Announcements:

- Today: ~~HW2 due~~ / HW3 release
  - All homeworks, including HW2, will be due Sunday 23:59pm **(no late periods)**
  - Or think of it as 4 free late days for everyone per homework
  - We will do our best to grade even faster!
  - We moved a OH to Friday but there will not be a weekend OH.
- Colab 4 due date extended to Sunday night (was not on Gradescope initially)
  - Same for future colabs
- Watch out for homework 2 feedback poll
- Project proposals – TAs will reach out with feedback
- Ed: Green checkmark means instructors “endorse” the answer
- Sun May 9 – Project Milestone deadline extended by three days (make sure to have dataset in hand/disk and demonstrate preliminary efforts)

Link Analysis: TrustRank and Web Spam

CSEP590A Machine Learning for Big Data

Tim Althoff

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

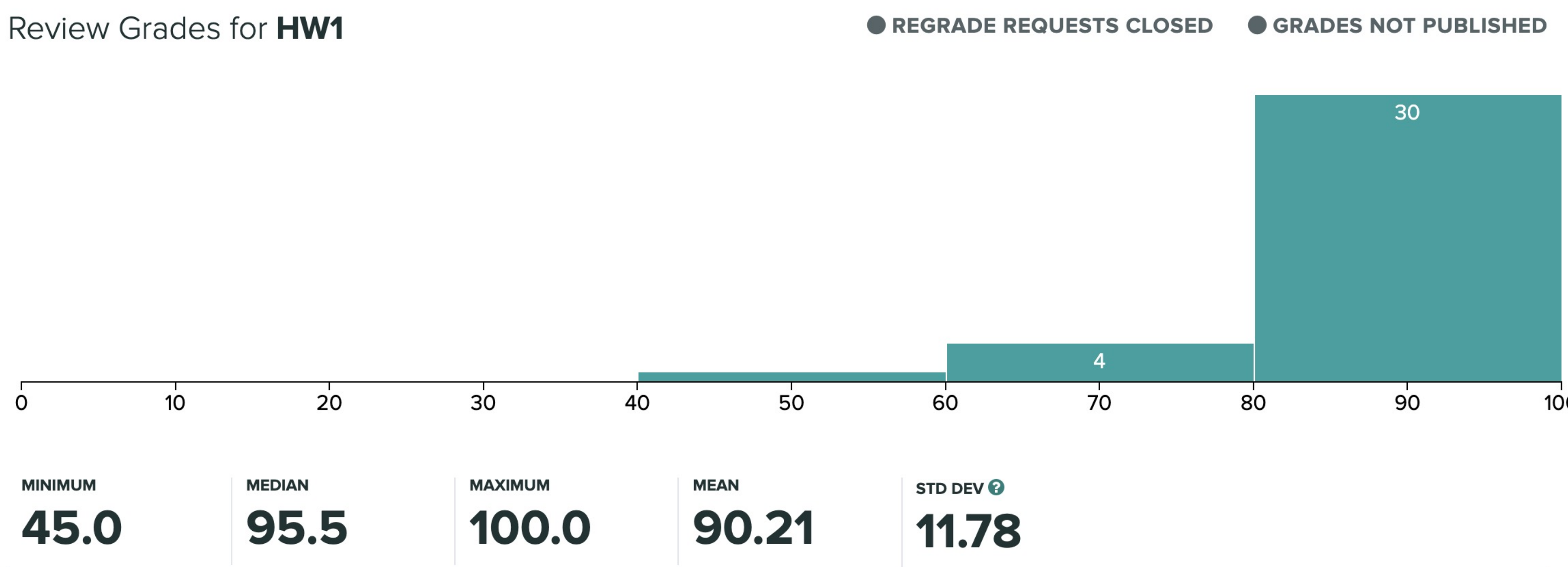
Midterm feedback

- Positive: lectures, slides, lecture recordings, office hours, homeworks, collabs, suggested (optional) reading help you learn

- Negative: workload

- Action:
  - All colabs and hw deadlines extended + OH change
  - We have dropped one problem from HW3 and HW4 relative to previous years
  - We replaced problems with more practice oriented simpler, shorter problems.
  - Checked in with Allen School and PMP leadership – recommendation was not to make any further changes

Grades so far



You do not need to get 100/100! It's okay not to finish everything.

Last year average homework was 85% -- at this average everyone got good grades

Project proposal grades also quite high!

New Topic: Graph Data!

High dim. data
- Locality sensitive hashing
- Clustering
- Dimensionality reduction

Graph data
- **PageRank,** SimRank
- Community Detection
- Spam Detection

Infinite data
- Sampling data streams
- Filtering data streams
- Queries on streams

Machine learning
- SVM
- Decision Trees
- Perceptron, kNN

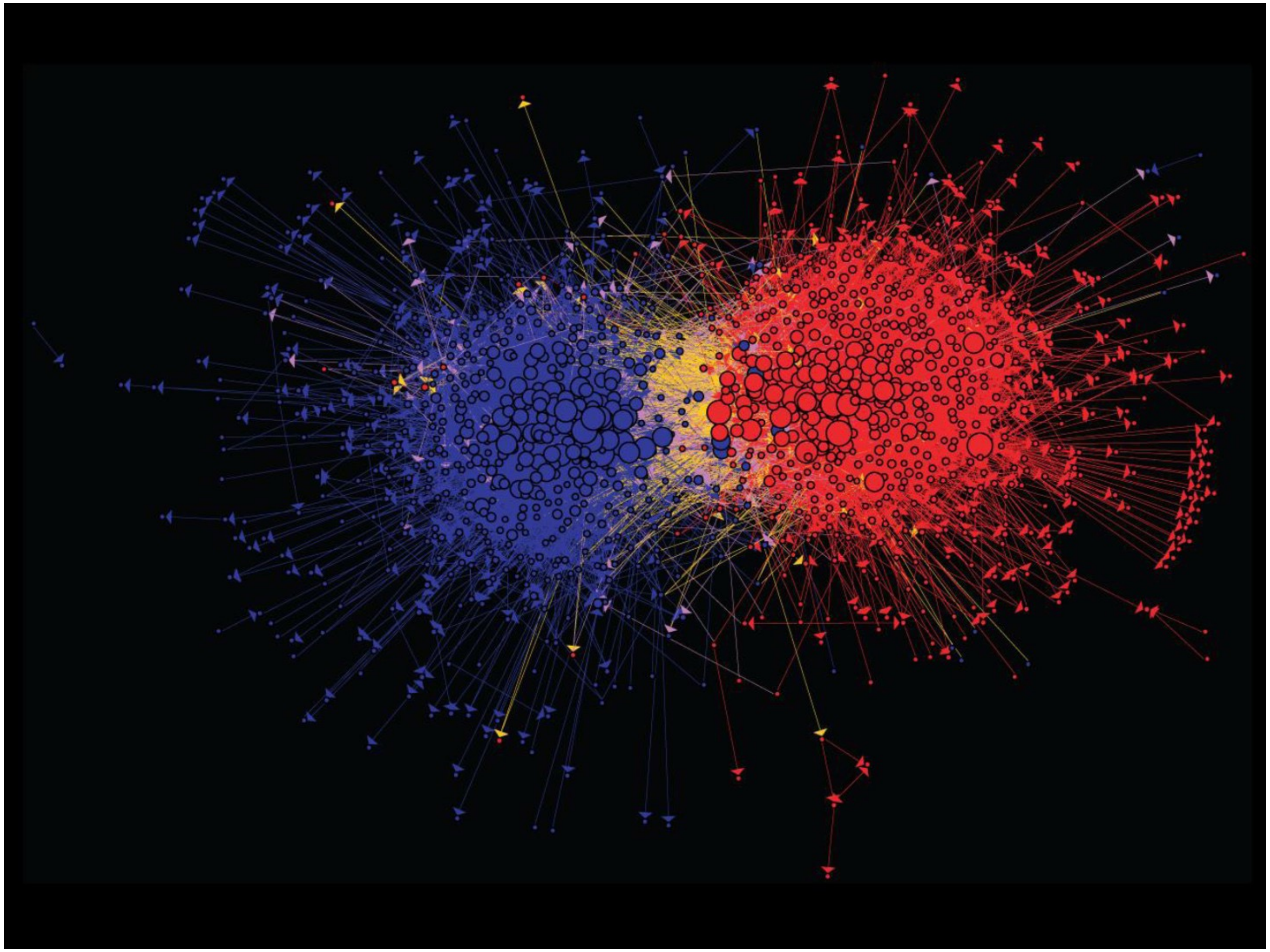
Apps
- Recommender systems
- Association Rules
- Duplicate document detection

Graph Data: Social Networks

Facebook social graph

4-degrees of separation [Backstrom-Boldi-Rosa-Ugander-Vigna, 2011]

