without having to store the Web graph** and keeping track of its changes
  - each page holds some **"cash"** that reflects its importance
  - when a page is visited, it **distributes its cash** among its successors
  - when a page is not visited, it **can still accumulate cash**
  - this **random process** has a **stationary limit** that captures the importance but is generally not the same as PageRank's stationary distribution

- Full details: [Abiteboul et al. '03]

OPIC (cont'd)

- **OPIC**: Online Page Importance Computation
- Maintain for each page i (out of n pages):
  - ***C*[*i*]** – cash that page i currently has and can distribute
  - ***H*[*i*]** – history of how much cash page has ever had in total
- Global counter
  - ***G*** – total amount of cash that has ever been distributed

```
G = 0; for each i do { C[i] = 1/n ; H[i] = 0 };
do forever {
        choose page i                          // (e.g., randomly or greedily)
        H[i] += C[i]                           // update history
        for each successor j of i do
                C[j] += C[i] / out(i)          // distribute cash
                G += C[i]                      // update global counter
                C[i] = 0                       // reset cash
}
```

OPIC (cont'd)

- Assumptions:
  - Web graph is strongly connected
  - for convergence, every page needs to be visited infinitely often
- At each step, an **estimate of the importance** of page *i* can be obtained as:

$$X[i] = \frac{H[i]}{G}$$

- Theorem: Let X_t denote the vector of cash fractions accumulated by pages until step t. The limit

$$X = \lim_{t \to \infty} X_t$$

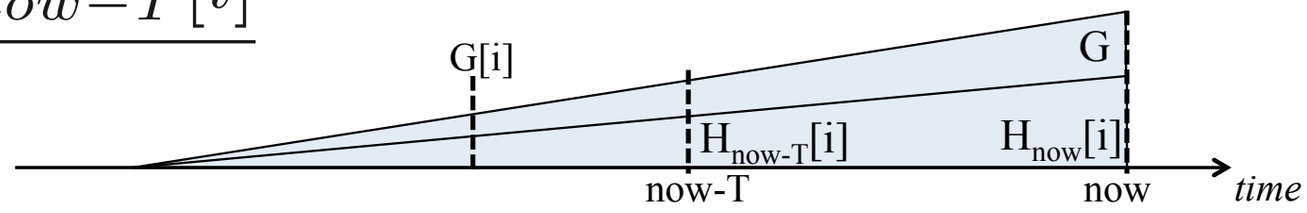
exists with

$$\|X\|_1 = \sum_i X_i = 1$$

Adaptive OPIC for Evolving Graphs

- Idea: Consider a time window [*now*-T, *now*] where time corresponds to the value of *G*
- Estimate importance of page *i* as

$$X_{now}[i] = \frac{H_{now}[i] - H_{now-T}[i]}{T}$$



- For crawl time *now*, update history $H_{now}[i]$ by **interpolation**
  - Let $H_{now\text{-}T}[i]$ be the cash acquired by page i until time (*now*-*T*)
  - $C_{now}[i]$ the current cash of page *i*
  - Let $G[i]$ denote the time *G* at which *i* was crawled previously

$$H_{now}[i] = \begin{cases} H_{now-T} \cdot \frac{T-(G-G[i])}{T} + C_{now}[i] & : G - G[i] < T \\ C_{now}[i] \cdot \frac{T}{G-G[i]} & : \text{otherwise} \end{cases}$$

Summary of IV.5

- **OPIC**
  integrates page importance computation into crawl process
  can be made adaptive to handle the evolving Web graph

Additional Literature for IV.5

- **S. Abiteboul, M. Preda, G. Cobena**: *Adaptive on-line page importance computation*, WWW 2003

IV.6 Similarity Search

- How can we use the links between objects (not only web pages) to figure out **which objects are similar to each other**?

