Web Information Retrieval

Lecture 12

Link analysis for ranking

Today's lecture

- Link analysis for ranking
  - Pagerank and variants
  - HITS

Why Link Analysis?

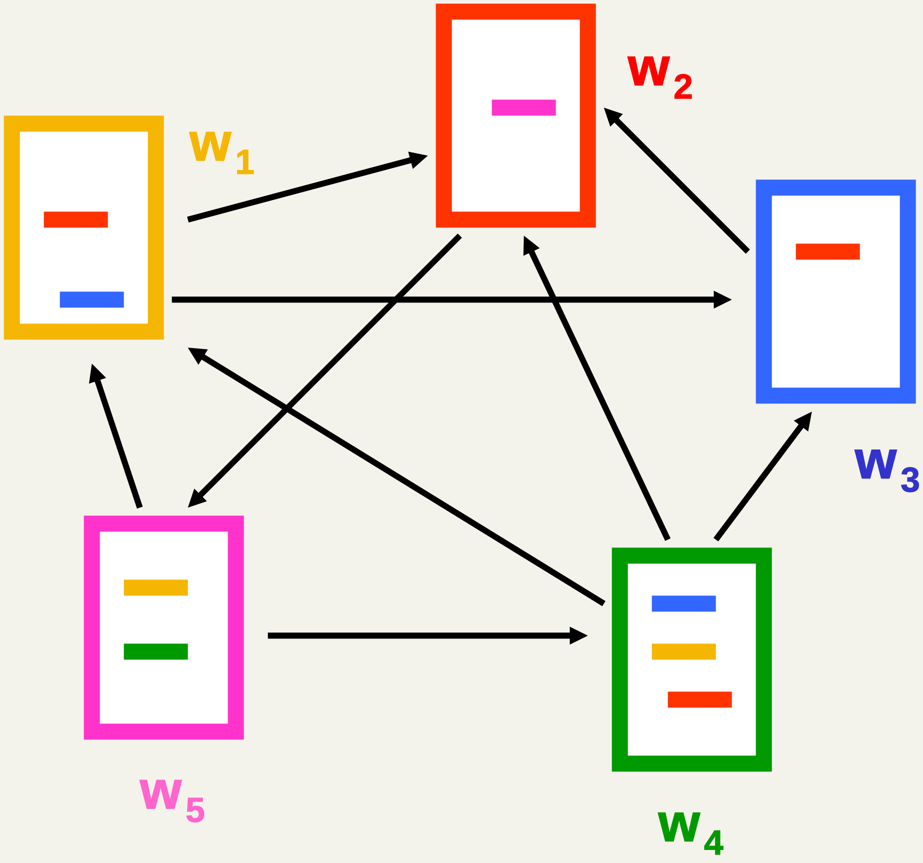
- First generation search engines
  - view documents as flat text files
  - could not cope with size, spamming, user needs
- Second generation search engines
  - Ranking becomes critical
  - use of Web specific data: Link Analysis
  - shift from relevance to authoritativeness
  - a success story for the network analysis

Link Analysis for ranking: Intuition

- A link from page p to page q denotes endorsement
  - page p considers page q an authority on a subject
  - mine the web graph of recommendations
  - assign an authority value to every page

Link Analysis Ranking Algorithms

- Start with a collection of web pages
- Extract the underlying hyperlink graph
- Run the LAR algorithm on the graph
- Output: an authority weight for each node



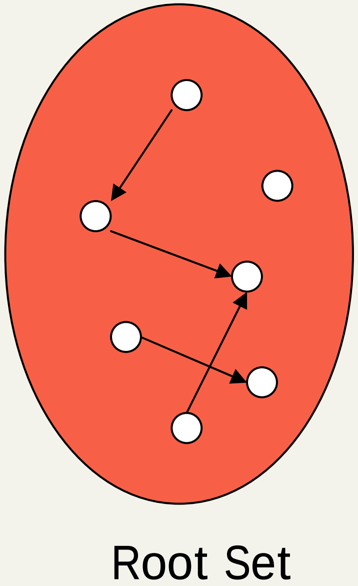
Algorithm input

- Query independent: rank the whole Web
  - PageRank (Brin and Page 98) was proposed as query independent
- Query dependent: rank a small subset of pages related to a specific query
  - HITS (Kleinberg 98) was proposed as query dependent

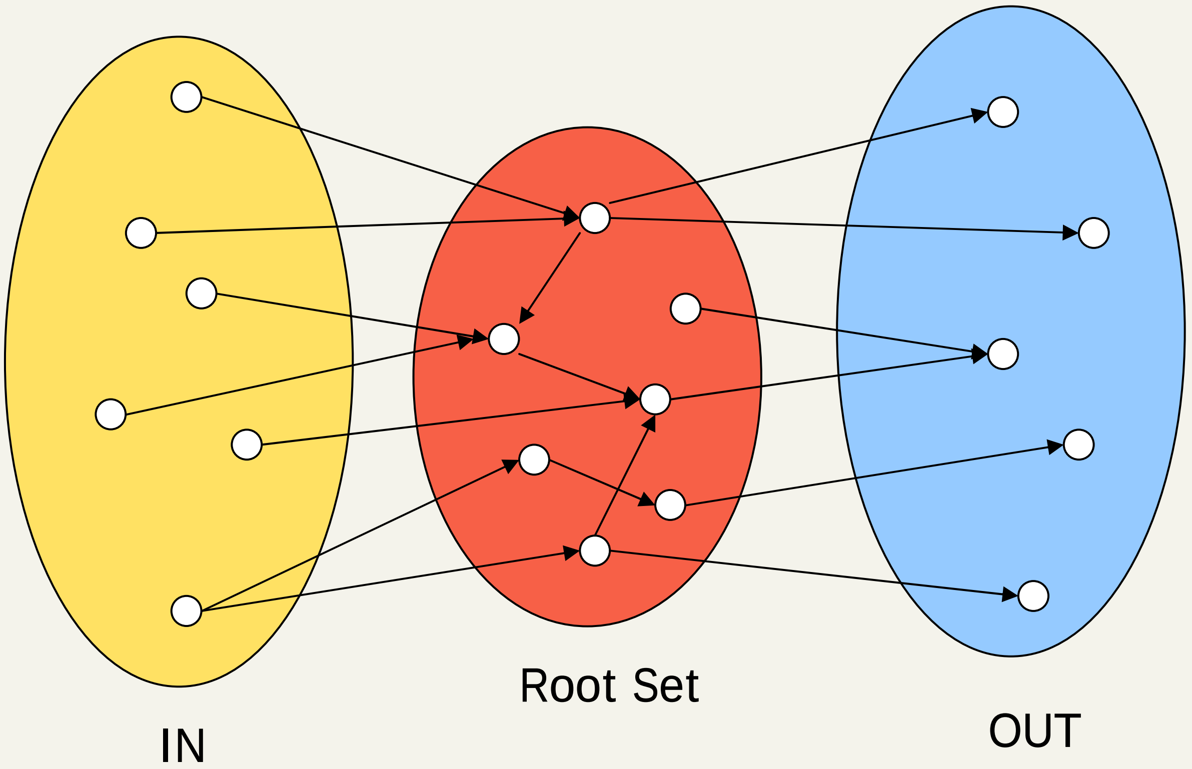
Query dependent analysis

- First retrieve all pages meeting the text query (say ***venture capital***).
- Order these by their link popularity

