Information Storage and Retrieval

CSCE 670
Texas A&M University
Department of Computer Science & Engineering
Instructor: Prof. James Caverlee

Link Analysis: PageRank
7 February 2017

PageRank

Origins of PageRank: Citation Analysis

- Citation analysis: analysis of citations in the scientific literature
- Example citation: "Miller (2001) has shown that physical activity alters the metabolism of estrogens."
- Two ways of measuring similarity of two scientific articles
  - **Cocitation similarity**: The two articles are cited by the same articles.
  - **Bibliographic coupling similarity**: The two articles cite the same articles

Origins of PageRank: Citation Analysis

- Citation frequency can be used to measure the impact of an article.
  - Each article gets one vote.
  - Not a very accurate measure
- Better measure: weighted citation frequency / citation rank
  - An article's vote is weighted according to its citation impact.
  - Sounds circular, but can be formalized in a well-defined way.
  - This is basically Pagerank.
  - Pagerank was invented in the context of citation analysis by Pinsker and Narin in the 1960s.
- Key observation: Citation in scientific literature = Web link

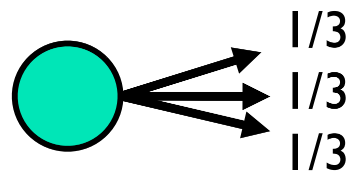
Link-based ranking

- Query processing with link-based ranking:
  - First retrieve all pages meeting the query (say venture capital)
  - Order these by their link popularity (= citation frequency, first generation)
- … or by Pagerank (second generation)

- Simple link popularity (= number of inlinks of a page) is easy to spam.
- Why?

Pagerank scoring

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



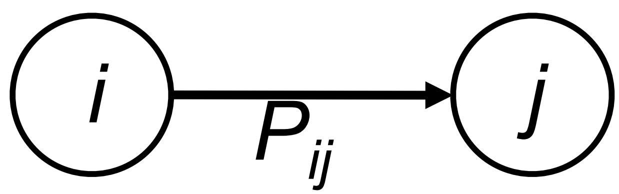
  - At each step, go out of the current page along one of the links on that page, equiprobably

- "In the steady state" each page has a long-term visit rate - use this as the page's score.

- **PageRank = steady state probability = long-term visit rate**

Markov chains

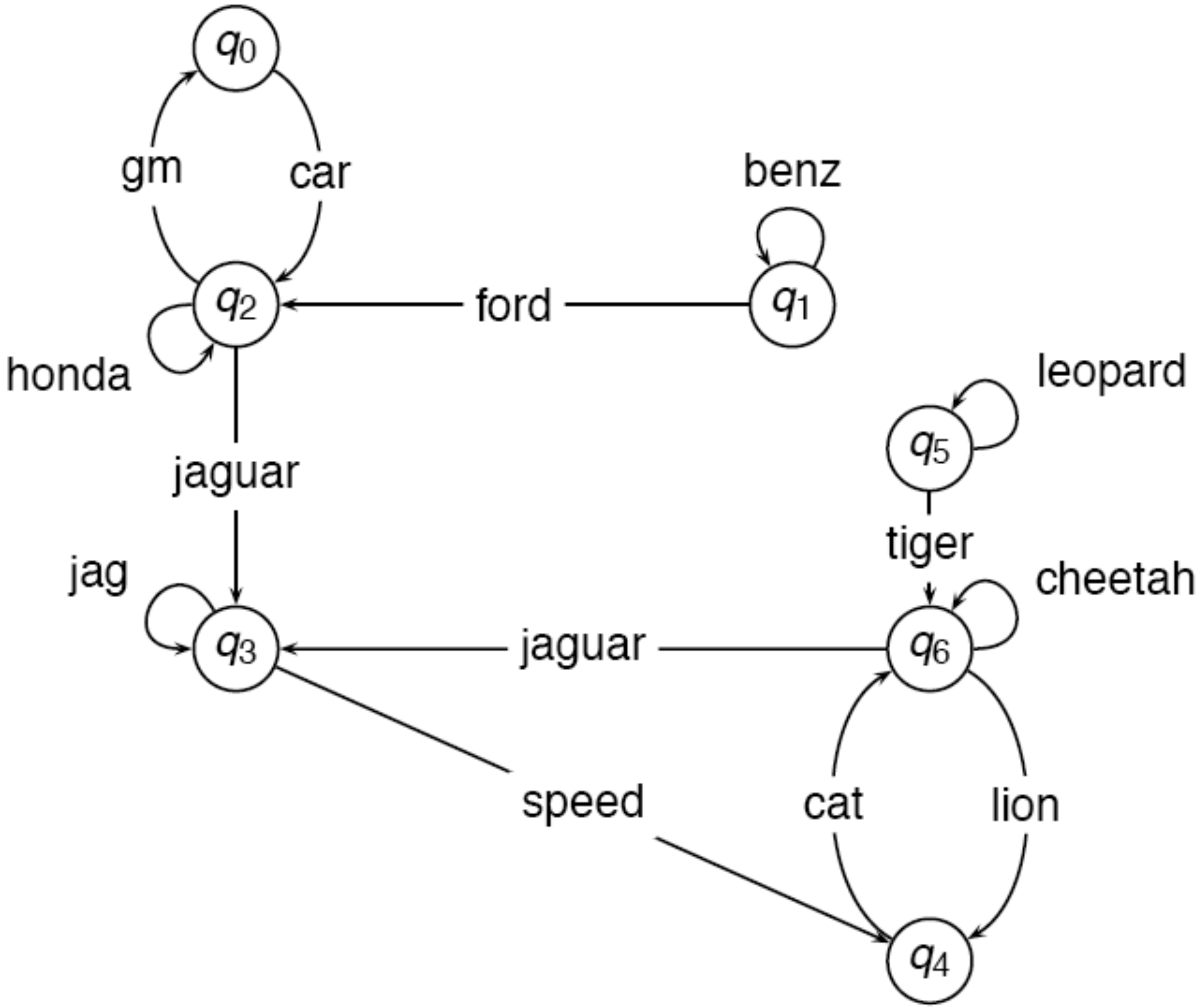
- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix **P**.
- **state = page**
- At each step, we are in exactly one of the states.
- For $1 \le i,j \le n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state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



