CSCE 670 - Information Storage and Retrieval

Lecture 5: Link Analysis (PageRank)

Yu Zhang

yuzhang@tamu.edu

September 9, 2025

Course Website: https://yuzhang-teaching.github.io/CSCE670-F25.html

Adapted from the slides by Prof. Jure Leskovec (Stanford)

Recap: BM25

$$\text{BM25}(q,d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{\text{TF}(t,d) \cdot (k_1 + 1)}{\text{TF}(t,d) + k_1\left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)}$$

- k_1 controls term frequency scaling
  - $k_1 = 0$: binary model
  - k_1 very large: raw term frequency
- b controls document length normalization
  - $b = 0$: no document length normalization
  - $b = 1$: relative frequency (full document length normalization)
- Typically, k_1 is set between 1.2 and 2; b is set around 0.75

- $|d|$ is the length of d (in words); avgdl = average document length (in words)

Our Plan: Ranking

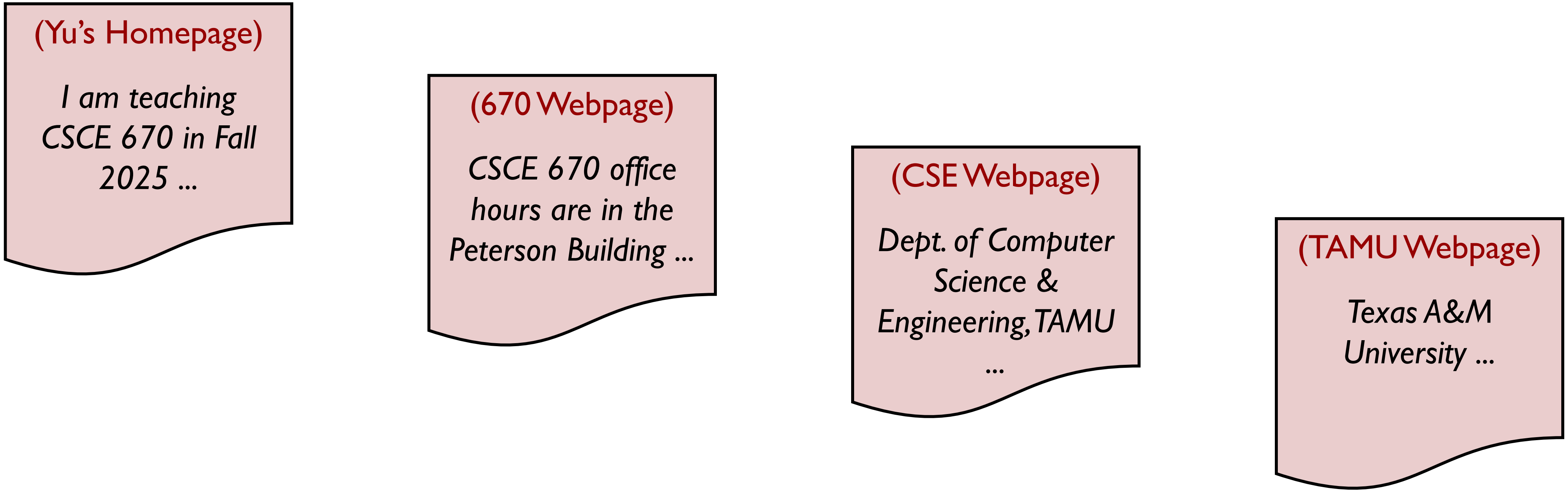
- ✅ Why is ranking important?
- ✅ What factors impact ranking?
- Two foundational text-based approaches
  - ✅ TF-IDF
  - ✅ BM25
- Two foundational link-based approaches
  - PageRank
  - HITS
- Machine-learned ranking ("learning to rank")

Recap: What factors impact ranking?

- Query: "*TAMU 2025 Fall Break*"
- Document 1: https://registrar.tamu.edu/academic-calendar/fall-2025
- Document 2: A social media post written by an account with 10 followers mentioning the time of TAMU 2025 Fall Break
- Document 1 should be ranked higher than Document 2 because it has a higher "reputation".
  - But how can we know the "reputation" of a website?

