# NADINE
# Part II

**UMIL**

# Software (M7)

# Open Tools

- WebGraph: representing and analyzing large compressed graphs; new parallel version
- fastutil: high-performance collections, foundations for WebGraph
- Used to analyze large crawls
- Used at Twente to perform experiments on large datasets
- BUbiNG: high-performance crawler (currently used by Information Technologies Institute in Thessaloniki, Greece in a EU project, Istituto per le Applicazioni del Calcolo di Roma, ENEA ICT DIVISION)
- Integrated with spam detector from Sztaki
- HyperBall: computing distance distributions and geometric centralities

# Creating large and diverse datasets (M9)

# Web datasets

- Large scale web crawls
- Target: 1B pages, ~100B links
- Single-source seed for better reproducibility, open-source high-performance crawler
- EU countries crawls for diversity inside the EU
- UK crawls for linguistic uniformity
- General shallow crawls for maximum diversity
- Additionally, collaboration to the creation of ClueWeb12 dataset

# Wiki datasets

- Several languages
- Freely available to the community in pre-packaged format
- WikipediaDocumentSequence: a single class (part of MG4J) that can transform a Wikipedia dump into a graph and a search engine over the content
- Basic statistics available from the site
- Foundation for the PLoS ONE paper (and others) with Toulouse
- http://wikirank.di.unimi.it/

# Understanding large graphs

# The problem

- Previous analysis on web graphs was flakey
- Unreproducible result, “eyeballing” on plots instead of statistical tests
- Spurred the “power-law” craze, but without actual foundations
- In collaboration with the Data and Web Science Group of the University of Mannheim, we unleashed the tools developed by NADINE for graph analysis

# The result

- The in-depth analysis of the largest available web graph (3.5 billion pages)
- Kolmogoroff–Smirnoff testing of power laws
- Several level of aggregation
- Published on the Web Science Track of WWW 2014, soon on Network Science.
- Andrei Broder mentioned the paper in his “crystal ball keynote” at WWW 2015 together with Clauset’s analysis of empirical power laws

# Ranking

- We published the first open ranking of the web
- The scale is unprecedented: we provide different rankings on the 100M hosts of a 3.5B pages crawl
- Available for browsing from a web site
- http://wwwranking.webdatacommons.org/
- Data downloadable (actually, we didn’t think anybody wanted it, but then some national libraries asked for it)

# Understanding ranking on (directed) networks (M13)

# The problem

- Understanding the correlation between different rankings
- Why results in information retrieval about exogenous rankings are so flakey?
- Taking care of ties is essential (indegree)
- Rank differences between important elements should be more relevant
- Large-scale target (whole graphs), not small sets of results

| Indegree | PageRank | Katz | Harmonic | Closeness |
|---|---|---|---|---|
| United States | United States | United States | United States | **Kharqan Rural District** |
| List of sovereign states | Animal | List of sovereign states | United Kingdom | **Talageh-ye Sofla** |
| Animal | List of sovereign states | United Kingdom | World War II | **Talageh-ye Olya** |
| England | France | France | France | **Greatest Remix Hits (Whigfield album)** |
| France | Germany | Animal | Germany | **Suzhou HSR New Town** |
| Association football | Association football | World War II | Association football | **Suzhou Lakeside New City** |
| United Kingdom | England | England | English language | **Mepirodipine** |
| Germany | India | Association football | China | **List of MPs ... M–N** |
| Canada | United Kingdom | Germany | Canada | **List of MPs ... O–R** |
| World War II | Canada | Canada | India | **List of MPs ... S–T** |
| India | Arthropod | India | **Latin** | **List of MPs ... U–Z** |
| Australia | Insect | Australia | World War I | **List of MPs ... J–L** |
| London | World War II | London | England | **List of MPs ... C** |
| Japan | Japan | Italy | Italy | **List of MPs ... F–I** |
| Italy | Australia | Japan | **Russia** | **List of MPs ... A–B** |
| Arthropod | Village | New York City | **Europe** | **List of MPs ... D–E** |
| Insect | Italy | English language | Australia | **Esmaili-ye Sofla** |
| New York City | Poland | China | **European Union** | **Esmaili-ye Olya** |
| English language | English language | Poland | **Catholic Church** | **Levels of organization (ecology)** |
| Village | **Nationa Reg. of Hist. Places** | World War I | London | **Jacques Moeschal (architect)** |