Markov chains

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

Example web graph



Link matrix for example

	q_0	q_1	q_2	q_3	q_4	q_5	q_6
q_0	0	0	1	0	0	0	0
q_1	0	1	1	0	0	0	0
q_2	1	0	1	1	0	0	0
q_3	0	0	0	1	1	0	0
q_4	0	0	0	0	0	0	1
q_5	0	0	0	0	0	1	1
q_6	0	0	0	1	1	0	1

Transition probability matrix P

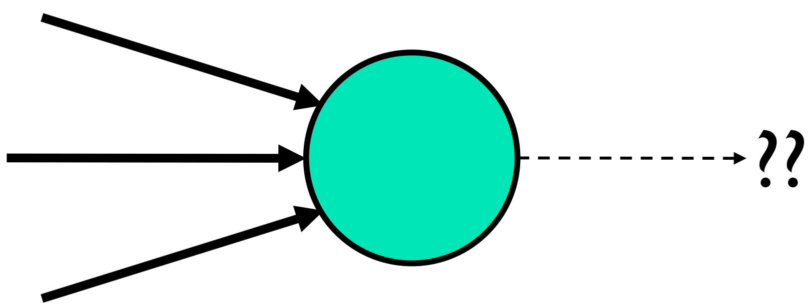
	q_0	q_1	q_2	q_3	q_4	q_5	q_6
q_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
q_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
q_2	0.33	0.00	0.33	0.33	0.00	0.00	0.00
q_3	0.00	0.00	0.00	0.50	0.50	0.00	0.00
q_4	0.00	0.00	0.00	0.00	0.00	0.00	1.00
q_5	0.00	0.00	0.00	0.00	0.00	0.50	0.50
q_6	0.00	0.00	0.00	0.33	0.33	0.00	0.33

Long-term visit rate

- Recall: PageRank = long-term visit rate
- Long-term visit rate of page ***d*** is the probability that a web surfer is at page ***d*** at a given point in time.
- Next: what properties must hold of the web graph for the long-term visit rate to be well defined?
- The web graph must correspond to an **ergodic Markov chain.**
- First a special case: The web graph must not contain dead ends.

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

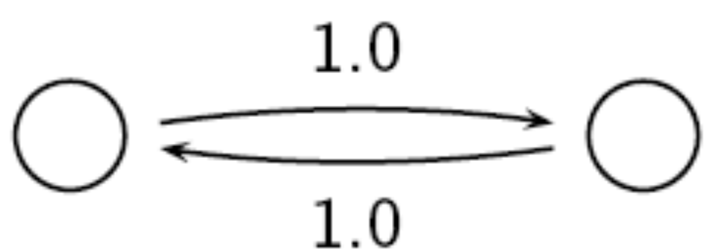
- At a dead end, jump to a random web page.
- At any non-dead end, with probability 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - 10% - a parameter.

Result of teleporting

- With teleporting, we cannot get stuck in a dead end
- Even without dead-ends, a graph may not have well-defined long-term visit rates
- More generally, we require that the Markov chain be ergodic

Ergodic Markov chains

- A Markov chain is ergodic iff it is irreducible and aperiodic
  - Irreducibility. Roughly: there is a path from any page to any other page
  - Aperiodicity. Roughly. The pages cannot be partitioned such that the random walker visits the partitions sequentially
  - A non-ergodic Markov chain:



Ergodic Markov chains

- For any ergodic Markov chain, there is a unique long-term visit rate for each state.
  - *Steady-state probability distribution.*
- Over a long time-period, we visit each state in proportion to this rate.
- It doesn't matter where we start.