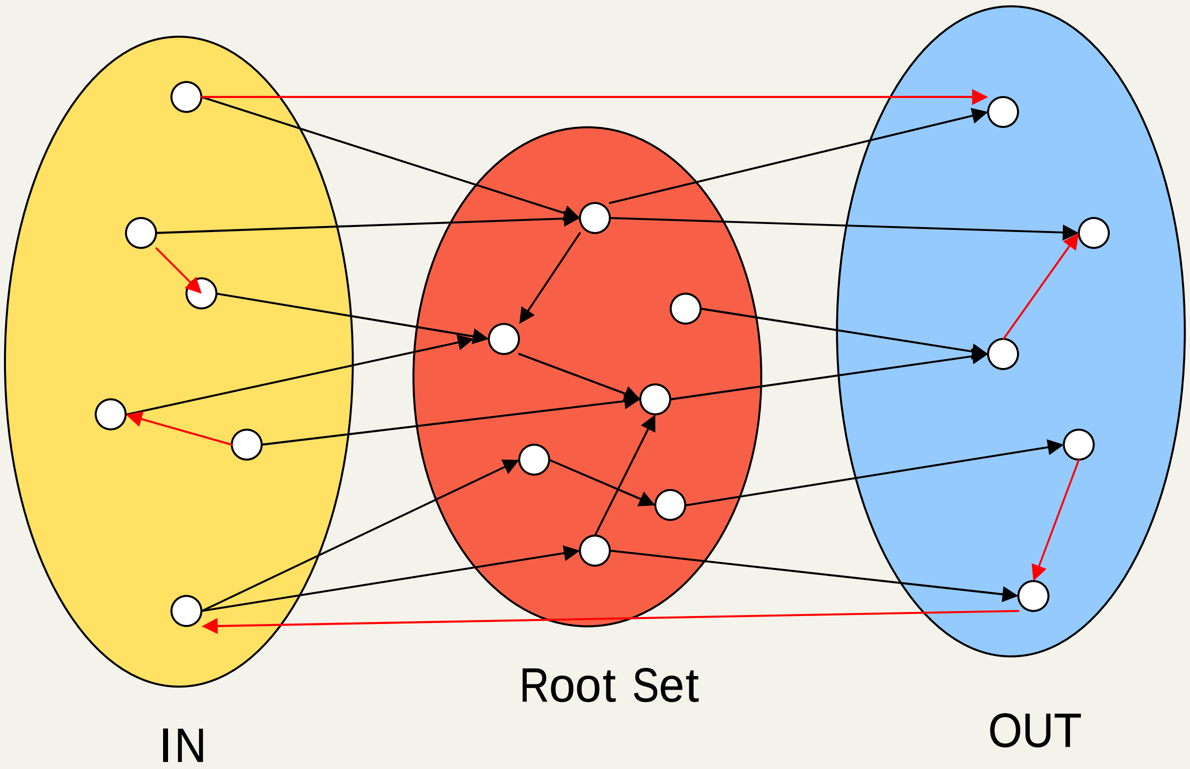
Query dependent input



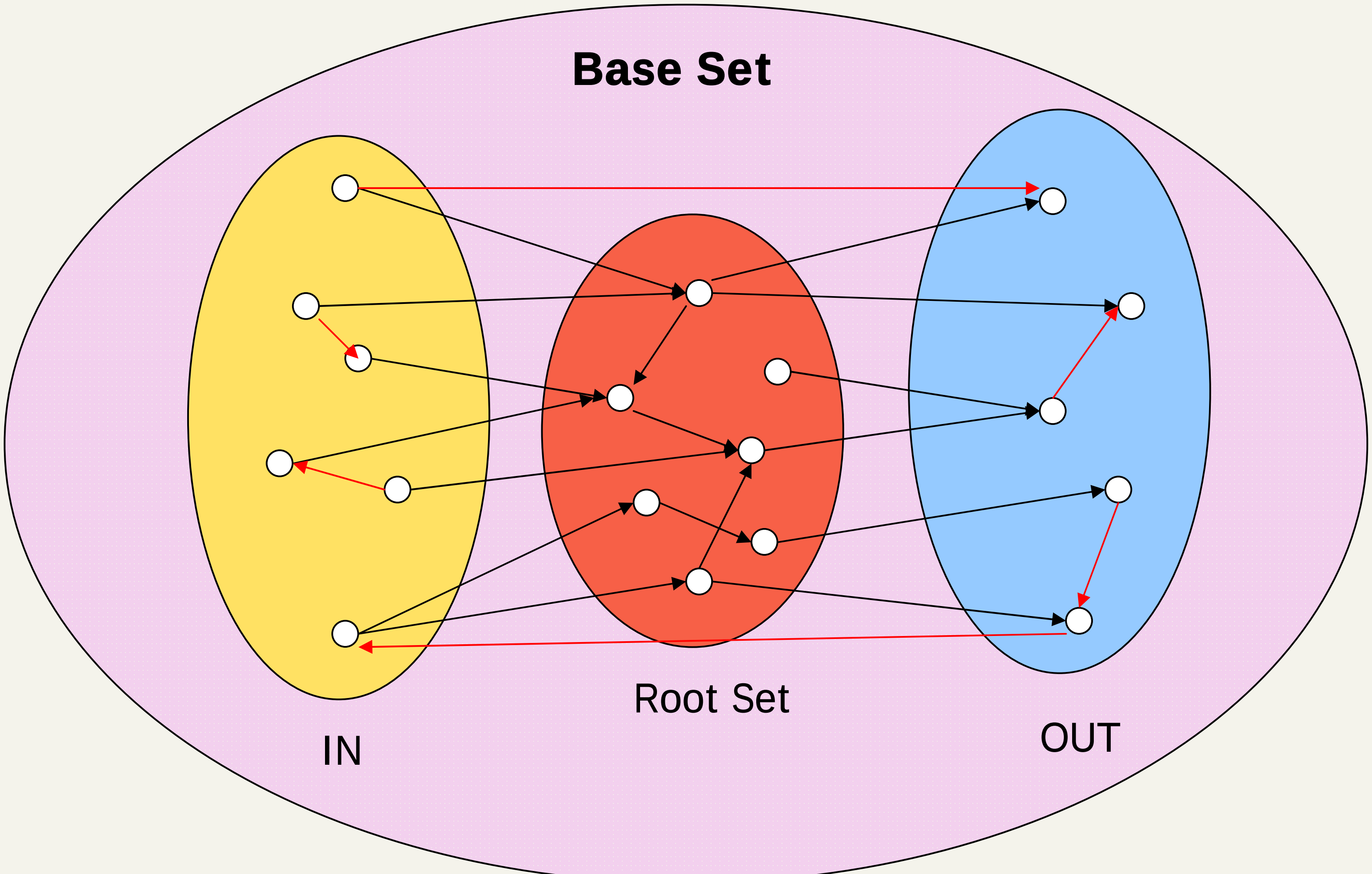
Query dependent input



Query dependent input



Query dependent input



Previous work

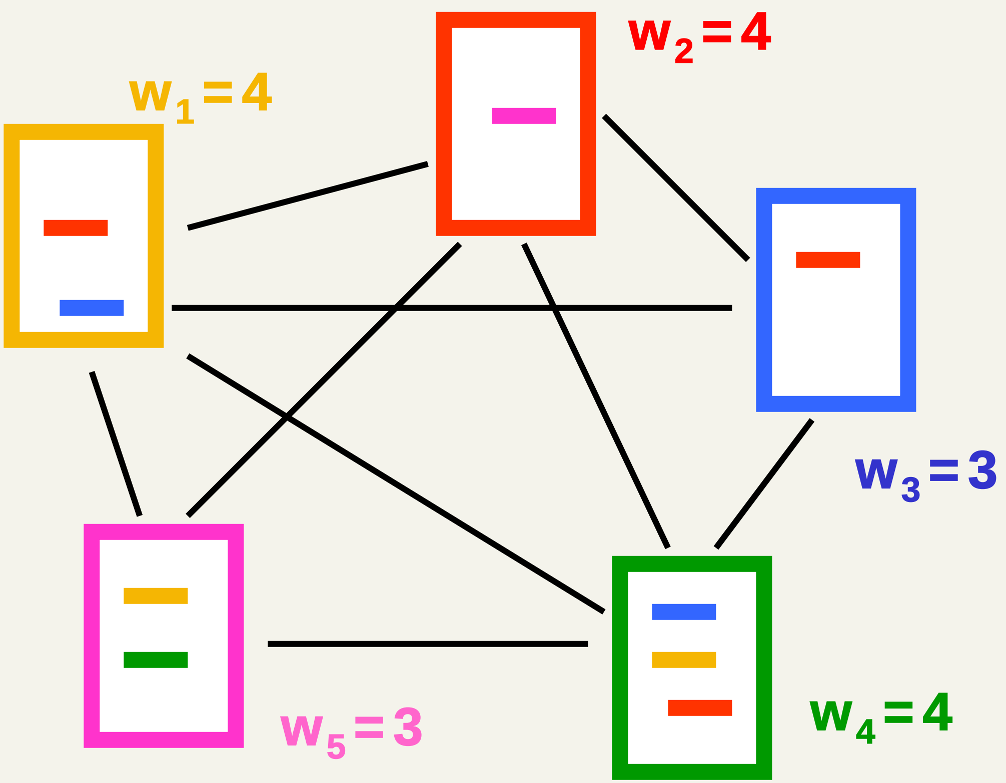
- The problem of identifying the most important nodes in a network has been studied before in social networks and bibliometrics
- The idea is similar
  - A link from node p to node q denotes endorsement
  - mine the network at hand
  - assign an centrality/importance/standing value to every node

Citation Analysis

- Citation frequency
- Co-citation coupling frequency
  - Cocitations with a given author measures “impact”
  - Cocitation analysis [Mcca90]
    - Convert frequencies to correlation coefficients, do multivariate analysis/clustering, validate conclusions
    - E.g., cocitation in the “Geography and GIS” web shows communities [Lars96]
- Bibliographic coupling frequency
  - Articles that co-cite the same articles are related
- Citation indexing
  - Who is a given author cited by? (Garfield [Garf72])
    - E.g., Science Citation Index (*http://www.isinet.com/*)
    - CiteSeer (*http://citeseer.ist.psu.edu*) [Lawr99a]
- Pagerank preview: Pinsker and Narin ‘60s

Undirected popularity

- Rank pages according to degree
  - w_i = | degree(i) |



1. Red Page
2. Yellow Page
3. Blue Page
4. Purple Page
5. Green Page

Spamming undirected popularity

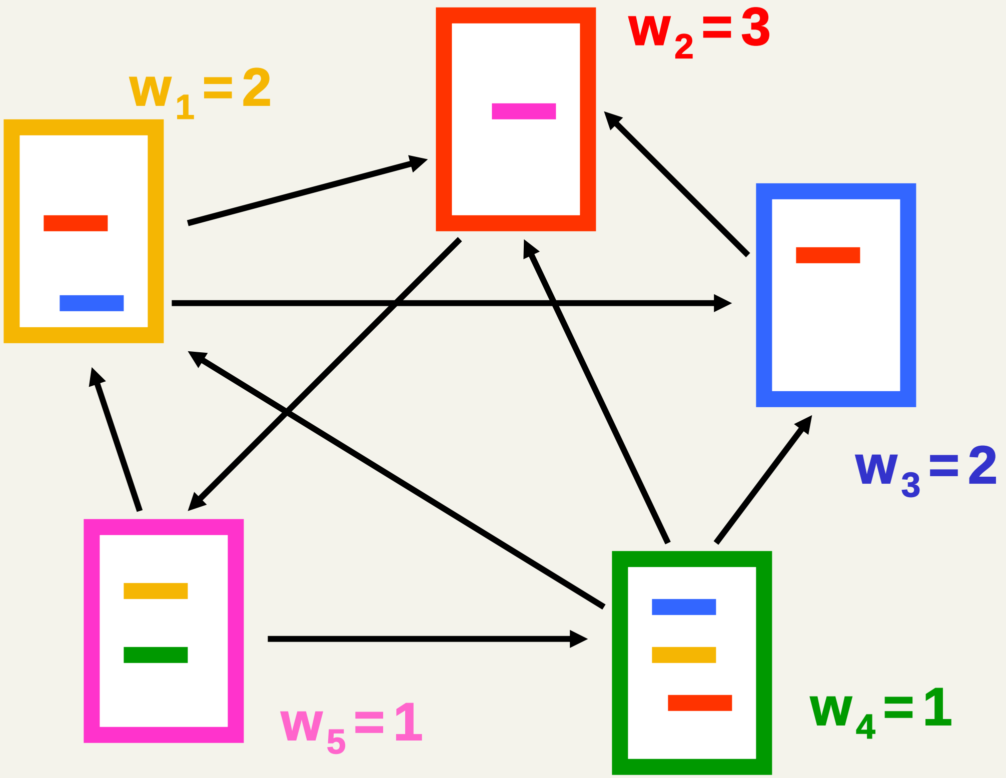
- *Exercise*: How do you spam the undirected popularity heurestic

Spamming undirected popularity

- *Exercise*: How do you spam the undirected popularity heurestic
- Add a lot of outlinks