Table 1: Top 20 pages of the English version of Wikipedia following five different centrality measures.

| | Ind. | PR | Katz | Harm. | Cl. |
|---|---|---|---|---|---|
| Indegree | 1 | 0.75 | 0.90 | 0.62 | 0.55 |
| PageRank | 0.75 | 1 | 0.75 | 0.61 | 0.56 |
| Katz | 0.90 | 0.75 | 1 | 0.70 | 0.62 |
| Harmonic | 0.62 | 0.61 | 0.70 | 1 | 0.92 |
| Closeness | 0.55 | 0.56 | 0.62 | 0.92 | 1 |

| | Ind. | PR | Katz | Harm. | Cl. |
|---|---|---|---|---|---|
| Indegree | 1 | 0.31 | 0.63 | 0.24 | 0.06 |
| PageRank | 0.31 | 1 | 0.27 | 0.10 | 0.10 |
| Katz | 0.63 | 0.27 | 1 | 0.50 | 0.20 |
| Harmonic | 0.24 | 0.10 | 0.50 | 1 | 0.65 |
| Closeness | 0.06 | 0.10 | 0.20 | 0.65 | 1 |

| Indegree | PageRank | Katz | Harmonic | Closeness |
|---|---|---|---|---|
| Carl Linnaeus | Carl Linnaeus | Carl Linnaeus | Aristotle | **Noël Bernard (botanist)** |
| Aristotle | Aristotle | Aristotle | Albert Einstein | **Charles Coquelin** |
| Thomas Jefferson | Thomas Jefferson | Thomas Jefferson | Thomas Jefferson | **Markku Kivinen** |
| Margaret Thatcher | Charles Darwin | Albert Einstein | Charles Darwin | **Angiolo Maria Colomboni** |
| Plato | Plato | Charles Darwin | Thomas Edison | **Om Prakash (historian)** |
| Charles Darwin | Albert Einstein | Karl Marx | **Alexander Graham Bell** | **Michel Mandjes** |
| Karl Marx | Karl Marx | Plato | **Nikola Tesla** | **Kees Posthumus** |
| Albert Einstein | Pliny the Elder | Margaret Thatcher | **William James** | **F. Wolfgang Schnell** |
| Vladimir Lenin | Vladimir Lenin | Vladimir Lenin | Isaac Newton | **Christof Ebert** |
| Sigmund Freud | Johann Wolfgang von Goethe | Isaac Newton | Karl Marx | **Reese Prosser** |
| J. R. R. Tolkien | Margaret Thatcher | Ptolemy | **Charles Sanders Peirce** | **David Tulloch** |
| Johann Wolfgang von Goethe | Ptolemy | Johann Wolfgang von Goethe | Noam Chomsky | **Kim Hawtrey** |
| **Spider-Man** | Sigmund Freud | Pliny the Elder | **Enrico Fermi** | **Patrick J. Miller** |
| Pliny the Elder | Isaac Newton | Benjamin Franklin | Ptolemy | **Mikel King** |
| Benjamin Franklin | Benjamin Franklin | J. R. R. Tolkien | **John Dewey** | **Albert Perry Brigham** |
| Leonardo da Vinci | J. R. R. Tolkien | Thomas Edison | Johann Wolfgang von Goethe | **Gordon Wagner (economist)** |
| Isaac Newton | Immanuel Kant | Sigmund Freud | **Bertrand Russell** | **George Henry Chase** |
| Ptolemy | Leonardo da Vinci | Immanuel Kant | Plato | **Charles C. Horn** |
| Immanuel Kant | **Pierre André Latreille** | Leonardo da Vinci | **John von Neumann** | **Paul Goldstene** |
| **George Bernard Shaw** | Thomas Edison | Noam Chomsky | Vladimir Lenin | **Robert Stanton Avery** |