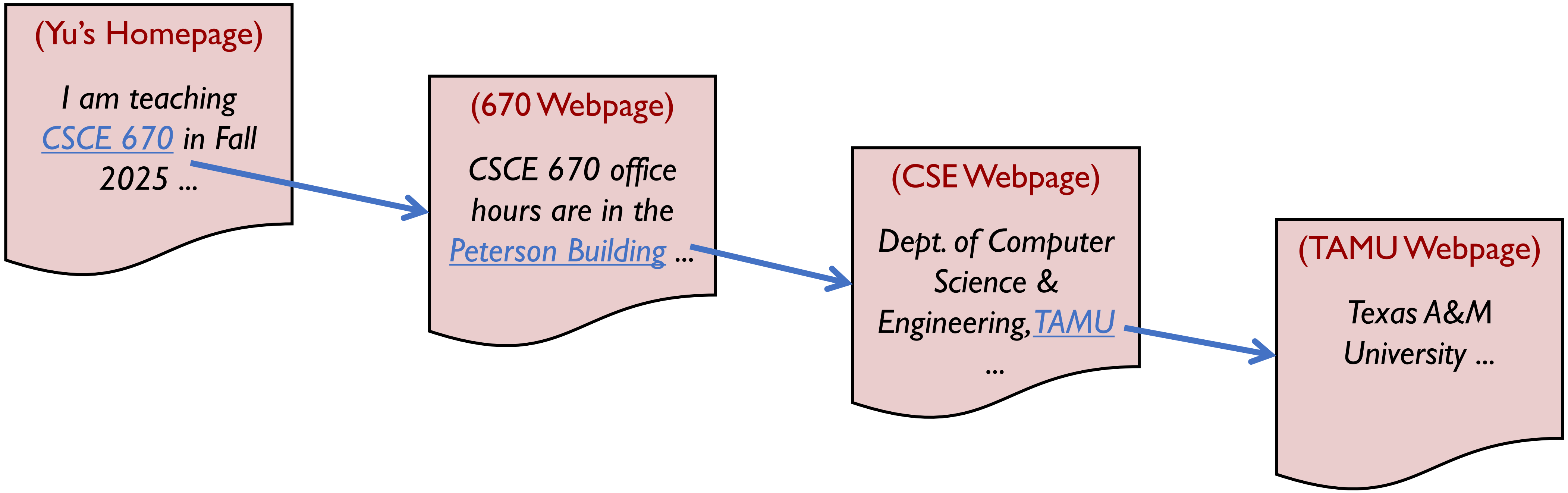
Web as a Directed Graph

- Nodes: Webpages

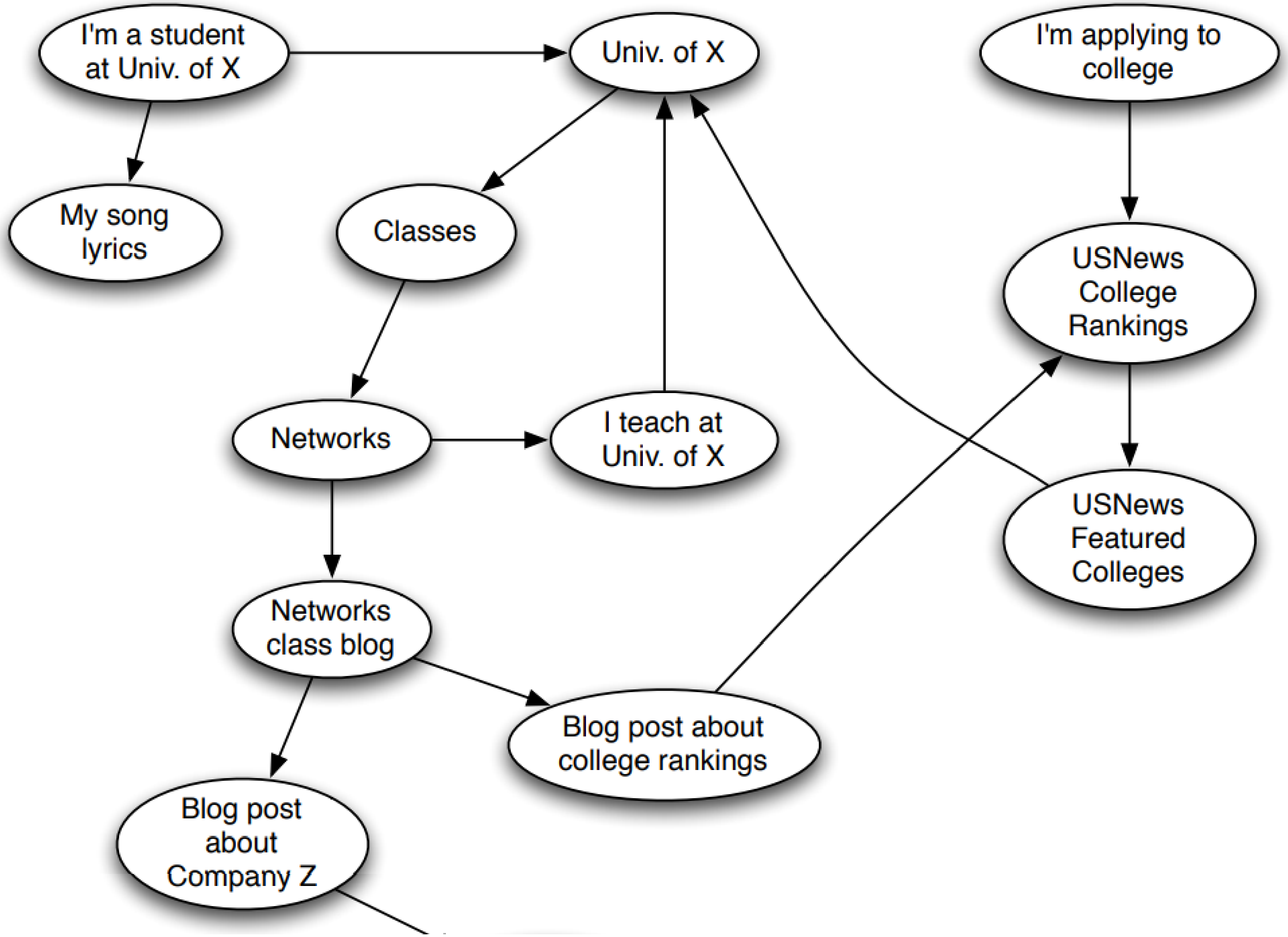


Web as a Directed Graph

- Nodes: Webpages
- Edges: Hyperlinks



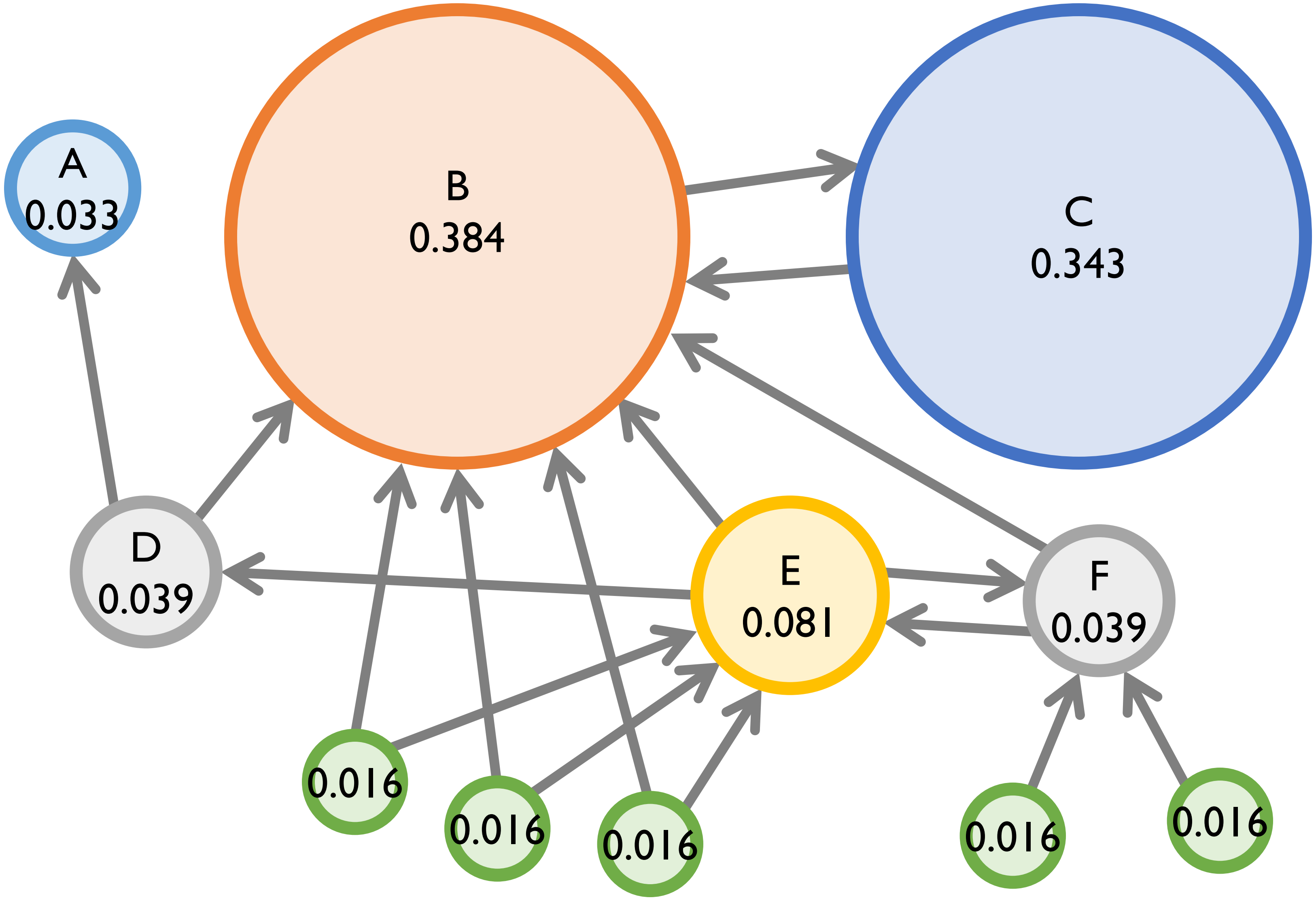
Web as a Directed Graph



Links as Votes

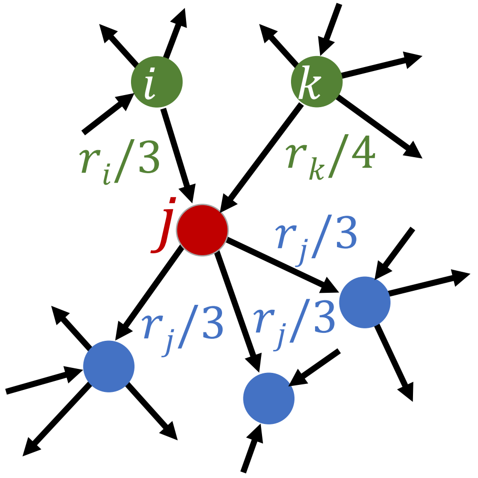
- Rough Idea: A webpage is more important if it has more links
  - In-coming links? Out-going links?
  - Out-going links can be easily manipulated by the webpage creator.
- Think of in-links as votes:
  - www.stanford.edu has 23,400 in-links
  - www.joe-schmoe.com has 1 in-link
- Are all in-links equal?
  - Links from important webpages count more.
  - Recursive question!

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 r_j/n votes
  - A vote from an important page is worth more.
- Page j's own importance is the sum of the votes on its in-links.
  - A page is important if it is pointed to by other important pages