- **Not limited to the Web graph** but also applicable to
  - *k*-partite graphs derived from relational database (students, lecture, etc.)
  - implicit graphs derived from observed user behavior
  - word co-occurrence graphs
  - …

- Applications:
  - Identification of similar pairs of objects (e.g., documents or queries)
  - Recommendation of similar objects (e.g., documents based on a query)

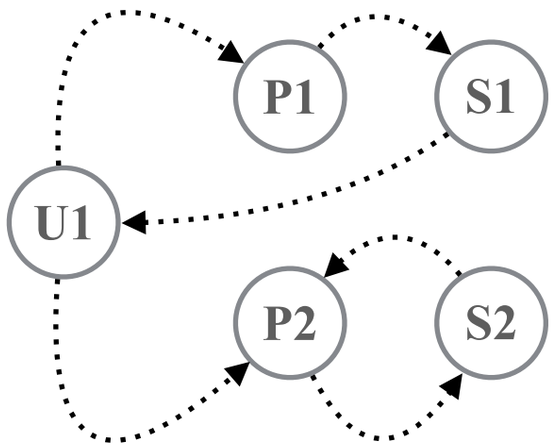
SimRank

- Intuition: Two objects are **similar if similar objects** point to them

$$s(u,v) = \frac{C}{|I(u)|\,|I(v)|} \sum_{i=1}^{|I(u)|} \sum_{j=1}^{|I(v)|} s(I_i(u), I_j(v))$$

with **confidence constant** $C < 1$, **in-neighbors** $I(u)$ and $I(v)$, and $I_i(u)$ and $I_j(v)$ as the ***i*-th and *k*-th in-neighbor** of u and v

- Example: **U**niversities, **P**rofessors, **S**tudents



With $C = 0.8$:

s(P1, P2)	=	0.414
s(S1, S2)	=	0.331
s(U1, P2)	=	0.132
s(P1, S2)	=	0.106
s(P2, S2)	=	0.088
s(P2, S1)	=	0.042
s(U1, S2)	=	0.034

SimRank (cont'd)

$s^{(0)}(u, v) = 1$ (for $u = v$) $s^{(0)}(u, v) = 0$ (for $u \neq v$)

Repeat until convergence:

$$s^{(k+1)}(u,v) = \frac{C}{|I(u)|\,|I(v)|} \sum_{i=1}^{|I(u)|} \sum_{j=1}^{|I(v)|} s^{(k)}(I_i(u), I_j(v)) \qquad \text{(for } u \neq v\text{)}$$

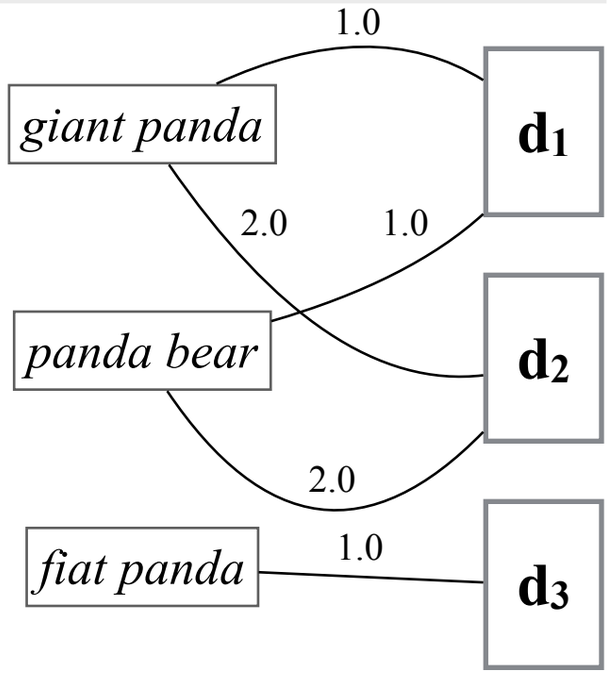
$$s^{(k+1)}(u,v) = 1 \qquad \text{(for } u = v\text{)}$$

- SimRank score $s(u, v)$ can be interpreted as the **expected number of steps** that it takes **two random surfers to meet** if they
  - **start** at nodes u and v
  - walk the graph **backwards in lock step** (i.e., their steps are synchronous)

- Full details: [Jeh and Widom '03]

Random Walks on the Click Graph

- Consider **bi-partite click graph** with queries and documents as **vertices** and **weighted edges** (q, d) indicating users' tendency to click on document d for query q



- Perform **PageRank-style random walk** with link-following probabilities proportional to edge weights and random jump to single query or document

- Applications:
  - **query-to-document search**
  - **query-to-query suggestion**
  - **document-by-query annotation**
  - **document-to-document suggestion**