Directed popularity

- Rank pages according to in-degree
  - w_i = | indegree(i) |



1. Red Page
2. Yellow Page
3. Blue Page
4. Purple Page
5. Green Page

Spamming directed popularity

- *Exercise*: How do you spam the directed popularity heurestic

Spamming directed popularity

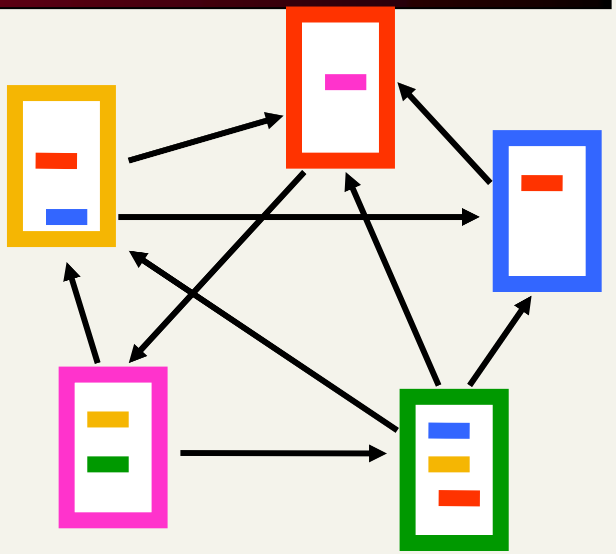
- *Exercise*: How do you spam the directed popularity heurestic

- Create a lot of web pages
- Add links to the page of interest

PageRank algorithm

High-level idea:

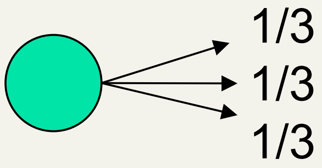
- A good page has a lot of endorsements by important (authoritative) pages
- **Good** authorities should be pointed by **good** authorities
- Count number of votes, but votes have different weights that depends on who votes for them, and so on
- Motivated also by the **random-surfer** model



1. **Red Page**
2. **Purple Page**
3. **Yellow Page**
4. **Blue Page**
5. **Green Page**

Pagerank scoring

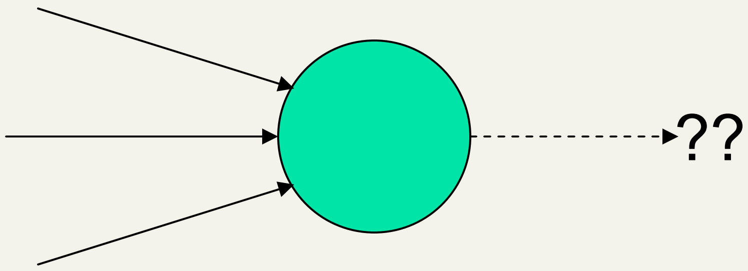
- Imagine a browser doing a random walk on web pages:
  - Start at a random page



  - At each step, go out of the current page along one of the links on that page, equiprobably
- "In the steady state" each page has a long-term visit rate - use this as the page's score.

Not quite enough

- The web is full of dead-ends.
  - Random walk can get stuck in dead-ends.
  - Makes no sense to talk about long-term visit rates.



Teleporting

- At a dead end, jump to a random web page.
- At any non-dead end, with probability α = 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - α = 10% – a parameter

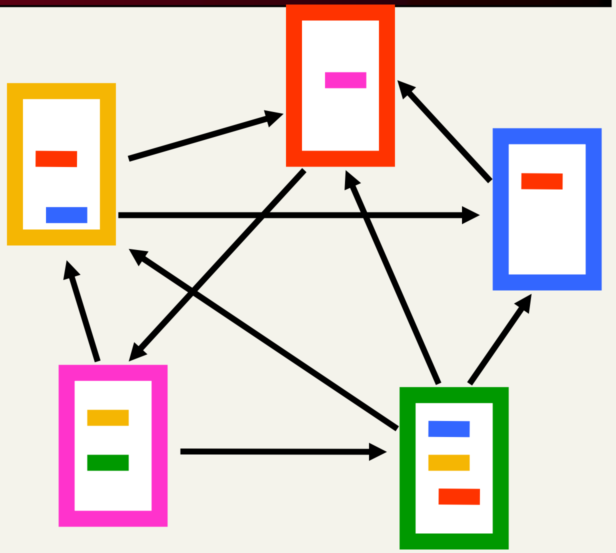
Result of teleporting

- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited (not obvious, will show this).
- How do we compute this visit rate?

PageRank algorithm

- Good authorities should be pointed by good authorities
- Random walk on the web graph
  - pick a page at random
  - Repeat
    - If dead end jump to a random page
    - with probability α jump to a random page
    - with probability 1-α follow a random outgoing link

- Pagerank weight of page **p** = Probability to be at page **p**

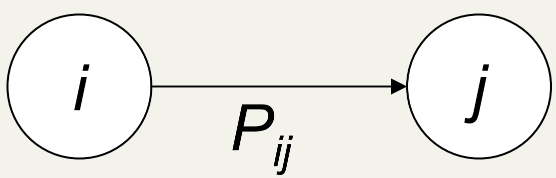


1. Red Page
2. Purple Page
3. Yellow Page
4. Blue Page
5. Green Page

Markov chains

- A Markov chain consists of n **states**, plus an $n \times n$ **transition probability matrix** **P**.
- At each step, we are in exactly one of the states.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.

$P_{ii}>0$ is OK.



Markov chains

- A Markov chain describes a **discrete time stochastic process** over a set of **states**

$$S = \{s_1, s_2, \dots s_n\}$$

according to a **transition probability** matrix

$$s_i \xrightarrow{P_{ij}} s_j$$

$$P = \{P_{ij}\}$$

  - P_{ij} = probability of moving to state s_j when at state s_i
    - $\sum_j P_{ij} = 1$ (**stochastic matrix**)

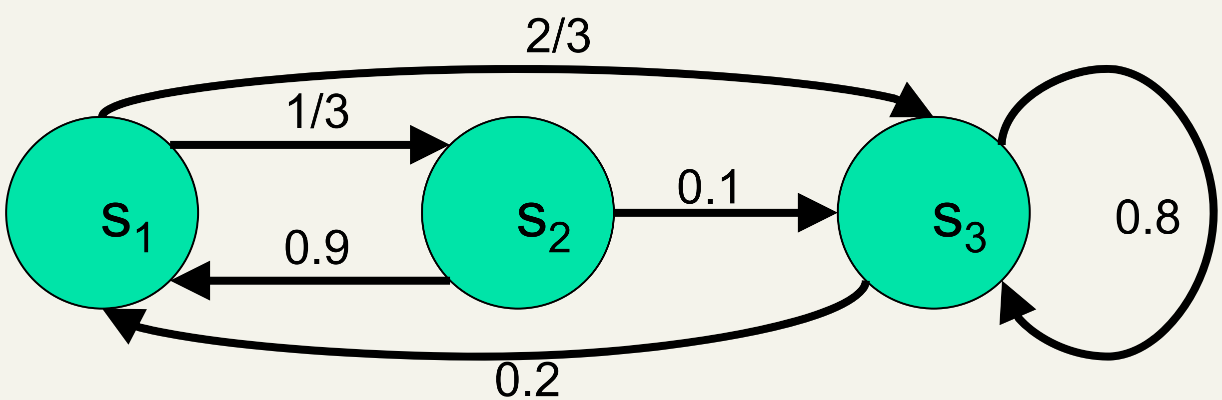
$P_{ii}>0$ is OK

- **Memorylessness property**: The next state of the chain depends only at the current state and not on the past of the process
- Markov chains are abstractions and generalizations of **random walks**.

Markov chain graph

- Often we represent a Markov chain as a graph
- Nodes = states
- Edge weights = transition probabilities

$$P = \begin{bmatrix} 0 & 1/3 & 2/3 \\ 0.9 & 0 & 0.1 \\ 0.2 & 0 & 0.8 \end{bmatrix}$$



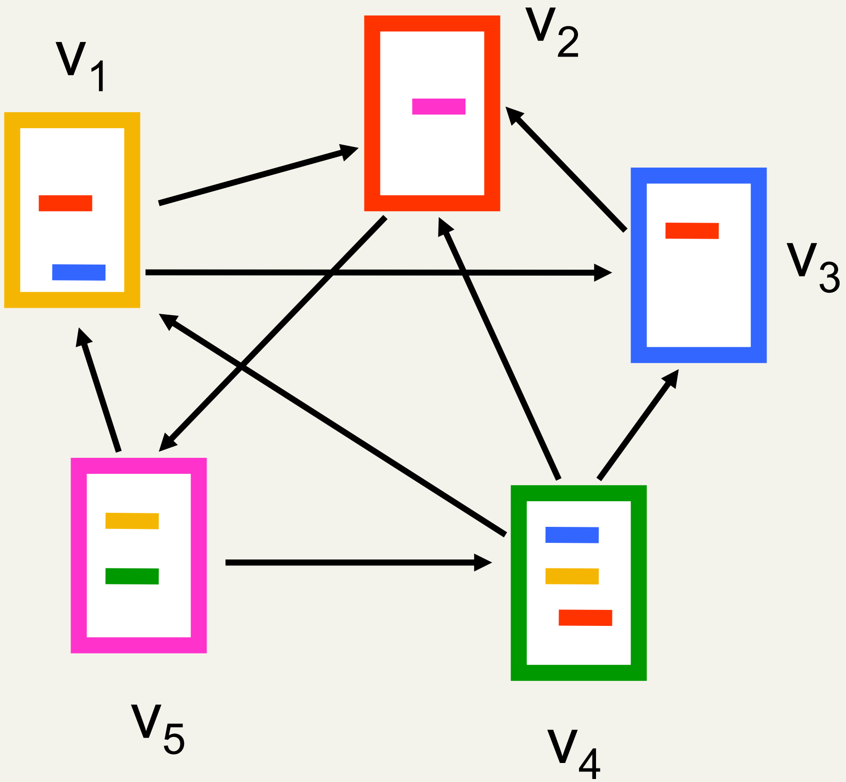
Random walks

- Random walks on graphs are examples of Markov chains
  - The set of states is the set of nodes of the graph **G**
  - The transition probability matrix is the probability that we follow an edge from one node to another
- Pagerank is **NOT** a random walk (but similar)
  - Why?

