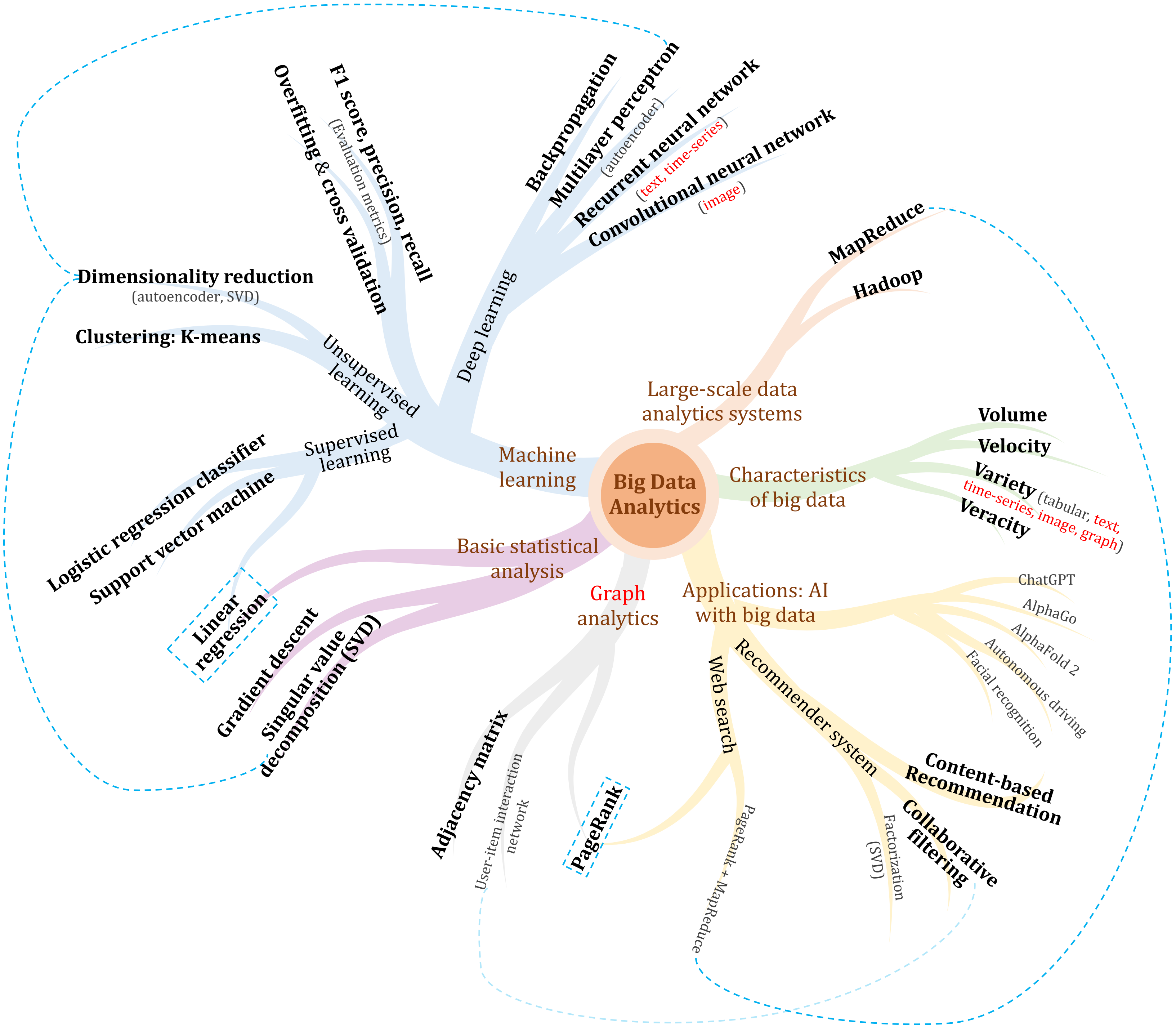
THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

COMP4434 Big Data Analytics

Lecture 11 PageRank

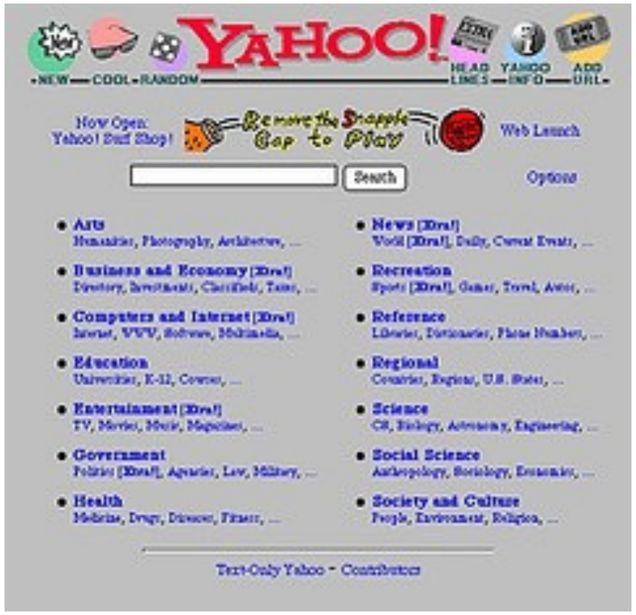
HUANG Xiao
xiaohuang@comp.polyu.edu.hk



Big Data Analytics
Machine learning
Deep learning
Backpropagation
Multilayer perceptron
(autoencoder)
Recurrent neural network
(text, time-series)
Convolutional neural network
(image)
F1 score, precision, recall
(Evaluation metrics)
Overfitting & cross validation
Unsupervised learning
Dimensionality reduction
(autoencoder, SVD)
Clustering: K-means
Supervised learning
Logistic regression classifier
Support vector machine
Linear regression
Large-scale data analytics systems
MapReduce
Hadoop
Characteristics of big data
Volume
Velocity
Variety (tabular, text, time-series, image, graph)
Veracity
Basic statistical analysis
Gradient descent
Singular value decomposition (SVD)
Graph analytics
Adjacency matrix
User-item interaction network
PageRank
Applications: AI with big data
ChatGPT
AlphaGo
AlphaFold 2
Autonomous driving
Facial recognition
Web search
PageRank + MapReduce
Recommender system
Content-based Recommendation
Collaborative filtering
Factorization (SVD)

PageRank Motivation: How to organize the Web?