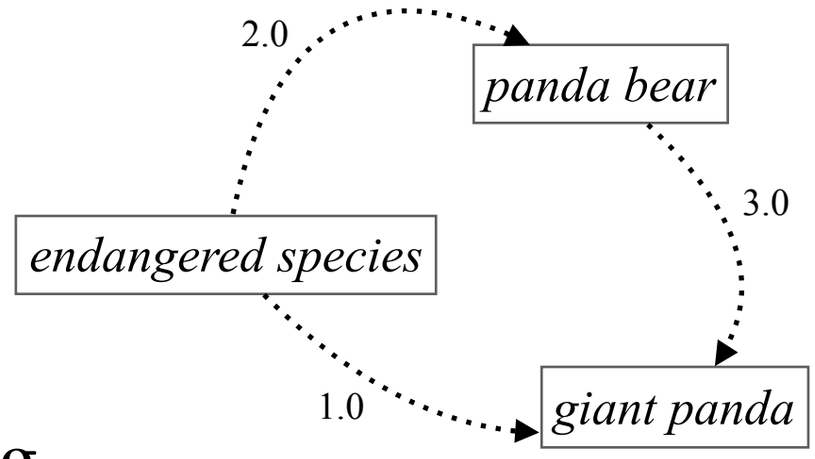
$k=$

Annotation using a random walk:

P	Query	Distance
0.075	boxer dog puppies	3
0.066	boxer puppy pics	3
0.060	boxer puppies	1
0.056	puppy boxer	3
0.056	boxer puppy pictures	3
0.049	boxer pups	3
0.049	boxer puppy	3
0.038	puppy boxers	5
0.034	boxer pup	3
0.030	baby boxer	3

Random Walks on the Query-Flow Graph

- Consider **query-flow graph** with queries as **vertices** and **weighted edges** (q, q') reflecting how often q' is issued after q



- Recommend related queries by performing **PageRank-style random walk** on the query-flow graph with **link-following probabilities** proportional to edge weights and **random jump** to current query (or last few queries)

`banana` → `apple`	`banana`
banana	banana
apple	eating bugs
usb no	banana holiday
banana cs	opening a banana
giant chocolate bar	banana shoe
where is the seed in anut	fruit banana
banana shoe	recipe 22 feb 08
fruit banana	banana jules oliver
banana cloths	banana cs
eating bugs	banana cloths

`beatles` → `apple`	`beatles`
beatles	beatles
apple	scarring
apple ipod	paul mcartney
scarring	yarns from ireland
srg peppers artwork	statutory instrument A55
ill get you	silver beatles tribute band
bashles	beatles mp3
dundee folk songs	GHOST'S
the beatles love album	ill get you
place lyrics beatles	fugees triger finger remix

- Full details: [Boldi et al. '08]

Summary of IV.6

- **SimRank**
  considers two objects similar if similar objects point to them
  is based on two lock-step backwards random walks

- **Click graph**
  a bi-partite graph capturing users' click behavior
  can be used to recommend similar queries or similar documents

- **Query-flow graph**
  a directed graph derived from users' query sessions
  can be used to recommend similar queries

Additional Literature for IV.6

- **G. Jeh and J. Widom**: *SimRank: A Measure of Structural-Contextual Similarity,* KDD 2002
- **N. Craswell and M. Szummer**: *Random Walks on the Click Graph*, SIGIR 2007
- **P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna**: *The Query-flow Graph: Model and Applications*, CIKM 2008

IV.7 Spam Detection

- **Discoverability** of web pages has often a direct **impact** on the **commercial success** of the business behind them
- **Search Engine Optimization** (SEO) seeks to optimize web pages to make them easier to discover for potential customers
  - "white hat" (optimizes for the user and respects to search engine policies)
  - "black hat" (manipulates search results by web spamming techniques)
- Web spamming techniques and search engines evolved in parallel
  - initially: only content spam, then: link spam, now: social media spam
  - 2004 DarkBlue SEO challenge: "*nigritude ultramarine*"
  - 2005 c't SEO challenge: "*Hommingberger Gepardenforelle*"

Content Spam vs. Link Spam vs. Content Hiding

- **Content spam**
  - **keyword stuffing** – add unrelated but often-sought keywords to page
  - **invisible content** – unrelated content invisible to users (like this)
- **Link spam**
  - **link farms** – collection of pages aiming to manipulate PageRank
  - **honeypots** – create valuable web pages with links to spam page
  - **link hijacking** – leave comments on reputable web pages or blogs
- **Content hiding**
  - **cloaking** – show different content to search engine's crawler and users
- More details: [Gyöngyi et al. '05]