An example

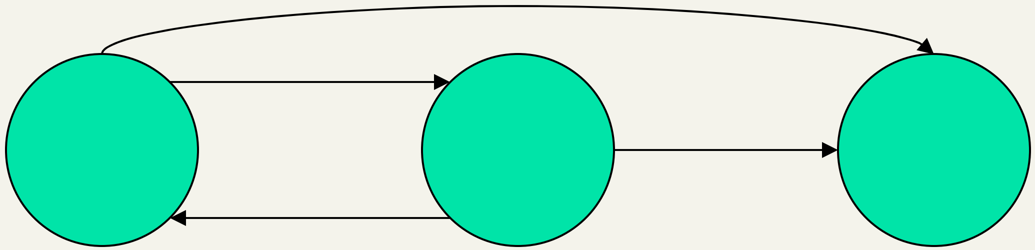
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P_{RW} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{bmatrix}$$



Markov chains

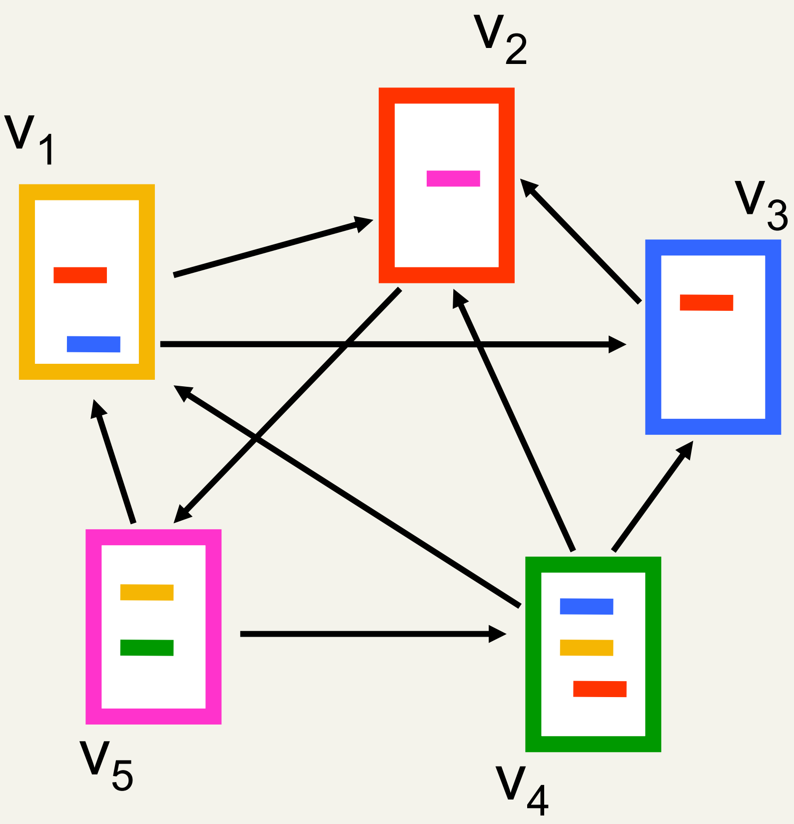
- Clearly, for all i, $\sum_{j=1}^{n} P_{ij} = 1.$
- Markov chains are abstractions and generalizations of **random walks**.
- *Exercise*: represent the teleporting random walk from 3 slides ago as a Markov chain, for this case:



The PageRank Markov chain

- Previous graph:

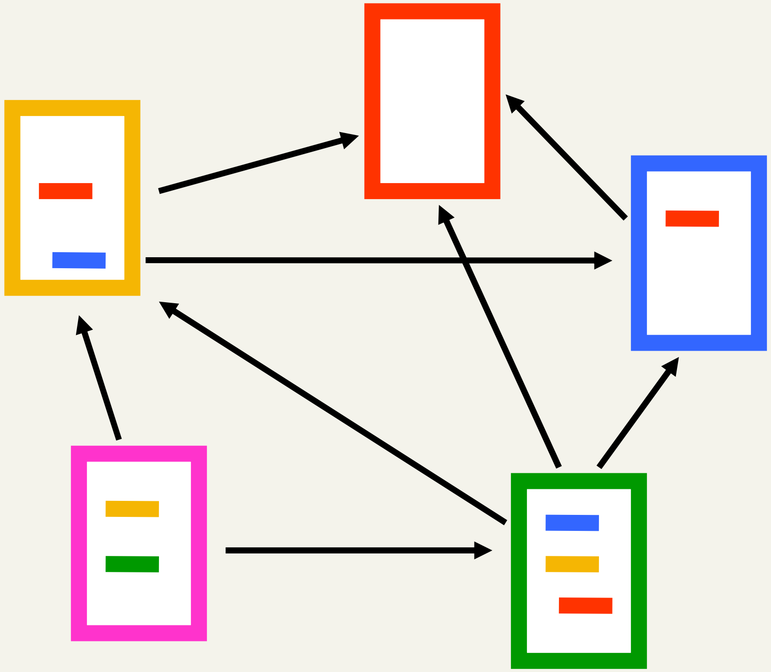
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



The PageRank Markov chain

- Let's consider a different example (assume that page 2 has no outlinks)

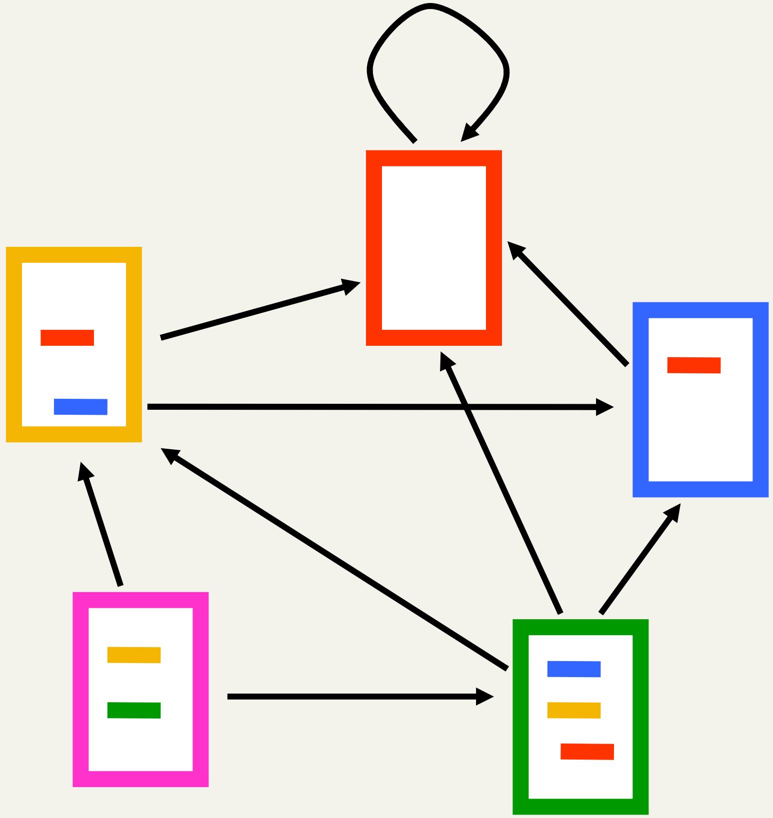
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



The PageRank Markov chain

- What about sink nodes?
  - what happens when the random walk moves to a node without any outgoing inks?

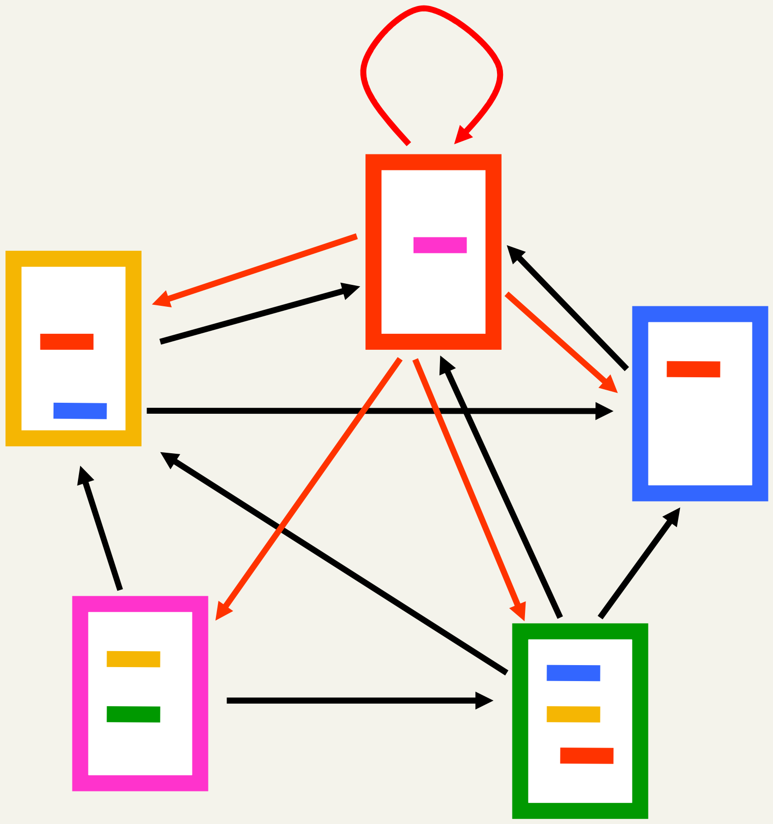
$$P_{RW} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ \boxed{0 \quad 1 \quad 0 \quad 0 \quad 0} \\ 0 & 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{bmatrix}$$



The PageRank Markov chain

- Replace these row vectors with a vector v
  - typically, the uniform vector

$$P_{RW} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ \boxed{1/5} & \boxed{1/5} & \boxed{1/5} & \boxed{1/5} & \boxed{1/5} \\ 0 & 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{bmatrix}$$



The PageRank Markov chain

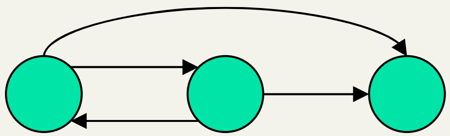
- How do we guarantee irreducibility?
  - add a random jump to vector v with prob α
    - typically, to a uniform vector

$$P_{PR} = (1-\alpha)\begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 0 & 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{bmatrix} + \alpha \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \end{bmatrix}$$

$P_{PR} = (1-\alpha)P_{RW} + \alpha U$, where U is the uniform matrix with rows summing to 1

Transition matrix for pagerank

- Take the adjacency matrix A
- If a line i has no 1s set P_{ij} = 1/N
- For the rest of the rows:
  - Set: $$P_{ij} = (1-\alpha)P_{RW} + \frac{\alpha}{N} = (1-\alpha)\frac{A_{ij}}{(\#\ \text{1s in line } i)} + \frac{\alpha}{N}$$



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

$$P_{RW} = \begin{bmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} \frac{\alpha}{3} & \frac{1}{2} - \frac{\alpha}{6} & \frac{1}{2} - \frac{\alpha}{6} \\ \frac{1}{2} - \frac{\alpha}{6} & \frac{\alpha}{3} & \frac{1}{2} - \frac{\alpha}{6} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

Probability vectors

- A probability (row) vector **q** = (q_1, … q_n) tells us where the walk is at any point.
- E.g., (000…1…000) means we're in state *i*.

