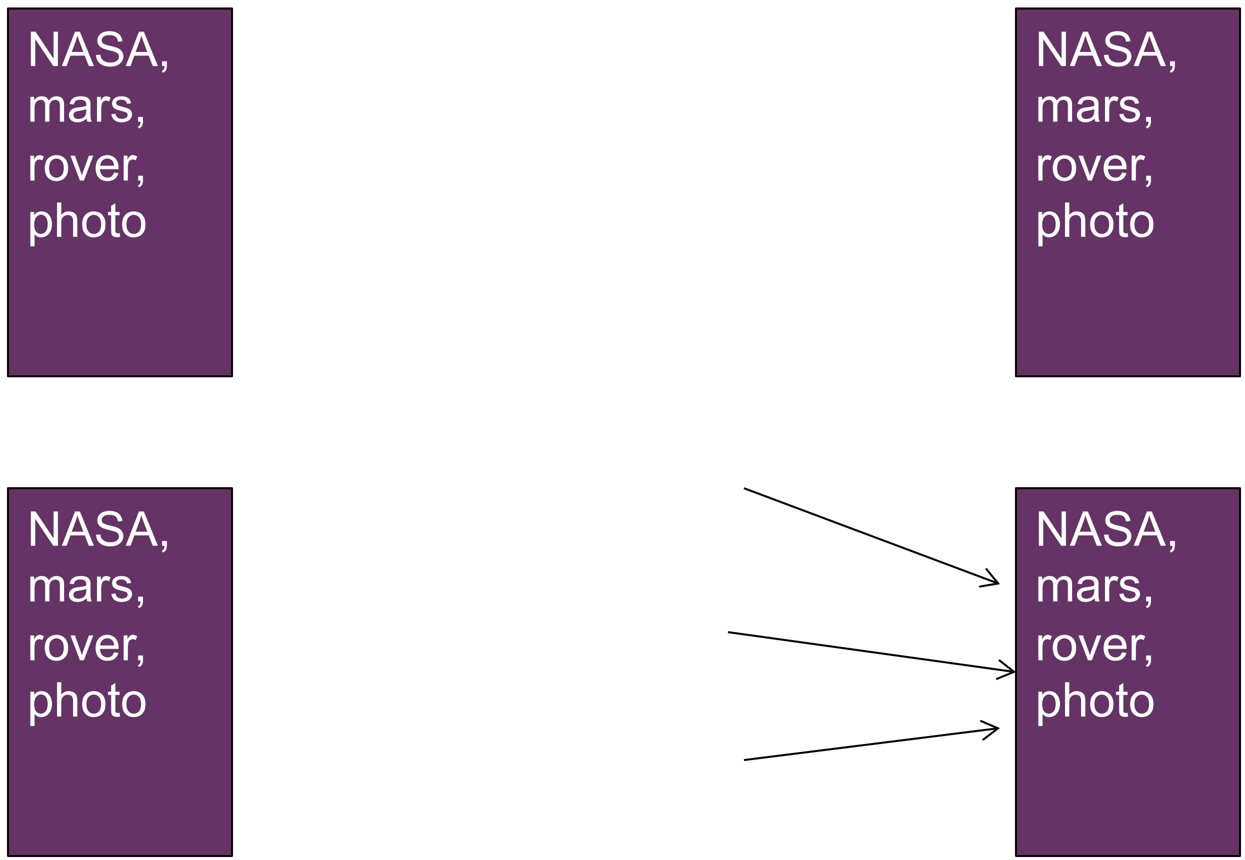
Recommender Systems
Node Ranking

Professor Robin Burke
Spring 2019

Node Ranking

- Non-personalized case
- Suppose our items are nodes in a directed network
  - We retrieve some of the items based on their features
  - But we want to rank them by "importance"
- Search engine problem
  - Not enough to retrieve pages based on content
    - Adversarial issues

