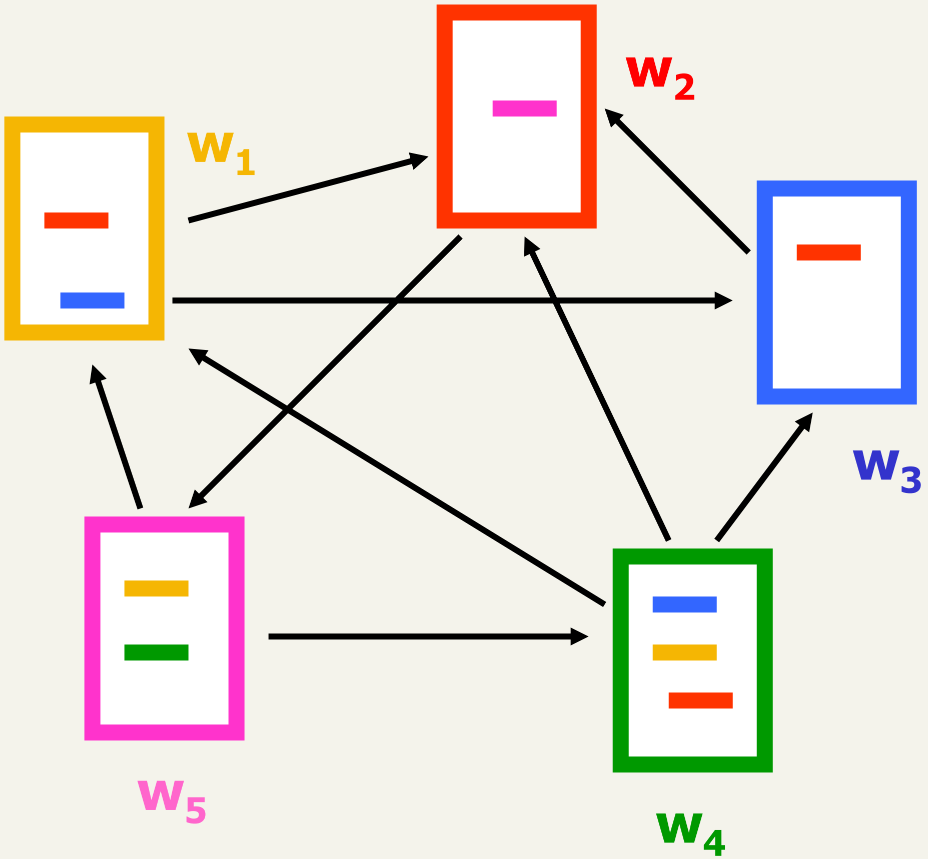
PageRank

The Web Graph



Why is it interesting to study the Web Graph?

- It is the largest artifact ever conceived by the human
- Exploit its structure of the Web for
  - Crawl strategies
  - Search
  - Spam detection
  - Discovering communities on the web
  - Classification/organization
- Predict the evolution of the Web
  - Mathematical models
  - Sociological understanding