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

In general, $$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

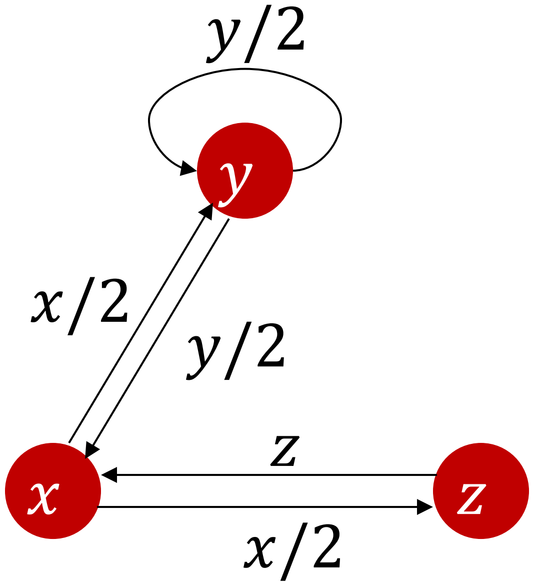
where d_i is the out-degree of i

Example

- $x = \frac{y}{2} + z$ (1)
- $y = \frac{y}{2} + \frac{x}{2}$ (2)
- $z = \frac{x}{2}$ (3)

- 3 equations, 3 unknowns. Looks like we can solve it!
- BUT if you add (1) and (2) together,
  - You will get (3).
  - Essentially, we have only 2 equations, so there exist infinitely many sets of solutions.

- Additional constraint forces uniqueness:
  - $x + y + z = 1$

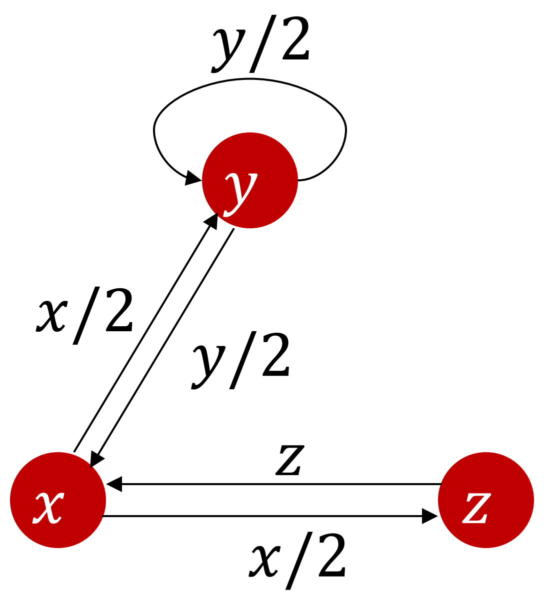


Example

- $x = \frac{y}{2} + z$ (1)
- $y = \frac{y}{2} + \frac{x}{2}$ (2)
- $x + y + z = 1$ (3)

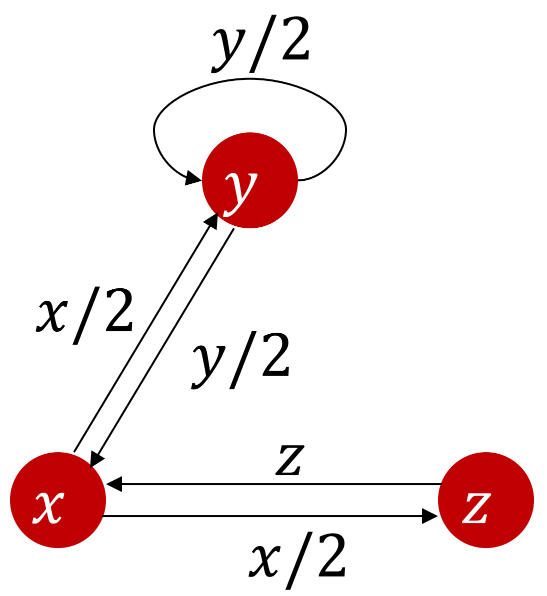
- Solution:
  - $x = \frac{2}{5}, y = \frac{2}{5}, z = \frac{1}{5}$.

- Gaussian elimination method works for small examples, but we need a better method for large web-size graphs.
  - We need a new formulation!