- **First try:** Human curated **Web directories**
  - Yahoo, DMOZ, LookSmart
- **Second try: Web search**
  - **Information Retrieval** investigates: Finding 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.



- **Arts**
  Humanities, Photography, Architecture, ...
- **Business and Economy** [Xtra!]
  Directory, Investments, Classifieds, Taxes, ...
- **Computers and Internet** [Xtra!]
  Internet, WWW, Software, Multimedia, ...
- **Education**
  Universities, K-12, Courses, ...
- **Entertainment** [Xtra!]
  TV, Movies, Music, Magazines, ...
- **Government**
  Politics [Xtra!], Agencies, Law, Military, ...
- **Health**
  Medicine, Drugs, Diseases, Fitness, ...
- **News** [Xtra!]
  World [Xtra!], Daily, Current Events, ...
- **Recreation**
  Sports [Xtra!], Games, Travel, Autos, ...
- **Reference**
  Libraries, Dictionaries, Phone Numbers, ...
- **Regional**
  Countries, Regions, U.S. States, ...
- **Science**
  CS, Biology, Astronomy, Engineering, ...
- **Social Science**
  Anthropology, Sociology, Economics, ...
- **Society and Culture**
  People, Environment, Religion, ...

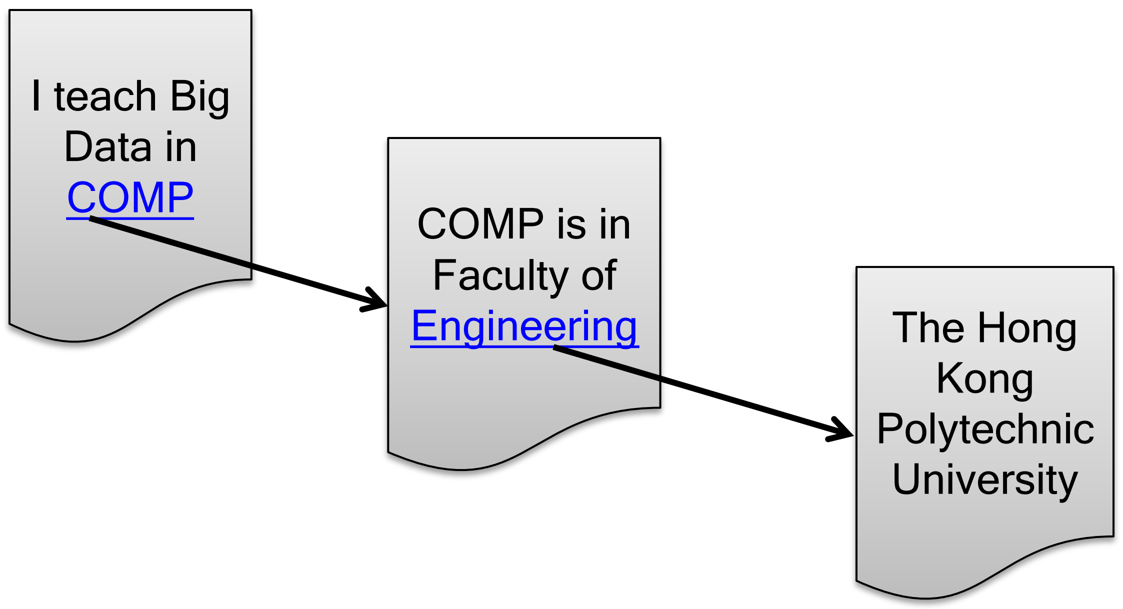
Text-Only Yahoo - Contributors

Challenges in Web Search

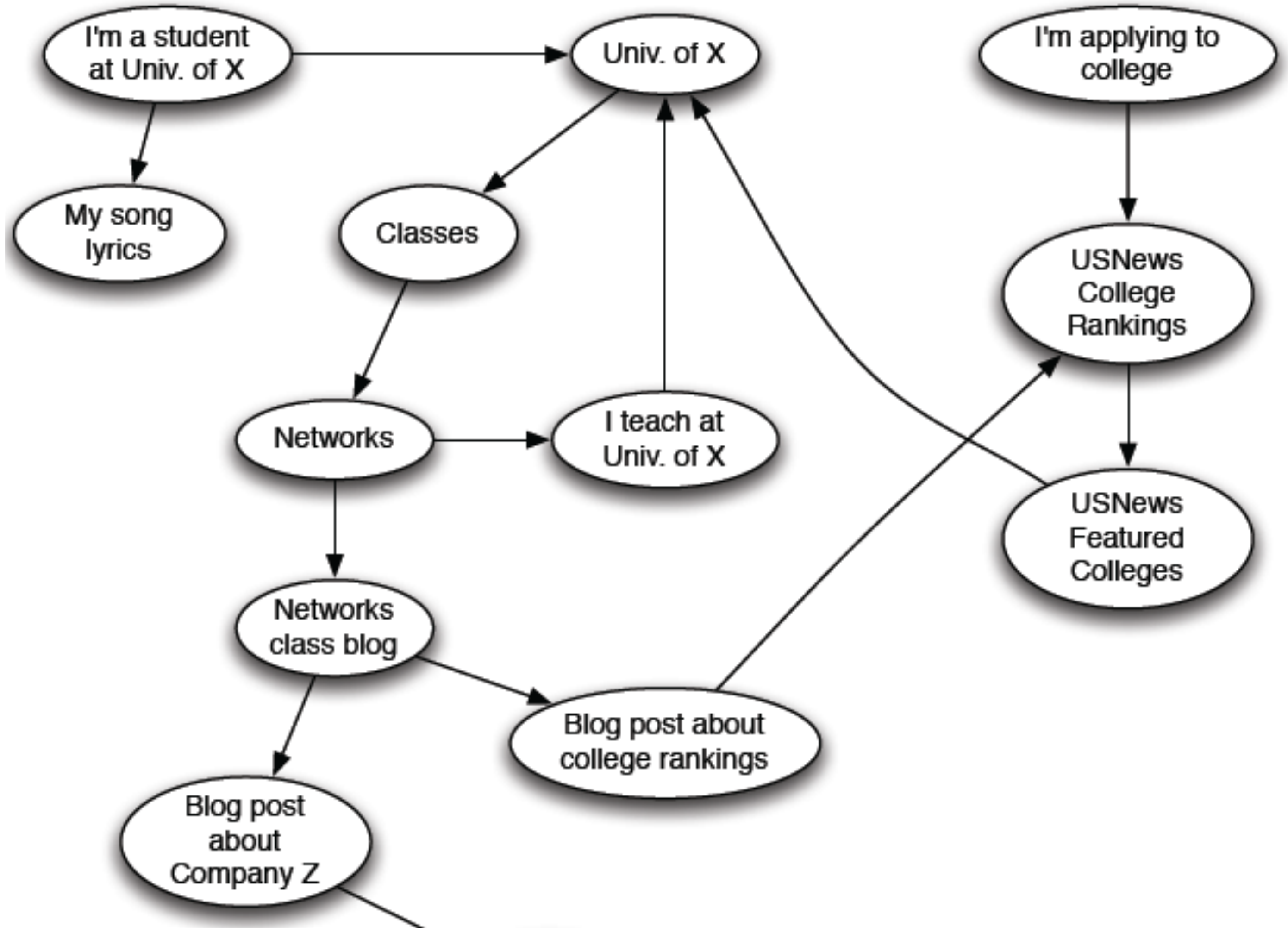
- **(1) Web contains many sources of information**
  Who to "trust"?
  - **Trick:** Trustworthy pages may point to each other!
- **(2) What is the "best" answer to query "newspaper"?**
  - No single right answer
  - **Trick:** Pages that actually know about newspapers might all be pointing to many newspapers