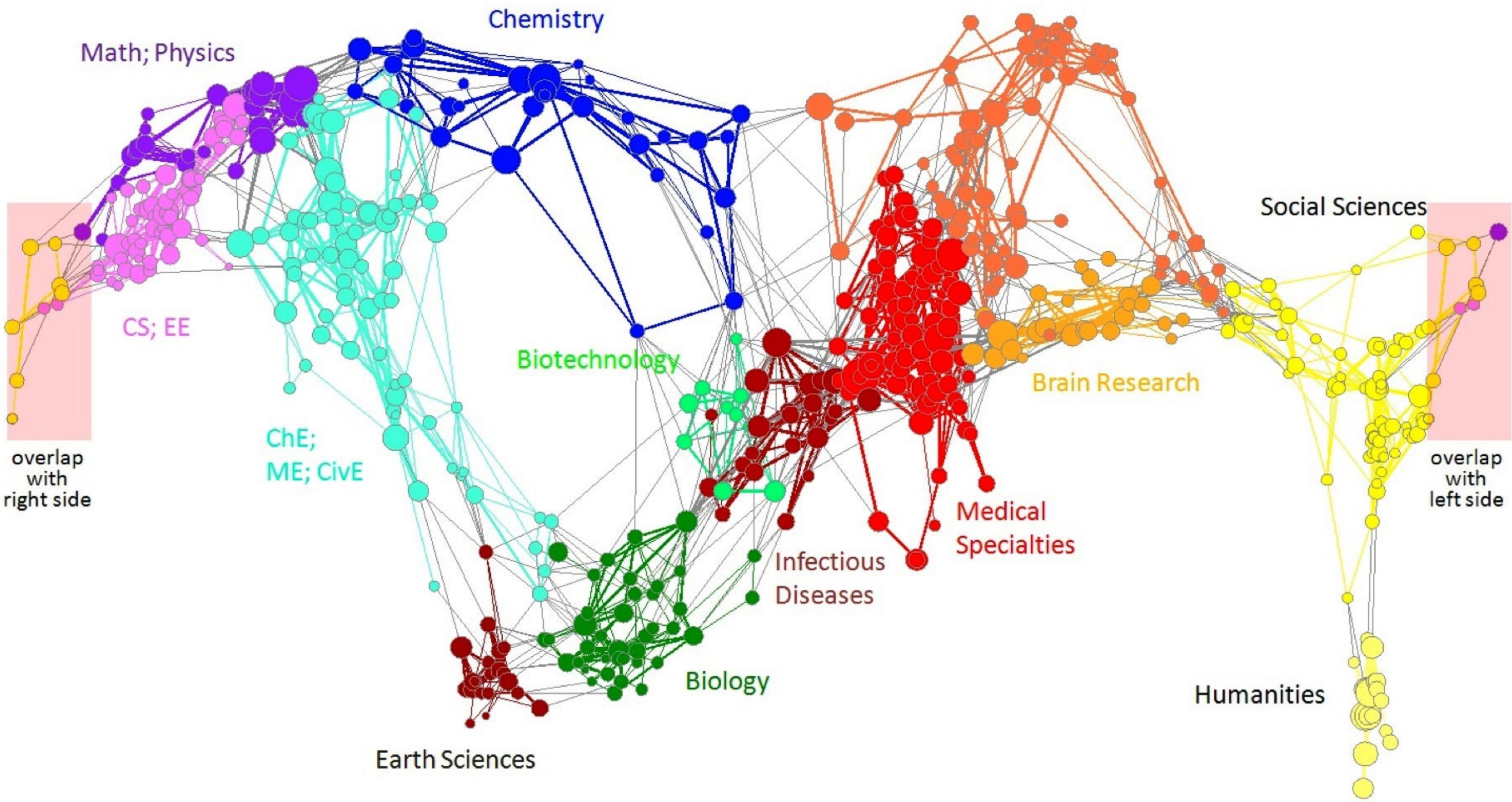
Graph Data: Media Networks



Connections between political blogs

Polarization of the network [Adamic-Glance, 2005]