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

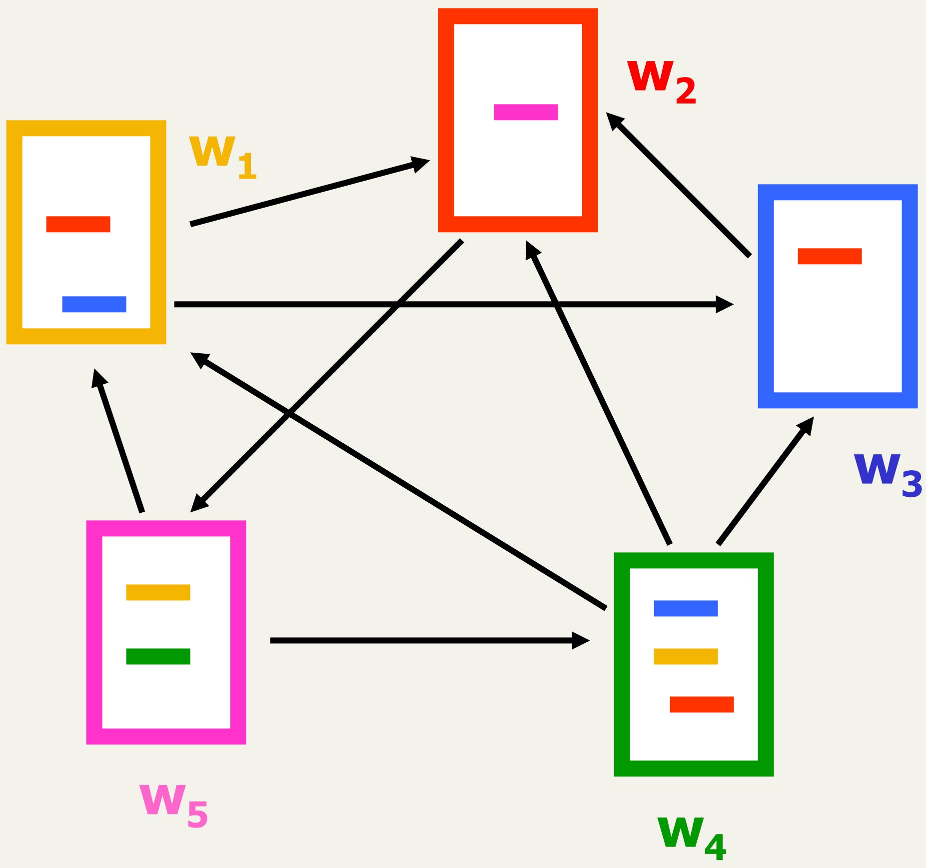
Link Analysis for ranking: Intuition

- 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

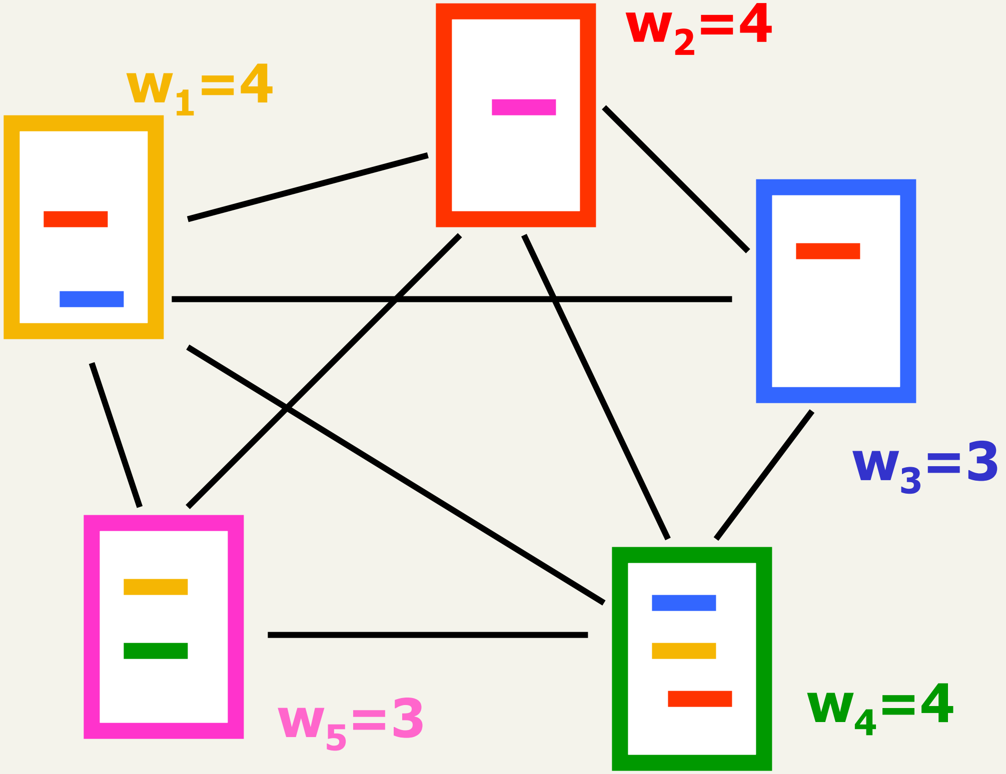
- Start with a collection of web pages
- Extract the underlying hyperlink graph
- Run a LAR algorithm on the graph
- Output: an authority weight for each node

- What is a good LAR algorithm?