Problem: Adversarial IR



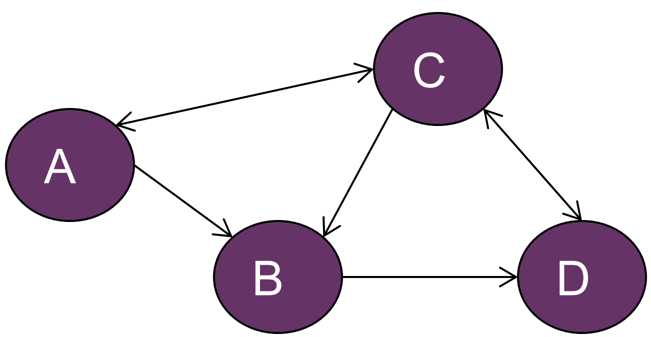
“Random surfer” model

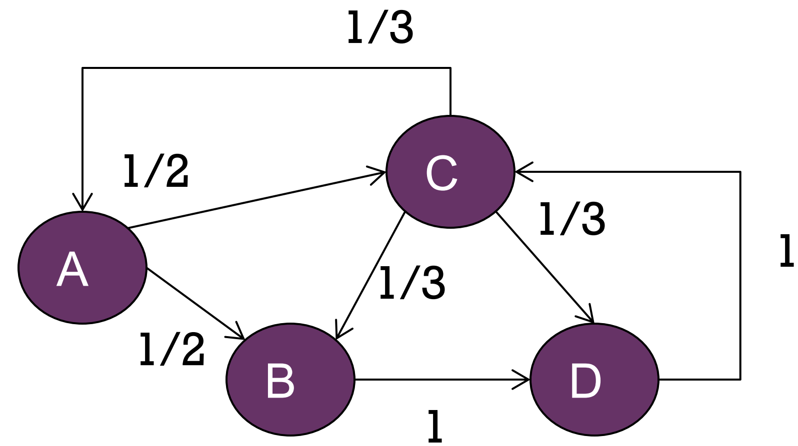
- We conduct a random walk on the web
- The number of times that we encounter a page = its “importance”

Matrix definition

- Instead of an adjacency matrix
  - we need a "transition probability" matrix
- A Markov model
  - for each node,
    - equal probability of moving to all linked nodes
- New adjacency matrix
  - each row is divided by degree

Weight = probability = 1/d





$$M = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad N = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Markov model

- A model of a process that can be in multiple states
- Transitions between states are defined by links
  - the transitions are probabilistic
- Always add up to 1
  - leaving any node

After k steps?

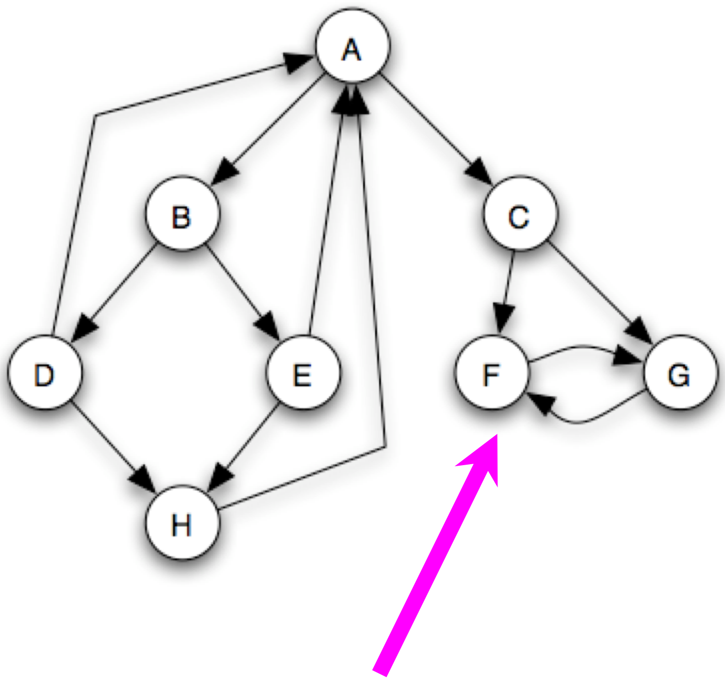
- $v_k = (N^T)^k v_0$
- We are interested in convergence
- So, at what vector v* does
  - $v^* = (N^T) v^*$
  - probability must sum to 1

Eigenvectors

- In a Markov matrix
  - Only one real positive non-zero eigenvalue = 1
- That's our (Basic) PageRank
  - Named after Larry Page

Problem

- Random surfer might get "stuck"



No way out...