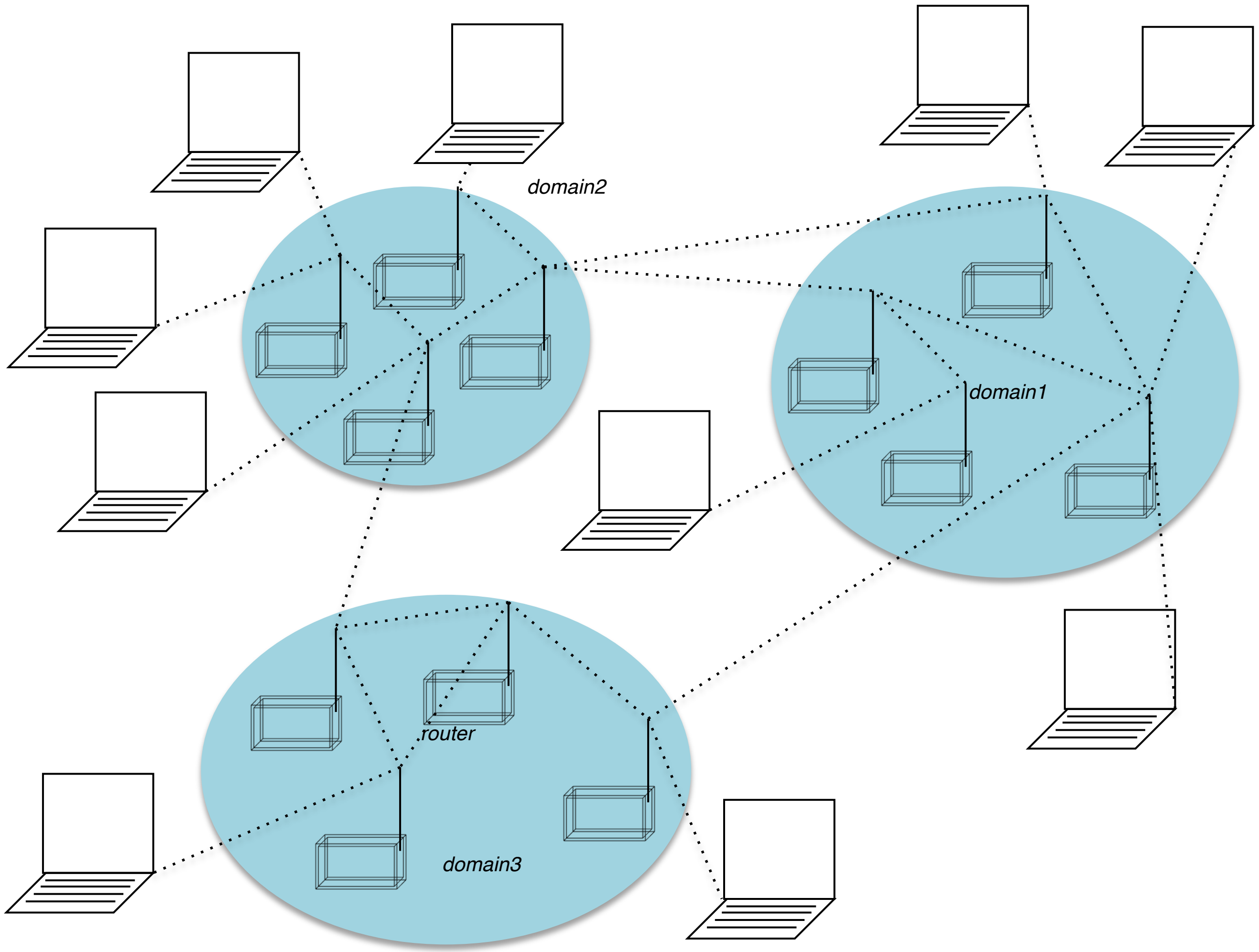
Graph Data: Information Nets



Citation networks and Maps of science

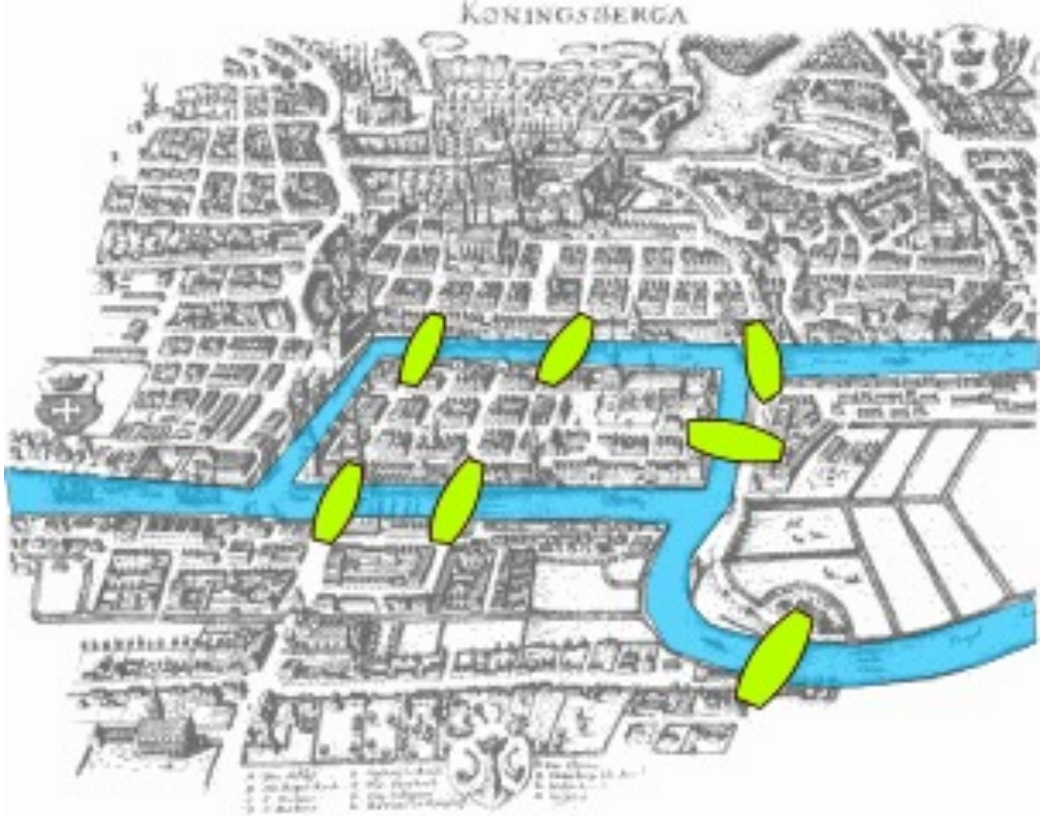
[Börner et al., 2012]

Graph Data: Communication Networks



Internet

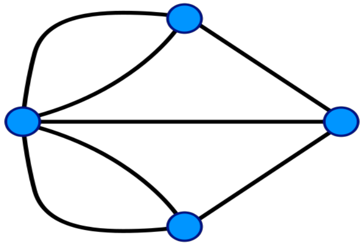
Graph Data: Technological Networks



Seven Bridges of Königsberg

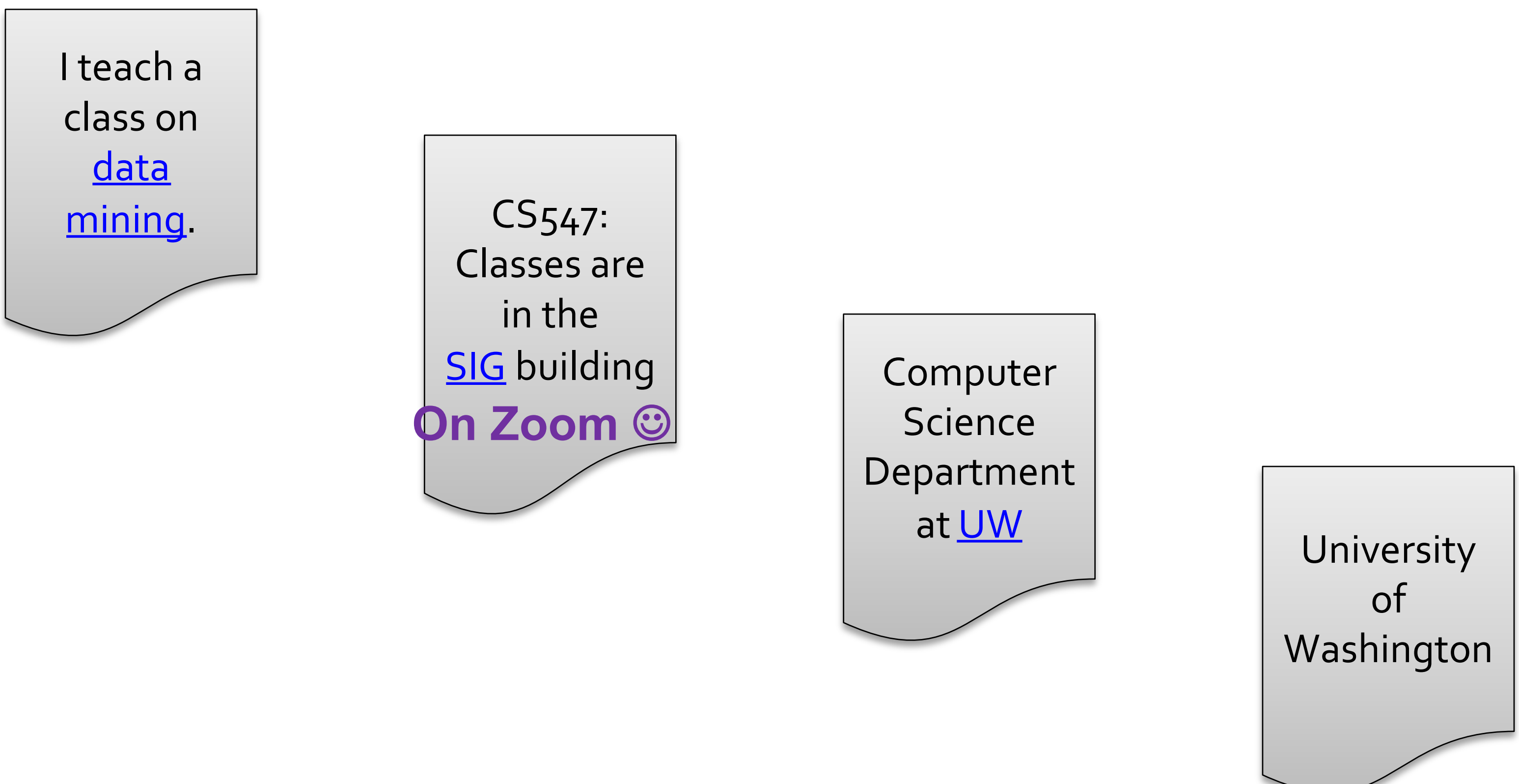
[Euler, 1735]

Return to the starting point by traveling each link of the graph once and only once.



