PageRank

CS5154/6054

Yizong Cheng

11/29/2022

21 *Link analysis*

21.2 PageRank

PAGERANK

We now focus on scoring and ranking measures derived from the link structure alone. Our first technique for link analysis assigns to every node in the web graph a numerical score between 0 and 1, known as its *PageRank*. The PageRank of a node will depend on the link structure of the web graph. Given a query, a web search engine computes a composite score for each web page that combines hundreds of features such as cosine similarity (Section 6.3) and term proximity (Section 7.2.2), together with the PageRank score. This composite score, developed using the methods of Section 15.4.1, is used to provide a ranked list of results for the query.

Introduction to

Information Retrieval

CS276

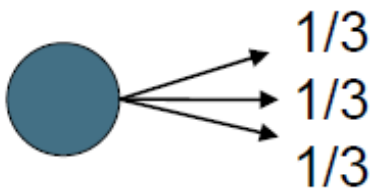
Information Retrieval and Web Search

Chris Manning and Pandu Nayak

Link analysis

Pagerank scoring

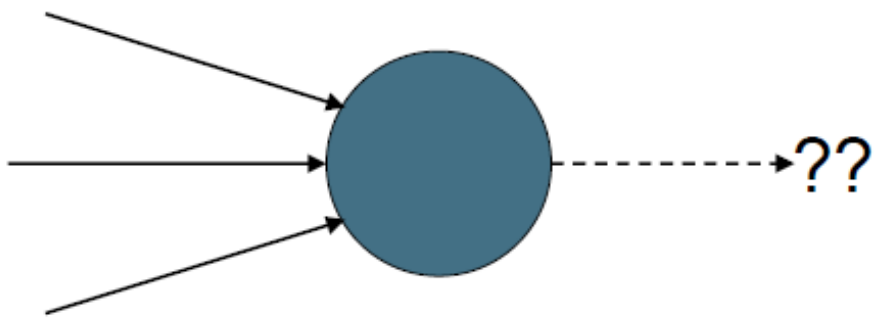
- Imagine a user doing a random walk on web pages:
  - Start at a random page
  - At each step, go out of the current page along one of the links on that page, equiprobably



- "In the long run" 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

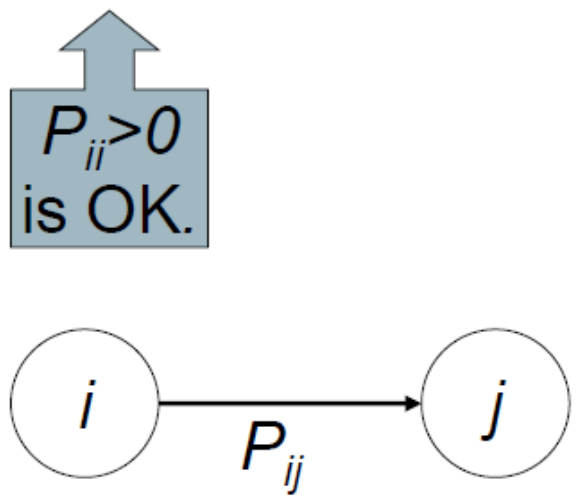
- At a dead end, jump to a random web page.
- At any non-dead end, with probability 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - 10% - a parameter.
    - "Teleportation" probability
    - Simulates a web users going somewhere else
    - Solves linear algebra problems….

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?

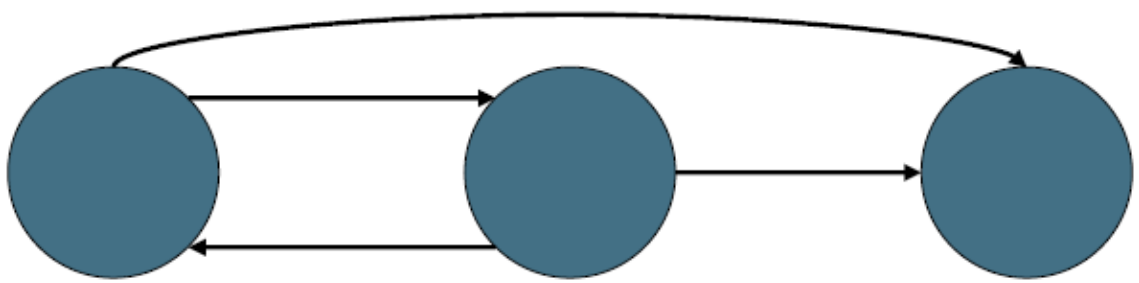
Markov chains

- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix **P**.
- At each step, we are in 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.