TrustRank & BadRank

- Idea: Pages linked to by trustworthy pages tend to be trustworthy
- **TrustRank** performs PageRank-style **random walk** with random jumps only to an explicitly selected set of **trusted pages** *T*

- Idea: Pages linking to spam pages tend to be spam themselves
- **BadRank** performs PageRank-style **backwards random walk** (i.e., following incoming links) with random jumps only to an explicitly selected set of **blacklisted pages** *B*

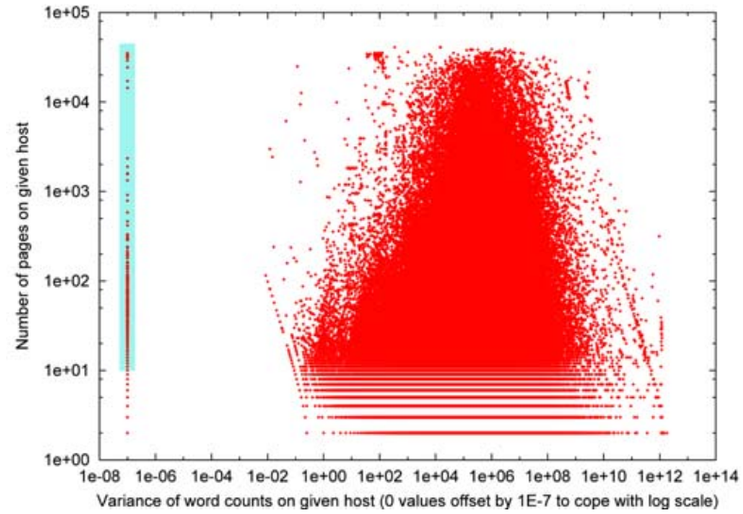
- Problems:
  - Sets of trusted and blacklisted pages are **difficult to maintain**
  - TrustRank and BadRank scores are **hard to interpret and combine**

- Full details: [Kamvar et al. ’03][Gyöngyi et al. ‘04]

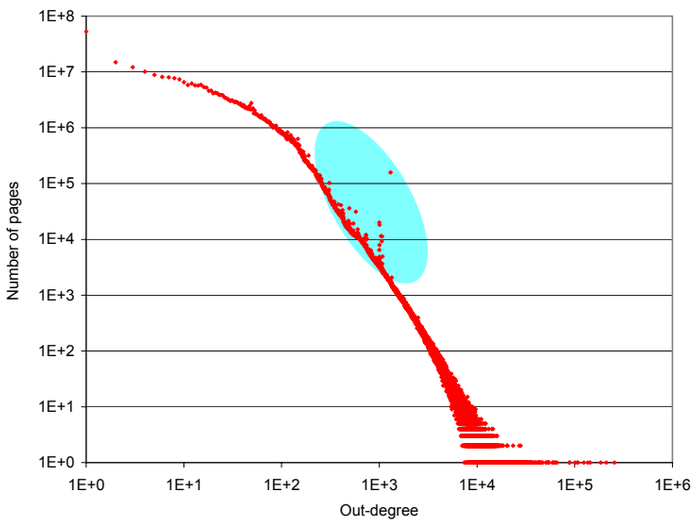
Spam, Damn Spam, and Statistics

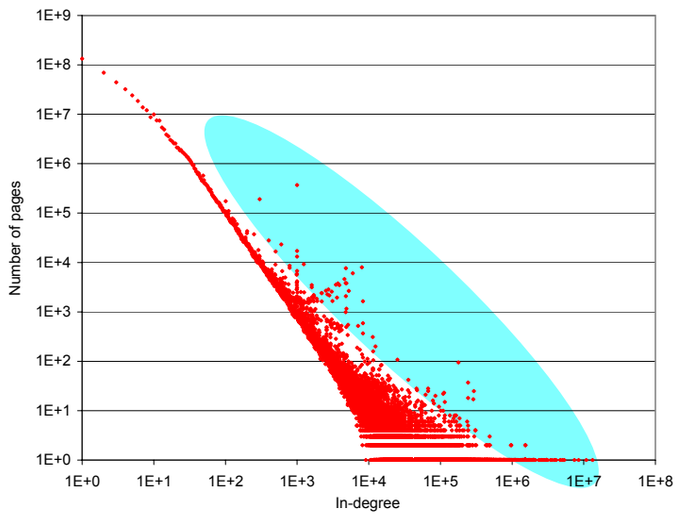
- Idea: Look for **statistical deviation**