1 *i* *n*

More generally, the vector **q** = (q_1, … q_n) means the walk is in state *i* with probability q_i.

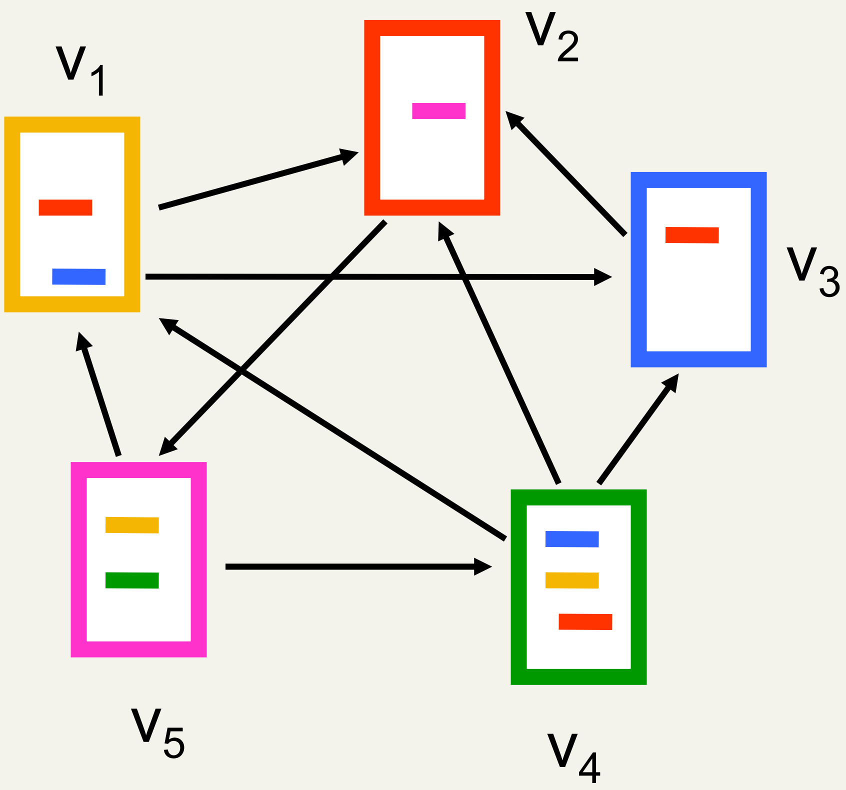
$$\sum_{i=1}^{n} q_i = 1.$$

Change in probability vector

- If the probability vector is $\mathbf{q} = (q_1, \ldots q_n)$ at this step, what is it at the next step?
- Recall that row *i* of the transition prob. Matrix **P** tells us where we go next from state *i*.
- So from **q**, our next state is distributed as **qP**.
- After t steps: $\mathbf{qP}^\mathbf{t}$

An example

$$P = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{bmatrix}$$



$q^{t+1}_1 = 1/3\ q^t_4 + 1/2\ q^t_5$

$q^{t+1}_2 = 1/2\ q^t_1 + q^t_3 + 1/3\ q^t_4$

$q^{t+1}_3 = 1/2\ q^t_1 + 1/3\ q^t_4$

$q^{t+1}_4 = 1/2\ q^t_5$

$q^{t+1}_5 = q^t_2$

Questions:

- What page should we start at?
- How does the probability depend on the starting page?
- How can we compute the probabilities?

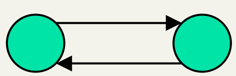
Stationary distribution

- A **stationary distribution** or **steady-state distribution** for a MC with transition matrix P, is a probability distribution π, such that $\pi = \pi P$

- If we start or arrive at the stationary distribution then we remain there

Stationary distribution

- A MC has a unique stationary distribution if
  - it is irreducible
    - From each state we can arrive to every other state
    - the underlying graph is strongly connected
  - it is aperiodic
    - After a number of steps, you can be in any state at every time step, with non-zero probability.

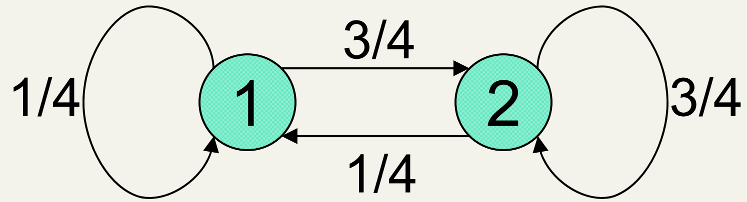


Not ergodic (even/odd).

- Such a MC is called **ergodic**
- Over a long time-period, we visit each state in proportion to this rate.
- **It doesn't matter where we start.**
- The probability π_i is the fraction of times that we visited state i as $t \to \infty$

Steady state example

- The steady state looks like a vector of probabilities $\boldsymbol{\pi} = (\pi_1, \ldots \pi_n)$:
  - π_i is the probability that we are in state i.



For this example, π_1=1/4 and π_2=3/4.

How do we compute this vector?

- Let $\boldsymbol{\pi} = (\pi_1, \ldots \pi_n)$ denote the row vector of steady-state probabilities.
- If we our current position is described by $\boldsymbol{\pi}$, then the next step is distributed as $\boldsymbol{\pi}\mathbf{P}$.
- But $\boldsymbol{\pi}$ is the steady state, so $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}$.
- Solving this matrix equation gives us $\boldsymbol{\pi}$
- (So $\boldsymbol{\pi}$ is the (left) eigenvector for $\mathbf{P}$)

One way of computing $\boldsymbol{\pi}$

- Recall, regardless of where we start, we eventually reach the steady state $\boldsymbol{\pi}$
- Start with any distribution (say $\mathbf{q}^{\mathbf{0}}=(10\ldots0)$)
- After one step, we're at $\mathbf{q}^{\mathbf{0}}\mathbf{P}$
- after two steps at $\mathbf{q}^{\mathbf{0}}\mathbf{P}^{2}$, then $\boldsymbol{\pi}^{\mathbf{T}}\mathbf{P}^{3}$ and so on
- "Eventually" means for "large" t, $\boldsymbol{\pi}\mathbf{P}^{t} = \boldsymbol{\pi}$
- Algorithm: multiply $\mathbf{q}^{\mathbf{0}}$ by increasing powers of **P** until the product looks stable

Pagerank summary

- Preprocessing:
  - Given graph of links, build matrix **P**.
  - From it compute $\boldsymbol{\pi}$.
  - The entry π_i is a number between 0 and 1: the pagerank of page *i*.
- Query processing:
  - Retrieve pages meeting query.
  - Rank them by their pagerank.
  - Order is query-*independent*.
  - Combine pagerank with other scores (e.g., IR based)

Effects of random jump

- Guarantees irreducibility
- Motivated by the concept of random surfer
- Offers additional flexibility
  - personalization
  - anti-spam
- Controls the rate of convergence
  - the second eigenvalue of matrix P is α

Pagerank: Issues and Variants

- How realistic is the random surfer model?
  - What if we modeled the back button? [Fagi00]
  - Surfer behavior sharply skewed towards short paths
  - Search engines, bookmarks & directories make jumps non-random.
- Biased Surfer Models
  - Weight edge traversal probabilities based on match with topic/query (non-uniform edge selection)
  - Bias jumps to pages on topic (e.g., based on personal bookmarks & categories of interest)

Research on PageRank

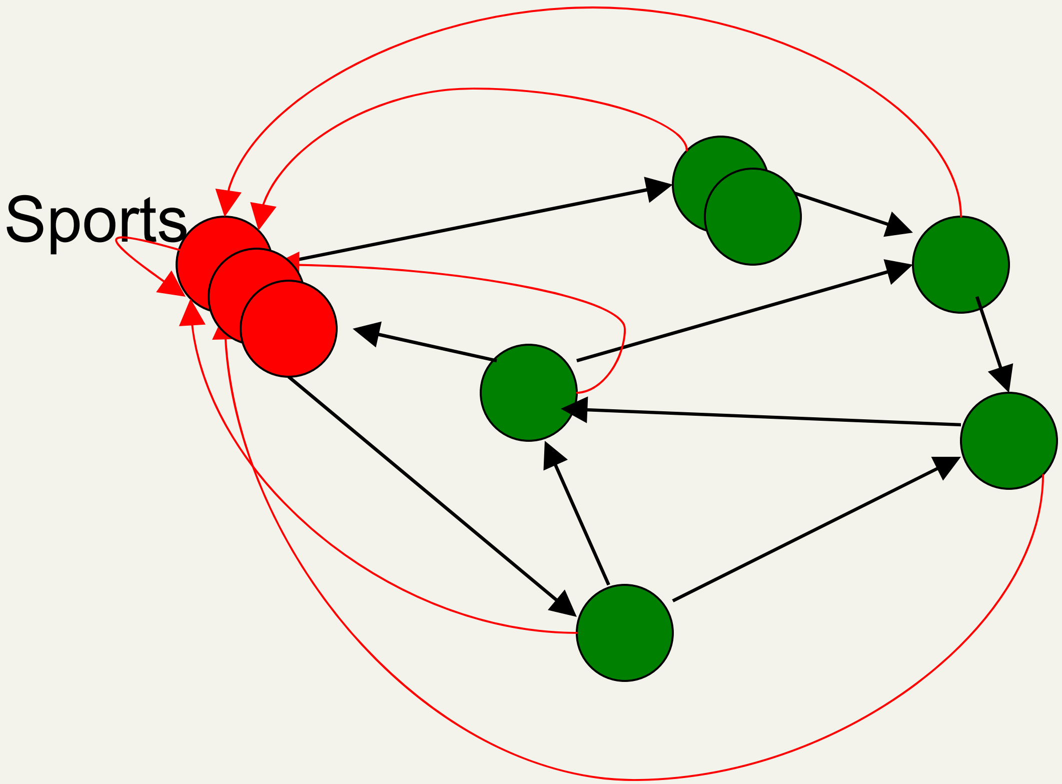
- Specialized PageRank
  - personalization [BP98]
    - instead of picking a node uniformly at random favor specific nodes that are related to the user
  - topic sensitive PageRank [H02]
    - compute many PageRank vectors, one for each topic
    - estimate relevance of query with each topic
    - produce final PageRank as a weighted combination
- Updating PageRank [Chien et al 2002]
- Fast computation of PageRank
  - numerical analysis tricks
  - node aggregation techniques
  - dealing with the “Web frontier”

Topic Specific Pagerank

- Assume that I am interested in a topic:
  - Sports, Art, etc.

- Can I bias Pagerank towards this topic?