Hint: Web as a Directed Graph

- **Nodes: Webpages**
- **Edges: Hyperlinks**


I teach Big Data in COMP
COMP is in Faculty of Engineering
The Hong Kong Polytechnic University

Web as a Directed Graph

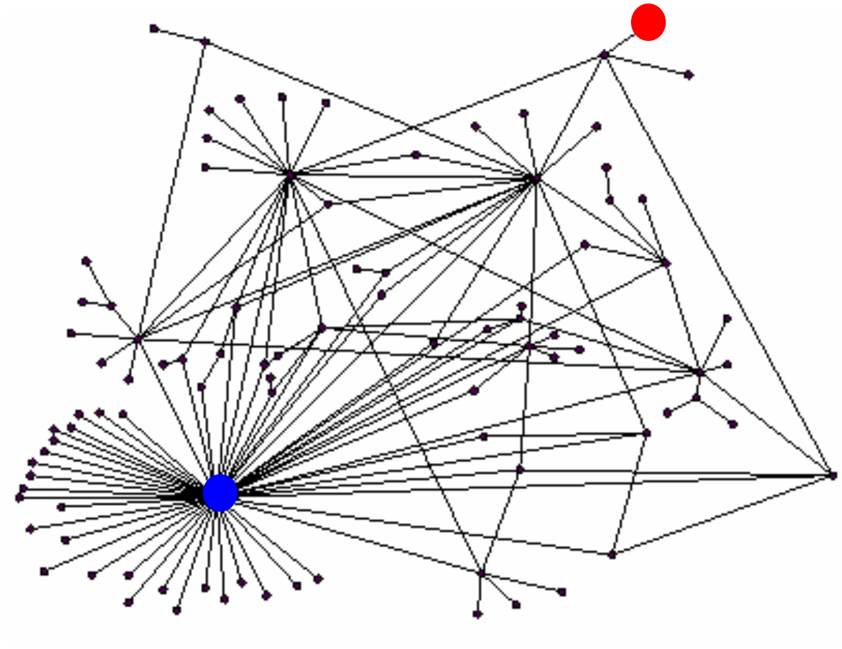


Ranking Nodes on the Graph

- **All web pages are not equally "important"**

  https://xhuang31.github.io vs. https://www.polyu.edu.hk

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



Example of Node Ranking

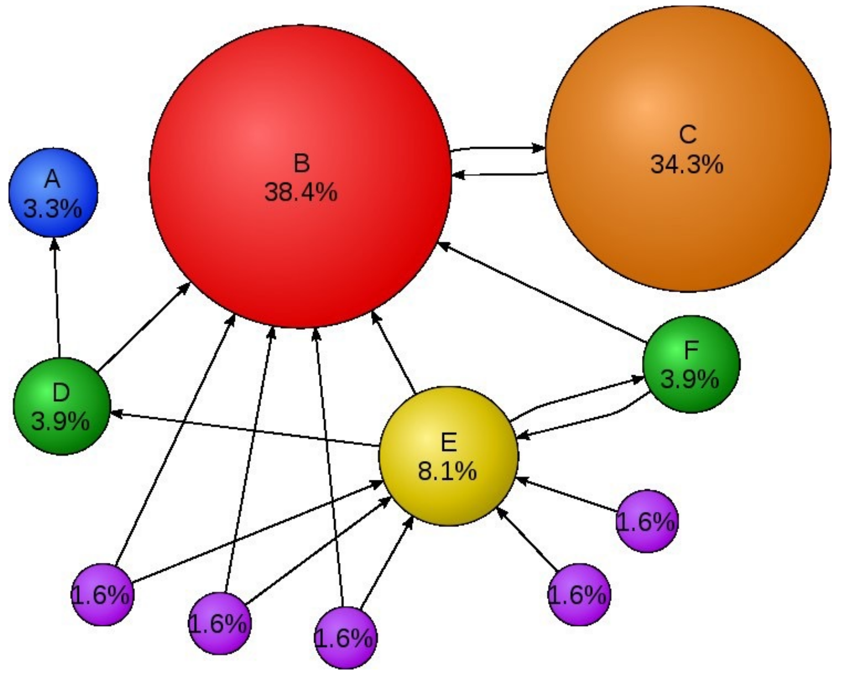
- Page Ranking
- Social Ranking
- Paper Ranking
- Scholar Ranking
- ……

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.stanford.edu has 23,400 in-links
  - https://xhuang31.github.io has 0 in-link
- **Are all in-links are equal?**
  - **Links from important pages count more**
  - Recursive question!

Google PageRank

- In-coming links! Out-going links?
- A page with high PageRank value
  - Many pages pointing to it, or
  - There are some pages that point to it and have high PageRank values
- Example:
  - Page C has a higher PageRank than Page E, even though it has fewer links to it
  - The link it has is of a much higher value



Is Page == “Webpage”?