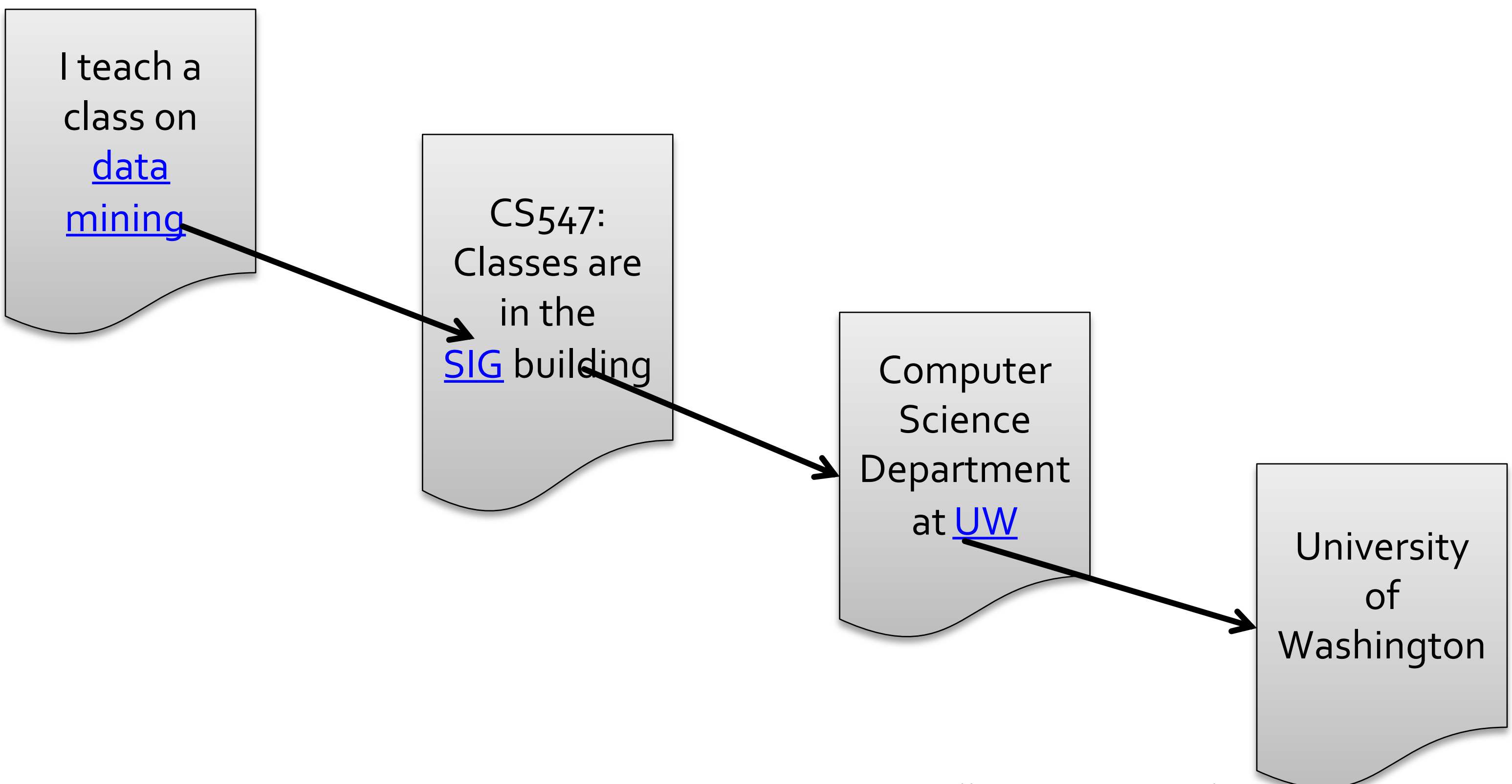
Web as a Graph

- **Web as a directed graph:**
  - **Nodes: Webpages**
  - **Edges: Hyperlinks**

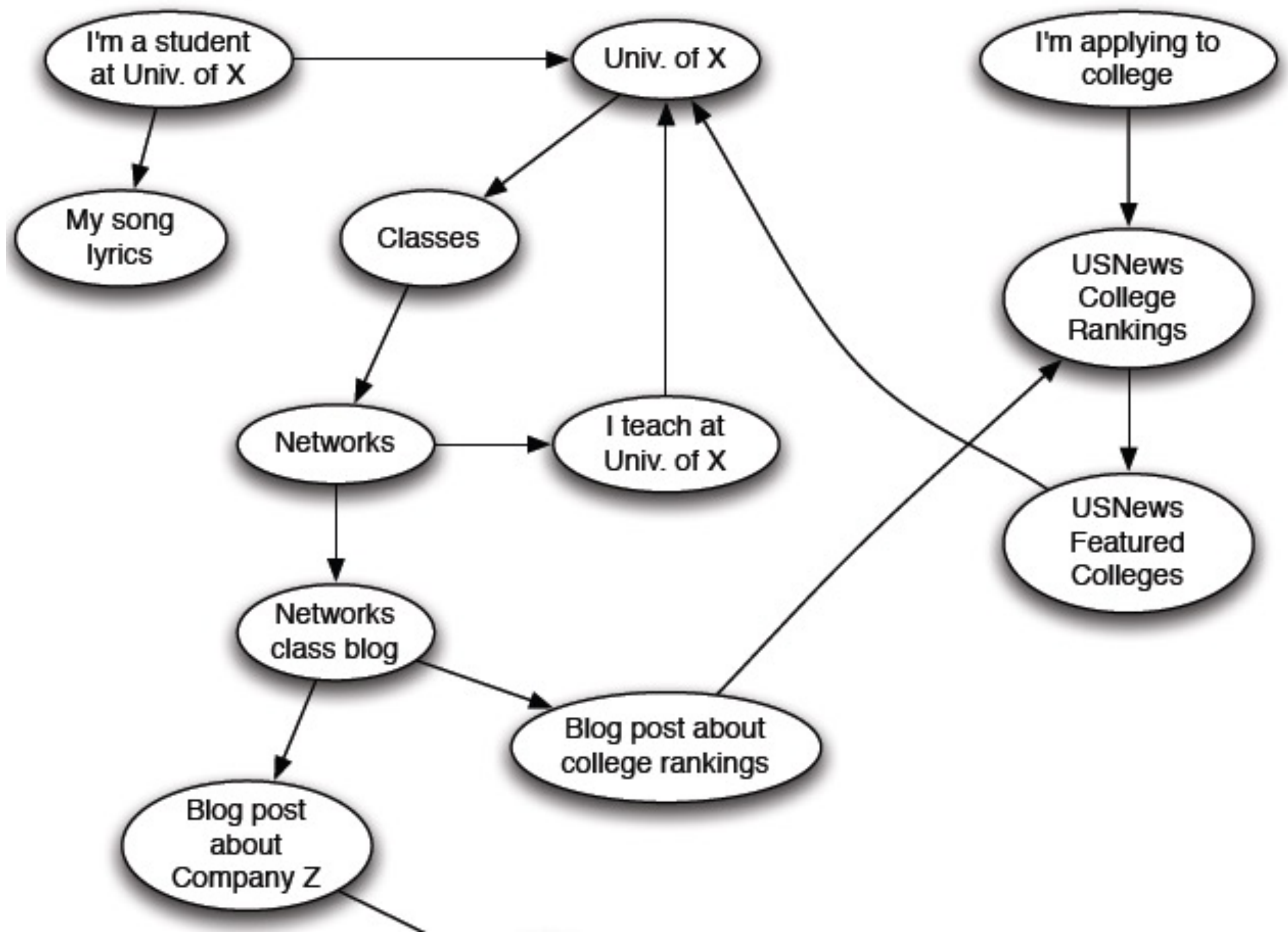


Web as a Graph

- **Web as a directed graph:**
  - **Nodes: Webpages**
  - **Edges: Hyperlinks**



Web as a Directed Graph



Broad Question

- **How to organize the Web?**
- **First try:** Human curated **Web directories**
  - Yahoo, DMOZ, LookSmart
- **Second try: Web Search**
  - **Information Retrieval** investigates: Find relevant docs in a small and trusted set
    - Newspaper articles, Patents, etc.
  - **But:** Web is **huge**, full of untrusted documents, random things, web spam, etc.

Web Search: 2 Challenges

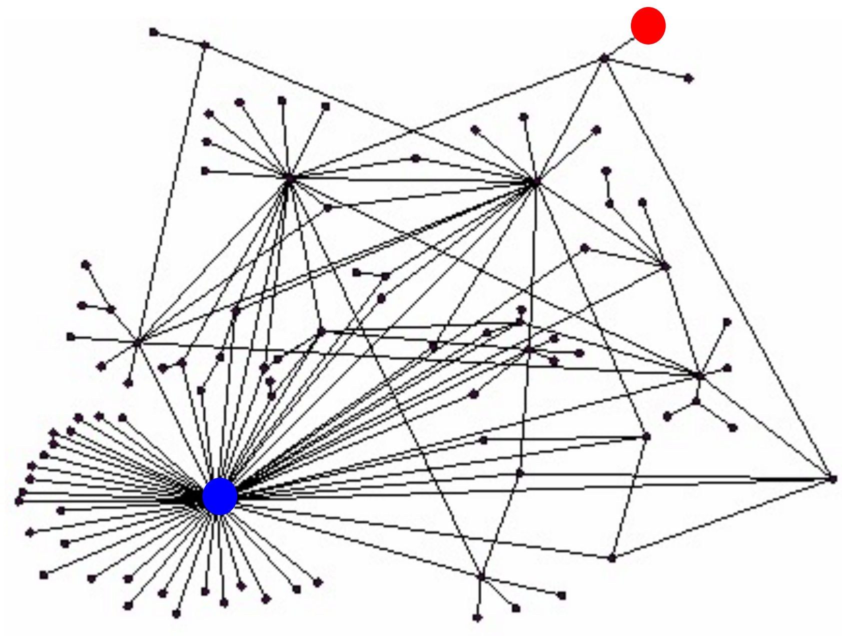
2 challenges of web search:

- **(1) Web contains many sources of information**
  Who to "trust"?
  - **Trick:** Trustworthy pages may point to each other!
- **(2) What is the "best" answer to query "newspaper"?**
  - No single right answer
  - **Trick:** Pages that actually know about newspapers might all be pointing to many newspapers

Ranking Nodes on the Graph

- **All web pages are not equally "important"**
  thispersondoesnotexist.com vs. www.uw.edu

- There is a large diversity in the web-graph node connectivity.
  Let's rank the pages by the link structure!



Link Analysis Algorithms

- We will cover the following **Link Analysis approaches** for computing **importances** of nodes in a graph:
  - Page Rank
  - Topic-Specific (Personalized) Page Rank
  - Web Spam Detection Algorithms

PageRank:
The “Flow” Formulation

Links as Votes

- **Idea: Links as votes**
  - **Page is more important if it has more links**
    - In-coming links? Out-going links?
- **Think of in-links as votes:**
  - www.uw.edu has **millions** in-links
  - thispersondoesnotexist.com has a **few hundreds (?)** in-links
- **Are all in-links equal?**
  - **Links from important pages count more**
  - Recursive question!