Non-uniform Teleportation



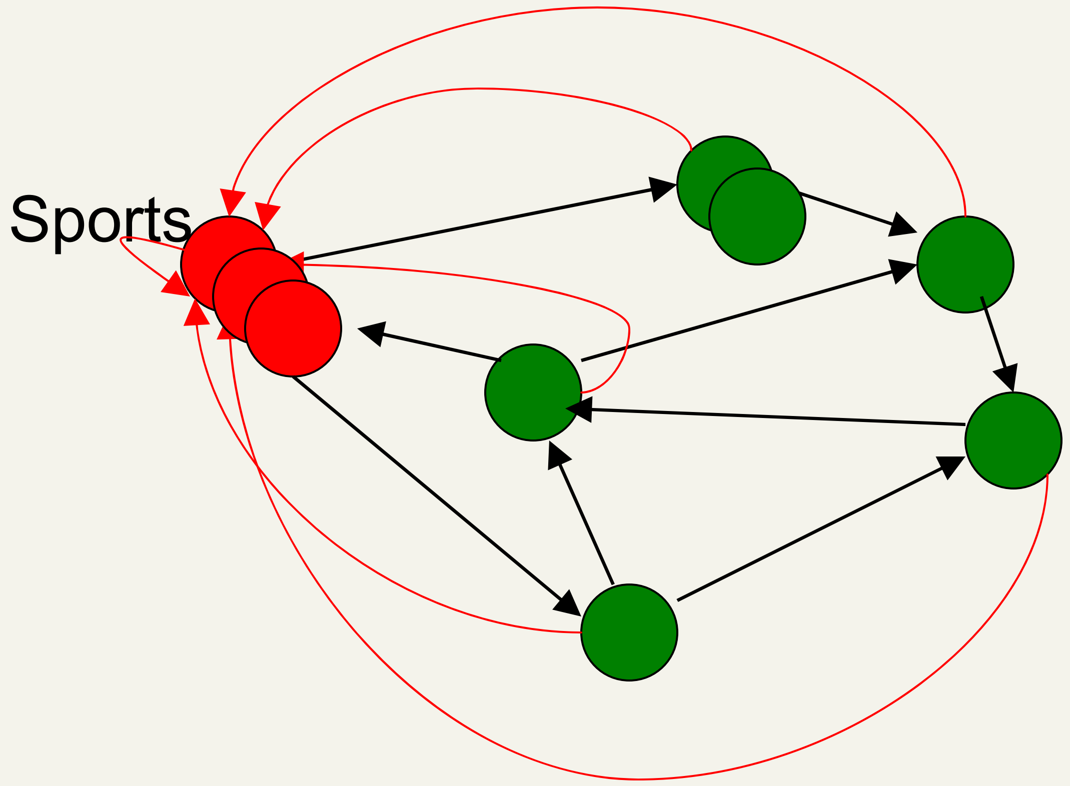
Teleport with 10% probability to a Sports page

Finding pages

- How do I know what pages are about Sports?
  - Use classification (Machine learning) – later
  - Use preclassified pages
    - Open Directory Project (ODP)

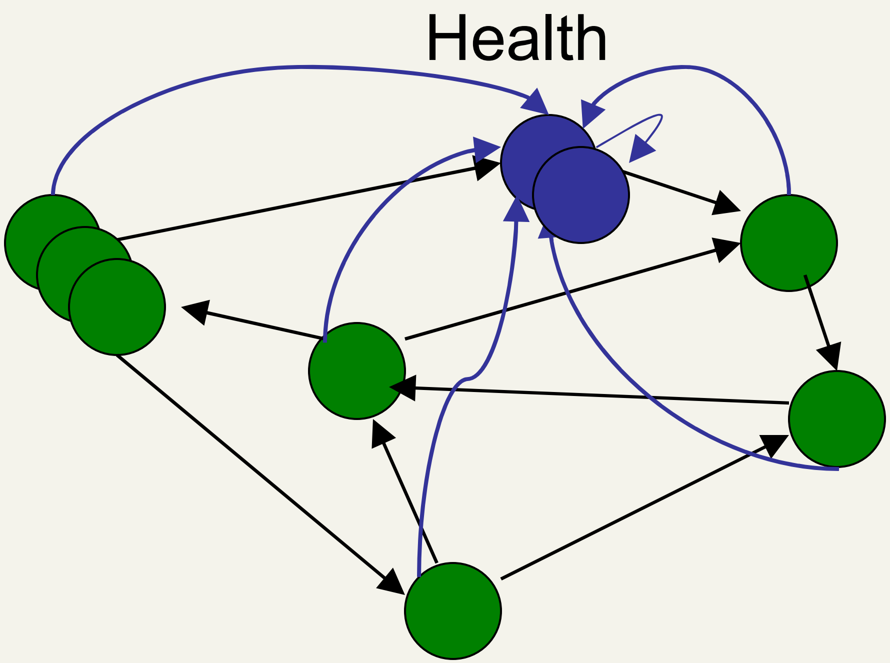
- Let PR(p, “sports”) = Pagerank with teleport towards sports pages

Non-uniform Teleportation



Teleport with 10% probability to a Sports page

Non-uniform Teleportation

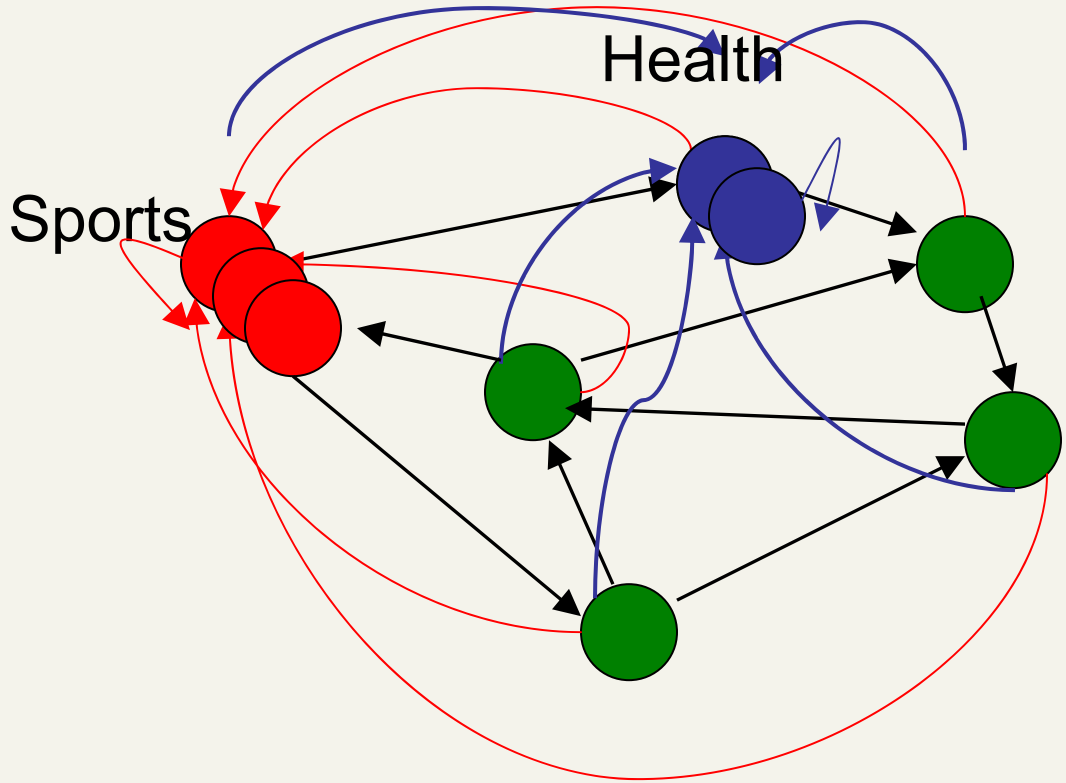


10% Health teleportation

General framework

- We have a set of categories C_j
  - C_1 = Sports, C_2 = health, C_3 = art, C_4 = politics, …
- A user is characterized by a distribution over categories
  - E.g.: 90% sports, 10% health
  - Profile: u = (0.9, 0.1, 0, 0, 0, …)
- We want for each page p: PR(p, u)
- We can compute the Pagerank as before but with different probabilities

Interpretation



If teleport probability α = 10%

We can have teleport: 9% to sport, 1% health

Problem

- We want:

  PR(p, u) = Pagerank with respect to user profile **u**

- **Problem:** If every user has different profiles we need a pagerank for every user

Solution

- We can precompute offline for each category (sports, health, art, …) $\mathrm{PR}(p, C_j)$

- Then, because of linearity, we have:

$$\begin{aligned} \mathrm{PR}(p,u) &= \mathrm{PR}(p, \sum_j u_j C_j) \\ &= \sum_j u_j \cdot \mathrm{PR}(p, C_j) \end{aligned}$$

(Handwaving notation)

- When a user u comes, we only sum the precomputed pagerank scores

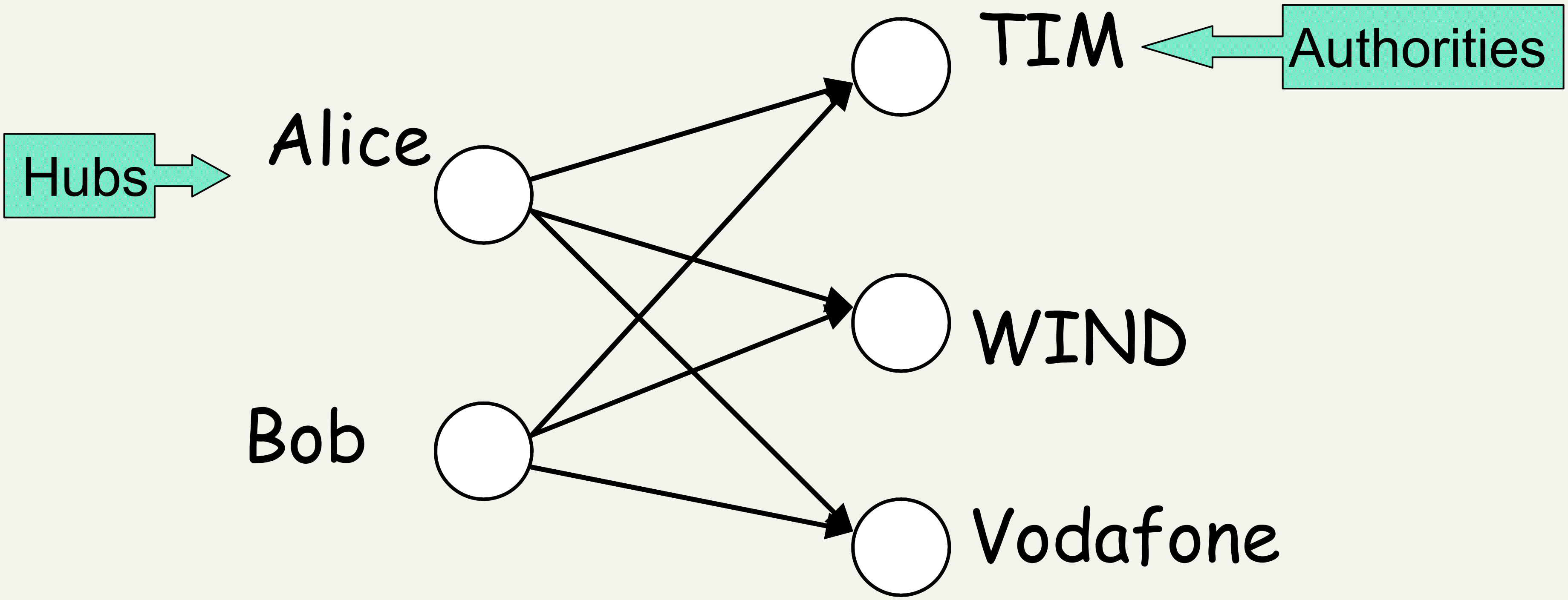
Hyperlink-Induced Topic Search (HITS) – Kleinberg 98