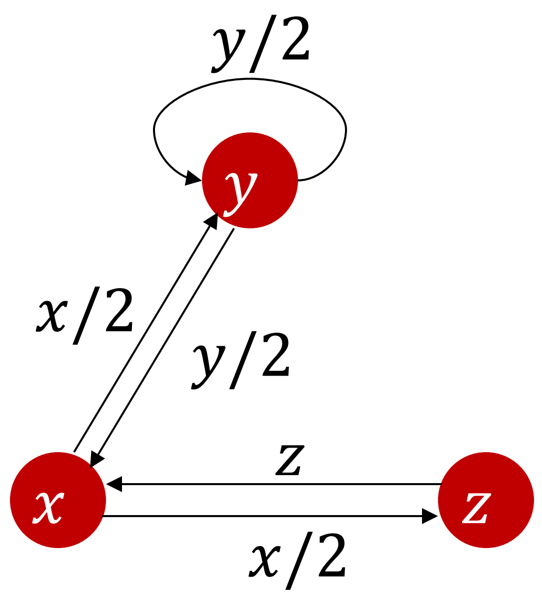
PageRank: Matrix Formulation

- Stochastic adjacency matrix $\boldsymbol{M}$
  - Assume page i has d_i out-links
  - If $i \rightarrow j$, then $M_{ji} = \frac{1}{d_i}$, else $M_{ji} = 0$.
  - Entries in each column of $\boldsymbol{M}$ sum to 1
  - Example: $\boldsymbol{M} = \begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$



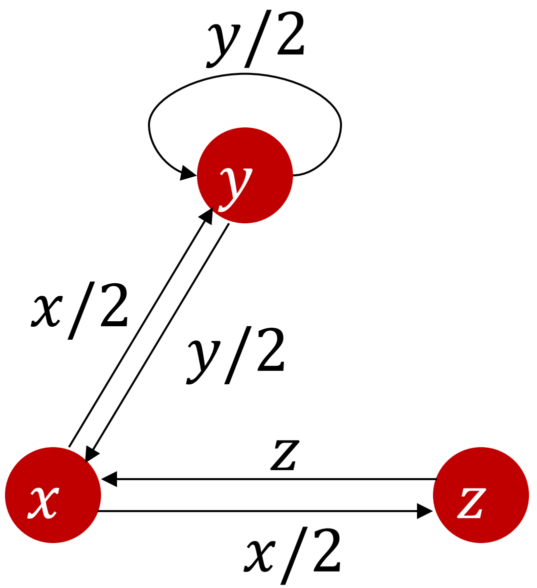
PageRank: Matrix Formulation

- Rank vector $\boldsymbol{r}$
  - r_i is the importance score of page i
  - Entries in $\boldsymbol{r}$ sum to 1
  - Example: $\boldsymbol{r} = \begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix}$



PageRank: Matrix Formulation

- Equations:
  - $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
  - Matrix form: $\boldsymbol{Mr} = \boldsymbol{r}$
  - Example: $\begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix} = \begin{bmatrix} 2/5 \\ 2/5 \\ 1/5 \end{bmatrix}$



- PageRank task:
  - Given the stochastic adjacency matrix $\boldsymbol{M}$, we need to find a rank vector $\boldsymbol{r}$ (whose entries sum to 1), so that

$$\boldsymbol{Mr} = \boldsymbol{r}$$

Solving $\boldsymbol{Mr} = \boldsymbol{r}$: Power Iteration Method

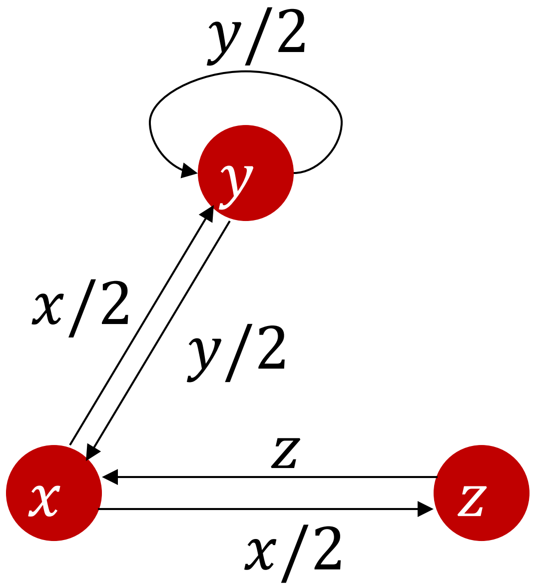
- *(Let's first assume this algorithm is correct. We will show why it works later.)*
- Power Iteration: a simple iterative scheme
  - Suppose there are N web pages in total
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots, 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$ (a very small number, e.g., 0.001)

- If the algorithm stops, we have a good solution $\boldsymbol{r}^{(t)}$
  - $\boldsymbol{M}\boldsymbol{r}^{(t)}$ is very close to $\boldsymbol{r}^{(t)}$

Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots., 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$

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

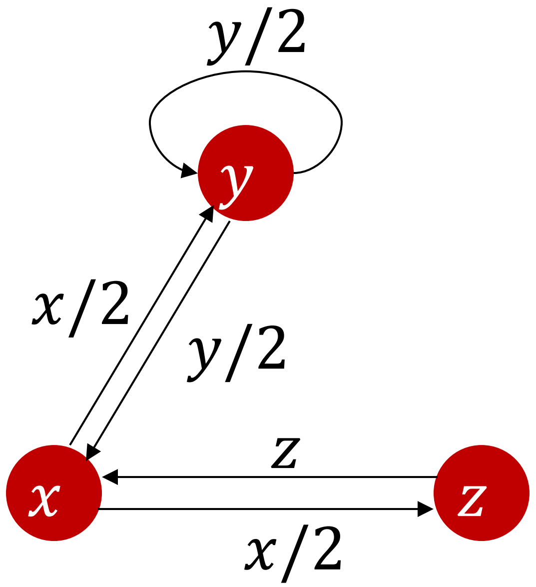


	$\boldsymbol{r}^{(0)}$
x	1/3 (0.33)
y	1/3 (0.33)
z	1/3 (0.33)

Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots, 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$