Markov chains

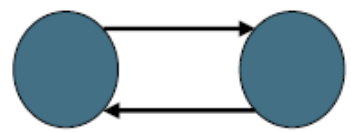
- Clearly, for all *i*, $\sum_{j=1}^{n} P_{ij} = 1.$
- Markov chains are abstractions of random walks.
- *Exercise*: represent the teleporting random walk from 3 slides ago as a Markov chain, for this case:



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.

- Ergodic: no periodic patterns
  - Teleportation ensures ergodicity



Not ergodic

Probability vectors

- A probability (row) vector $\mathbf{x} = (x_1, \ldots x_n)$ tells us where the walk is at any point.
- E.g., (000…1…000) means we're in state i.
  $1 \quad i \quad n$

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

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

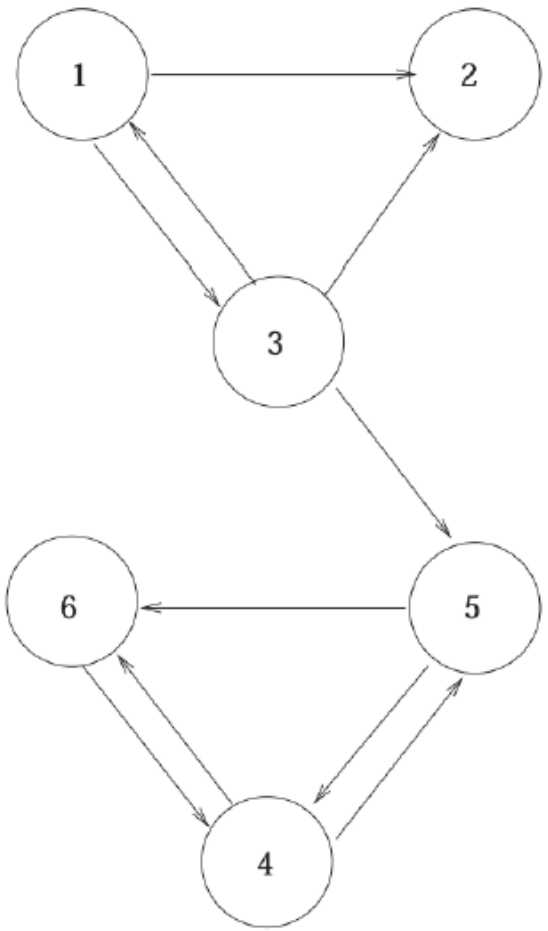
Change in probability vector

- 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**
  - The one after that is $\mathbf{xP}^2$, then $\mathbf{xP}^3$, etc.
  - (Where) Does this converge?
  - Running this and finding out is "the power method"
    - It's actually the method of choice, done with sparse P

How do we compute this vector?

- Let **a** = $(a_1, \ldots a_n)$ denote the row vector of steady-state probabilities.
- If 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. (See: Perron-Frobenius theorem.)
  - Transition probability matrices always have largest eigenvalue 1.

Example: Mini web graph



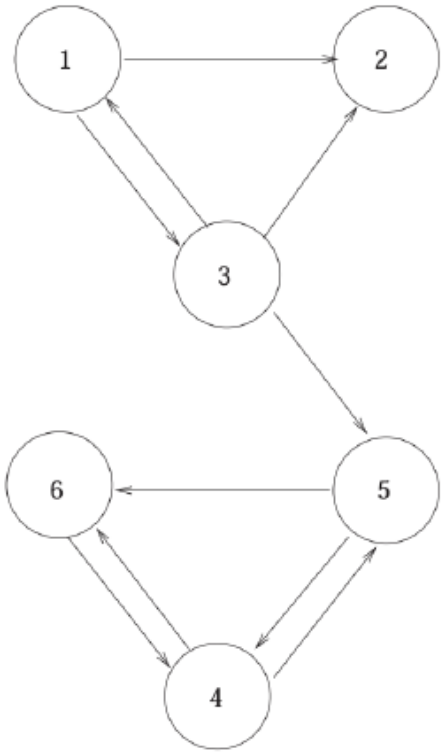
$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Example: Fixing sinks and teleporting

$$\bar{\mathbf{P}} = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{e}\mathbf{e}^T/n = \begin{pmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{pmatrix}$$