Formalization of "visit": Probability vector

- A probability (row) vector **x** $= (x_1, \ldots x_n)$ tells us where the walk is at any point.
- E.g., $\overset{1}{(0}00\ldots\overset{i}{1}\ldots00\overset{n}{0})$ means we're in state *i*.

More generally, the vector **x** $= (x_1, \ldots x_n)$ means the walk is in state *i* with probability x_i.

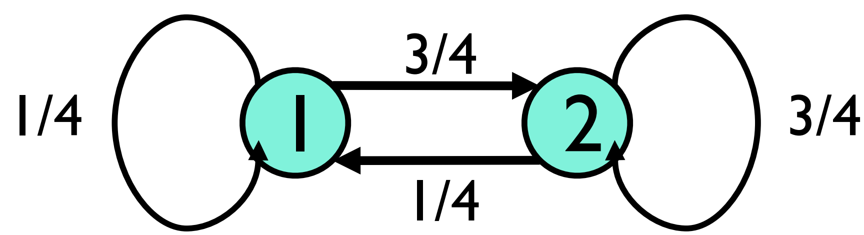
$$\sum_{i=1}^{n} x_i = 1.$$

Change in probability vector

- If the probability vector is $\mathbf{x} = (x_1, \ldots x_n)$ 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 **x**, our next state is distributed as **xP**.

Steady state example

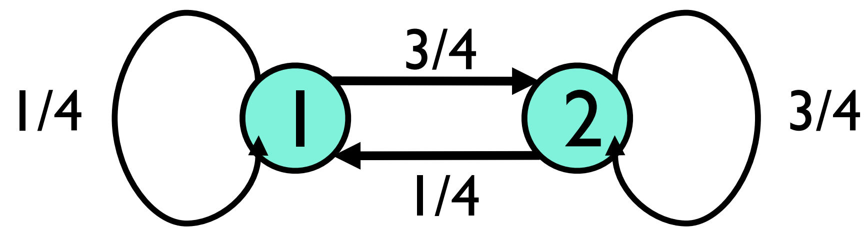
- The steady state looks like a vector of probabilities $\mathbf{a} = (a_1, \ldots a_n)$:
  - a_i is the probability that we are in state i.



What is the steady state in this example?

Steady state example

- The steady state looks like a vector of probabilities $\mathbf{a} = (a_1, \ldots a_n)$:
  - a_i is the probability that we are in state i.



For this example, $a_1=1/4$ and $a_2=3/4$.

How do we compute this vector?

- Let **a** = $(a_1, \ldots a_n)$ denote the row vector of steady-state probabilities.
- If we our current position is described by **a**, then the next step is distributed as **aP**.
- But **a** is the steady state, so **a**=**aP**.
- Solving this matrix equation gives us **a**.
  - So **a** is the (left) eigenvector for **P**.
  - (Corresponds to the "principal" eigenvector of **P** with the largest eigenvalue.)
  - Transition probability matrices always have largest eigenvalue 1.

One way of computing **a**

- Recall, regardless of where we start, we eventually reach the steady state **a**.
- Start with any distribution (say $\mathbf{x}=(1\,0\ldots0)$).
- After one step, we're at $\mathbf{xP}$;
- after two steps at $\mathbf{xP}^2$, then $\mathbf{xP}^3$ and so on.
- "Eventually" means for "large" k, $\mathbf{xP}^k = \mathbf{a}$.
- Algorithm: multiply **x** by increasing powers of **P** until the product looks stable.

Power method: example

Two-node example: $\vec{x} = (0.5, 0.5)$, $P = \begin{pmatrix} 0.25 & 0.75 \\ 0.25 & 0.75 \end{pmatrix}$