Fix

- Add a random jump probability
- Each iteration
  - small probability of jumping to a random node anywhere in the network
  - if not, then random surf
    - choose an edge randomly from the current node
- This is enough to avoid sinks

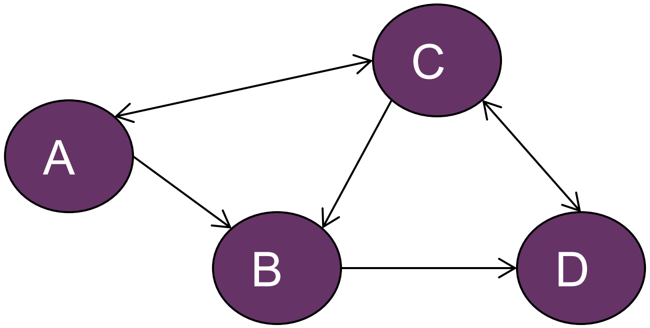
New equation

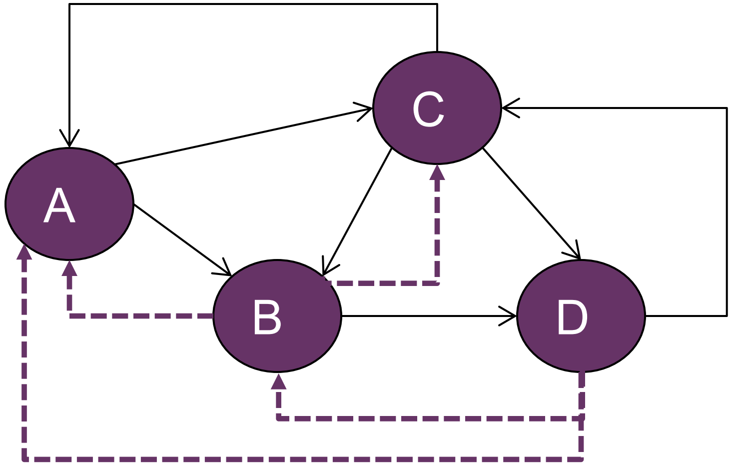
- $v_t = Pv_{t-1}$
- where P has entries

$$p_{i,j} = \begin{cases} \alpha \dfrac{m_{i,j}}{deg_{out}(i)}, if\ m_{i,j} = 1 \\ \dfrac{\beta}{n - deg_{out}(i)}, if\ m_{i,j} = 0 \end{cases}$$

- $\alpha+\beta=1$
  - β is the random jump probability
    - 0.15 suggested in the original article
  - Still probabilistic, but now non-connected nodes can be reached
- Solve as before
  - eigenvalues of new matrix
  - largest eigenvector = PageRank vector
  - Now the matrix is dense
    - Solution is harder to compute efficiently

Weight = $p_{i,j}$





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

$$P = \begin{bmatrix} 0.075 & 0.425 & 0.425 & 0.075 \\ 0.05 & 0.05 & 0.05 & 0.85 \\ 0.283 & 0.283 & 0.15 & 0.283 \\ 0.05 & 0.05 & 0.85 & 0.05 \end{bmatrix}$$

PageRank

- Similar to eigenvector centrality
  - accounts for "attention" effect
    - 1 friend in 50 < 1 friend in 5
- Makes sense in information networks
  - models connected edges
- A node is central IF
  - if it connected to other central nodes
  - who are selective
- Very effective for adversarial IR
- But NOT personalized
  - The same for all users

One way that spammers try to get their pages ranked more highly is to create “link farms” where they create many web pages with links to a desired target page. What would you expect relative to this technique?

- A. It won’t work very well because the pages linking to the target page won’t have high PageRank themselves.
- B. It will work well because the target page will attract a lot of PageRank value by having such high in-degree.
- C. It will work well because the link farm creates a circular system that traps the PageRank value and allows it to build up.
- D. It will not work well because such link configurations can be easily detected.