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

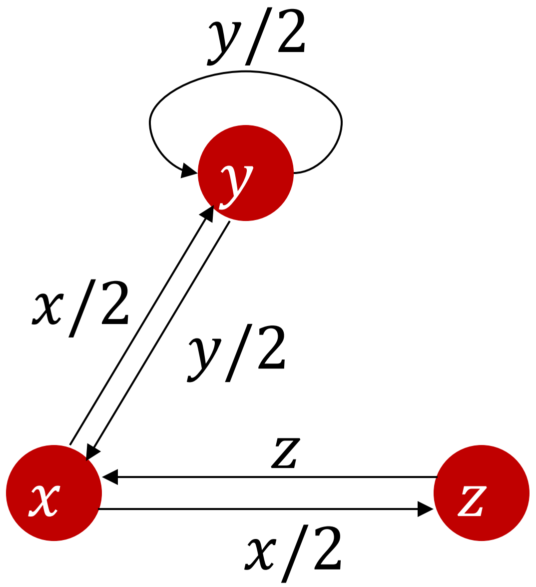


	$\boldsymbol{r}^{(0)}$	$\boldsymbol{r}^{(1)}$
x	1/3 (0.33)	1/2 (0.50)
y	1/3 (0.33)	1/3 (0.33)
z	1/3 (0.33)	1/6 (0.17)

Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \dots, 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\| < \epsilon$

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



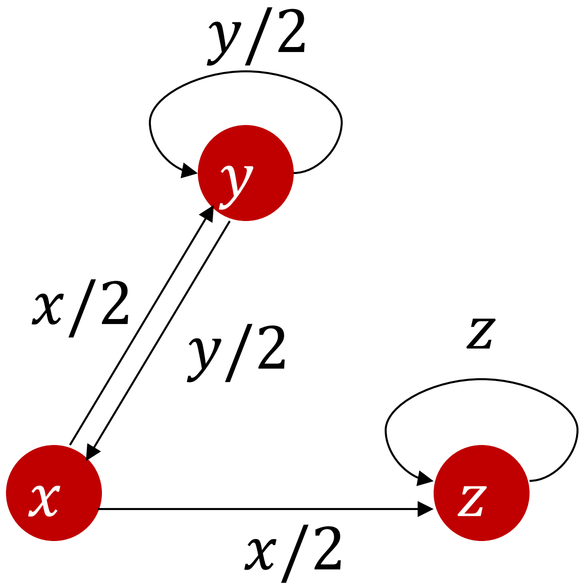
	$\boldsymbol{r}^{(0)}$	$\boldsymbol{r}^{(1)}$	$\boldsymbol{r}^{(2)}$	$\boldsymbol{r}^{(3)}$	…	Finally
x	1/3 (0.33)	1/2 (0.50)	1/3 (0.33)	11/24 (0.46)	…	0.40
y	1/3 (0.33)	1/3 (0.33)	5/12 (0.42)	3/8 (0.38)	…	0.40
z	1/3 (0.33)	1/6 (0.17)	1/4 (0.25)	1/6 (0.17)	…	0.20

Questions?

Another Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \dots, 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$

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

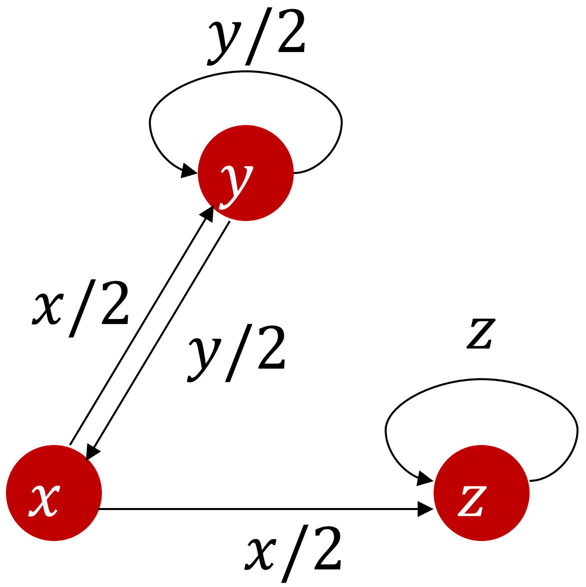


	$\boldsymbol{r}^{(0)}$
x	1/3 (0.33)
y	1/3 (0.33)
z	1/3 (0.33)

Another Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots., 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$

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



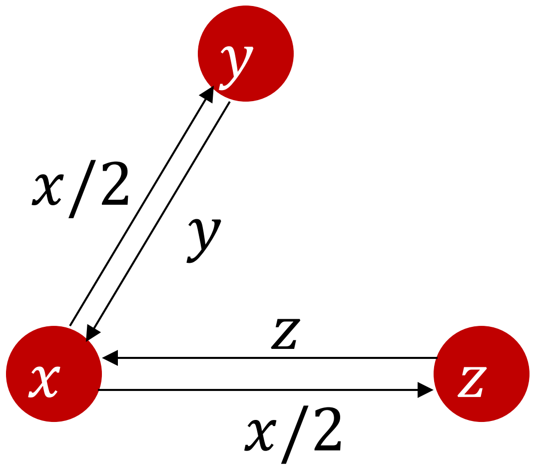
All the PageRank scores get "trapped" in node z.

	$\boldsymbol{r}^{(0)}$	$\boldsymbol{r}^{(1)}$	$\boldsymbol{r}^{(2)}$	$\boldsymbol{r}^{(3)}$	…	Finally
x	1/3 (0.33)	1/6 (0.17)	1/6 (0.17)	1/8 (0.13)	…	0.00
y	1/3 (0.33)	1/3 (0.33)	1/4 (0.25)	5/24 (0.21)	…	0.00
z	1/3 (0.33)	1/2 (0.50)	7/12 (0.58)	2/3 (0.67)	…	1.00

An Even Worse Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots., 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$

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



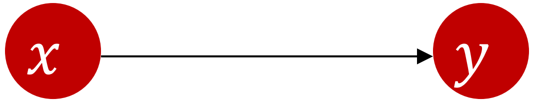
The algorithm falls into an infinite loop and will not terminate!
Root cause: the graph is bipartite.

	$\boldsymbol{r}^{(0)}$	$\boldsymbol{r}^{(1)}$	$\boldsymbol{r}^{(2)}$	$\boldsymbol{r}^{(3)}$	…	Finally
x	1/3	2/3	1/3	2/3	…	?
y	1/3	1/6	1/3	1/6	…	?
z	1/3	1/6	1/3	1/6	…	?

Yet Another Even Worse Example

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots., 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M}\boldsymbol{r}^{(t)}$
  - Stop when $\left\|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^{(t)}\right\| < \epsilon$

$$\boldsymbol{M} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

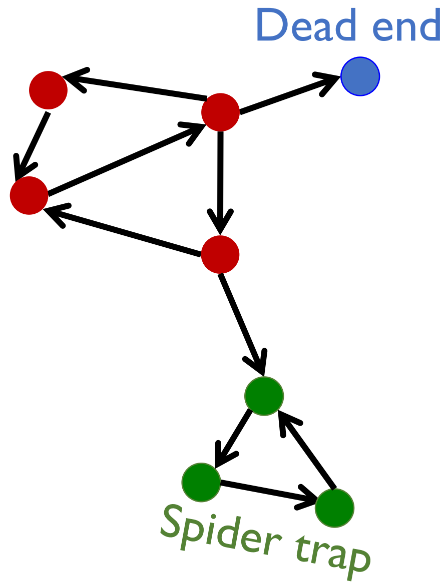


All the PageRank scores get "leaked"!
Root cause: the graph has a dead-end node (i.e., no out-links).