$\vec{x}P = (0.25, 0.75)$

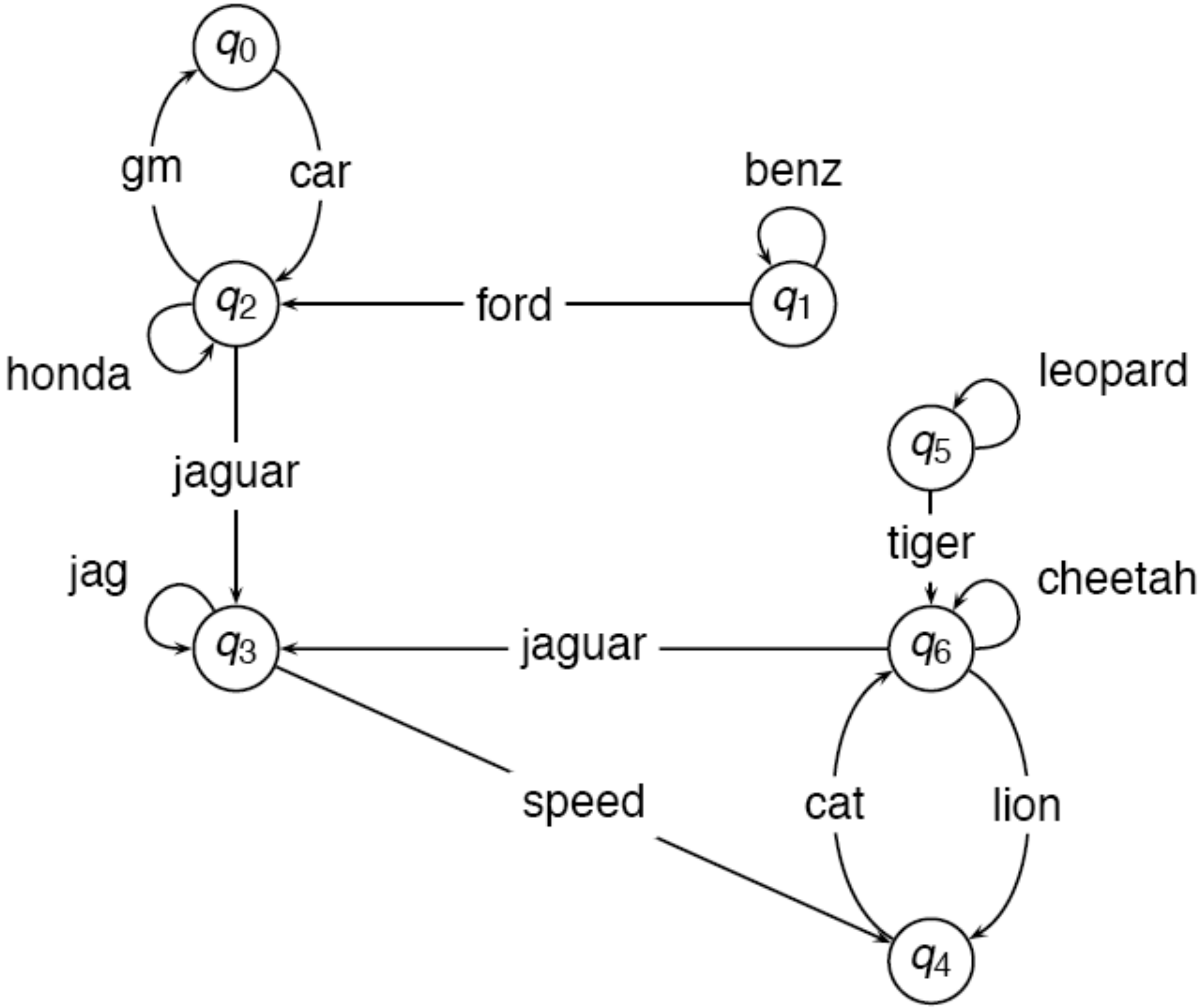
$\vec{x}^2 P = (0.25, 0.75)$

Convergence in one iteration!

Pagerank summary

- Preprocessing:
  - Given graph of links, build matrix **P**.
  - From it compute **a**.
  - The entry a_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*.

Web graph example



Transition matrix

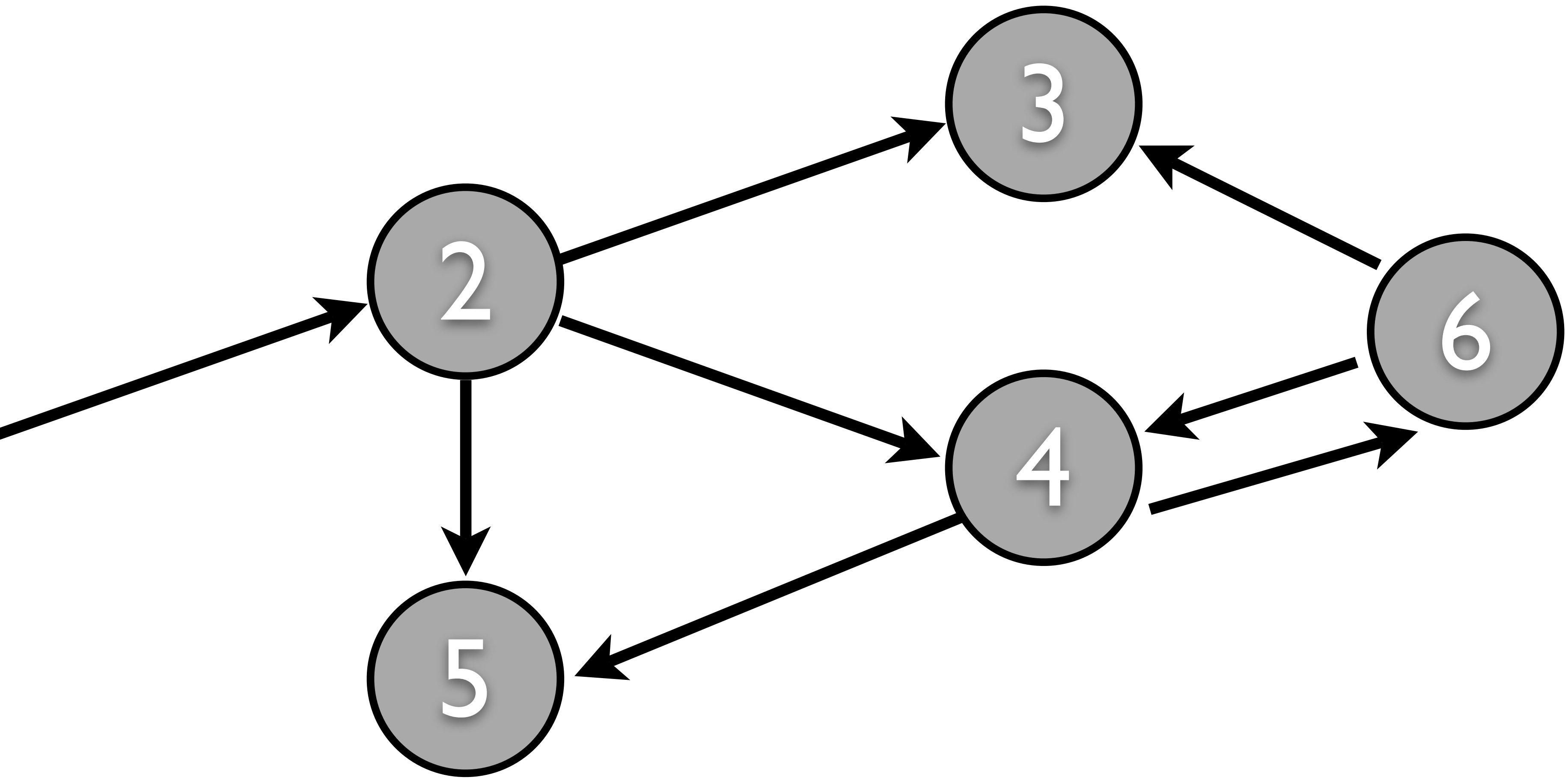
	q_0	q_1	q_2	q_3	q_4	q_5	q_6
q_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
q_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
q_2	0.33	0.00	0.33	0.33	0.00	0.00	0.00
q_3	0.00	0.00	0.00	0.50	0.50	0.00	0.00
q_4	0.00	0.00	0.00	0.00	0.00	0.00	1.00
q_5	0.00	0.00	0.00	0.00	0.00	0.50	0.50
q_6	0.00	0.00	0.00	0.33	0.33	0.00	0.33