- In response to a query, instead of an ordered list of pages each meeting the query, find **two** sets of inter-related pages:
  - *Hub pages* are good lists of links on a subject.
    - e.g., “Bob’s list of cancer-related links.”
  - *Authority pages* occur recurrently on good hubs for the subject.
- Best suited for “broad topic” queries rather than for page-finding queries (navigational queries).
- Gets at a broader slice of common *opinion*.

Hubs and Authorities

- Thus, a good hub page for a topic *points* to many authoritative pages for that topic.
- A good authority page for a topic is *pointed* to by many good hubs for that topic.
- Circular definition - will turn this into an iterative computation.

The hope



Cell phone providers

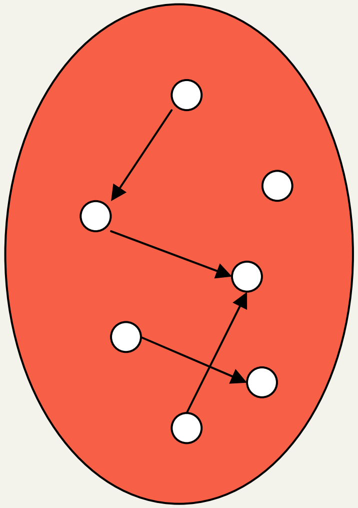
High-level scheme

- Extract from the web a **base set** of pages that *could* be good hubs or authorities
- From these, identify a small set of top hub and authority pages
  - iterative algorithm

Base set

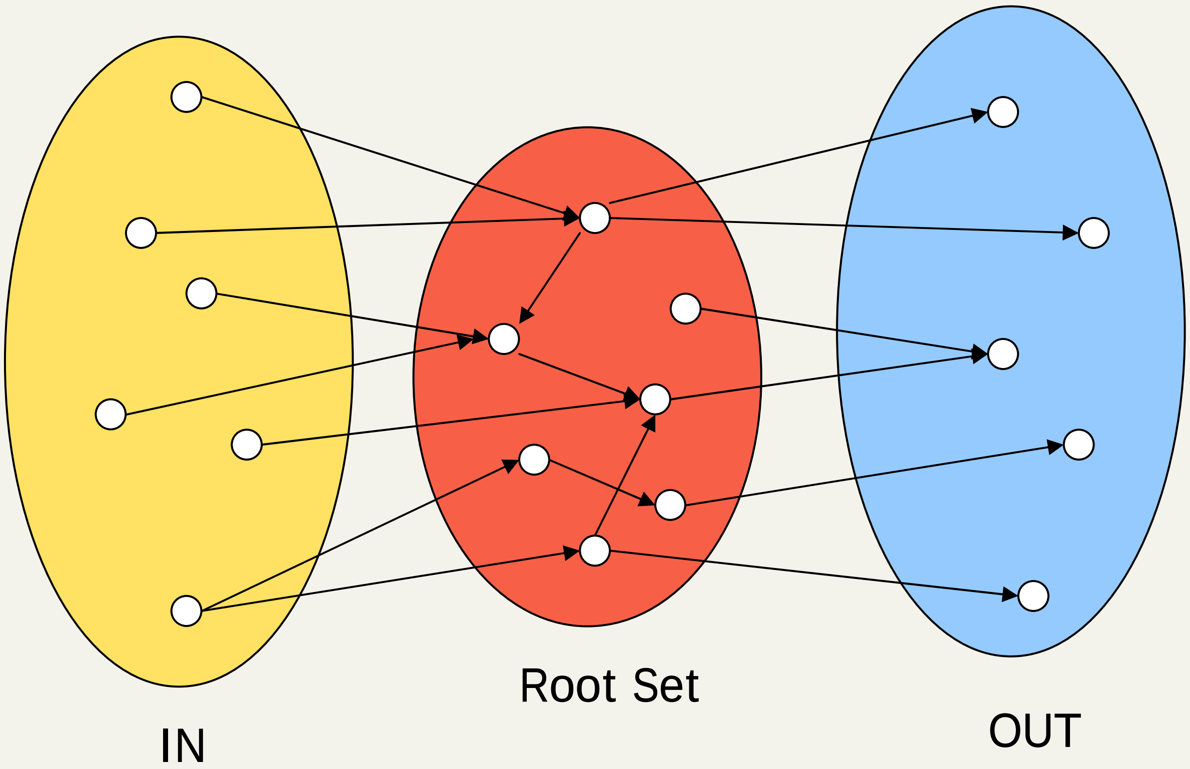
- Given text query (say ***browser***), use a text index to get all pages containing ***browser.***
  - Call this the **root set** of pages.
- Add in any page that either
  - points to a page in the root set, or
  - is pointed to by a page in the root set.
- Call this the **base set**.