- Content spam: Compare **word frequency distribution** to distribution in "good hosts"



- Link spam: Identify **outliers in out-degree and in-degree distributions** and inspect intersection





Typical for the Web:

$$P[deg = k] \propto \frac{1}{k^s}$$

$$s_{in} \approx 2.10$$

$$s_{out} \approx 2.72$$

- Full details: [Fetterly et al. '04]

SpamMass

- Idea: Measure **spam mass** as the amount of PageRank score that a web page receives **from web pages known to be spam**
- Assume that web pages are partitioned into **good pages** V^+ and **bad pages** V^- and that a **"good core"** $C \subseteq V^+$ is known
- **Absolute spam mass** of page p is then estimated as

$$SM(p) = \boldsymbol{\pi}(p) - \boldsymbol{\pi}_C(p)$$

with $\boldsymbol{\pi}(p)$ as its PageRank score and $\boldsymbol{\pi}_C(p)$ as its PageRank score with random jumps only to pages in the good core

- **Relative spam mass** of page p *is*

$$rSM(p) = SM(p)/\boldsymbol{\pi}(p)$$

- Full details: [Gyöngyi et al. '05] [Gyöngyi et al. '06]

Learning Spam Features

- Idea: Use **classifier** (e.g., Naïve Bayes or SVM) to classify pages into *Spam* and *NoSpam* based on context- and content-features
- **Discriminative context features** [Drost and Scheffer '05]:
  - tf.idf weights in page *p* and in-neighbors *in*(*p*)
  - average in-degree and out-degree of pages in *in*(p)
  - average number of words in title of pages in *out*(p)
  - number of pages in *in*(*p*) with same length as some other page in *in*(*p*)
  - sum of in-degree and out-degree of pages in *in*(*p*)
  - clustering coefficient of pages in *in*(*p*) (existing edges / possible edges)
  - number of pages in *in*(*p*) with same IP address as *p*
  - …

Learning Spam Features (cont'd)

- **Discriminative content features** [Ntoulas et al. '06]
  - average word length in page
  - percentage of page content that is anchor text
  - percentage of page content that is visible
  - percentage of page content in popular words (e.g., stopwords)
  - compressibility of page content (e.g., using Zip compression)
  - …
- Problem: *It's an arms race!* Spammers adjust to counter measures
- Full details: [Drost and Scheffer '05][Ntoulas et al. '06]

Summary of IV.7

- **Link spam**
  targets link analysis methods like PageRank

- **Statistical deviation**
  spam sites have different degree and word-frequency distributions

- **TrustRank & BadRank**
  perform PageRank-style from/to trusted/bad web pages

- **SpamMass**
  determines how much of a page's PageRank score is due to spam

Additional Literature for IV.7

- **A. Benczur, K. Csalongany, T. Sarlos, and M. Uher**: *SpamRank – Fully Automatic Link Spam Detection*, AIRWeb Workshop 2005

- **L. Becchetti, C. Castillo, D. Donato, R. Baeza-Yates, and S. Leonardi**: *Link analysis for Web spam detection,* ACM TWEB 2(1):1:42, 2008

- **C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri**: *Know your neighbors: Web spam detection using the web topology*, SIGIR 2007

- **I. Drost and T. Scheffer**: *Thwarting the Nigritude Ultramarine: Learning to Identify Link Spam*,
  ECML 2005

- **D. Fetterly, M. Manasse, and M. Najork**: *Spam, Damn Spam, and Statistics*,
  WebDB‘05

- **Z. Gyöngyi and H. Garcia-Molina**: *Spam: It‘s Not Just for Inboxes Anymore*,
  IEEE Computer 2005

- **Z. Gyöngyi, P. Berkhin, H. Garcia-Molina, and J. Pedersen**: *Link Spam Detection based on Mass Estimation*,
  VLDB 2006

IV.8 Social Networks

- **Social networks**
  - **diverse relations** (e.g., friendship, liking, check-in, following) between
  - **diverse types of objects** (e.g., people, entities, posts, images, videos)

- **Folksonomies** (~ folk + taxonomy)
  - allow users to organize objects by **tagging**
  - no centrally controlled vocabulary