Transition matrix with teleporting

	q_0	q_1	q_2	q_3	q_4	q_5	q_6
q_0	0.02	0.02	0.88	0.02	0.02	0.02	0.02
q_1	0.02	0.45	0.45	0.02	0.02	0.02	0.02
q_2	0.31	0.02	0.31	0.31	0.02	0.02	0.02
q_3	0.02	0.02	0.02	0.45	0.45	0.02	0.02
q_4	0.02	0.02	0.02	0.02	0.02	0.02	0.88
q_5	0.02	0.02	0.02	0.02	0.02	0.45	0.45
q_6	0.02	0.02	0.02	0.31	0.31	0.02	0.31

Power method

	$\vec{x}$	$\vec{x}P^1$	$\vec{x}P^2$	$\vec{x}P^3$	$\vec{x}P^4$	$\vec{x}P^5$	$\vec{x}P^6$	$\vec{x}P^7$	$\vec{x}P^8$	$\vec{x}P^9$	$\vec{x}P^{10}$	$\vec{x}P^{11}$
q_0	0.14	0.06	0.09	0.07	0.07	0.06	0.06	0.06	0.06	0.05	0.05	0.05
q_1	0.14	0.08	0.06	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
q_2	0.14	0.25	0.18	0.17	0.15	0.14	0.13	0.12	0.12	0.12	0.12	0.11
q_3	0.14	0.16	0.23	0.24	0.24	0.24	0.24	0.25	0.25	0.25	0.25	0.25
q_4	0.14	0.12	0.16	0.19	0.19	0.20	0.21	0.21	0.21	0.21	0.21	0.21
q_5	0.14	0.08	0.06	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
q_6	0.14	0.25	0.23	0.25	0.27	0.28	0.29	0.29	0.30	0.30	0.30	0.30



3
2
6
4
5

- Write down the probability transition matrix, assuming alpha = 5/6
- Suppose we use the power method to solve, what is one possible initial distribution we could use as input?
- Suppose we use an initial distribution different from the one you suggested. Will the choice have any impact on the PageRank calculation?

PageRank issues

- Real surfers are not random surfers – Markov model is not a good model of surfing.
    - Issues: back button, short vs. long paths, bookmarks, directories – and search!
- Simple PageRank ranking (as described on previous slide) produces bad results for many pages.
    - Consider the query ***video service***
    - The Yahoo home page (i) has a very high PageRank and (ii) contains both words.
    - If we rank all Boolean hits according to PageRank, then the Yahoo home page would be top-ranked.
    - Clearly not desirable
- In practice: rank according to weighted combination of many factors, including raw text match, anchor text match, PageRank and many other factors

How important is PageRank?