- Born in March 26, 1973
- Found Google at September 4, 1998
- As of Nov 2024, own an estimated net worth of $163 billion (No.15 Richest)
- Begins from ”Larry Page and Sergey Brin developed PageRank at Stanford University in 1996” as part of a research project about a new kind of search engine.

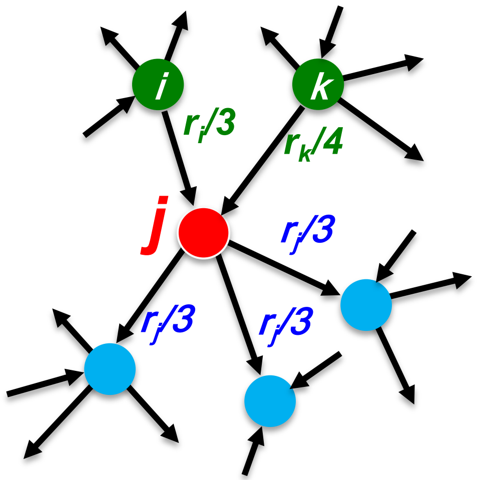
Larry Page

Co-founder of Google

Simple Recursive Formulation

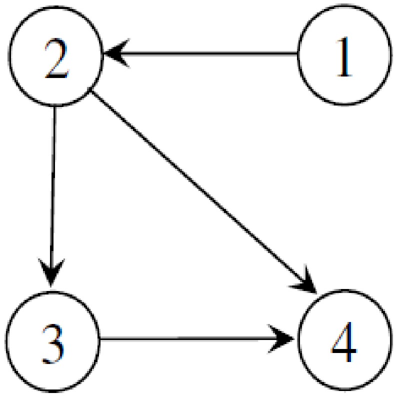
- Each link's vote is proportional to the **importance** of its source page
- If page ***j*** with importance ***PR(j)*** has ***n*** out-links, each link gets ***PR(j) / n*** votes
- Page ***j***'s own importance is the sum of the votes on its in-links

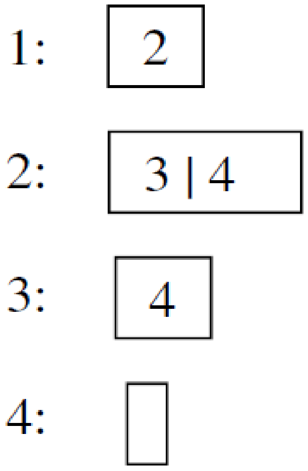
$$PR(j) = PR(i)/3 + PR(k)/4$$



How to Represent a Graph

- **Graph model** $G = (V, E)$
  - V is a set of pages
  - E is a set of edges
  - Each edge $(u, v) \in E$ represents that page u points/references to page v
- **Adjacent List**
  - A data structure for a graph
  - $Adj[u] = \{v: (u, v) \in E\}$ contains each vertex v being adjacent to u
  - Example: $Adj[2]$ = {3, 4}



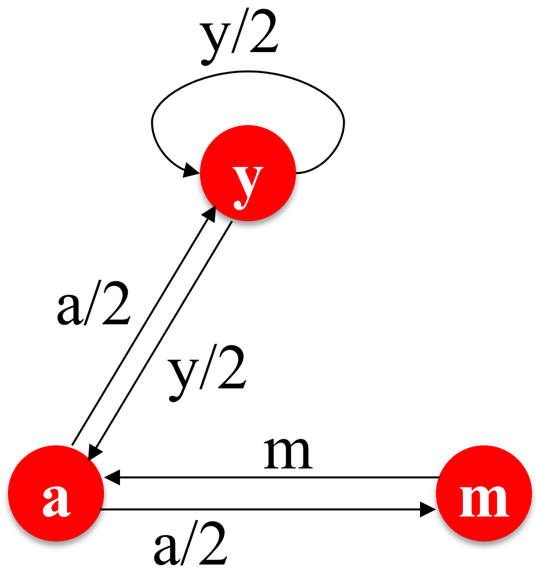


PageRank: The "Flow" Model

- 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



"Flow" equations:

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

Solving the Flow Equations

Flow equations:

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

- **3 equations, 3 unknowns, no constants**
  - No unique solution
  - All solutions equivalent modulo the scale factor
- **Additional constraint forces uniqueness:**
  - $r_y + r_a + r_m = 1$
  - **Solution:** $r_y = \frac{2}{5},\ r_a = \frac{2}{5},\ r_m = \frac{1}{5}$
- **But, we need a better method for large web-size graphs**

PageRank: Matrix Formulation

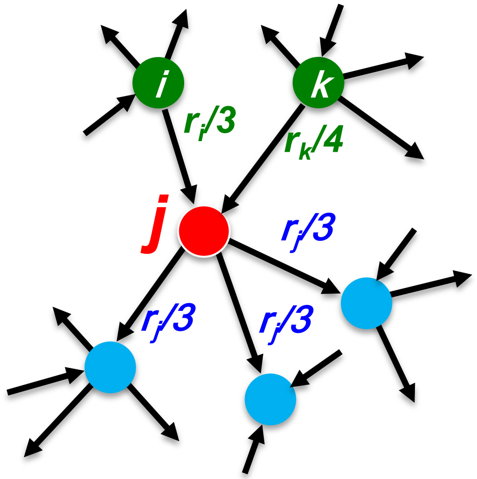
- **Stochastic adjacency matrix $\boldsymbol{M}$**
  - Let page i has d_i out-links
  - If $i \rightarrow j$, then $M_{ji} = \frac{1}{d_i}$ else $M_{ji} = 0$
    - $\boldsymbol{M}$ is a **column stochastic matrix**
      - Columns sum to 1
- **Rank vector $\boldsymbol{r}$:** vector with an entry per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