Query dependent input

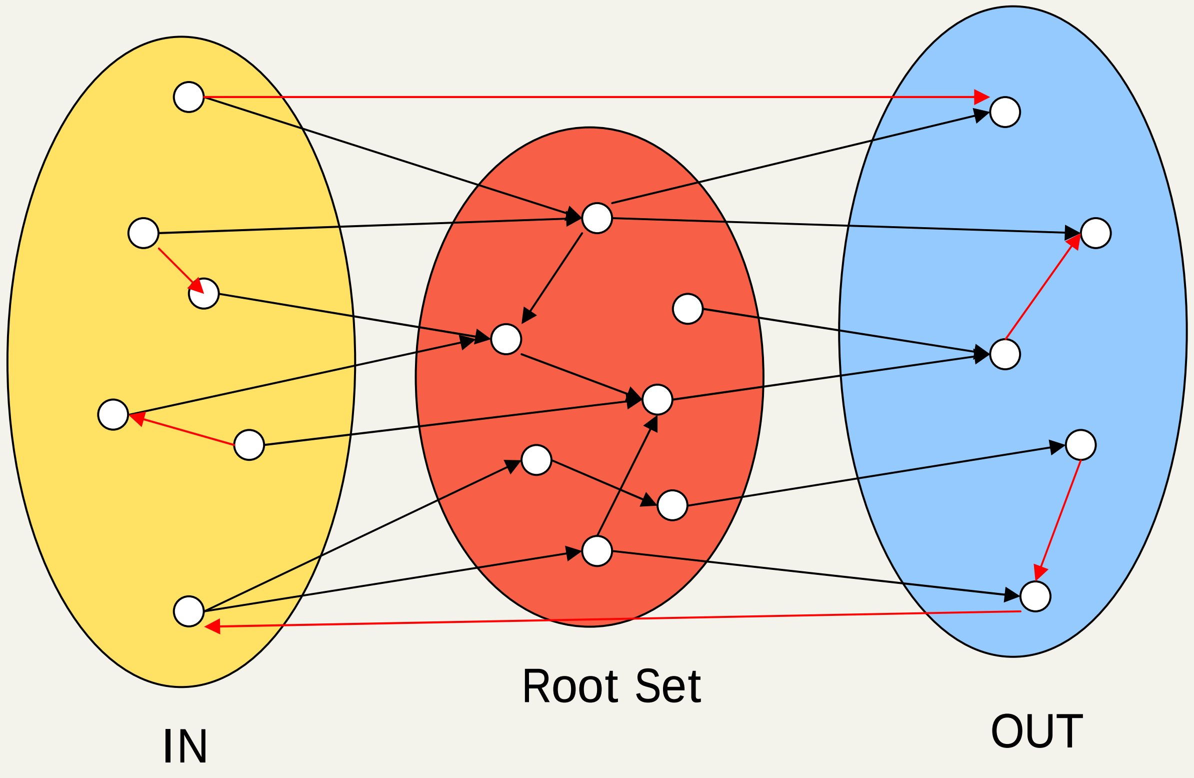


Root Set

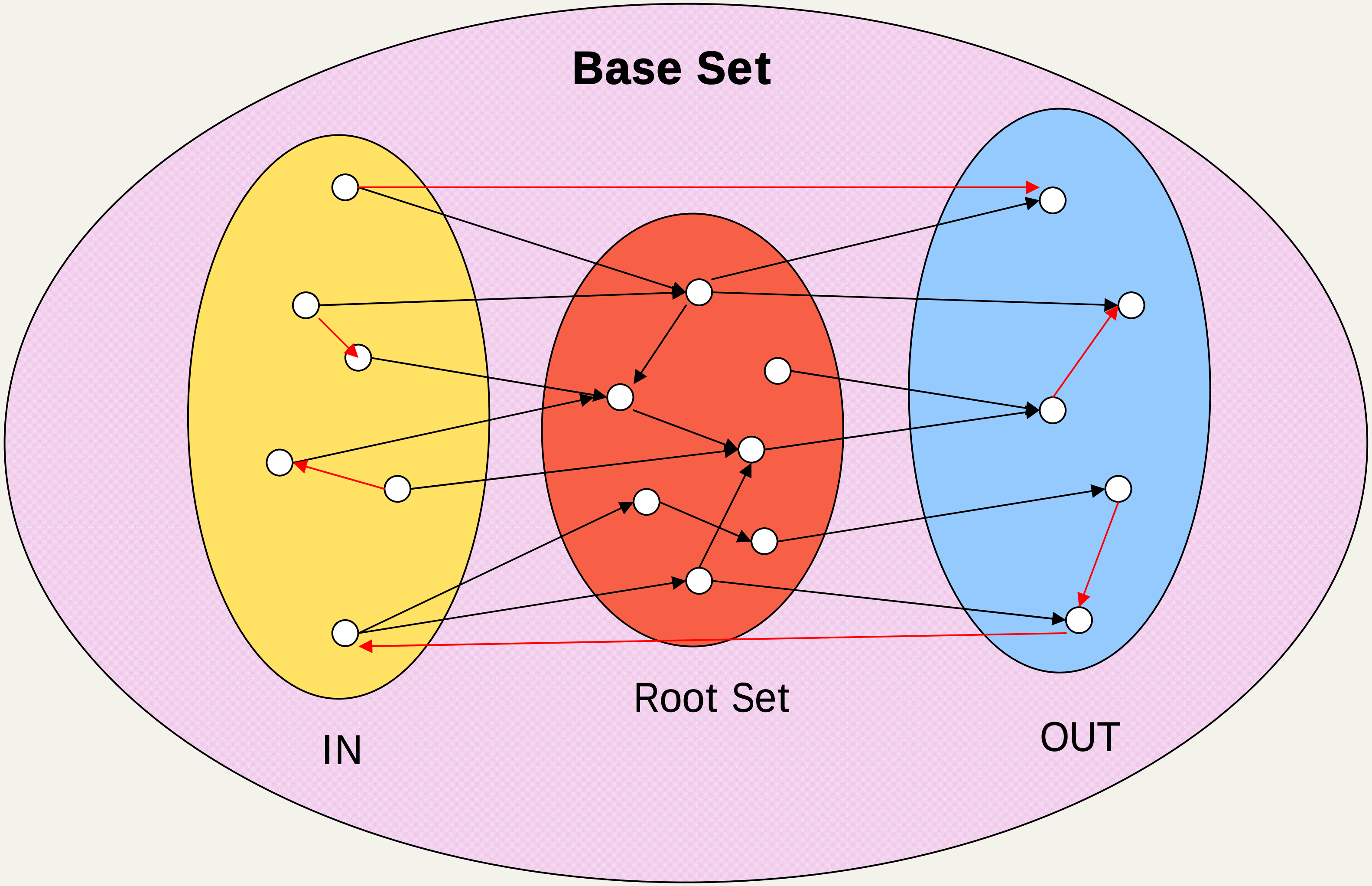
Query dependent input



Query dependent input



Query dependent input



Assembling the base set [Klei98]

- Root set typically 200-1000 nodes.
- Base set may have up to 5000 nodes.
- How do you find the base set nodes?
  - Follow out-links by parsing root set pages.
  - Get in-links (and out-links) from a *connectivity server.*
  - (Actually, suffices to text-index strings of the form ***href="URL"*** to get in-links to *URL*.)

Distilling hubs and authorities

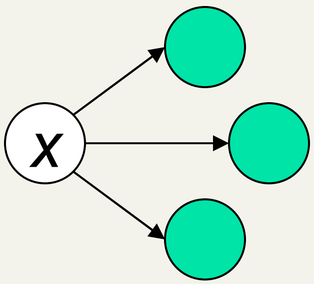
- Compute, for each page x in the base set, a **hub score *h*(*x*)** and an **authority score *a*(*x*)**
- Initialize: for all x, $h(x) \leftarrow 1$; $a(x) \leftarrow 1$;
- Iteratively update all $h(x)$, $a(x)$;
- After iterations
  - output pages with highest $h()$ scores as top hubs
  - highest $a()$ scores as top authorities.

Key

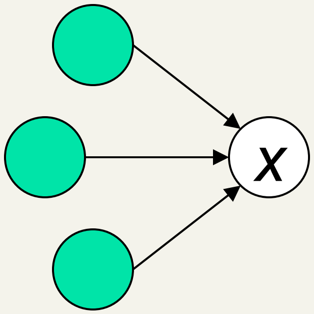
Iterative update

- Repeat the following updates, for all *x*:

$$h(x) \leftarrow \sum_{x \mapsto y} a(y)$$



$$a(x) \leftarrow \sum_{y \mapsto x} h(y)$$



Scaling

- To prevent the *h*() and *a*() values from getting too big, can scale down after each iteration.
  - E.g.: $h(x) \leftarrow h(x) / \max_x h(x)$
    $a(y) \leftarrow a(y) / \max_y a(y)$
- Scaling factor doesn't really matter:
  - we only care about the **relative** values of the scores.

How many iterations?

- Claim: relative values of scores will converge after a few iterations:
  - suitably scaled, *h*() and *a*() scores settle into a steady state!
  - proof of this comes later.
- We only require the **relative orders** of the *h*() and *a*() scores - not their absolute values.
- In practice, ~5 iterations get you close to stability.

HITS Algorithms

- Input: Graph G = (V,E)
- Output: h(v), a(v) for each v $\in$ V

- For all (v $\in$ V) set $h^0(v) \leftarrow 1$, $a^0(v) \leftarrow 1$
- Repeat until convergence

(E.g. $\max_{v\in V}\{|h^t(v) - h^{t-1}(v)|\} < \epsilon, \quad \max_{v\in V}\{|a^t(v) - a^{t-1}(v)|\} < \epsilon$)

  - Authorities collect the weight of the hubs

$$\forall u \in V: \quad a^t(u) \leftarrow \sum_{(v,u)\in E} h^{t-1}(v)$$

  - Hubs collect the weight of the authorities

$$\forall v \in V: \quad h^t(v) \leftarrow \sum_{(v,u)\in E} a^t(u)$$

  - Normalize weights:

$$\forall v \in V: \quad h^t(v) \leftarrow \frac{h^t(v)}{\max_v h^t(v)}, \quad a^t(v) \leftarrow \frac{a^t(v)}{\max_v a^t(v)}$$

Japan Elementary Schools

Hubs

- schools
- LINK Page-13
- “ú–{,ÌŠw□Z
- □a‰„□¬Šw□Zƒz□[ƒ□ƒy□[ƒW
- 100 Schools Home Pages (English)
- K-12 from Japan 10/...rnet and Education)
- http://www...iglobe.ne.jp/~IKESAN
- ‚l‚f‚j□¬Šw□Z‚U”N‚P‘g•¨Œê
- □ÒŠ—’¬—§□ÒŠ—“Œ□¬Šw□Z
- Koulutus ja oppilaitokset
- TOYODA HOMEPAGE
- Education
- Cay's Homepage(Japanese)
- –y“ì□¬Šw□Z‚Ìƒz□[ƒ□ƒy□[ƒW
- UNIVERSITY
- ‰J—³□¬Šw□Z DRAGON97-TOP
- □Â‰ª□¬Šw□Z‚T”N‚P‘gƒz□[ƒ□ƒy□[ƒW
- ¶µ°é¼ÂÁ© ¥á¥Ë¥å¡¼ ¥á¥Ë¥å¡¼

Authorities

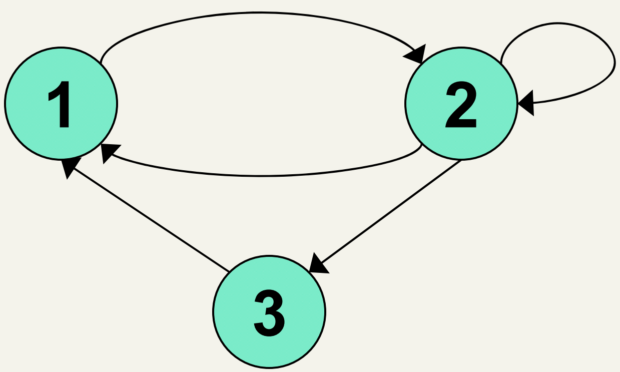
- The American School in Japan
- The Link Page
- ‰ª□è□s—§ˆä“c□¬Šw□Zƒz□[ƒ□ƒy□[ƒW
- Kids' Space
- ˆÀ□é□s—§ˆÀ□é□¼•”□¬Šw□Z
- ‹{□é‹³ˆç‘åŠw•□‘®□¬Šw□Z
- KEIMEI GAKUEN Home Page (Japanese)
- Shiranuma Home Page
- fuzoku-es.fukui-u.ac.jp
- welcome to Miasa E&J school
- □_“Þ□ìŒ§□E‰¡•l□s—§’†□ì□¼□¬Šw□Z‚Ìƒy
- http://www...p/~m_maru/index.html
- fukui haruyama-es HomePage
- Torisu primary school
- goo
- Yakumo Elementary,Hokkaido,Japan
- FUZOKU Home Page
- Kamishibun Elementary School...

Things to note

- Pulled together good pages regardless of language of page content.
- Use **only** link analysis **after** base set assembled
  - iterative scoring is query-dependent.
- Iterative computation **after** text index retrieval - significant overhead.

Proof of convergence

- $n \times n$ adjacency matrix **A**:
  - each of the n pages in the base set has a row and column in the matrix.
  - Entry $A_{ij} = 1$ if page i links to page j, else = 0.



$$
\begin{array}{c|ccc|}
 & 1 & 2 & 3 \\
\hline
1 & 0 & 1 & 0 \\
2 & 1 & 1 & 1 \\
3 & 1 & 0 & 0 \\
\hline
\end{array}
$$

Hub/authority vectors

- View the hub scores *h*() and the authority scores *a*() as vectors with *n* components.
- Recall the iterative updates

$$h(x) \leftarrow \sum_{x \mapsto y} a(y)$$

$$a(x) \leftarrow \sum_{y \mapsto x} h(y)$$

HITS and eigenvectors

- We can write the HITS algorithm in vector terms:
  - $a^t = A^T h^{t-1} / c_a$ and $h^t = A a^t / c_h$ (where c_a and c_h are the normalization constants)
- So:
  - $a^t = A^T h^{t-1} / c_a = A^T (A a^{t-1}) / c_a c_h = A^T A\, a^{t-1} / c_a c_h$
  - $h^t = A a^t / c_h = A(A^T h^{t-1}) / c_a c_h = A A^T\, h^{t-1} / c_a c_h$
- After convergence to values a and h we have
  - $a = (1/\lambda_a)\, A^T A\, a$ for a constant λ_a
  - $h = (1/\lambda_h)\, A A^T\, h$ for a constant λ_h
- The authority weight vector a is the eigenvector of $A^T A$ and the hub weight vector h is the eigenvector of $A A^T$
- The HITS algorithm is a power-method eigenvector computation

Guaranteed to converge.

Resources

- IIR Chapters 21.2, 21.3