- Frequent claim: Pagerank is the most important component of web ranking.
- The reality:
  - There are several components that are at least as important: e.g., anchor text indexing and zone weighting, phrases . . .
  - Rumor has it that Pagerank in its original form (as presented here) has a negligible impact on ranking!
  - However, variants of a page's pagerank are still an essential part of ranking.
  - Addressing link spam is difficult and crucial

Topic-specific PageRank

Topic Specific Pagerank [Have02]

- Conceptually, we use a random surfer who teleports, with say 10% probability, using the following rule:
  - Selects a category (say, one of the 16 top level ODP categories) based on a query & user -specific distribution over the categories
  - Teleport to a page uniformly at random within the chosen category
- Sounds hard to implement: can't compute PageRank at query time!

Topic Specific Pagerank [Have02]

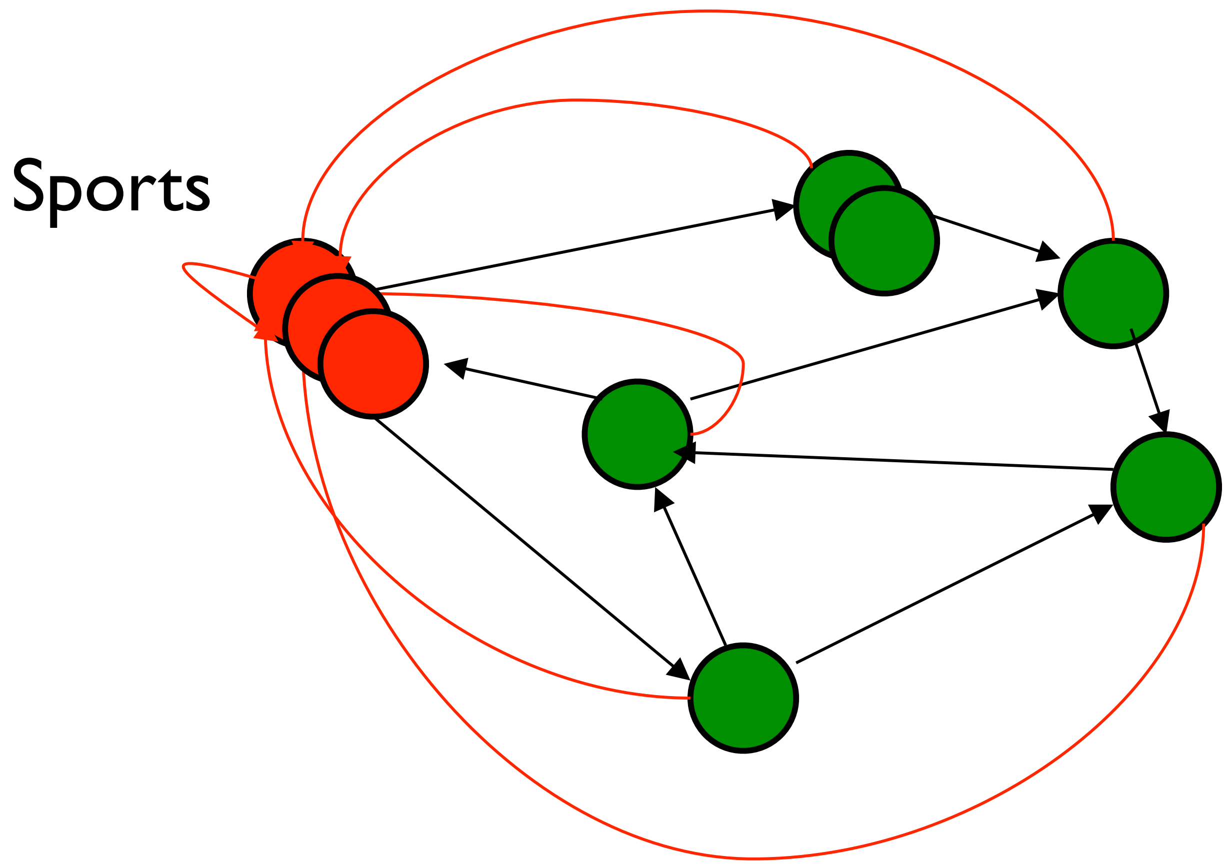
- Implementation
  - **offline:** Compute pagerank distributions wrt to individual categories
  - Query independent model as before
  - Each page has multiple pagerank scores – one for each ODP category, with teleportation only to that category
- **online:** Distribution of weights over categories computed by query context classification
  - Generate a dynamic pagerank score for each page - weighted sum of category-specific pageranks

Influencing PageRank (“Personalization”)

- Input:
  - Web graph W
  - influence vector $\mathbf{v}$

    $\mathbf{v}$: (page $\rightarrow$ degree of influence)
- Output:
  - Rank vector $\mathbf{r}$: (page $\rightarrow$ page importance wrt $\mathbf{v}$)
- $\mathbf{r} = \mathrm{PR}(W, \mathbf{v})$