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

- **The flow equations can be written**

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

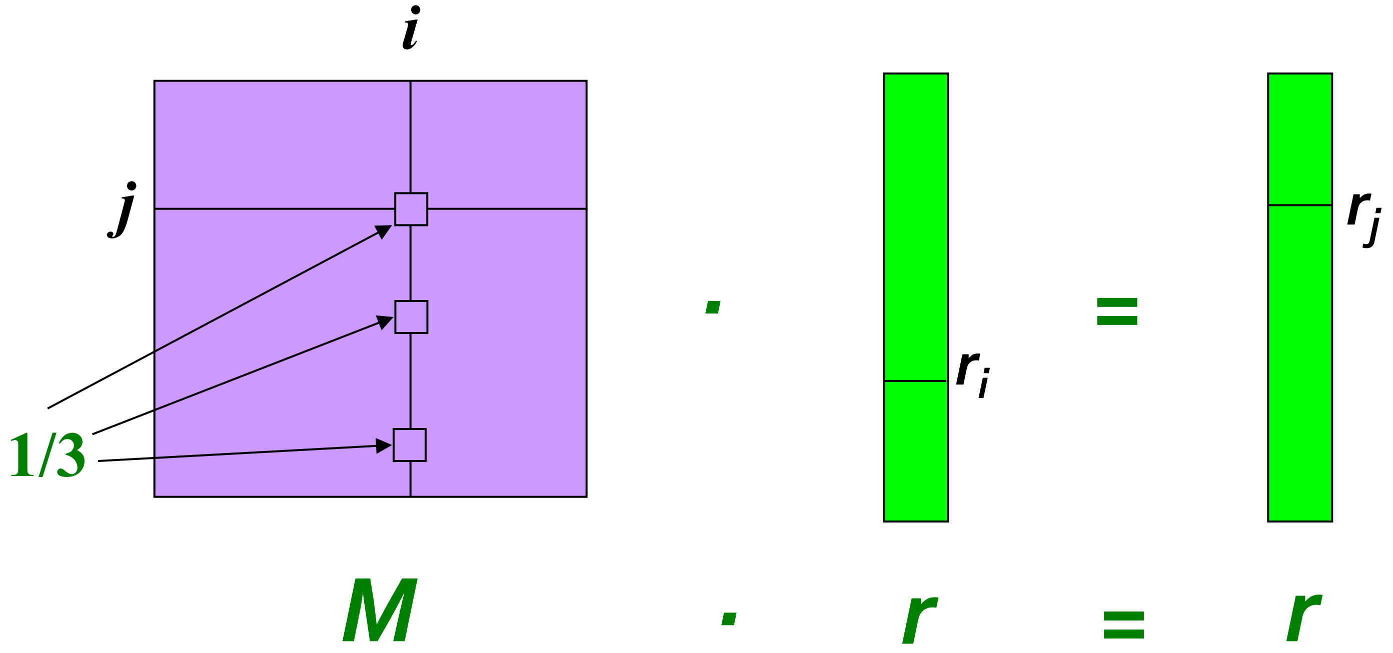
Example



- Remember the flow equation:
- Flow equation in the matrix form

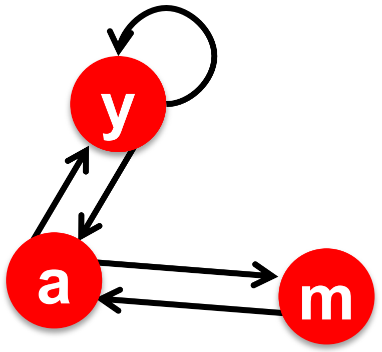
$$\boldsymbol{M} \cdot \boldsymbol{r} = \boldsymbol{r}$$

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



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

Example: Flow Equations & M



$$\begin{array}{c|ccc} & y & a & m \\ \hline y & \frac{1}{2} & \frac{1}{2} & 0 \\ a & \frac{1}{2} & 0 & 1 \\ m & 0 & \frac{1}{2} & 0 \end{array}$$

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

$$\begin{aligned} r_y &= r_y/2 + r_a/2 \\ r_a &= r_y/2 + r_m \\ r_m &= r_a/2 \end{aligned}$$

$$\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} y \\ a \\ m \end{bmatrix}$$

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 $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \varepsilon$

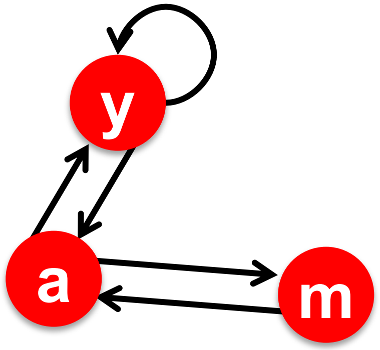
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

d_i …. out-degree of node i

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

PageRank: How to solve?

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



$$\begin{array}{c|ccc} & y & a & m \\ \hline y & \frac{1}{2} & \frac{1}{2} & 0 \\ a & \frac{1}{2} & 0 & 1 \\ m & 0 & \frac{1}{2} & 0 \end{array}$$

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

- **Example:**

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \quad \begin{matrix} 1/3 & 1/3 & 5/12 & 9/24 & & 6/15 \\ 1/3 & 3/6 & 1/3 & 11/24 & \ldots & 6/15 \\ 1/3 & 1/6 & 3/12 & 1/6 & & 3/15 \end{matrix}$$

Iteration 0, 1, 2, …

Why Power Iteration works? (1)

Details!

- **Power iteration:**
  A method for finding dominant eigenvector (the vector corresponding to the largest eigenvalue)
  - $\boldsymbol{r}^{(1)} = \boldsymbol{M} \cdot \boldsymbol{r}^{(0)}$
  - $\boldsymbol{r}^{(2)} = \boldsymbol{M} \cdot \boldsymbol{r}^{(1)} = \boldsymbol{M}\left(\boldsymbol{M}\boldsymbol{r}^{(0)}\right) = \boldsymbol{M}^2 \cdot \boldsymbol{r}^{(0)}$
  - $\boldsymbol{r}^{(3)} = \boldsymbol{M} \cdot \boldsymbol{r}^{(2)} = \boldsymbol{M}\left(\boldsymbol{M}^2\boldsymbol{r}^{(0)}\right) = \boldsymbol{M}^3 \cdot \boldsymbol{r}^{(0)}$
- **Claim:**
  Sequence $\boldsymbol{M} \cdot \boldsymbol{r}^{(0)}, \boldsymbol{M}^2 \cdot \boldsymbol{r}^{(0)}, \ldots \boldsymbol{M}^k \cdot \boldsymbol{r}^{(0)}, \ldots$ approaches the dominant eigenvector of $\boldsymbol{M}$ ($\boldsymbol{M}$ is stochastic/Markov matrix)
- **NOTE:** ***x*** is an eigenvector with the corresponding eigenvalue **λ** if:
  $$\boldsymbol{M}\boldsymbol{x} = \boldsymbol{\lambda}\boldsymbol{x}$$
  Optimal r is the first or principal eigenvector of M, with corresponding eigenvalue 1

Why Power Iteration works? (2)

Details!