| Indegree | PageRank | Katz | Harmonic | Closeness |
|---|---|---|---|---|
| Martini (cocktail) | Martini (cocktail) | Irish coffee | Irish coffee | **Magie Noir** |
| Piña colada | Caipirinha | Caipirinha | Caipirinha | **Batini (drink)** |
| Mojito | Mojito | Martini (cocktail) | Kir (cocktail) | **Scorpion bowl** |
| Caipirinha | Piña colada | Piña colada | Martini (cocktail) | **Poinsettia (cocktail)** |
| Cuba Libre | Irish coffee | Kir (cocktail) | Piña colada | Irish coffee |
| Irish coffee | Kir (cocktail) | Mojito | Mojito | Caipirinha |
| Singapore Sling | Cosmopolitan (cocktail) | Mai Tai | Beer cocktail | Kir (cocktail) |
| Manhattan (cocktail) | Manhattan (cocktail) | Cuba Libre | Shaken, not stirred | Martini (cocktail) |
| Windle (sidecar) | IBA Official Cocktail | Singapore Sling | Pisco Sour | Piña colada |
| Cosmopolitan (cocktail) | Beer cocktail | Long Island Iced Tea | Mai Tai | Mojito |
| Mai Tai | Mai Tai | Shaken, not stirred | Spritz (alcoholic beverage) | Beer cocktail |
| IBA Official Cocktail | Singapore Sling | Beer cocktail | Long Island Iced Tea | Shaken, not stirred |
| Kir (cocktail) | Cuba Libre | Manhattan (cocktail) | Sazerac | Mai Tai |
| Shaken, not stirred | **Tom Collins** | Cosmopolitan (cocktail) | Fizz (cocktail) | Spritz (alcoholic beverage) |
| Beer cocktail | Long Island Iced Tea | Windle (sidecar) | Flaming beverage | Pisco Sour |
| Pisco Sour | Sour (cocktail) | Pisco Sour | Cuba Libre | Long Island Iced Tea |
| Long Island Iced Tea | Shaken, not stirred | White Russian (cocktail) | Wine cocktail | Sazerac |
| Sour (cocktail) | **Negroni** | IBA Official Cocktail | Singapore Sling | Flaming beverage |
| White Russian (cocktail) | Flaming beverage | Moscow mule | Moscow mule | Fizz (cocktail) |
| Vesper (cocktail) | **Lillet** | Vesper (cocktail) | White Russian (cocktail) | Wine cocktail |

# Kendall's τ 1938

- Score vectors ***r***, ***s***
- Concordances: pairs (*i*, *j*), *i* < *j*, such that the ranks for *i* and *j* in ***r*** and ***s*** are in the same order (assuming no ties)
- $\tau$: Concordances minus discordances divided by concordances plus discordances (i.e., the number of ordered pairs)
- Ties cannot be solved by random assignment!
- <0,0,0,...,1,1,1,...> and <1,1,1,...,2,2,2,...> give correlation ≈0.5!

# Kendall's τ 1945

$$\langle \boldsymbol{r}, \boldsymbol{s} \rangle := \sum_{i<j} \operatorname{sgn}(r_i - r_j) \operatorname{sgn}(s_i - s_j)$$

$$\|\boldsymbol{r}\| := \sqrt{\langle \boldsymbol{r}, \boldsymbol{r} \rangle}$$

$$\tau(\boldsymbol{r}, \boldsymbol{s}) := \frac{\langle \boldsymbol{r}, \boldsymbol{s} \rangle}{\|\boldsymbol{r}\| \cdot \|\boldsymbol{s}\|}$$

# Now

$$\langle \boldsymbol{r}, \boldsymbol{s} \rangle_w := \sum_{i<j} \operatorname{sgn}(r_i - r_j) \operatorname{sgn}(s_i - s_j) w(i, j)$$

$$\tau_w(\boldsymbol{r}, \boldsymbol{s}) := \frac{\langle \boldsymbol{r}, \boldsymbol{s} \rangle_w}{\|\boldsymbol{r}\|_w \cdot \|\boldsymbol{s}\|_w}$$

$$|\langle \boldsymbol{r}, \boldsymbol{s} \rangle_w| \leq \|\boldsymbol{r}\|_w \|\boldsymbol{s}\|_w$$