Non-uniform Teleportation



Teleport with 10% probability to a Sports page

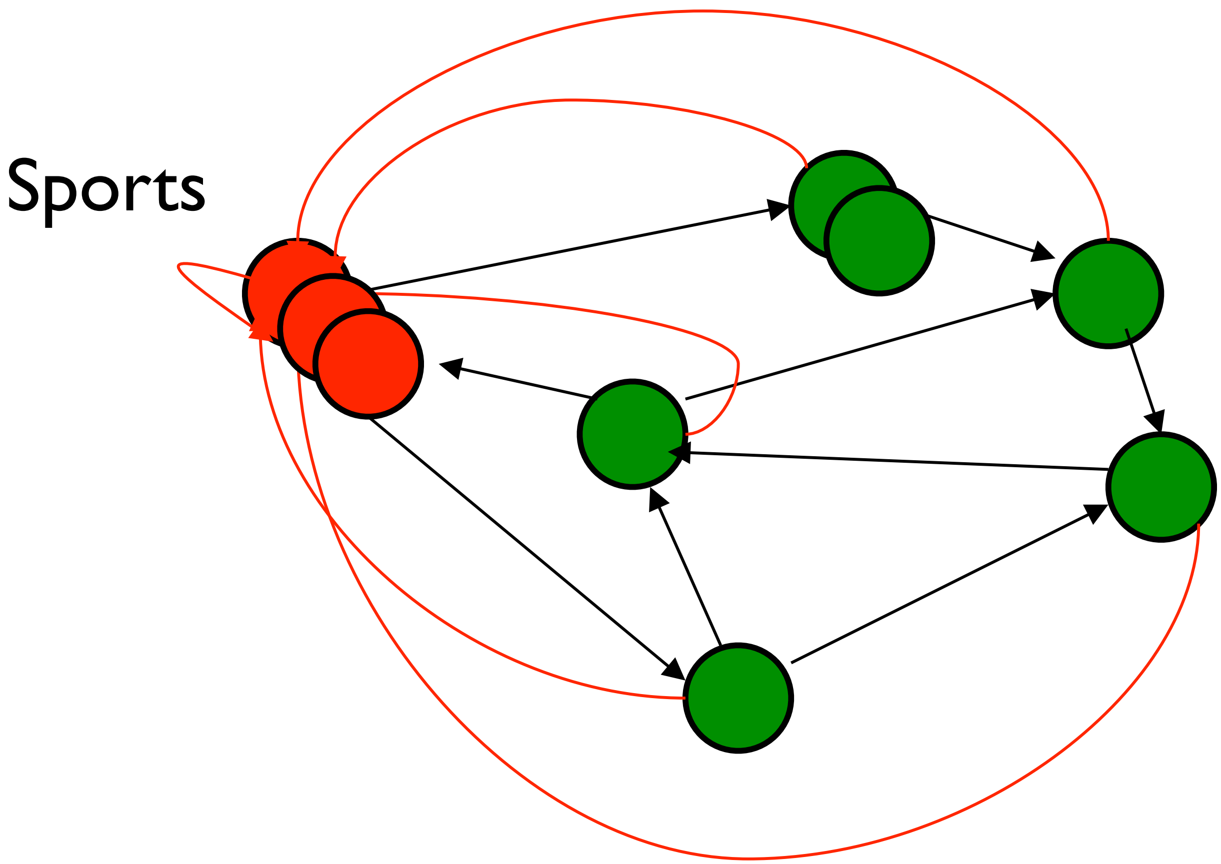
Interpretation of Composite Score

- For a set of personalization vectors $\{\mathbf{v_j}\}$

  $\sum_j [w_j \cdot PR(W, \mathbf{v}_j)] = PR(W, \sum_j [w_j \cdot \mathbf{v}_j])$

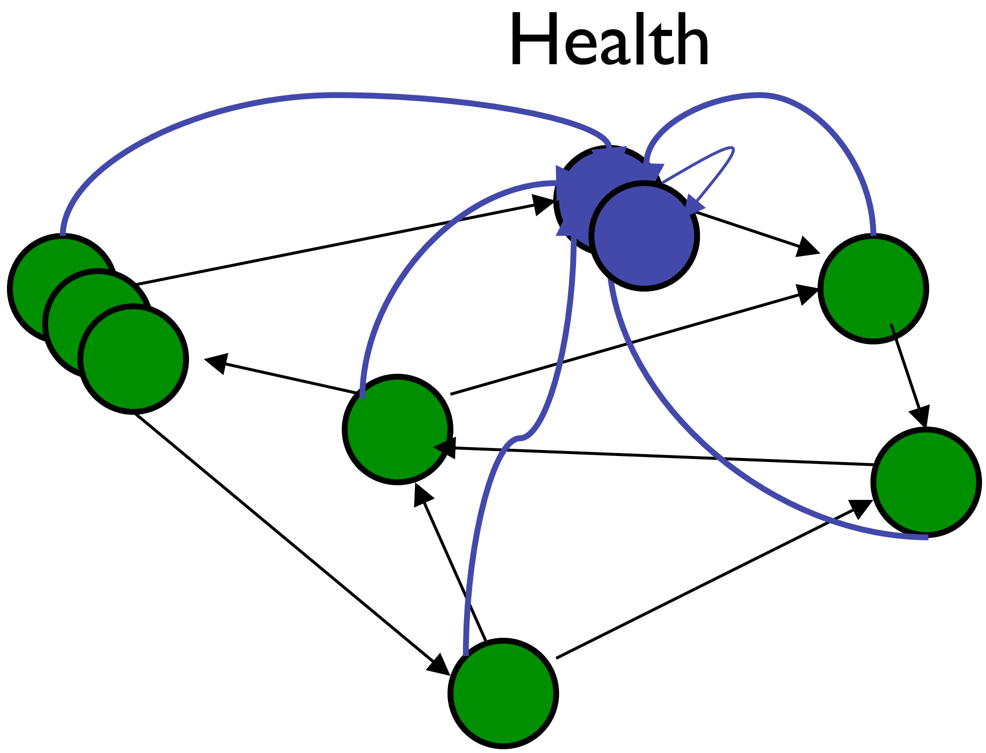
- Weighted sum of rank vectors itself forms a valid rank vector, because PR() is linear wrt $\mathbf{v_j}$