Example: Doing power iteration




```
import numpy as np
```

```
x0 = np.matrix([1/6, 1/6, 1/6, 1/6, 1/6, 1/6])
```

```
P = np.matrix([[1/60, 7/15, 7/15, 1/60, 1/60, 1/60],
               [1/6, 1/6, 1/6, 1/6, 1/6, 1/6],
               [19/60, 19/60, 1/60, 1/60, 19/60, 1/60],
               [1/60, 1/60, 1/60, 1/60, 7/15, 7/15],
               [1/60, 1/60, 1/60, 7/15, 1/60, 7/15],
               [1/60, 1/60, 1/60, 11/12, 1/60, 1/60]])
```

```
print(x0 * P)
```

```
[[0.09166667 0.16666667 0.11666667 0.26666667 0.16666667 0.19166667]]
```

```
print(x0 * P * P)
```

```
[[0.07666667 0.11791667 0.08291667 0.28916667 0.19666667 0.23666667]]
```

```
print(x0 * P * P * P * P)
```

```
[[0.05138229 0.07803542 0.05737917 0.34361667 0.20251667 0.26706979]]
```

```
print(x0 * P * P * P * P * P * P * P * P)
```

```
[[0.0391419  0.05730065 0.04374176 0.37100521 0.20527182 0.28353866]]
```

```
print(x0 * P * P * P * P * P * P * P * P * P * P * P * P * P * P * P * P)
```

```
[[0.03724891 0.05402154 0.04154868 0.37500616 0.20598094 0.28619378]]
```

21.2.1 Markov chains

A Markov chain is a *discrete-time stochastic process:* a process that occurs in a series of time-steps in each of which a random choice is made. A Markov chain consists of N *states*. Each web page will correspond to a state in the Markov chain we will formulate.

A Markov chain is characterized by an $N \times N$ *transition probability matrix* P each of whose entries is in the interval $[0, 1]$; the entries in each row of P add up to 1. The Markov chain can be in one of the N states at any given time-step; then, the entry P_{ij} tells us the probability that the state at the next time-step is j, conditioned on the current state being i. Each entry P_{ij} is known as a transition probability and depends only on the current state i; this is known as the Markov property. Thus, by the Markov property,

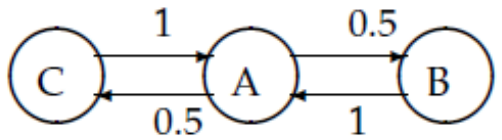
$$\forall i, j, P_{ij} \in [0, 1]$$

and

$$\forall i, \sum_{j=1}^{N} P_{ij} = 1. \tag{21.1}$$

STOCHASTIC MATRIX
PRINCIPAL LEFT EIGENVECTOR

A matrix with non-negative entries that satisfies Equation (21.1) is known as a *stochastic matrix*. A key property of a stochastic matrix is that it has a *principal left eigenvector* corresponding to its largest eigenvalue, which is 1.



▶ **Figure 21.2** A simple Markov chain with three states; the numbers on the links indicate the transition probabilities.

In a Markov chain, the probability distribution of next states for a Markov chain depends only on the current state, and not on how the Markov chain arrived at the current state. Figure 21.2 shows a simple Markov chain with three states. From the middle state A, we proceed with (equal) probabilities of 0.5 to either B or C. From either B or C, we proceed with probability 1 to A. The transition probability matrix of this Markov chain is then

$$\begin{pmatrix} 0 & 0.5 & 0.5 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

PROBABILITY VECTOR

A Markov chain's probability distribution over its states may be viewed as a *probability vector*: a vector all of whose entries are in the interval $[0, 1]$, and the entries add up to 1. An N-dimensional probability vector each of whose components corresponds to one of the N states of a Markov chain can be viewed as a probability distribution over its states. For our simple Markov chain of Figure 21.2, the probability vector would have 3 components that sum to 1.