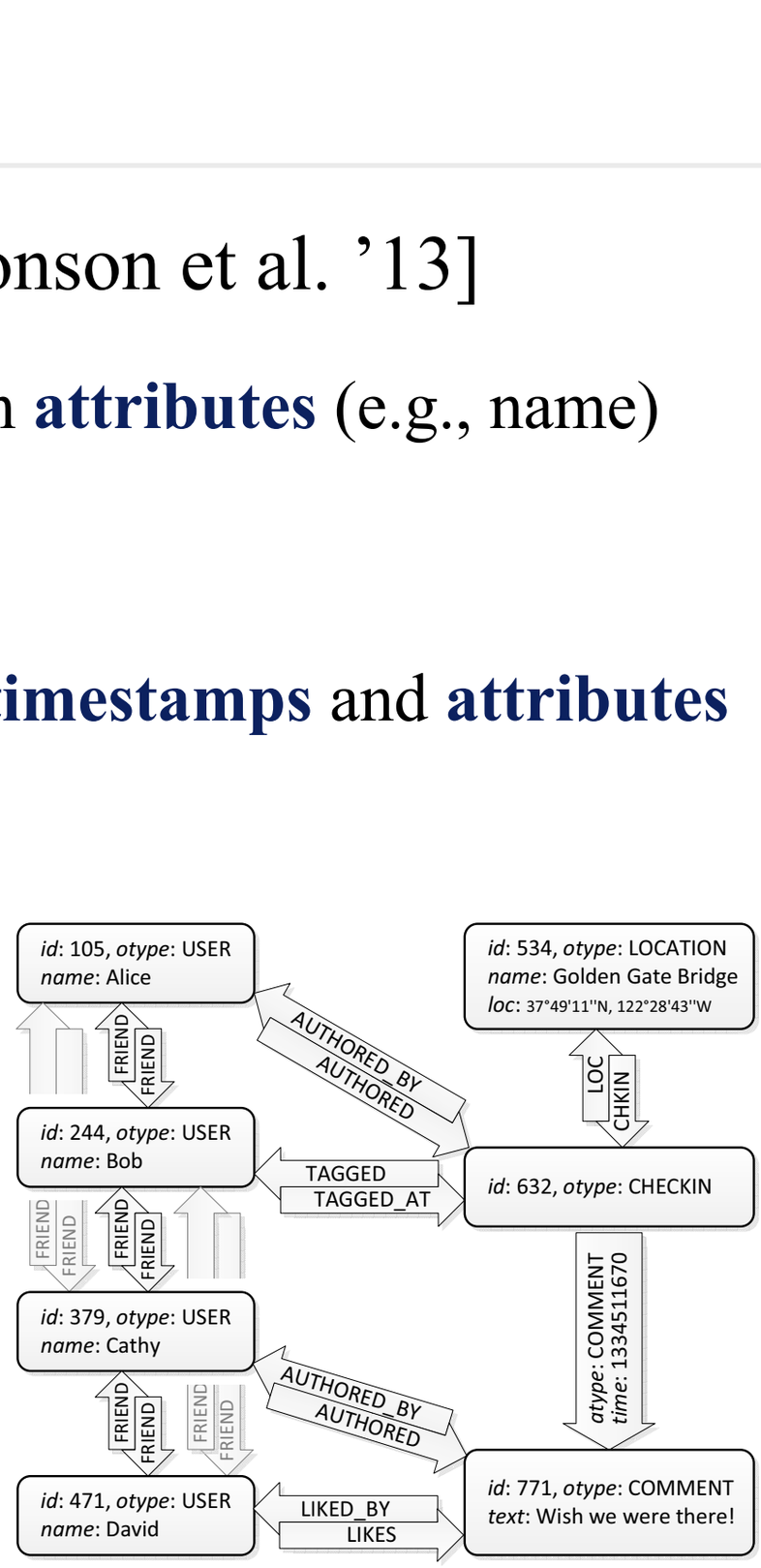
- Link analysis methods give insights into **importance** and **similarity** of objects (e.g., for ranking or recommendation)

More Than Directed Graphs…

- Example: **Facebook's Social Graph** [Bronson et al. '13]
  - **typed objects** (e.g., USER, LOCATION) with **attributes** (e.g., name)

    (*id*) => (*otype*, (*key* => *value*)*)
  - **typed directed relations** (e.g., LIKES) with **timestamps** and **attributes**

    (*id1*, *atype*, *id2*) => (*time*, (*key* => *value*)*)



- Full details: [Bronson et al. '13]