# Computable Quickly

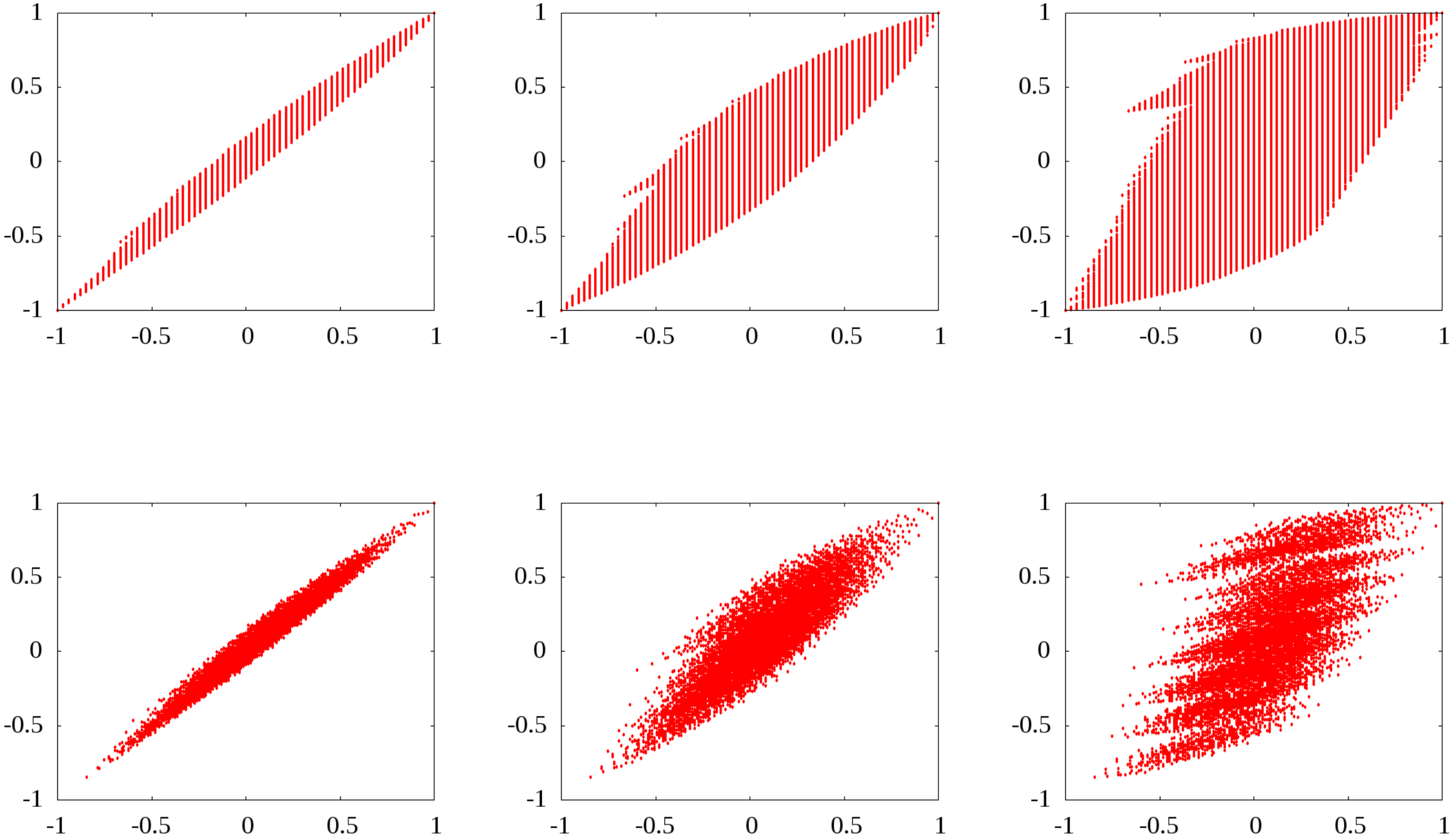
- $O(n \log n)$ variant of Knight's algorithm (highly parallelizable, distributable—it's a MergeSort)

- Open-source implementation in Java (scales to billion items)

- Works for any scheme $w(i, j) := f(i) \odot g(j)$ with suitable operation $\odot$ (e.g., addition, multiplication)

- We suggest additive hyperbolic weighting, weighting $(i, j)$ by$1 / (i + 1) + 1 / (j + 1)$: $\tau_h$!