Intuition – (1)

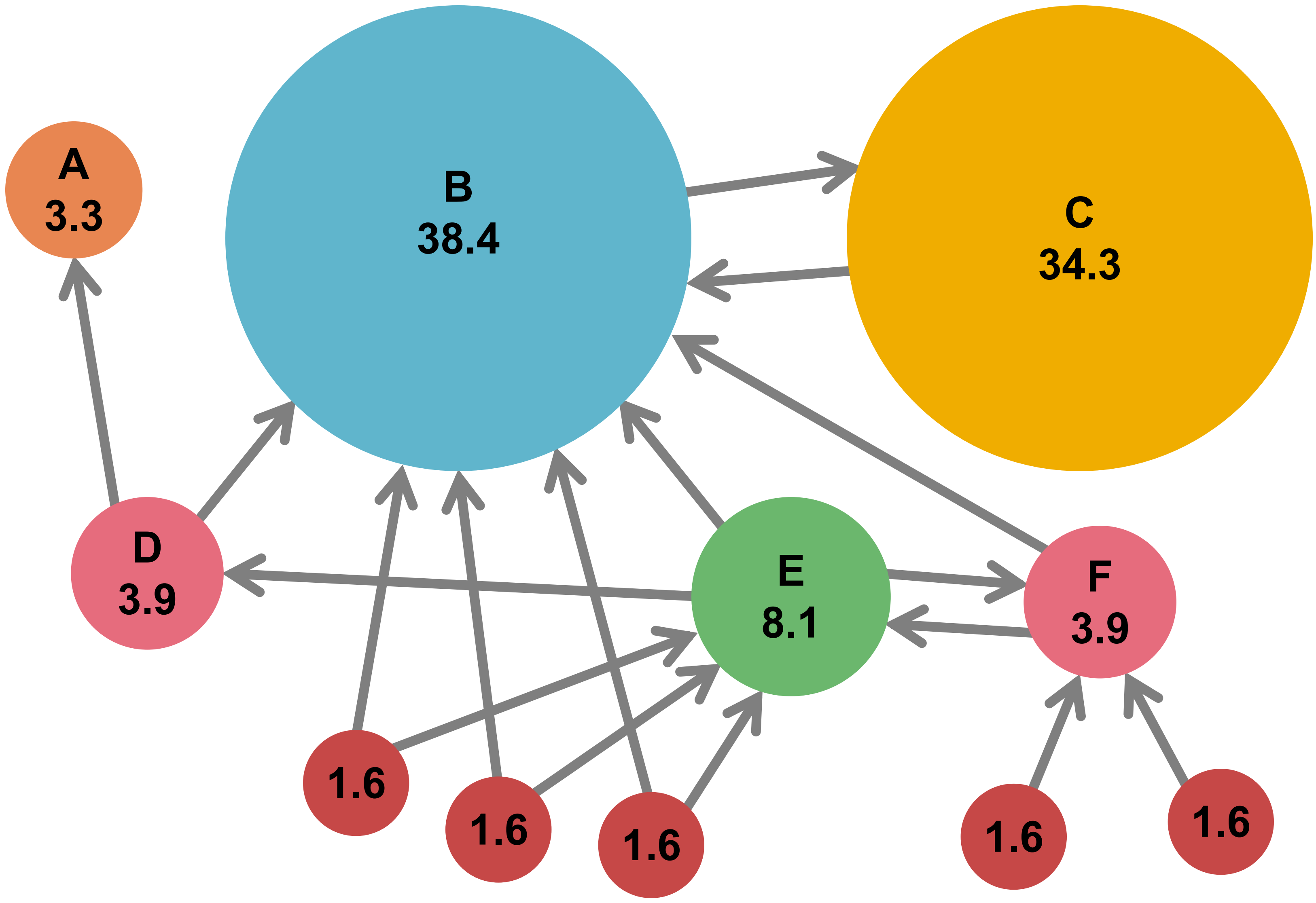
- Web pages are important if people visit them a lot.
- But we can't watch everybody using the Web.
- A good surrogate for visiting pages is to assume people follow links randomly.
- Leads to *random surfer* model:
  - Start at a random page and follow random out-links repeatedly, from whatever page you are at.
  - *PageRank* = limiting probability of being at a page.

Intuition – (2)

- **Solve the recursive equation:** “importance of a page = its share of the importance of each of its predecessor pages”
  - Equivalent to the random-surfer definition of PageRank

- Technically, *importance* = the principal eigenvector of the transition matrix of the Web
  - A few fix-ups needed

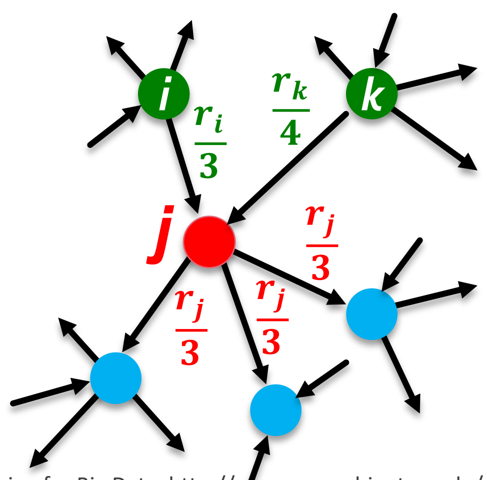
Example: PageRank Scores



Simple Recursive Formulation

- Each link's vote is proportional to the **importance** of its source page
- If page j with importance r_j has n out-links, each link gets $\frac{r_j}{n}$ votes
- Page j's own importance is the sum of the votes on its in-links

$$r_j = \frac{r_i}{3} + \frac{r_k}{4}$$



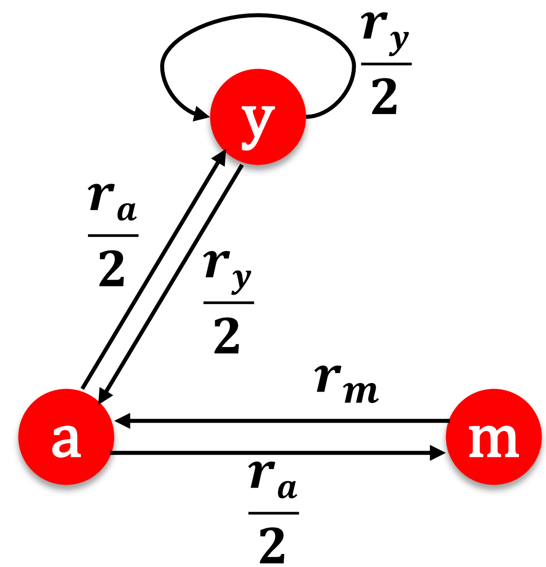
PageRank: The "Flow" Model

- **A "vote" from an important page is worth more**
- **A page is important if it is pointed to by other important pages**
- **Define a "rank" r_j for page j**

$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d}_i}$$

d_i … **out-degree of node i**

The web in 1839



"Flow" equations:

$$r_y = \frac{r_y}{2} + \frac{r_a}{2}$$

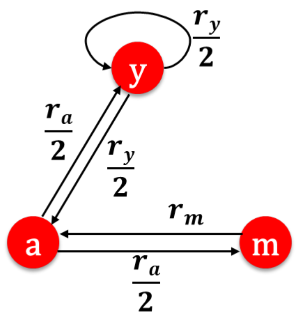
$$r_a = \frac{r_y}{2} + r_m$$

$$r_m = \frac{r_a}{2}$$

Solving the Flow Equations

"Flow" equations:

$$r_y = \frac{r_y}{2} + \frac{r_a}{2}$$
$$r_a = \frac{r_y}{2} + r_m$$
$$r_m = \frac{r_a}{2}$$



- **3 equations, 3 unknowns, no constants**
  - No unique solution
  - All solutions equivalent modulo a scale factor
- **Additional constraint forces uniqueness:**
  - $r_y + r_a + r_m = 1$
  - **Solution:** $r_y = \frac{2}{5}, \; r_a = \frac{2}{5}, \; r_m = \frac{1}{5}$
- **Gaussian elimination method works for small examples, but we need a better method for large web-size graphs**
- **We need a new formulation!**