	$\boldsymbol{r}^{(0)}$	$\boldsymbol{r}^{(1)}$	$\boldsymbol{r}^{(2)}$	$\boldsymbol{r}^{(3)}$
x	1/2	0	0	0
y	1/2	1/2	0	0

Summary of the Challenges

- Spider traps
  - All out-links are within the group
  - Can have more than one node
  - Eventually spider traps absorb all importance
- Dead ends
  - The node has no out-links, therefore its importance score has nowhere to go
  - Eventually dead ends cause all importance to "leak out"
- Bipartite graph
  - If the graph is bipartite and the two partitions have different numbers of nodes, the algorithm will not converge.



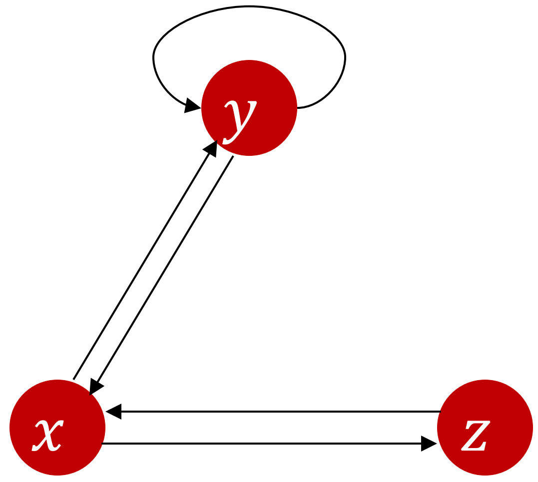
PageRank: Google Formulation

- Google's solution for spider traps: Teleportation!
  - Each node must contribute a portion of its importance score and distribute it evenly to all other nodes.

- Without teleports, $\boldsymbol{M} = \begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix}$

- With teleports, $\boldsymbol{M} = \beta \begin{bmatrix} 0 & 1/2 & 1 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \end{bmatrix} + (1-\beta) \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$

- In practice, $\beta = 0.8, 0.85$, or 0.9

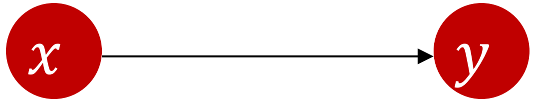


How about dead ends?

- Dead ends must contribute ALL of its importance score and distribute it evenly to all other nodes.

- Without teleports, $\boldsymbol{M} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$

- Without teleports, $\boldsymbol{M} = \beta \begin{bmatrix} 0 & 1/2 \\ 1 & 1/2 \end{bmatrix} + (1-\beta) \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{bmatrix}$



- Why do we call this solution “teleportation”?
  - Part of the importance score still flows according to the graph's defined neighborhoods
  - While the other part can instantly “teleport” to any node in the graph

Why does teleportation solve the problems?

- Spider traps: with traps, PageRank scores are not what we want
  - Solution: Never get stuck in a spider trap by teleporting out of it
- Dead ends: the matrix $\boldsymbol{M}$ is no longer column-stochastic (entries in a column may sum to 0 rather than 1)
  - Solution: Make $\boldsymbol{M}$ column-stochastic by always teleporting when there is nowhere else to go
- Wait, how about the bipartite-graph issue?
  - Teleportation makes the graph fully-connected (with different edge weights) and naturally non-bipartite.

PageRank: Google Formulation [Brin and Page, WWW 1998]

- Node-wise form:

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

- Note 1: Each node i in the graph teleports a score of $(1-\beta)\frac{1}{N}r_i$ to node j, so the total score node j receives through teleportation is exactly $(1-\beta)\frac{1}{N}\sum_i r_i = (1-\beta)\frac{1}{N}$.

- Note 2: This formulation assumes the graph has no dead ends. If there is a dead end, we can first link it to all the nodes (include itself).

PageRank: Google Formulation [Brin and Page, WWW 1998]

- Matrix form:

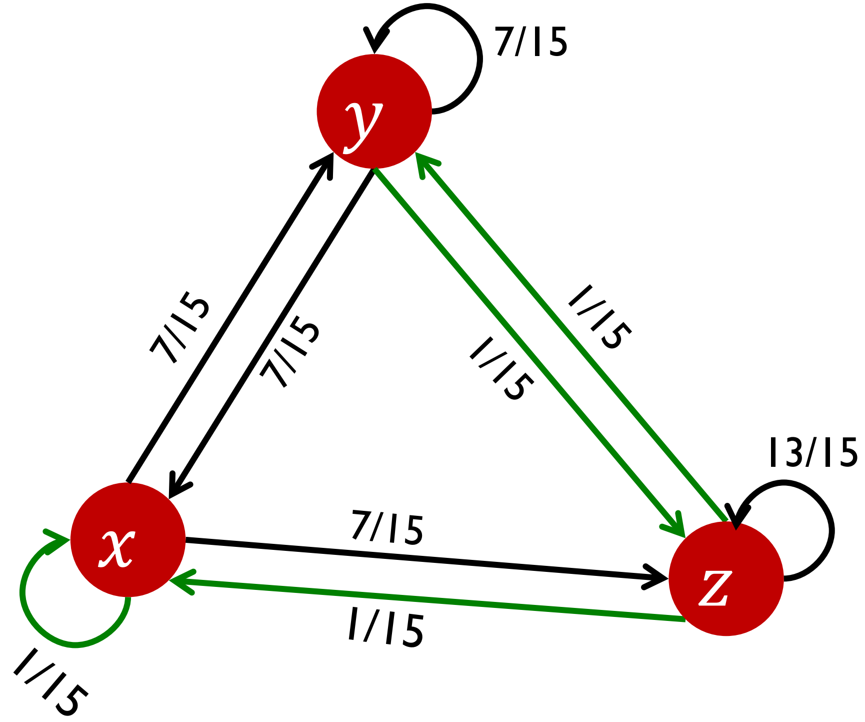
$$\boldsymbol{A} = \beta \boldsymbol{M} + (1 - \beta) \frac{\mathbf{1}}{N}$$

- Note: $\mathbf{1}$ is an $N \times N$ matrix where all entries are 1.