# Correlation with Kendall's τ



# Wikipedia

| | Ind. | PR | Katz | Harm. | Cl. |
|---|---|---|---|---|---|
| Indegree | 1 | 0.75 | 0.90 | 0.62 | 0.55 |
| PageRank | 0.75 | 1 | 0.75 | 0.61 | 0.56 |
| Katz | 0.90 | 0.75 | 1 | 0.70 | 0.62 |
| Harmonic | 0.62 | 0.61 | 0.70 | 1 | 0.92 |
| Closeness | 0.55 | 0.56 | 0.62 | 0.92 | 1 |

| | Ind. | PR | Katz | Harm. | Cl. |
|---|---|---|---|---|---|
| Indegree | 1 | 0.95 | 0.98 | 0.90 | 0.27 |
| PageRank | 0.95 | 1 | 0.96 | 0.92 | 0.65 |
| Katz | 0.98 | 0.96 | 1 | 0.93 | 0.26 |
| Harmonic | 0.90 | 0.92 | 0.93 | 1 | 0.28 |
| Closeness | 0.27 | 0.65 | 0.26 | 0.28 | 1 |

Table 6: $\tau_h$ on Wikipedia.

# Voting in social networks (M12)

# Recommendation by voting

- Recommendation happens usually by some form of *collaborative filtering*

- LiquidFM is a Facebook application that tries to shift the power to the users

- The basic underlying mechanism is that of a *liquid democracy*, in which users can take decisions or delegate them

# Viscous democracy

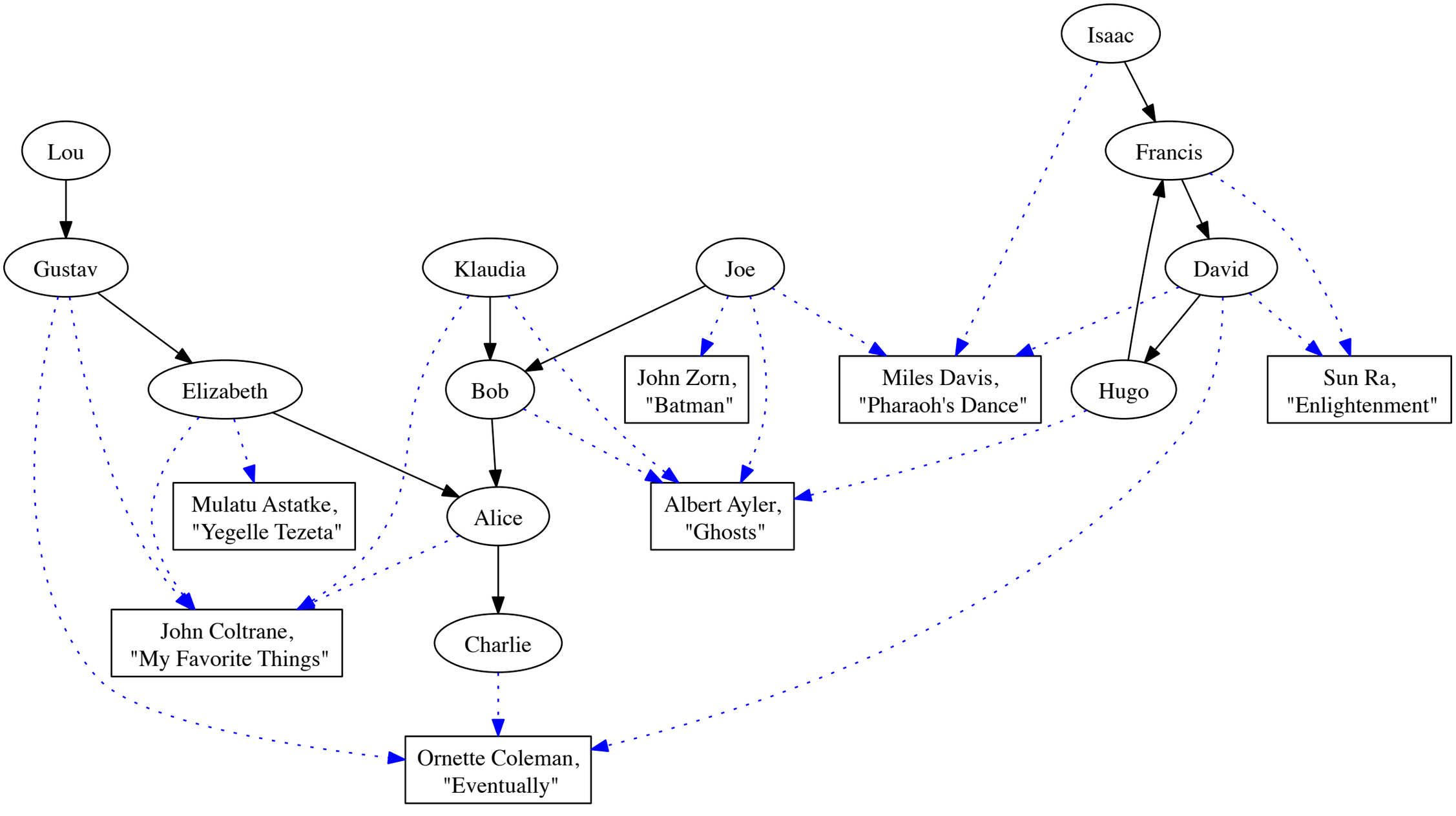
- In pure liquid democracy, votes are transferred exactly