- **Claim:** Sequence $\boldsymbol{M} \cdot \boldsymbol{r}^{(0)}, \boldsymbol{M}^2 \cdot \boldsymbol{r}^{(0)}, \ldots \boldsymbol{M}^k \cdot \boldsymbol{r}^{(0)}, \ldots$ approaches the dominant eigenvector of $\boldsymbol{M}$

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

- **Proof:**
  - Assume ***M*** has ***n*** linearly independent eigenvectors, $x_1, x_2, \ldots, x_n$ with corresponding eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$, where $\lambda_1 > \lambda_2 > \cdots > \lambda_n$
  - Vectors $x_1, x_2, \ldots, x_n$ form a basis and thus we can write: $r^{(0)} = c_1\, x_1 + c_2\, x_2 + \cdots + c_n\, x_n$
  - $\boldsymbol{Mr^{(0)}} = \boldsymbol{M(c_1\, x_1 + c_2\, x_2 + \cdots + c_n\, x_n)}$
    $= c_1(Mx_1) + c_2(Mx_2) + \cdots + c_n(Mx_n)$
    $= c_1(\lambda_1 x_1) + c_2(\lambda_2 x_2) + \cdots + c_n(\lambda_n x_n)$
  - **Repeated multiplication on both sides produces**

$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

Why Power Iteration works? (3)

Details!

- **Claim:** Sequence $\boldsymbol{M} \cdot \boldsymbol{r}^{(0)}, \boldsymbol{M}^{2} \cdot \boldsymbol{r}^{(0)}, \ldots \boldsymbol{M}^{k} \cdot \boldsymbol{r}^{(0)}, \ldots$ approaches the dominant eigenvector of $\boldsymbol{M}$
- **Proof (continued):**
  - Repeated multiplication on both sides produces

$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \cdots + c_n(\lambda_n^k x_n)$$

  - $M^k r^{(0)} = \lambda_1^k \left[c_1 x_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k x_n \right]$
  - Since $\lambda_1 > \lambda_2$ then fractions $\frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1} \ldots < 1$

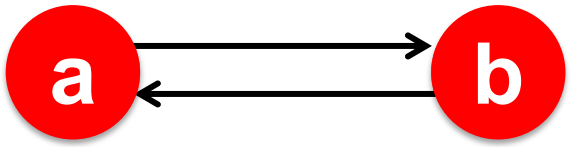
    and so $\left(\frac{\lambda_i}{\lambda_1}\right)^k = 0$ as $k \to \infty$ (for all $i = 2 \ldots n$).
  - **Thus:** $\boldsymbol{M^k r^{(0)} \approx c_1(\lambda_1^k x_1)}$
    - Note if $c_1 = 0$ then the method won't converge
  - The largest eigenvalue of a stochastic matrix is always 1.

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?



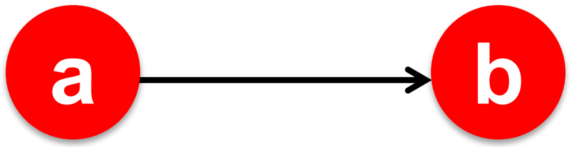
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_i}$$

- **Example:**

r_a	=	1	0	1	0
r_b		0	1	0	1

Iteration 0, 1, 2, …

Does it converge to what we want?



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

- **Example:**

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

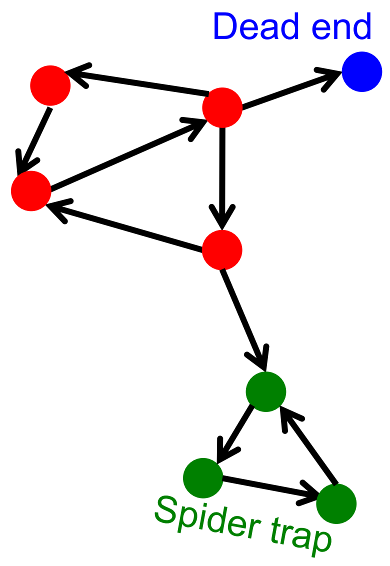
Iteration 0, 1, 2, …

PageRank: Problems

2 problems:

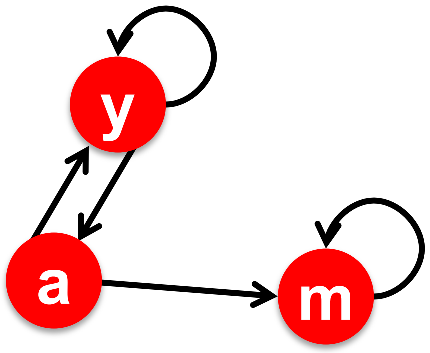
- **(1)** Some pages are **dead ends** (have no out-links)
  - "Vote" has "nowhere" to go to
  - Such pages cause importance to "leak out"

- **(2) Spider traps:** (all out-links are within the group)
  - "Vote" 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

$$\begin{array}{c|ccc} & y & a & m \\ \hline y & ½ & ½ & 0 \\ a & ½ & 0 & 0 \\ m & 0 & ½ & 1 \end{array}$$

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

$\mathbf{r_a = r_y/2}$

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

- **Example:**

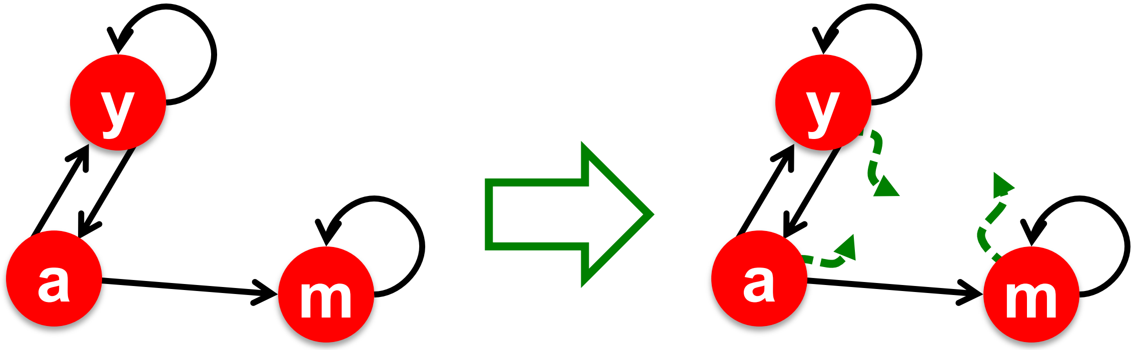
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \dots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{matrix}$$

Iteration 0, 1, 2, …

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

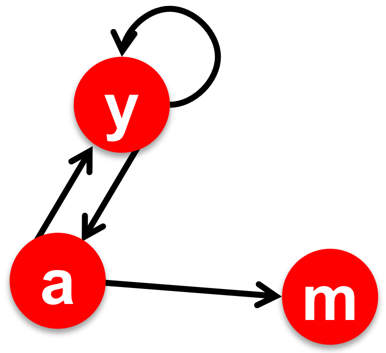
Solution: Teleports!