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

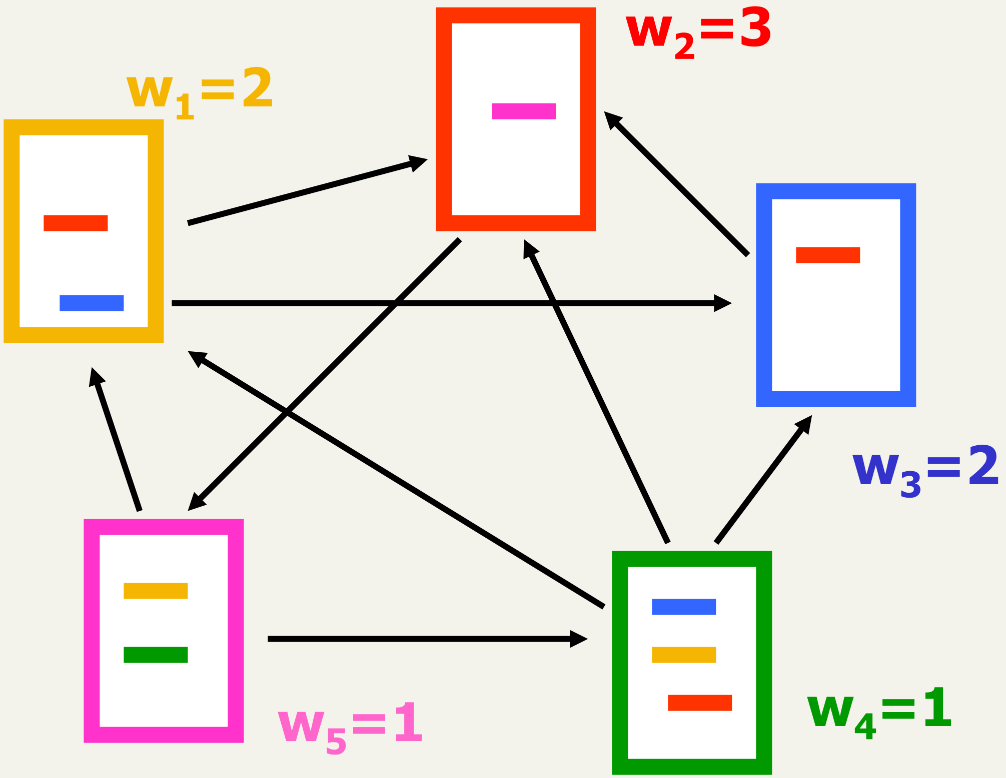
Spamming undirected popularity

- *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

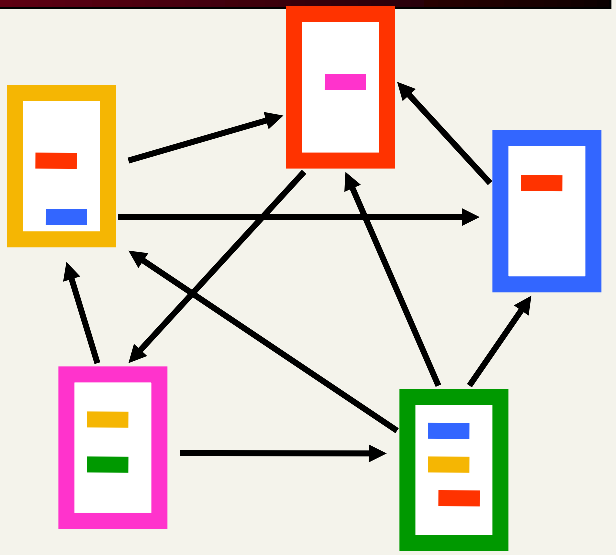
- *Exercise*: How do you spam the directed popularity heurestic

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

PageRank algorithm

High-level idea:

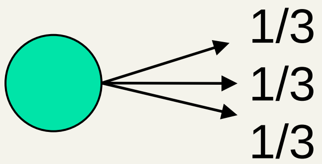
- A good page has a lot of endorsements by important (authoritative) pages
- **Good** authorities should be pointed by **good** authorities
- Count number of votes, but votes have different weights that depends on who votes for them, and so on
- Motivated also by the **random-surfer** model



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

Pagerank scoring

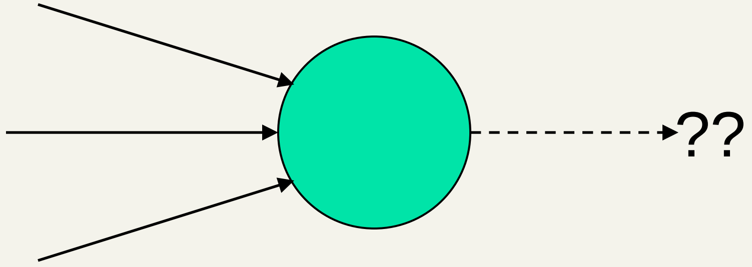
- Imagine a browser doing a random walk on web pages:
  - Start at a random page