- Now we need to solve $\boldsymbol{Ar} = \boldsymbol{r}$
  - We can still use Power Iteration

Example ($\beta = 0.8$)

$$A = 0.8 \times \begin{bmatrix} 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \end{bmatrix} + 0.2 \times \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$
$$= \begin{bmatrix} 1/15 & 7/15 & 1/15 \\ 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 13/15 \end{bmatrix}$$



	$r^{(0)}$	$r^{(1)}$	$r^{(2)}$	$r^{(3)}$	…	Finally
x	1/3	0.20	0.20	0.18	…	0.15
y	1/3	0.33	0.28	0.26	…	0.21
z	1/3	0.47	0.52	0.56	…	0.64

Extended Content
(will not appear in quizzes or the exam)

Why does Power Iteration work?

- $\boldsymbol{Ar} = \boldsymbol{r}$
- In other words, $\boldsymbol{r}$ is an eigenvector of $\boldsymbol{A}$ with the corresponding eigenvalue $\lambda = 1$

- Why does $\boldsymbol{A}$ necessarily have an eigenvalue of 1?
- How about other eigenvalues of $\boldsymbol{A}$?

- Perron–Frobenius Theorem: Let $\boldsymbol{A}$ be a square matrix with all entries strictly positive, and entries in each column sum to 1, then
  - $\boldsymbol{A}$ has an eigenvalue of 1
  - 1 is the unique "largest" eigenvalue of $\boldsymbol{A}$. That is, for all other eigenvalues λ of $\boldsymbol{A}$, we have $|\lambda| < 1$.

Why does Power Iteration work?

- Power Iteration:
  - Initialize: $\boldsymbol{r}^{(0)} = [1/N, \ldots., 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{A}\boldsymbol{r}^{(t)}$

$$\boldsymbol{r}^{(1)} = \boldsymbol{A}\boldsymbol{r}^{(0)}$$
$$\boldsymbol{r}^{(2)} = \boldsymbol{A}\boldsymbol{r}^{(1)} = \boldsymbol{A}\left(\boldsymbol{A}\boldsymbol{r}^{(1)}\right) = \boldsymbol{A}^2\boldsymbol{r}^{(0)}$$
$$\boldsymbol{r}^{(3)} = \boldsymbol{A}\boldsymbol{r}^{(2)} = \boldsymbol{A}\left(\boldsymbol{A}^2\boldsymbol{r}^{(0)}\right) = \boldsymbol{A}^3\boldsymbol{r}^{(0)}$$
$$\ldots$$

- We have a sequence of vectors $\boldsymbol{A}\boldsymbol{r}^{(0)}, \boldsymbol{A}^2\boldsymbol{r}^{(0)}, \boldsymbol{A}^3\boldsymbol{r}^{(0)}, \ldots$
- We need to prove that this sequence converges to the eigenvector of $\boldsymbol{A}$ with the eigenvalue $\lambda = 1$

Proof

- Let's assume $\boldsymbol{A}$ has eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_N$, where $1 = \lambda_1 > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_N|$
- The eigenvectors corresponding to $\lambda_1, \lambda_2, \dots, \lambda_N$ are $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$
  - Let's also assume that $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ are linearly independent
  - If $\lambda_1, \lambda_2, \dots, \lambda_N$ are different from each other, this assumption always holds.

- $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ form a basis, so we can write $\boldsymbol{r}^{(\mathbf{0})} = c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2 + \cdots + c_N\boldsymbol{x}_N$
- $\boldsymbol{A}\boldsymbol{r}^{(\mathbf{0})} = \boldsymbol{A}(c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2 + \cdots + c_N\boldsymbol{x}_N)$
  $= c_1\boldsymbol{A}\boldsymbol{x}_1 + c_2\boldsymbol{A}\boldsymbol{x}_2 + \cdots + c_N\boldsymbol{A}\boldsymbol{x}_N$
  $= c_1\lambda_1\boldsymbol{x}_1 + c_2\lambda_2\boldsymbol{x}_2 + \cdots + c_N\lambda_N\boldsymbol{x}_N$

- Repeated multiplication on both sides
- $\boldsymbol{A}^2\boldsymbol{r}^{(\mathbf{0})} = c_1\lambda_1^2\boldsymbol{x}_1 + c_2\lambda_2^2\boldsymbol{x}_2 + \cdots + c_N\lambda_N^2\boldsymbol{x}_N$
- $\boldsymbol{A}^k\boldsymbol{r}^{(\mathbf{0})} = c_1\lambda_1^k\boldsymbol{x}_1 + c_2\lambda_2^k\boldsymbol{x}_2 + \cdots + c_N\lambda_N^k\boldsymbol{x}_N$

Proof

- Let's assume $\boldsymbol{A}$ has eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_N$, where $1 = \lambda_1 > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_N|$
- The eigenvectors corresponding to $\lambda_1, \lambda_2, \dots, \lambda_N$ are $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$

- Repeated multiplication on both sides
- $\boldsymbol{A}^k \boldsymbol{r}^{(\mathbf{0})} = c_1 \lambda_1^k \boldsymbol{x}_1 + c_2 \lambda_2^k \boldsymbol{x}_2 + \cdots + c_N \lambda_N^k \boldsymbol{x}_N$

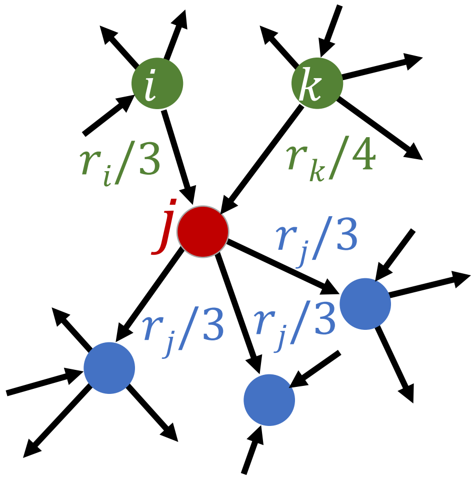
$$= \lambda_1^k \left(c_1 \boldsymbol{x}_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k \boldsymbol{x}_2 + \cdots + c_N \left(\frac{\lambda_N}{\lambda_1}\right)^k \boldsymbol{x}_N \right)$$

- Note that $\left|\left(\frac{\lambda_i}{\lambda_1}\right)^k\right| = \left|\frac{\lambda_i}{\lambda_1}\right|^k \to 0$ when $k \to \infty$ (because $|\lambda_i| < |\lambda_1|$)
- Therefore, $\boldsymbol{A}^k \boldsymbol{r}^{(\mathbf{0})} \to \lambda_1^k (c_1 \boldsymbol{x}_1 + 0 + \cdots + 0) = c_1 \boldsymbol{x}_1$, which is the eigenvector of $\boldsymbol{A}$ with the eigenvalue $\lambda_1 = 1$.

Note: This proof does not apply to the case where $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ are NOT linearly independent, which may happen when $\boldsymbol{A}$ does not have N distinct eigenvalues.