- **The Google solution for spider traps: At each time step, the "vote" has two options**
    - With prob. β, follow a link at random
    - With prob. **1-**β, jump to some random page
    - Common values for β are in the range 0.8 to 0.9
- **"Vote" 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:**

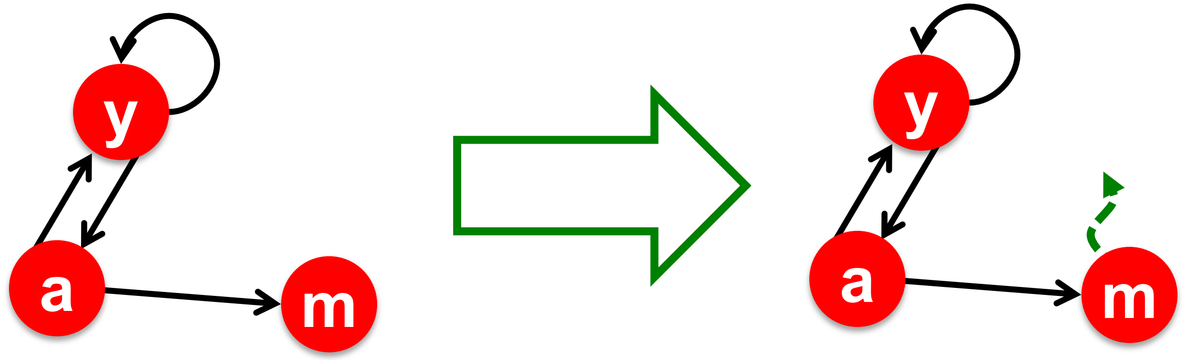
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{array}{cccccc} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{array}$$

Iteration 0, 1, 2, …

Here the PageRank "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



$$\begin{array}{c|c|c|c|}
 & y & a & m \\ \hline
y & \frac{1}{2} & \frac{1}{2} & 0 \\ \hline
a & \frac{1}{2} & 0 & 0 \\ \hline
m & 0 & \frac{1}{2} & 0 \\ \hline
\end{array}
\qquad
\begin{array}{c|c|c|c|}
 & y & a & m \\ \hline
y & \frac{1}{2} & \frac{1}{2} & \frac{1}{3} \\ \hline
a & \frac{1}{2} & 0 & \frac{1}{3} \\ \hline
m & 0 & \frac{1}{2} & \frac{1}{3} \\ \hline
\end{array}$$

Why 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

- <u>**Google's solution that does it all:**</u>
  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** [Larry Page and Sergey Brin 1998]

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

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]

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

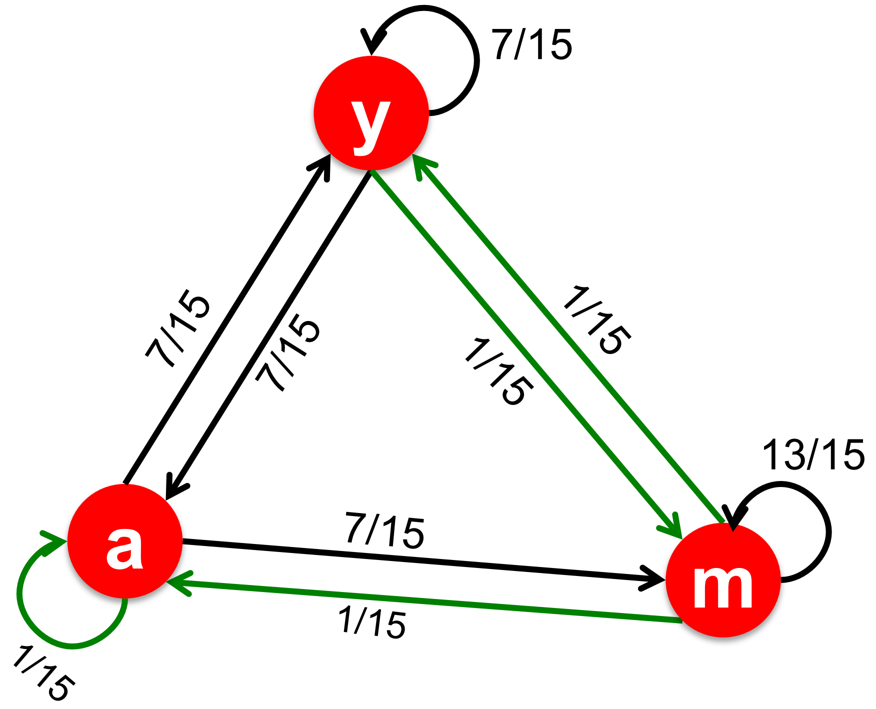
- **The Google Matrix *A*:**

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

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

- **We have a recursive problem: $\boldsymbol{r} = \boldsymbol{A} \cdot \boldsymbol{r}$**
  And the Power method still works!
- **What is β ?**
  - In practice $\beta = 0.8 \sim 0.9$ (make 5 steps on avg., jump)

Random Teleports ($\beta = 0.8$)



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

$$\begin{array}{c|ccc|} y & 7/15 & 7/15 & 1/15 \\ a & 7/15 & 1/15 & 1/15 \\ m & 1/15 & 7/15 & 13/15 \end{array}$$