PageRank: Matrix Formulation

- **Stochastic adjacency matrix $\boldsymbol{M}$**
  - Let page i has d_i out-links
  - If $i \rightarrow j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
    - $\boldsymbol{M}$ is a **column stochastic matrix**
      - Columns sum to 1
- **Rank vector $\boldsymbol{r}$:** vector with an entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$
- **The flow equations can be written**

$$r_j = \sum_{i \rightarrow j} \frac{r_i}{\mathrm{d}_i}$$

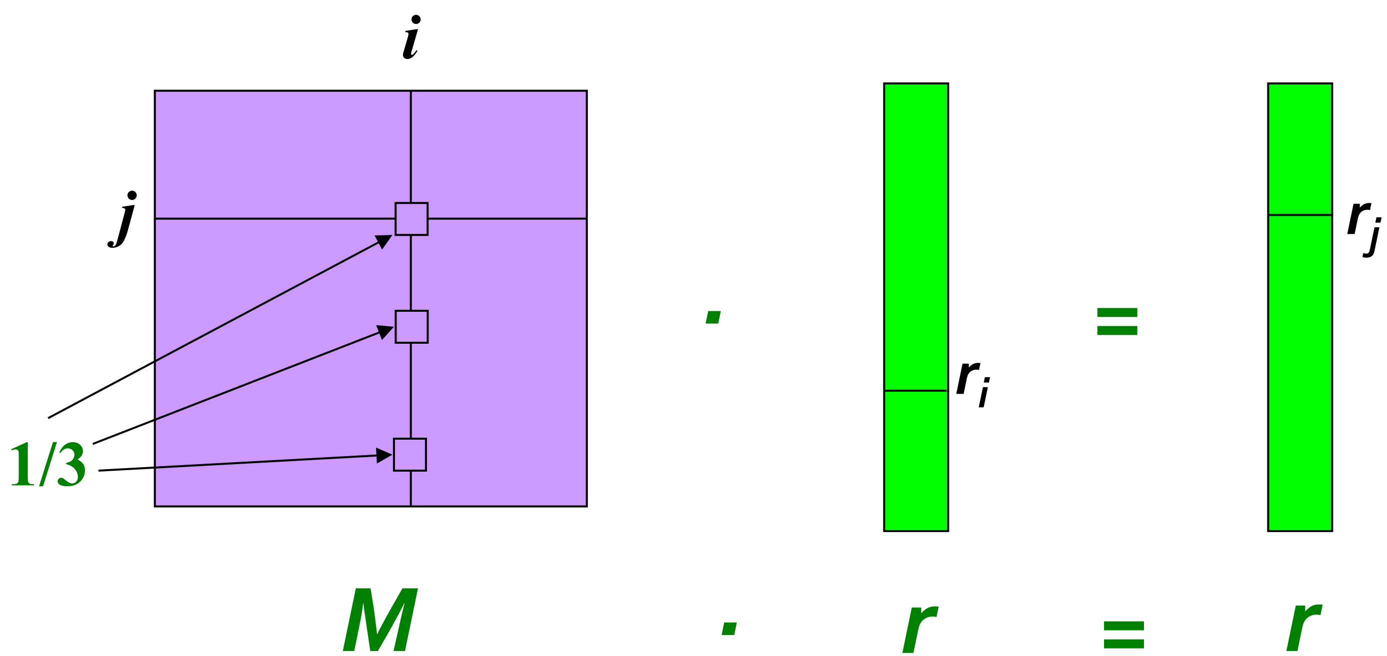
$$\boldsymbol{r} = \boldsymbol{M} \cdot \boldsymbol{r}$$

Example

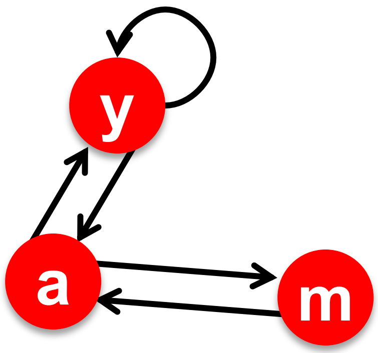
- **Remember the flow equation:** $r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d_i}}$
- **Flow equation in the matrix form**

$$\boldsymbol{M} \cdot \boldsymbol{r} = \boldsymbol{r}$$

  - **Suppose page *i* links to 3 pages, including *j***



Example: Flow Equations & M



	r_y	r_a	r_m
r_y	½	½	0
r_a	½	0	1
r_m	0	½	0

$$r = M \cdot r$$

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}$$

Eigenvector Formulation

- **The flow equations can be written**

$$\boldsymbol{r} = \boldsymbol{M} \cdot \boldsymbol{r}$$

- So the **rank vector** ***r*** is an **eigenvector** of the stochastic web matrix ***M***
  - Starting from any vector $\boldsymbol{u}$, the limit $\boldsymbol{M}(\boldsymbol{M}(\ldots\boldsymbol{M}(\boldsymbol{M}\ \boldsymbol{u})))$ is the long-term distribution of the surfers.
    - The math: limiting distribution = principal eigenvector of M = PageRank.
      - Note: If $\boldsymbol{r}$ satisfies the equation $\boldsymbol{r} = \boldsymbol{M}\boldsymbol{r}$, then r is an eigenvector of $\boldsymbol{M}$ with eigenvalue 1

NOTE: ***x*** is an eigenvector with the corresponding eigenvalue **λ** if: $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{\lambda}\boldsymbol{x}$

- **We can now efficiently solve for *r*!**

The method is called Power iteration

Power Iteration Method

- **Given a web graph with *n* nodes, where the nodes are pages and edges are hyperlinks**
- **Power iteration:** a simple iterative scheme
  - Suppose there are *N* web pages
  - Initialize: $\mathbf{r}^{(0)} = [1/N,\ldots.,1/N]^T$
  - Iterate: $\mathbf{r}^{(t+1)} = \mathbf{M} \cdot \mathbf{r}^{(t)}$
  - Stop when $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \varepsilon$

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

d_i …. out-degree of node i

$|\mathbf{x}|_1 = \sum_{1 \leq i \leq N} |x_i|$ is the $\mathbf{L_1}$ norm
Can use any other vector norm, e.g., Euclidean

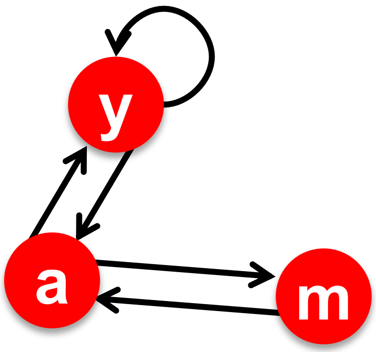
About 50 iterations is sufficient to estimate the limiting solution.

PageRank: How to solve?

- **Power Iteration:**
  - Set $r_j = 1/N$
  - **1:** $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** $r = r'$
  - Goto **1**
- **Example:**

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 \\ 1/3 \\ 1/3 \end{matrix}$$

Iteration 0, 1, 2, …

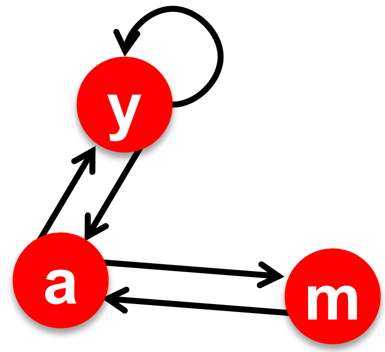


	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$\mathbf{r_y = r_y/2 + r_a/2}$$
$$\mathbf{r_a = r_y/2 + r_m}$$
$$\mathbf{r_m = r_a/2}$$

PageRank: How to solve?

- **Power Iteration:**
  - Set $r_j = 1/N$
  - **1:** $r'_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** $r = r'$
  - Goto **1**



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

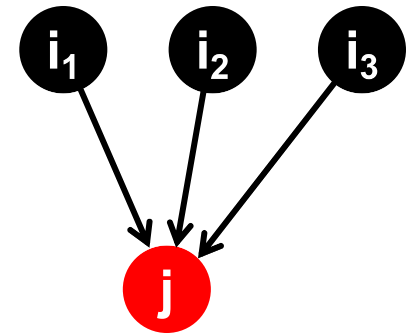
$$\mathbf{r_y} = \mathbf{r_y}/2 + \mathbf{r_a}/2$$
$$\mathbf{r_a} = \mathbf{r_y}/2 + \mathbf{r_m}$$
$$\mathbf{r_m} = \mathbf{r_a}/2$$

- **Example:**

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & & \mathbf{2/5} \\ 1/3 & 3/6 & 1/3 & 11/24 & \ldots & \mathbf{2/5} \\ 1/3 & 1/6 & 3/12 & 1/6 & & \mathbf{1/5} \end{matrix}$$