  - 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

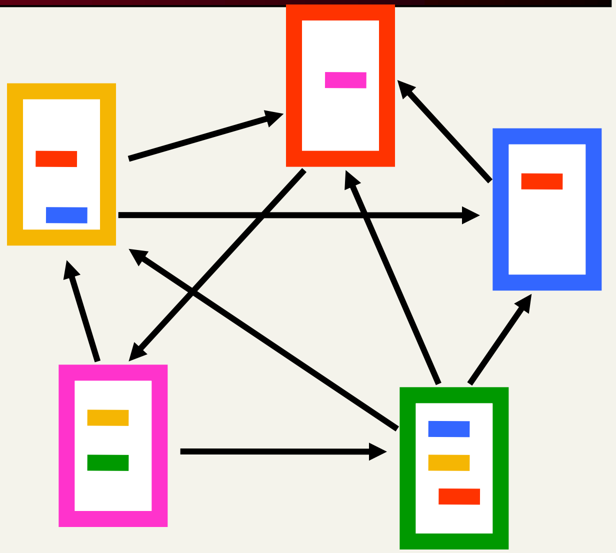
Result of teleporting

- 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 process

- 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 $1-\alpha$ follow a random outgoing link

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



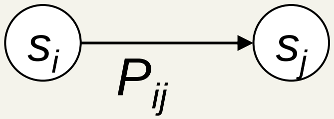
1. Red Page
2. Purple Page
3. Yellow Page
4. Blue Page
5. Green Page

Markov chains

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

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

according to a **transition probability** matrix



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

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

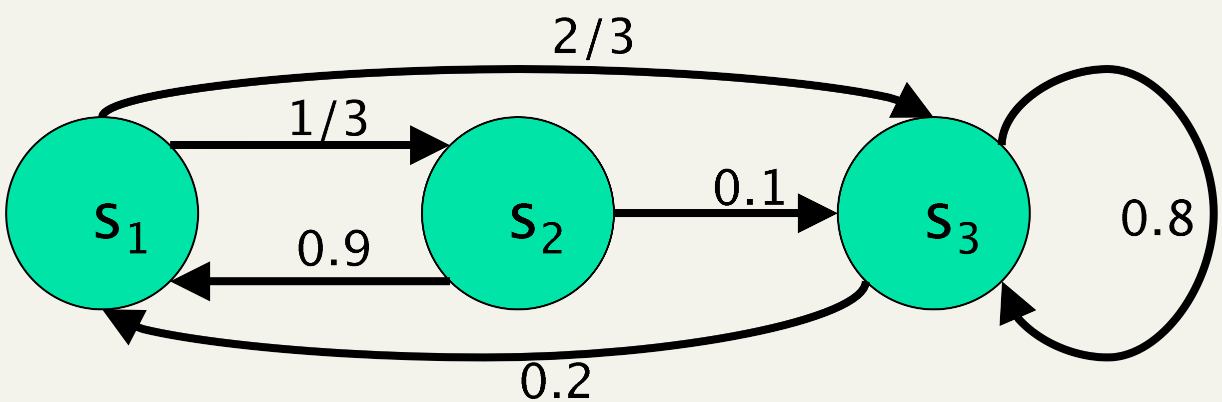
$P_{ii}>0$ is OK

- **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

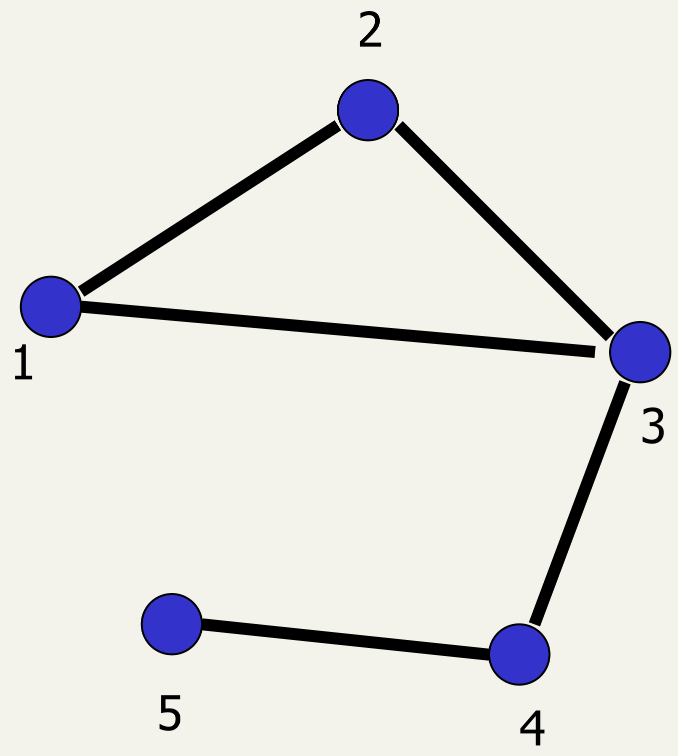
- Random walks on graphs are examples of Markov chains
  - The set of states is the set of nodes of the graph **G**
  - The transition probability matrix is the probability that we follow an edge from one node to another

- Pagerank is **NOT** a random walk (but similar)
  - Why?