A

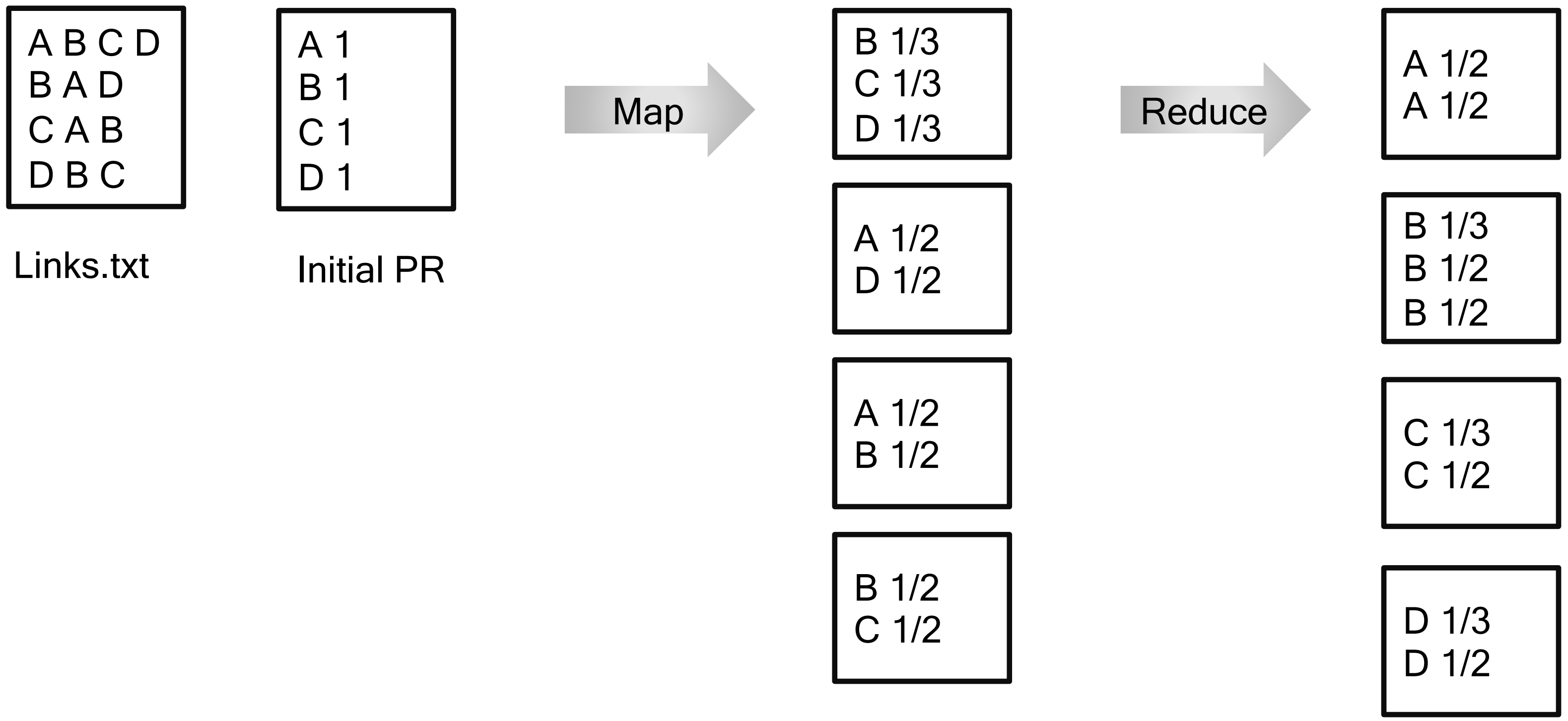
$$\begin{matrix} y \\ a \\ m \end{matrix} = \begin{matrix} 1/3 & 0.33 & 0.24 & 0.26 & & 7/33 \\ 1/3 & 0.20 & 0.20 & 0.18 & \ldots & 5/33 \\ 1/3 & 0.46 & 0.52 & 0.56 & & 21/33 \end{matrix}$$

MapReduce Program for PageRank

```
Map(key, value) {
    // key:        a page,
    // value:      page rank of the page

    For each page in Adj[key]
        emit(page, PR(key)/sizeof(Adj[key]);
}
```

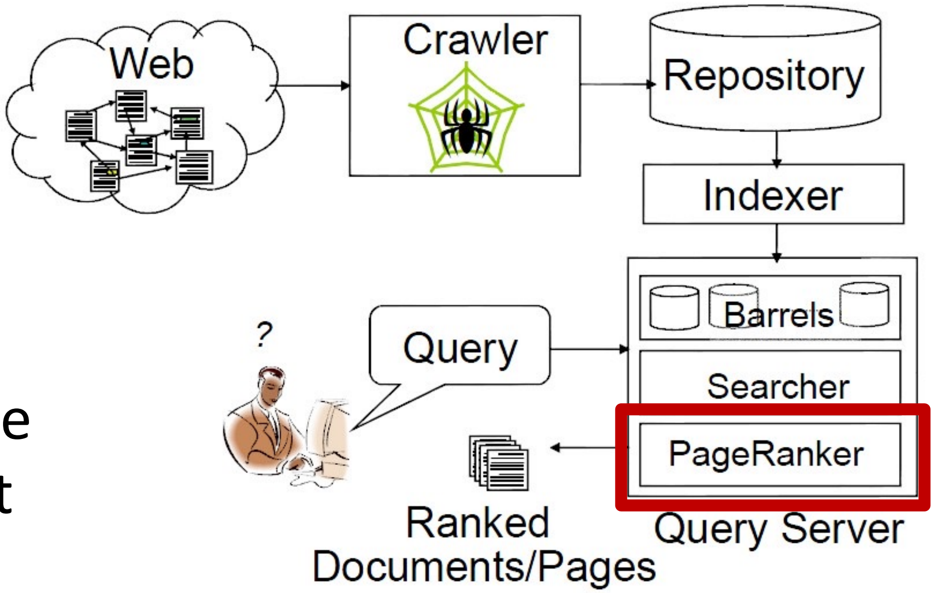
```
Reduce(key, values) {
    // key:        a page,
    // values:     a list of page ranks from all its incoming pages

    PR(key)=1-$\beta$;
    For each pagerank in values
        PR(key) = PR(key) + $\beta$*pagerank;
    emit(key, PR(key));
}
```

MapReduce Program for PageRank



Web Search Engines

- Indexer
  - Process the retrieved pages/documents and represents them in efficient search data structures (inverted files)
- Query Server
  - Accept the query from the user and return the result pages by consulting the search data structure


Web
Crawler
Repository
Indexer
Barrels
Searcher
PageRanker
?
Query
Ranked Documents/Pages
Query Server