Personalized PageRank

- Instead of jumping to a random location
  - Jump back to a "root" node
  - Has the effect of finding "important" items in a given network neighborhood
- Example
  - You could use this to recommend people to follow on LinkedIn
  - Random surfer model
    - But always re-starting at the user node
    - Personalized PageRank
  - Nodes with a high Personalized PageRank
    - Would be nodes highly associated with user's network neighborhood

Personalized equation

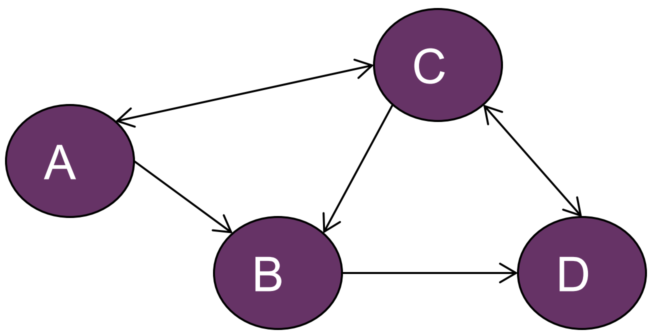
Original PageRank for direct neighbors

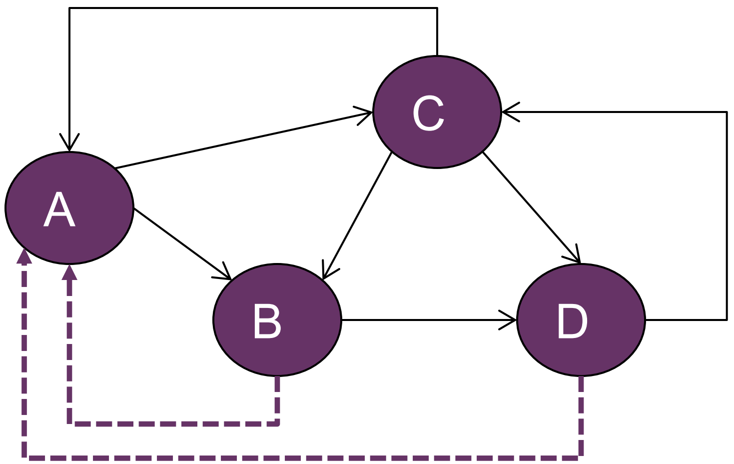
- $v_t = P_k v_{t-1}$
- where P_k has entries

$$p_{i,j} = n_{i,j}, if\ m_{i,k} = 1, else$$
$$\alpha \frac{m_{i,j}}{deg_{out}(i)}, if\ m_{i,j} = 1$$
$$\beta, if\ m_{i,j} = 0\ and\ j = k$$
$$0, otherwise$$

- $\alpha+\beta=1$
  - β is the random jump probability
    - 0.15 suggested in the original article
  - Still probabilistic, but now non-connected nodes can be reached

Weight = $P_A(i,j)$





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

$$P_A = \begin{bmatrix} 0.15 & 0.425 & 0.425 & 0.0 \\ 0.15 & 0.0 & 0.0 & 0.85 \\ 0.333 & 0.333 & 0.0 & 0.333 \\ 0.15 & 0.0 & 0.85 & 0.0 \end{bmatrix}$$

Personalized PageRank

- Also has a solution
  - Still Markov matrix
  - Still has largest eigenvalue = 1
  - Use associated eigenvector
- Different solution for each user

+

In a network with approx. 10,000 nodes, there is one node D that has high (regular) PageRank, approx. 10x larger than the next highest node. You compute the personalized PageRank (PPR) for two randomly chosen nodes A and B in the network. You would expect

- A. $PPR_A(D)$ would be very different from $PPR_B(D)$
- B. $PPR_A(D)$ would be similar to $PPR_B(D)$ but quite different from PR(D)
- C. $PPR_A(D)$, $PPR_B(D)$ and PR(D) would all be similar in value
- D. It depends on the network's structure