SocialPageRank

- Considers **pages** P, **tags** T, and **users** U
  - $\mathbf{M}_{PU}$ capturing **page-user associations** (# tags assigned by u to p)
  - $\mathbf{M}_{UT}$ capturing **user-tag associations** (# pages tagged by u with t)
  - $\mathbf{M}_{TP}$ capturing **tag-page associations** (# users who put t on p)
- Iterative computation of importance vectors $\mathbf{r}_P$, $\mathbf{r}_T$, and $\mathbf{r}_U$ as

$$\mathbf{r}_U = \mathbf{M}_{PU}^T \, \mathbf{r}_P$$

$$\mathbf{r}_T = \mathbf{M}_{UT}^T \, \mathbf{r}_U$$

$$\mathbf{r}_P = \mathbf{M}_{TP}^T \, \mathbf{r}_T$$

  with renormalization after every iteration until convergence
- Full details: [Bao et al. ’07]

FolkRank

- Considers **pages** P, **tags** T, and **users** U
  - $\mathbf{M}_{PU}$ capturing **page-user associations** (# tags assigned by u to p)
  - $\mathbf{M}_{UT}$ capturing **user-tag associations** (# pages tagged by u with t)
  - $\mathbf{M}_{TP}$ capturing **tag-page associations** (# users who put t on p)
- Merges $\mathbf{M}_{PU}$, $\mathbf{M}_{UT}$, and $\mathbf{M}_{TP}$ into a single graph $G(V, E)$
- Assumes that each user has a **preference vector** $\mathbf{p}$
- Iterative computation of **importance vector** $\mathbf{r}$ as

$$\mathbf{r} = \alpha\,\mathbf{r} + \beta\,\mathbf{A}^T\,\mathbf{r} + \gamma\,\mathbf{r}$$

  with $\mathbf{A}$ as right-stochastic adjacency matrix of $G(V, E)$
- Full details: [Hotho et al. '06]

TunkRank

- Idea: Measure a Twitter user's influence as the **expected number of people who will read** a tweet (including re-tweets) by the user
- Considers **Twitter's follower graph** $G(V, E)$ consisting of users as vertices V and directed edges E with edge (i, j) indicating that user i follows user j
- Assumptions:
  - if i follows j, i reads tweet by j with probability $1 / out(i)$
  - constant **re-tweeting probability** p

$$r(j) = \sum_{(i,j) \in E} \frac{(1 + p \cdot r(i))}{|out(i)|}$$

- Full details: [Tunkelang '09]

TwitterRank

- Considers **Twitter's follower graph** $G(V, E)$ consisting of users as vertices V and directed edges E with edge (i, j) indicating that user i follows user j

- **PageRank-style random walk** with link-following probabilities

$$\mathbf{T}_{ij} = \begin{cases} \frac{|N_j|}{\sum_{(i,k)\in E} |N_k|} \cdot sim(i,j) & : \quad (i,j) \in E \\ 0 & : \quad \text{otherwise} \end{cases}$$

  with Ni as the **number of tweets** published by user i and $sim(i, j)$ reflecting **similarity** between tweets by i and j

- Extension considers **topics obtained by LDA** and factors them into random jump probabilities $\mathbf{j}_t$ and similarity $sim_t(i, j)$

- Full details: [Weng et al '10]

Summary of IV.8

- **Social networks**
  as complex graphs with diverse types of objects, diverse relations in-between, timestamps, and associated attributes

- **Link analysis methods**
  can be used to measure importance and similarity
  with applications in search and recommendation

Additional Literature for IV.8

- **S. Bao, G.-R. Xue, X. Wu, Y. Yu, B. Fei, and Z. Su**: *Optimizing web search using social annotations*, WWW 2007

- **N. Bronson et al.**: *TAO: Facebook's Distributed Data Store for the Social Graph*, USENIX ATC 2013

- **A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme**: *FolkRank: A Ranking Algorithm for Folksonomies*, LWA 2006

- **A. Kashyap, R. Amini, and V. Hristidis**: *SonetRank: leveraging social networks to personalize search*, CIKM 2012

- **D. Tunkelang**: *A Twitter Analog to PageRank*, 2009 http://thenoisychannel.com/2009/01/13/a-twitter-analog-to-pagerank/

- **J. Weng, E.-P. Lim, J. Jiang, and Q. He**: *TwitterRank: finding topic-sensitive influential twitterers*, WSDM 2010