Adjacency matrix

- Adjacency matrix
  - symmetric matrix for undirected graphs

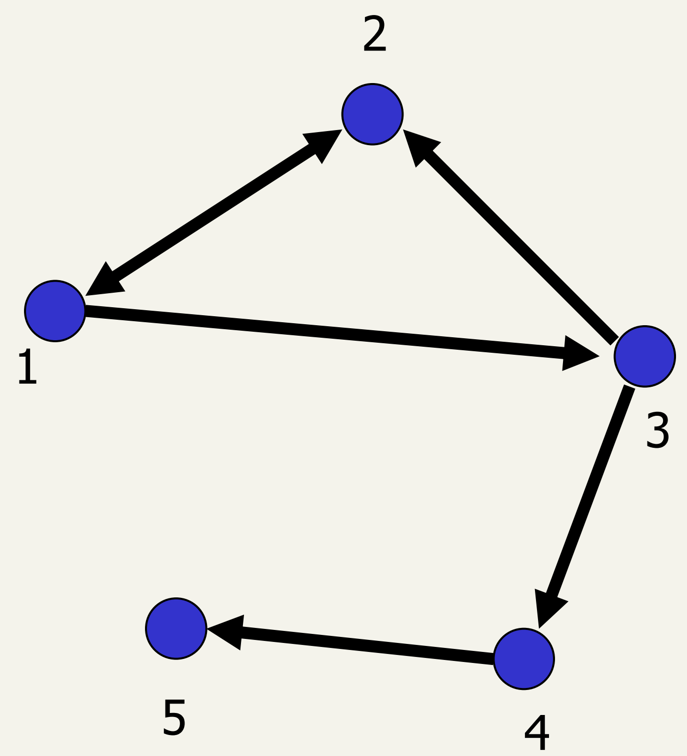
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$



Adjacency matrix

- Adjacency matrix
  - unsymmetric matrix for undirected graphs

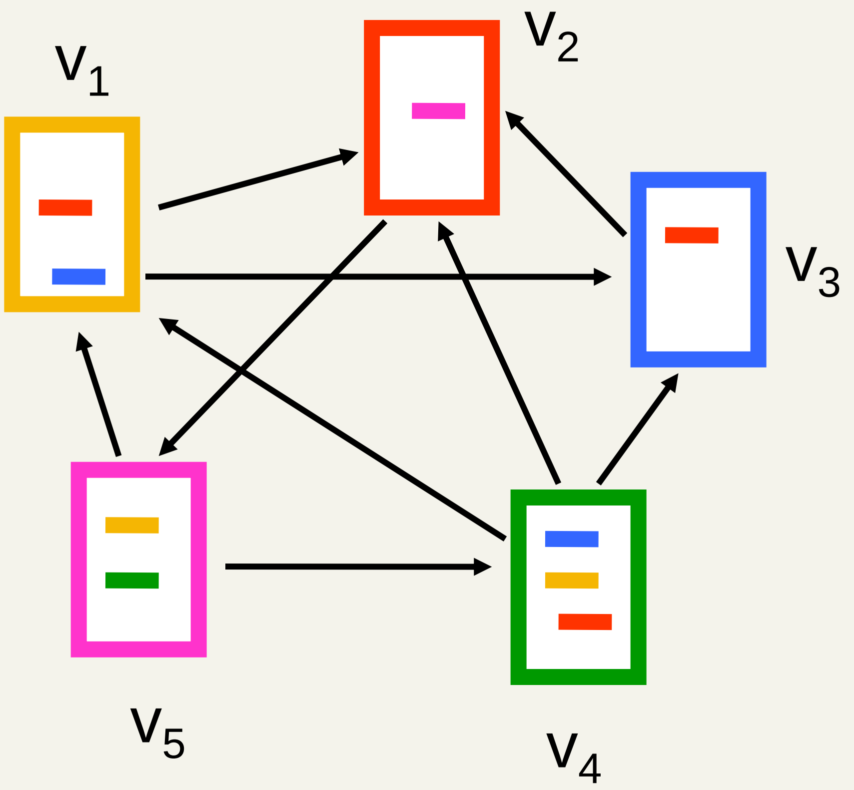
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