- *Viscous democracy* has been proposed by Boldi *et al.* [CACM 2011] as a way to introduce some *friction* in vote transmission

- The idea is that a vote will conserve just a fraction α of its power when it is transferred to a delegate

# Basic setting

- We have a set of user *U* and a set of songs *S*
- There is an underlying *friendship graph* having *U* as set of nodes, i.e., $F \subseteq U \times U$
- Every user expresses votes for some song
- Every user can *delegate* at most a friend as an expert, giving rise to a *delegation graph* $D \subseteq F$


Lou
Gustav
Elizabeth
Klaudia
Joe
Bob
John Zorn,
"Batman"
Isaac
Francis
David
Miles Davis,
"Pharaoh's Dance"
Hugo
Sun Ra,
"Enlightenment"
Mulatu Astatke,
"Yegelle Tezeta"
Alice
Albert Ayler,
"Ghosts"
John Coltrane,
"My Favorite Things"
Charlie
Ornette Coleman,
"Eventually"

# Computing votes

- In liquid democracy, one assumes that there are no cycles and the "power" of a user is simply the size of its in-tree

- In viscous democracy, a vote traveling $k$ hops has weight $\alpha^k$

- The score of a user $u$ is thus $\sum_v \alpha^{-d(v,u)}$

- Note that this is just Katz's index (or PageRank); actually, cycles are possible and the formula becomes an infinite path sum

- The score of a song is the sum of the scores of the users voting it

# Implementation

- Katz's index computed in Java (periodically)
- MongoDB to store data
- A MusicBrainz local server to provide suggestions and unique references to music
- The resulting score is used in convex combination with the global score
- http://bit.ly/liquidfm

# Main problem

- Not surprisingly: *user engagement*
- Chicken-and-egg: if LiquidFM was famous, people would like to have an “expert” label
- Without that, people have no incentive to add delegations and suggestions
- This is particularly bad for the “active” nature of the recommendation

# On the positive side

- High privacy: you decide what to make visible of your music taste

- High serendipity: even in our small set of user (a hundred) it is evident that people tend to insert songs that are not “obvious”

- (Actually, there are a few records that entered my listening list from LiquidFM.)