Iteration 0, 1, 2, …

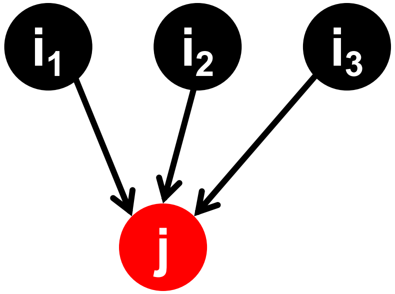
Random Walk Interpretation



$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d_{out}}(\mathrm{i})}$$

- **Imagine a random web surfer:**
  - At any time $\boldsymbol{t}$, surfer is on some page $\boldsymbol{i}$
  - At time $\boldsymbol{t+1}$, the surfer follows an out-link from $\boldsymbol{i}$ uniformly at random
  - Ends up on some page $\boldsymbol{j}$ linked from $\boldsymbol{i}$
  - Process repeats indefinitely
- **Let:**
  - $\boldsymbol{p(t)}$ … vector whose $\boldsymbol{i}^{\text{th}}$ coordinate is the prob. that the surfer is at page $\boldsymbol{i}$ at time $\boldsymbol{t}$
  - So, $\boldsymbol{p(t)}$ is a probability distribution over pages

The Stationary Distribution



$$p(t+1) = \mathrm{M} \cdot p(t)$$

- **Where is the surfer at time $t+1$?**
  - Follows a link uniformly at random

    $$\boldsymbol{p}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{M} \cdot \boldsymbol{p}(\boldsymbol{t})$$

- Suppose the random walk reaches a state

  $$\boldsymbol{p}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{M} \cdot \boldsymbol{p}(\boldsymbol{t}) = \boldsymbol{p}(\boldsymbol{t})$$

  then $\boldsymbol{p}(\boldsymbol{t})$ is **stationary distribution** of a random walk
- **Our original rank vector** $\boldsymbol{r}$ satisfies $\boldsymbol{r} = \boldsymbol{M} \cdot \boldsymbol{r}$
  - **So, r is a stationary distribution for the random walk**

Existence and Uniqueness

- **A central result from the theory of random walks (a.k.a. Markov processes):**

For graphs that satisfy **certain conditions**, the **stationary distribution is unique** and eventually will be reached no matter what is the initial probability distribution at time **t = 0**

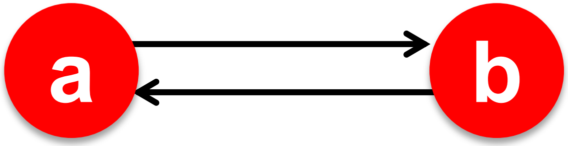
PageRank:
The Google Formulation

PageRank: Three Questions

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_\mathrm{i}}$$ or equivalently $$r = Mr$$

- **Does this converge?**
- **Does it converge to what we want?**
- **Are results reasonable?**

Does this converge?



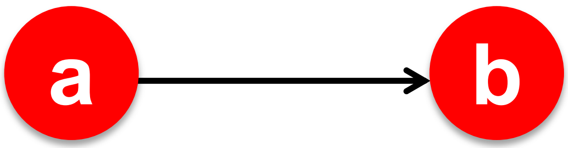
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

- **Example:**

$$\begin{matrix} r_a \\ r_b \end{matrix} = \begin{matrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{matrix}$$

Iteration 0, 1, 2, …

Does it converge to what we want?



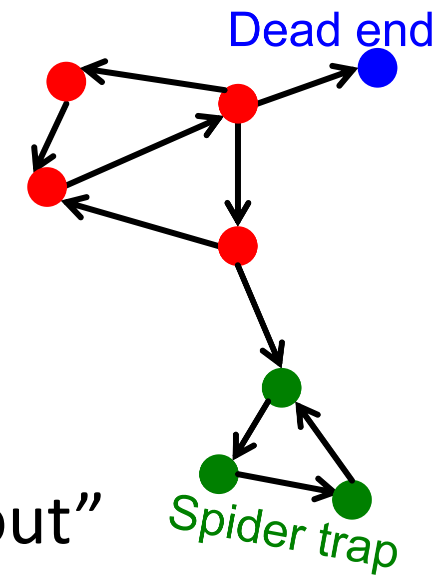
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d_i}}$$

- **Example:**

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}$$

Iteration 0, 1, 2, …

PageRank: Problems



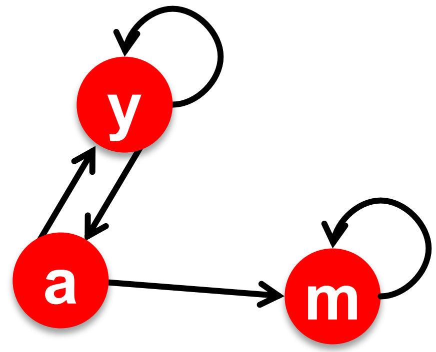
2 problems:

- **(1) Dead ends:** Some pages have no out-links
  - Random walk has "nowhere" to go to
  - Such pages cause importance to "leak out"

- **(2) Spider traps:** (all out-links are within the group)
  - Random walk gets "stuck" in a trap
  - And eventually spider traps absorb all importance

Problem: Spider Traps

- **Power Iteration:**
  - Set $r_j = 1$
  - $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
    - And iterate



m is a spider trap

	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

$$\mathbf{r_y} = \mathbf{r_y}/2 + \mathbf{r_a}/2$$
$$\mathbf{r_a} = \mathbf{r_y}/2$$
$$\mathbf{r_m} = \mathbf{r_a}/2 + \mathbf{r_m}$$

- **Example:**

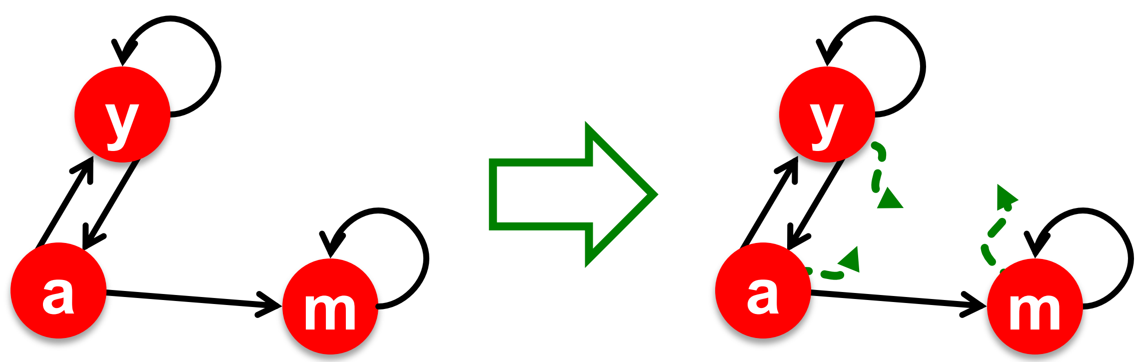
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{matrix}$$

Iteration 0, 1, 2, …

All the PageRank score gets "trapped" in node m.

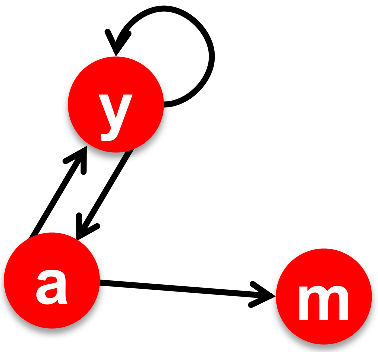
Solution: Teleports!

- **The Google solution for spider traps: At each time step, the random surfer has two options**
  - With prob. β, follow a link at random
  - With prob. **1-**β, jump to some random page
  - β is typically in the range 0.8 to 0.9
- **Surfer will teleport out of spider trap within a few time steps**



Problem: Dead Ends

- **Power Iteration:**
  - Set $r_j = 1$
  - $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
    - And iterate



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$r_y = r_y/2 + r_a/2$

$r_a = r_y/2$

$r_m = r_a/2$

- **Example:**

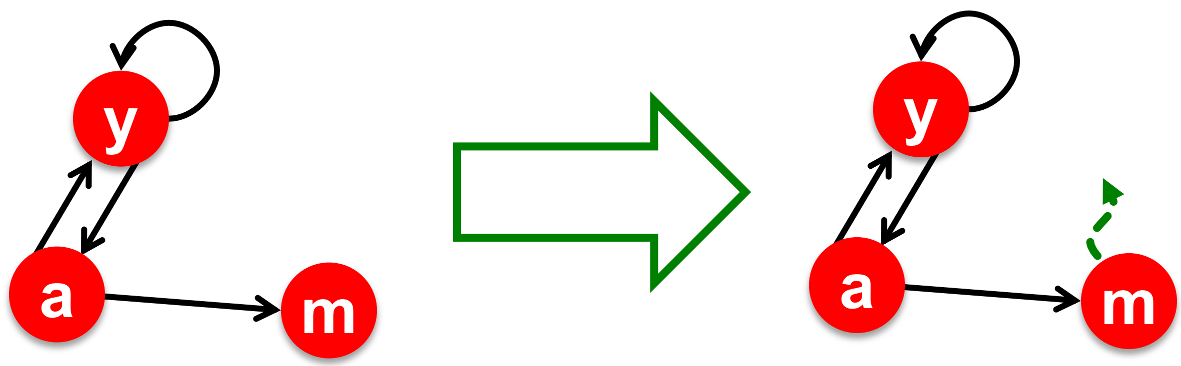
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{matrix}$$

Iteration 0, 1, 2, …

Here the PageRank score “leaks” out since the matrix is not stochastic.