An example

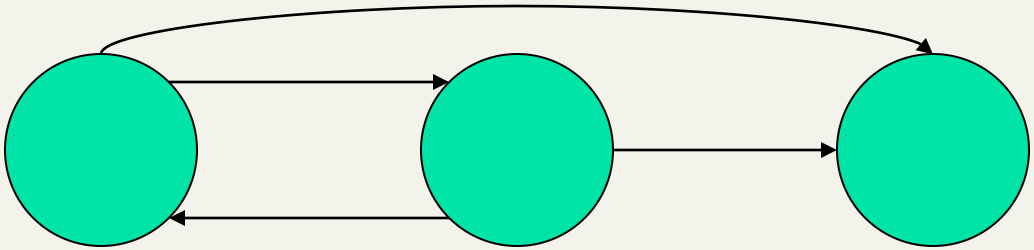
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

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



Markov chains

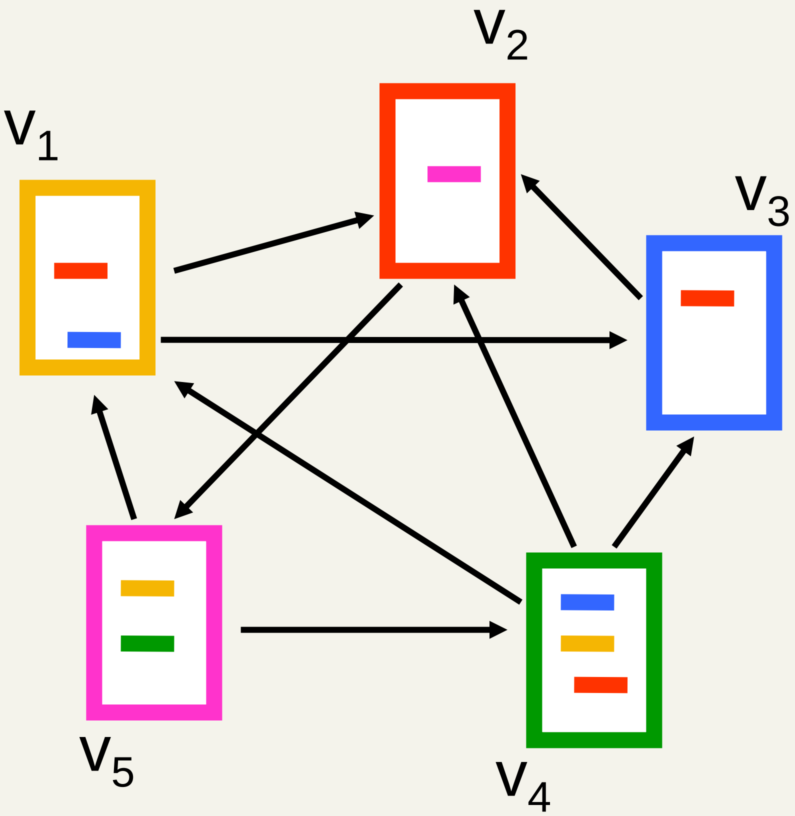
- Clearly, for all i, $\sum_{j=1}^{n} P_{ij} = 1.$
- Markov chains are abstractions and generalizations of **random walks**.



The PageRank Markov chain

- Previous graph:

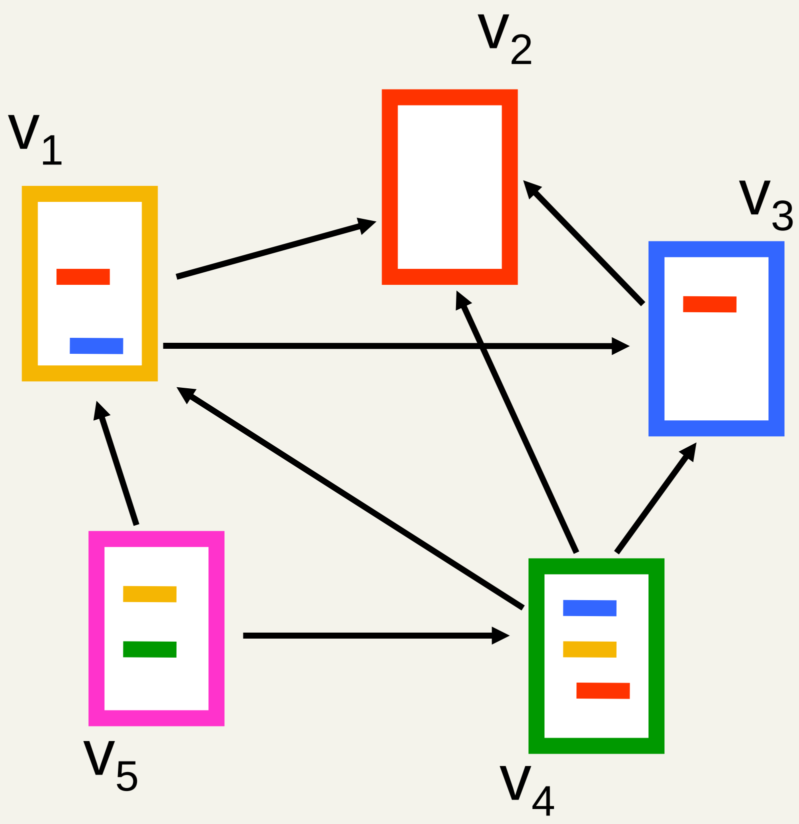
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