# Foundations for Monte–Carlo (and randomized) algorithms (M10)

# A new, old family of PRNGs

- Most algorithmic software used in NADINE is Monte–Carlo or randomized in nature
- Such software needs very fast PRNGs of high quality
- In some cases (e.g., generating random permutation) the PRNG cost can be dominant
- Current available generators suffer unfortunately from an “academic slant” syndrome

# Developing new PRNGs

- Starting point: Marsaglia's well-known xorshift family
- Almost trivial generators using just three shifts and three xors (which reflect linear operations on **Z** / 2**Z**)
- Need just a little "bump" to hide linear artefacts
- xorshift* generators multiply by a constant
- xorshift+ generators add part of the state

# Fast & good

- xorshift128+ is the fastest known generator passing the BigCrush statistical test suite

- Scheduled to be the new generator of the Erlang and Julia language (actually, xorshift112+)

- xorshift1024+ offers a quality superior to the Mersenne Twister or WELL1024 at twice the speed

```
uint64_t s[ 2 ];

uint64_t next(void) {
   uint64_t s1 = s[ 0 ];
   const uint64_t s0 = s[ 1 ];
   s[ 0 ] = s0;
   s1 ^= s1 << 23; // a
   return ( s[ 1 ] = ( s1 ^ s0 ^ ( s1 >> 17 ) ^ ( s0 >> 26 ) ) ) + s0; // b, c
}
```

```
#define W 32
#define R 32
#define M1 3
#define M2 24
#define M3 10

#define MAT0POS(t,v) (v^(v>>t))
#define MAT0NEG(t,v) (v^(v<<(-(t))))
#define Identity(v) (v)

#define V0            STATE[state_i                   ]
#define VM1           STATE[(state_i+M1) & 0x0000001fU]
#define VM2           STATE[(state_i+M2) & 0x0000001fU]
#define VM3           STATE[(state_i+M3) & 0x0000001fU]
#define VRm1          STATE[(state_i+31) & 0x0000001fU]
#define newV0         STATE[(state_i+31) & 0x0000001fU]
#define newV1         STATE[state_i                   ]

static unsigned int state_i = 0;
static unsigned int STATE[R];
static unsigned int z0, z1, z2;

static unsigned long int next( void *unused0, void *unused1 ) {
  z0    = VRm1;
  z1    = Identity(V0)       ^ MAT0POS (8, VM1);
  z2    = MAT0NEG (-19, VM2) ^ MAT0NEG(-14,VM3);
  newV1 = z1                 ^ z2;
  newV0 = MAT0NEG (-11,z0)   ^ MAT0NEG(-7,z1)    ^ MAT0NEG(-13,z2) ;
  state_i = (state_i + 31) & 0x0000001fU;
  return REV( STATE[state_i] );
}
```

# Analysis and prediction on directed networks

# LlamaFur

- Wikipedia = Directed knowledge base
- Wikipedia pages (*concepts*) are tagged by *categories*
- There exists a latent (unknown) relation between categories

# LlamaFur: phase 1

- Llama = Learning Latent Matrix
- Extract a latent Tag x Tag **category matrix *W*** that "explains" wikipedia links
    - E.g.: actor → movie is *typical* because many actor pages link to the movies they acted in
- We use the Passive/Aggressive learning algorithm

# LlamaFur: phase 1

+1/-1 depending on whether it is a positive or negative example

The categories tagging the concept

positive and negative examples (i.e., existent
en concepts) is built:

$$(d_1, d'_1), \ldots, (d_T, d'_T)$$

$$\xi_{t+1}$$

- subject to

$$\sigma(d_t, d'_t) \cdot \sum_{c \in C_{d_t}} \sum_{c' \in C_{d'_t}} w_{t+1}(c, c') \geq 1 - \xi_{t+1}$$

- Objective function: keep some memory of the past

- Constraint: learn correctly the *t*-th pair ( $\xi_t$ allows for some error)

# LlamaFur: phase 2

- Fur = to Find Unexpected Relations
- Use *W* to assign a score of *expectedness* to each link of the original knowledge base
- E.g.: did you know that George Clooney used to have a pig pet named Oscar?
- Applications: better ranking algorithms, diversifying search results, building restricted knowledge bases for serendipitous search

# Local Ranking on the BrowseGraph

- **Local Ranking Problem** (LRP): divergence between PageRank computed on a known subgraph (local) and that computed on the large unknown graph (global)

- Here: study the problem on the BrowseGraph (Liu *et al.*, SIGIR 2008)

- BrowseGraph: a weighted graph that reflects the users' transition among pages (of a given portion of the web)

# Entry points

- Users enter the domain of interest from different **entry points** (e.g., from a search engine, from facebook, from a news website)
- Each entry point defines a different BrowseGraph
- How much do they look alike?