PageRank: Random Walk Interpretation

- Imagine there is a random web surfer
  - At time t, the surfer is on a page i
  - At time $t + 1$, the surfer has two options
    - With probability β, it follows an out-link from i uniformly at random (i.e., ends up on some page j linked from i)
    - With probability $1 - \beta$, it jumps to a random page in the graph (can be i, j, or any other node)



- The process repeats indefinitely
- Let $\boldsymbol{p}(t)$ be the vector whose i-th coordinate is the probability that the surfer is at page i at time t
  - So $\boldsymbol{p}(t)$ is a probability distribution over pages

The Stationary Distribution

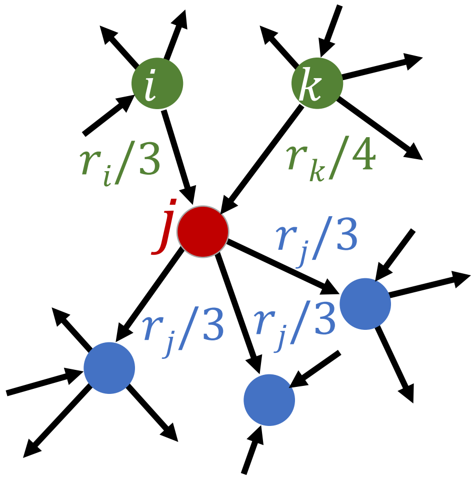
- Where is the surfer at time $t + 1$?

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

- Suppose the random walk reaches a state

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

then $\boldsymbol{p}(\boldsymbol{t})$ is stationary distribution for the random walk



- The PageRank vector $\boldsymbol{r}$ satisfies $\boldsymbol{r} = \boldsymbol{A} \cdot \boldsymbol{r}$
  - So $\boldsymbol{r}$ is a stationary distribution for the random walk

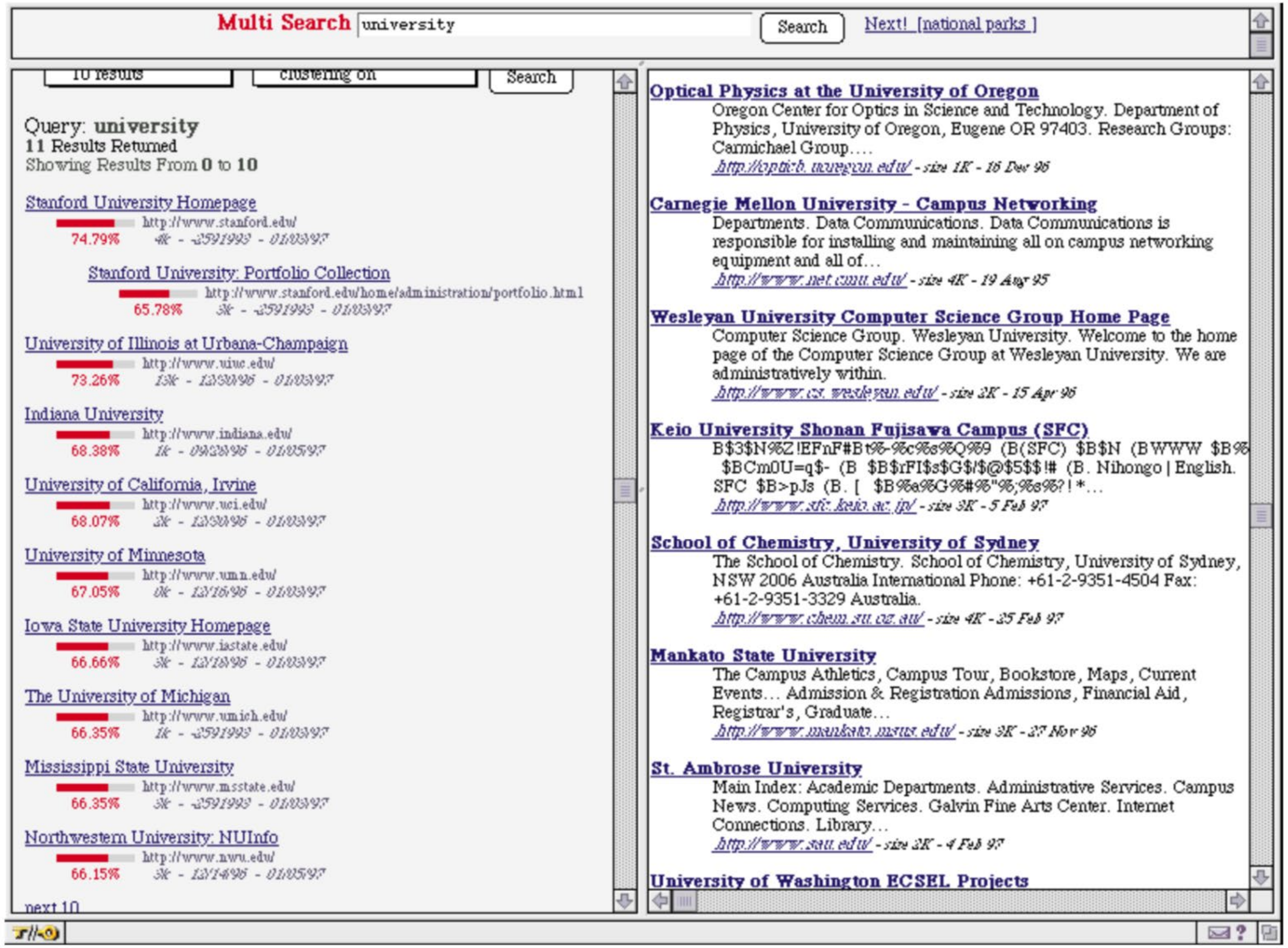
A central result from the theory of random walks (Markov processes):
For graphs that satisfy certain conditions (connected and non-bipartite), the stationary distribution is unique and eventually will be reached no matter what the initial probability distribution is at time $t = 0$

Back to the Broader Story of Ranking

Boolean + PageRank results for the query "*university*" [Page et al., 1999]

- With the rise of the Web, traditional text-based signals (e.g., TF-IDF and BM25) may not be sufficient.

- Many early web search engines relied on classic text-based ranking plus some rudimentary link-based signals.



Back to the Broader Story of Ranking

- In practice, we will build a scoring function that considers many features.
- Typically, we have:
  - Query-dependent features: e.g., TF-IDF, BM25, # of times a query word occurs in a document, …
  - Query-independent features: e.g., PageRank, # of in-links to a webpage, popularity of an album, …
    - Many query-independent features are proxies for "reputation"

- How to jointly consider these features?
  - Week 5

Thank You!

Course Website: https://yuzhang-teaching.github.io/CSCE670-F25.html