The PageRank Markov chain

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

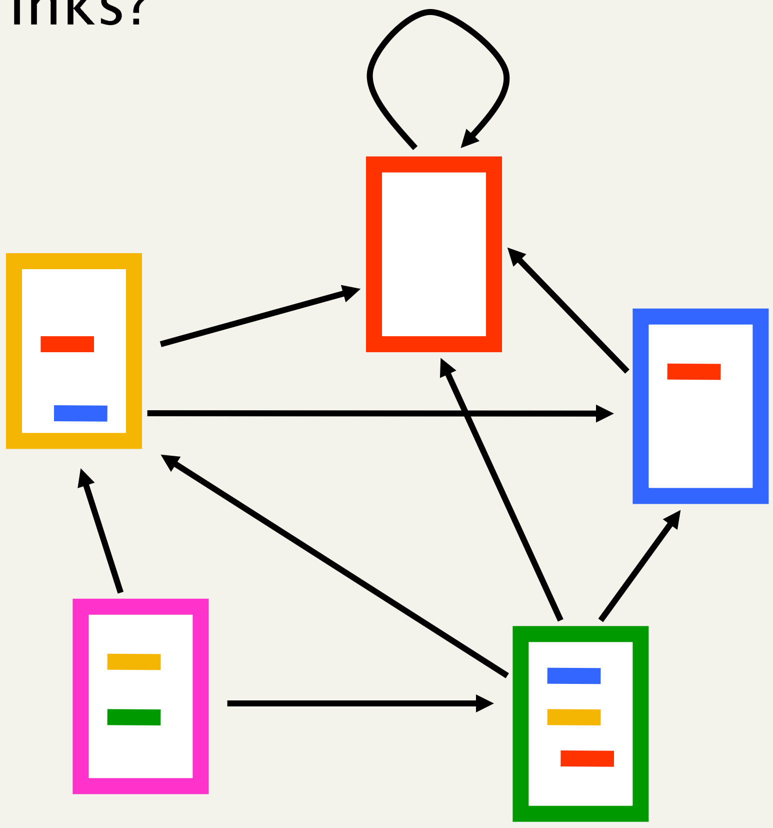
$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



The PageRank Markov chain

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

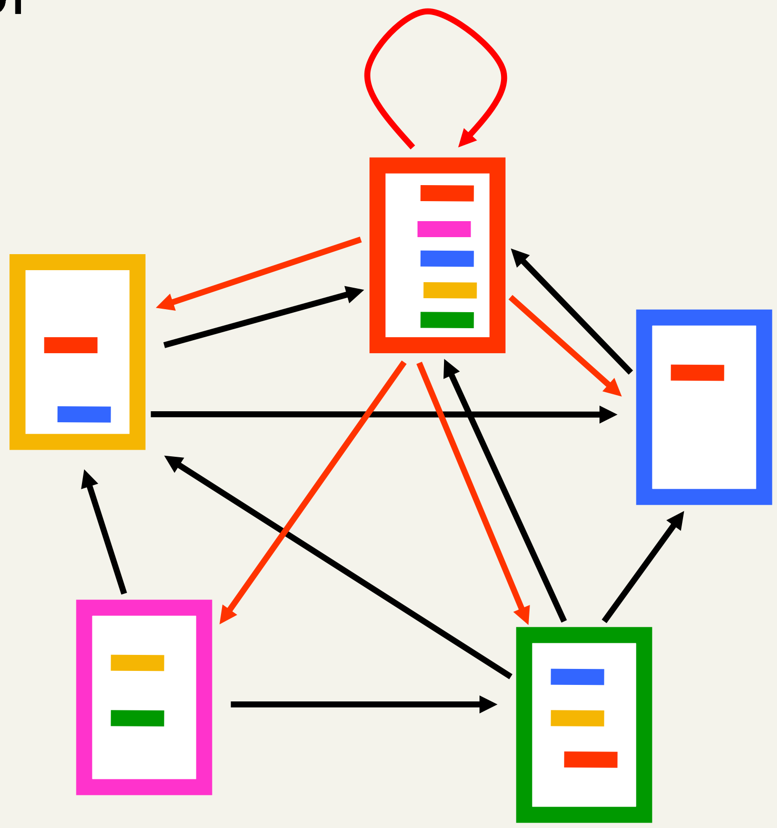
$$P_{RW} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 0 \end{bmatrix}$$



The PageRank Markov chain

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

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



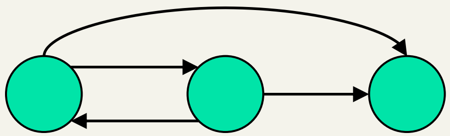
The PageRank Markov chain

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

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

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.