# Rank comparison

| | full | facebook | google | bing | ysearch | reddit | ymy | ynews | twitter |
|---|---|---|---|---|---|---|---|---|---|
| full | 1.0000 | 0.1791 | 0.3931 | 0.3278 | 0.3548 | 0.0656 | 0.2931 | 0.2797 | 0.0764 |
| facebook | 0.1791 | 1.0000 | 0.3146 | 0.4111 | 0.3430 | 0.2616 | 0.3895 | 0.4070 | 0.3026 |
| google | 0.3931 | 0.3146 | 1.0000 | 0.5815 | 0.5860 | 0.1088 | 0.4960 | 0.4217 | 0.1297 |
| bing | 0.3278 | 0.4111 | 0.5815 | 1.0000 | 0.6624 | 0.1469 | 0.5741 | 0.5238 | 0.1688 |
| search | 0.3548 | 0.3430 | 0.5860 | 0.6624 | 1.0000 | 0.1245 | 0.5168 | 0.4632 | 0.1386 |
| reddit | 0.0656 | 0.2616 | 0.1088 | 0.1469 | 0.1245 | 1.0000 | 0.1513 | 0.1534 | 0.2309 |
| my | 0.2931 | 0.3895 | 0.4960 | 0.5741 | 0.5168 | 0.1513 | 1.0000 | 0.5040 | 0.1506 |
| news | 0.2797 | 0.4070 | 0.4217 | 0.5238 | 0.4632 | 0.1534 | 0.5040 | 1.0000 | 0.1523 |
| t.co | 0.0764 | 0.3026 | 0.1297 | 0.1688 | 0.1386 | 0.2309 | 0.1506 | 0.1523 | 1.0000 |

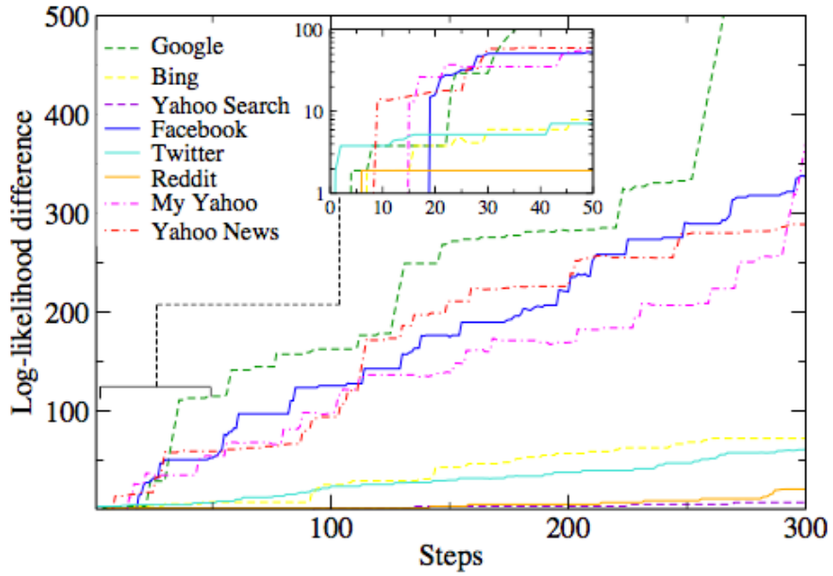
Table 3: Kendall's $\tau$ correlations between PageRank values ($\alpha = 0.85$).

Poor correlation between local ranks!

⇒ The behaviour of users is *different* depending on the entry point

# Prediction

- Is it possible to guess the entry point, observing the user's behaviour?




- After observing 5 → 15 steps, the entry point can be guessed with extremely high accuracy
- Once more, the accuracy depends on the entry point

# Conclusion

- Software tools in collaboration with other nodes
- Many new open datasets for the community, used throughout the project
- Entirely generated and ranked by open-source software
- Significantly deeper understanding of the structure of web graphs
- Open Wikipedia ranking by category / Open WWW ranking
- Facebook app for voting using spectral graph algorithms
- New algorithms to predict links and behaviour on directed graphs (Wikipedia, BrowseGraph)