Interpretation



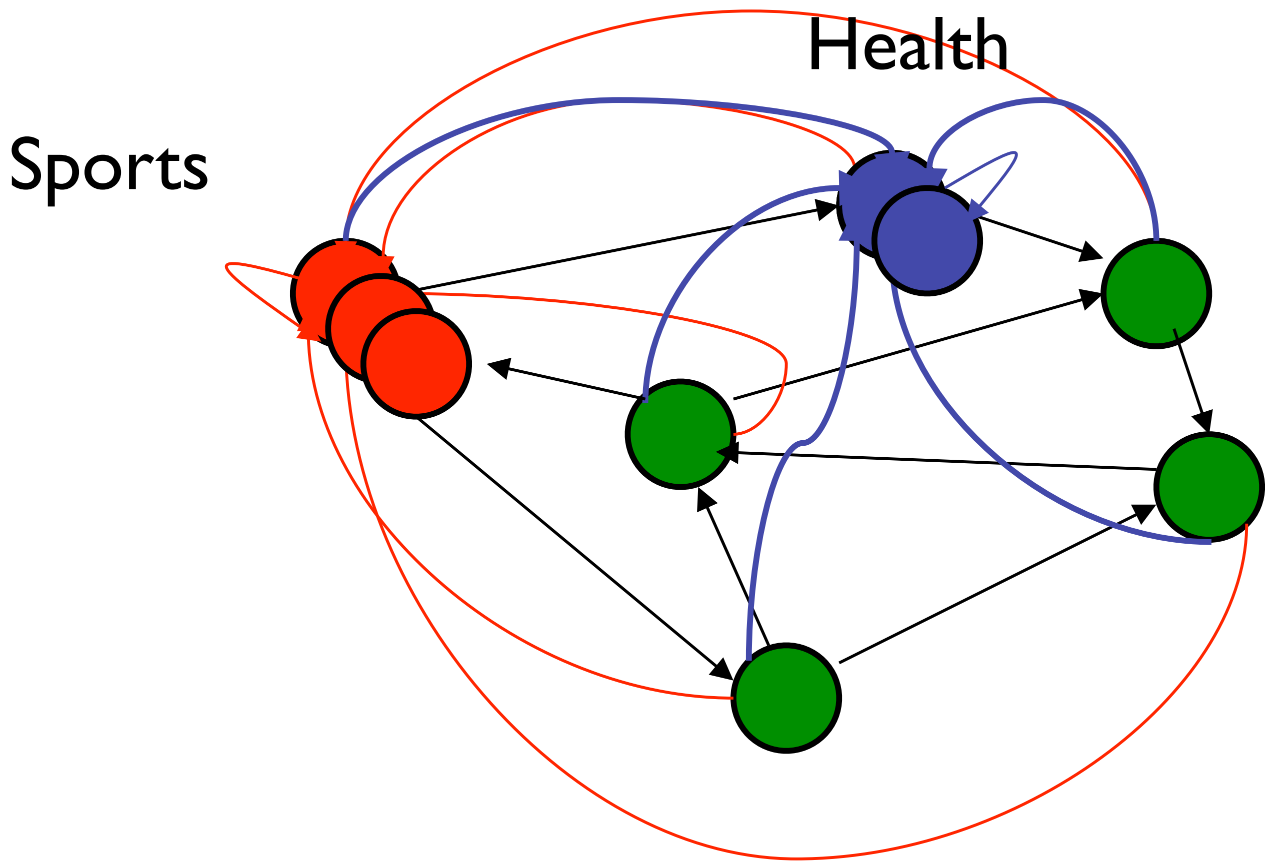
Teleport with 10% probability to a Sports page

Interpretation



10% Health teleportation

Interpretation



pr = (0.9 PR_{sports} + 0.1 PR_{health}) gives you:

9% sports teleportation, 1% health teleportation