Solution: Always Teleport!

- **Teleports:** Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matrix accordingly



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

Why Teleports Solve the Problem?

Why are dead-ends and spider traps a problem and why do teleports solve the problem?

- **Spider-traps** are not a problem, but with traps PageRank scores are **not** what we want
  - **Solution:** Never get stuck in a spider trap by teleporting out of it in a finite number of steps
- **Dead-ends** are a problem
  - The matrix is not column stochastic so our initial assumptions are not met
  - **Solution:** Make matrix column stochastic by always teleporting when there is nowhere else to go

Solution: Random Teleports

- **Google's solution that does it all:**
  At each step, random surfer has two options:
  - With probability β, follow a link at random
  - With probability $1\text{-}\beta$, jump to some random page

- **PageRank equation** [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

d_i ... out-degree of node i

This formulation assumes that M has no dead ends. We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

The Google Matrix

- **PageRank equation** [Brin-Page, '98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

- **The Google Matrix *A*:**

[1/N]$_{N\times N}$…N by N matrix where all entries are 1/N

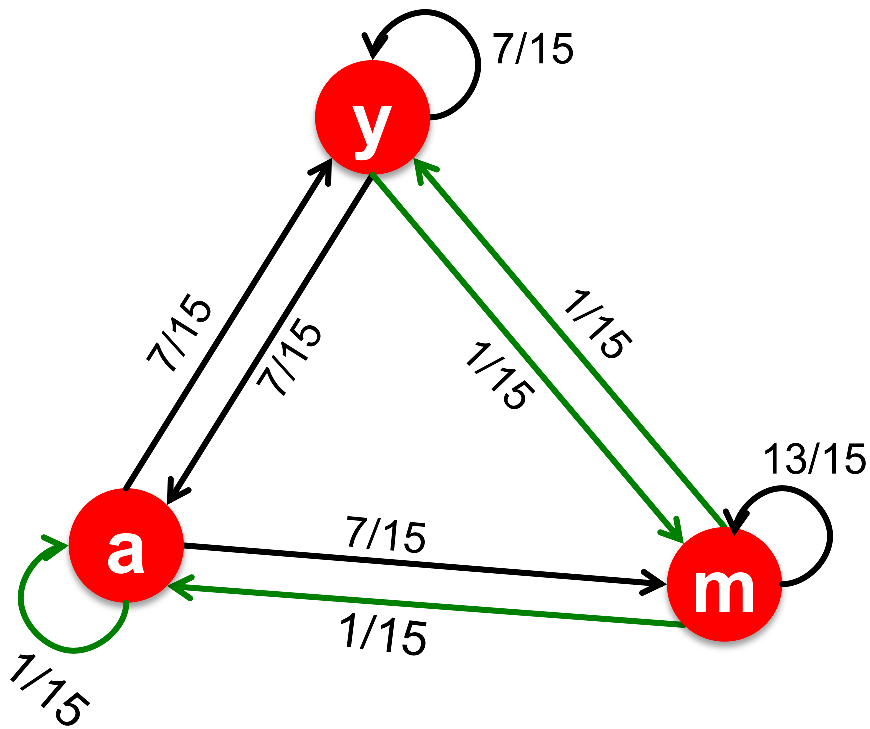
$$A = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N\times N}$$

- **We have a recursive problem:** $\boldsymbol{r = A \cdot r}$

And the Power method still works!

- **What is β?**
  - In practice $\beta = 0.8, 0.9$ (make 5 steps on avg., jump)

Random Teleports ($\beta = 0.8$)



$$
0.8 \overset{\textbf{M}}{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}} + 0.2 \overset{[1/N]_{NxN}}{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}
$$

$$
\begin{array}{c} y \\ a \\ m \end{array}
\underset{\textbf{A}}{\begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}}
$$

$$
\begin{array}{c} y \\ a \\ m \end{array} = \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 0.33 \\ 0.20 \\ 0.46 \end{array} \quad \begin{array}{c} 0.24 \\ 0.20 \\ 0.52 \end{array} \quad \begin{array}{c} 0.26 \\ 0.18 \\ 0.56 \end{array} \quad \dots \quad \begin{array}{c} 7/33 \\ 5/33 \\ 21/33 \end{array}
$$

How do we actually compute the PageRank?

Computing PageRank

- **Key step is matrix-vector multiplication**
  - $\boldsymbol{r}^{new} = \boldsymbol{A} \cdot \boldsymbol{r}^{old}$
- Easy if we have enough main memory to hold **A**, $\mathbf{r}^{old}$, $\mathbf{r}^{new}$
- **Say N = 1 billion pages**
  - We need 4 bytes for each entry (say)
  - 2 billion entries for vectors, approx 8GB
  - Matrix **A** has N^2 entries
    - 10^{18} is a large number!

$$\mathbf{A} = \beta \cdot \mathbf{M} + (1-\beta)\,[1/N]_{N\times N}$$

$$\boldsymbol{A} = 0.8 \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$$= \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

Rearranging the Equation

- $\boldsymbol{r} \;=\boldsymbol{A}\cdot\boldsymbol{r},$ **where** $\boldsymbol{A_{ji}=\beta\, M_{ji}+\frac{1-\beta}{N}}$
- $r_j \;=\sum_{\mathrm{i}=1}^{N} A_{ji}\cdot r_i$
- $r_j \;=\sum_{i=1}^{N}\left[\beta\, M_{ji}+\frac{1-\beta}{N}\right]\cdot r_i$

  $=\sum_{\mathrm{i}=1}^{N}\beta\, M_{ji}\cdot r_i+\frac{1-\beta}{N}\sum_{\mathrm{i}=1}^{N} r_i$

  $=\sum_{\mathrm{i}=1}^{N}\beta\, M_{ji}\cdot r_i+\frac{1-\beta}{N}$ since $\sum r_i=1$
- **So we get:** $\boldsymbol{r}=\boldsymbol{\beta\, M}\cdot\boldsymbol{r}+\left[\boldsymbol{\frac{1-\beta}{N}}\right]_{\boldsymbol{N}}$

Note: Here we assume **M** has no dead-ends

$[x]_N$ … a vector of length N with all entries x

Sparse Matrix Formulation

- We just rearranged the **PageRank equation**

$$r = \beta M \cdot r + \left[\frac{1-\beta}{N}\right]_N$$

  - where $[(1-\beta)/N]_N$ is a vector with all N entries $(1-\beta)/N$
- ***M*** is a **sparse matrix!** (with no dead-ends)
  - 10 links per node, approx $10N$ entries
- So in each iteration, we need to:
  - Compute $r^{new} = \beta M \cdot r^{old}$
  - Add a constant value $(1-\beta)/N$ to each entry in r^{new}
    - **Note if M contains dead-ends then $\sum_j r_j^{new} < 1$ and we also have to renormalize r^{new} so that it sums to 1**

PageRank: The Complete Algorithm

- **Input: Graph G and parameter β**
  - Directed graph G (can have **spider traps** and **dead ends**)
  - Parameter β
- **Output: PageRank vector r^{new}**
  - **Set:** $r_j^{old} = \frac{1}{N}$
  - **repeat until convergence:** $\sum_j \left| r_j^{new} - r_j^{old} \right| < \varepsilon$
    - $\forall j$: $\boldsymbol{r'^{new}_j = \sum_{i \to j} \beta \frac{r_i^{old}}{d_i}}$

      $\boldsymbol{r'^{new}_j = 0}$ if in-degree of $\boldsymbol{j}$ is $\mathbf{0}$
    - **Now re-insert the leaked PageRank:**

      $\forall \boldsymbol{j}$: $\boldsymbol{r_j^{new} = r'^{new}_j + \frac{1-S}{N}}$ where: $S = \sum_j r'^{new}_j$
    - $\boldsymbol{r^{old} = r^{new}}$

If the graph has no dead-ends then the amount of leaked PageRank is **1-β**. But since we have dead-ends the amount of leaked PageRank may be larger. We have to explicitly account for it by computing **S**.

Some Problems with PageRank

- **Measures generic popularity of a page**
  - Biased against topic-specific authorities
  - **Solution:** Topic-Specific PageRank (**after break**)
- **Uses a single measure of importance**
  - Other models of importance
  - **Solution:** Hubs-and-Authorities
- **Susceptible to Link spam**
  - Artificial link topographies created in order to boost page rank
  - **Solution:** TrustRank (**after break**)