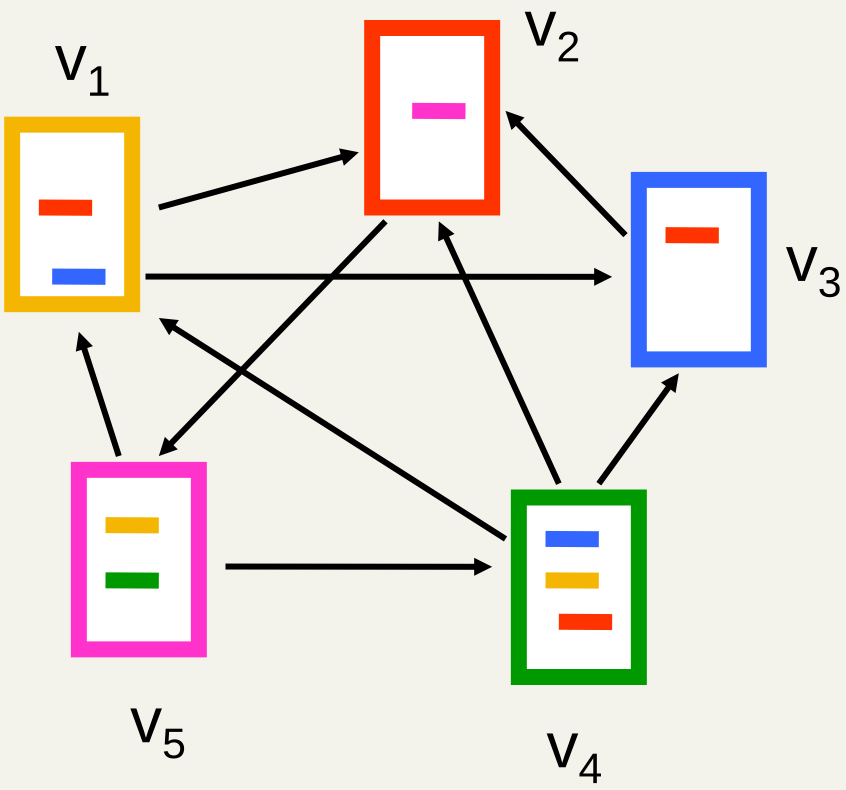
$$\sum_{i=1}^{n} q_i = 1.$$

Change in probability vector

- If the probability vector is **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: $\mathbf{qP}^t$

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?

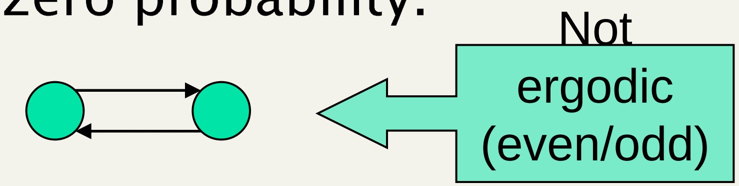
Stationary distribution

- 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.



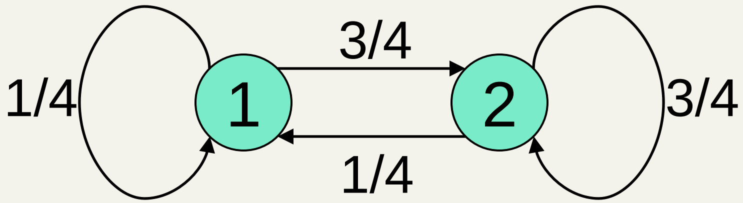
- Such a MC is called **ergodic**
- Over a long time-period, we visit each state in proportion to this rate.
- **It doesn't matter where we start.**
- The probability π_i is the fraction of times that we visited state i as $t \rightarrow \infty$

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 **π**, then the next step is distributed as **πP**.
- But **π** is the steady state, so **π** = **πP**.
- Solving this matrix equation gives us **π**
- (So **π** is a (left) eigenvector for **P**)

One way of computing π

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