We can view a random surfer on the web graph as a Markov chain, with one state for each web page, and each transition probability representing the probability of moving from one web page to another. The teleport operation contributes to these transition probabilities. The adjacency matrix A of the web graph is defined as follows: if there is a hyperlink from page i to page j, then $A_{ij} = 1$, otherwise $A_{ij} = 0$. We can readily derive the transition probability matrix P for our Markov chain from the $N \times N$ matrix A:

1. If a row of A has no 1's, then replace each element by 1/N. For all other rows proceed as follows.

2. Divide each 1 in A by the number of 1's in its row. Thus, if there is a row with three 1's, then each of them is replaced by 1/3.

3. Multiply the resulting matrix by $1 - \alpha$.

4. Add α/N to every entry of the resulting matrix, to obtain P.

We can depict the probability distribution of the surfer's position at any time by a probability vector $\vec{x}$. At $t = 0$ the surfer may begin at a state whose corresponding entry in $\vec{x}$ is 1 while all others are zero. By definition, the surfer's distribution at $t = 1$ is given by the probability vector $\vec{x}P$; at $t = 2$ by $(\vec{x}P)P = \vec{x}P^2$, and so on. We will detail this process in Section 21.2.2. We can thus compute the surfer's distribution over the states at any time, given only the initial distribution and the transition probability matrix P.

Exercise 21.6

Consider a web graph with three nodes 1, 2 and 3. The links are as follows: $1 \to 2, 3 \to 2, 2 \to 1, 2 \to 3$. Write down the transition probability matrices for the surfer's walk with teleporting, for the following three values of the teleport probability: (a) $\alpha = 0$; (b) $\alpha = 0.5$ and (c) $\alpha = 1$.

We consider the web graph in Exercise 21.6 with $\alpha = 0.5$. The transition probability matrix of the surfer's walk with teleportation is then

$$P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}. \tag{21.3}$$

ERGODIC MARKOV CHAIN

Definition: A Markov chain is said to be *ergodic* if there exists a positive integer T_0 such that for all pairs of states i, j in the Markov chain, if it is started at time 0 in state i then for all $t > T_0$, the probability of being in state j at time t is greater than 0.

For a Markov chain to be ergodic, two technical conditions are required of its states and the non-zero transition probabilities; these conditions are known as *irreducibility* and *aperiodicity*. Informally, the first ensures that there is a sequence of transitions of non-zero probability from any state to any other, while the latter ensures that the states are not partitioned into sets such that all state transitions occur cyclically from one set to another.

STEADY-STATE

Theorem 21.1. *For any ergodic Markov chain, there is a unique* steady-state *probability vector $\vec{\pi}$ that is the principal left eigenvector of P, such that if $\eta(i,t)$ is the number of visits to state i in t steps, then*

$$\lim_{t\to\infty} \frac{\eta(i,t)}{t} = \pi(i),$$

where $\pi(i) > 0$ is the steady-state probability for state i.

It follows from Theorem 21.1 that the random walk with teleporting results in a unique distribution of steady-state probabilities over the states of the induced Markov chain. This steady-state probability for a state is the PageRank of the corresponding web page.

21.2.2 The PageRank computation

How do we compute PageRank values? Recall the definition of a left eigenvector from Equation 18.2; the left eigenvectors of the transition probability matrix P are N-vectors $\vec{\pi}$ such that

$$\vec{\pi} P = \lambda \vec{\pi}. \tag{21.2}$$

The N entries in the principal eigenvector $\vec{\pi}$ are the steady-state probabilities of the random walk with teleporting, and thus the PageRank values for the corresponding web pages. We may interpret Equation (21.2) as follows: if $\vec{\pi}$ is the probability distribution of the surfer across the web pages, he remains in the steady-state distribution $\vec{\pi}$. Given that $\vec{\pi}$ is the steady-state distribution, we have that $\pi P = 1\pi$, so 1 is an eigenvalue of P. Thus if we were to compute the principal left eigenvector of the matrix P — the one with eigenvalue 1 — we would have computed the PageRank values.

$\vec{x}_0$	1	0	0
$\vec{x}_1$	1/6	2/3	1/6
$\vec{x}_2$	1/3	1/3	1/3
$\vec{x}_3$	1/4	1/2	1/4
$\vec{x}_4$	7/24	5/12	7/24
...	...	...	...
$\vec{x}$	5/18	4/9	5/18

► **Figure 21.3** The sequence of probability vectors.

We consider the web graph in Exercise 21.6 with $\alpha = 0.5$. The transition probability matrix of the surfer's walk with teleportation is then

$$P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}. \tag{21.3}$$

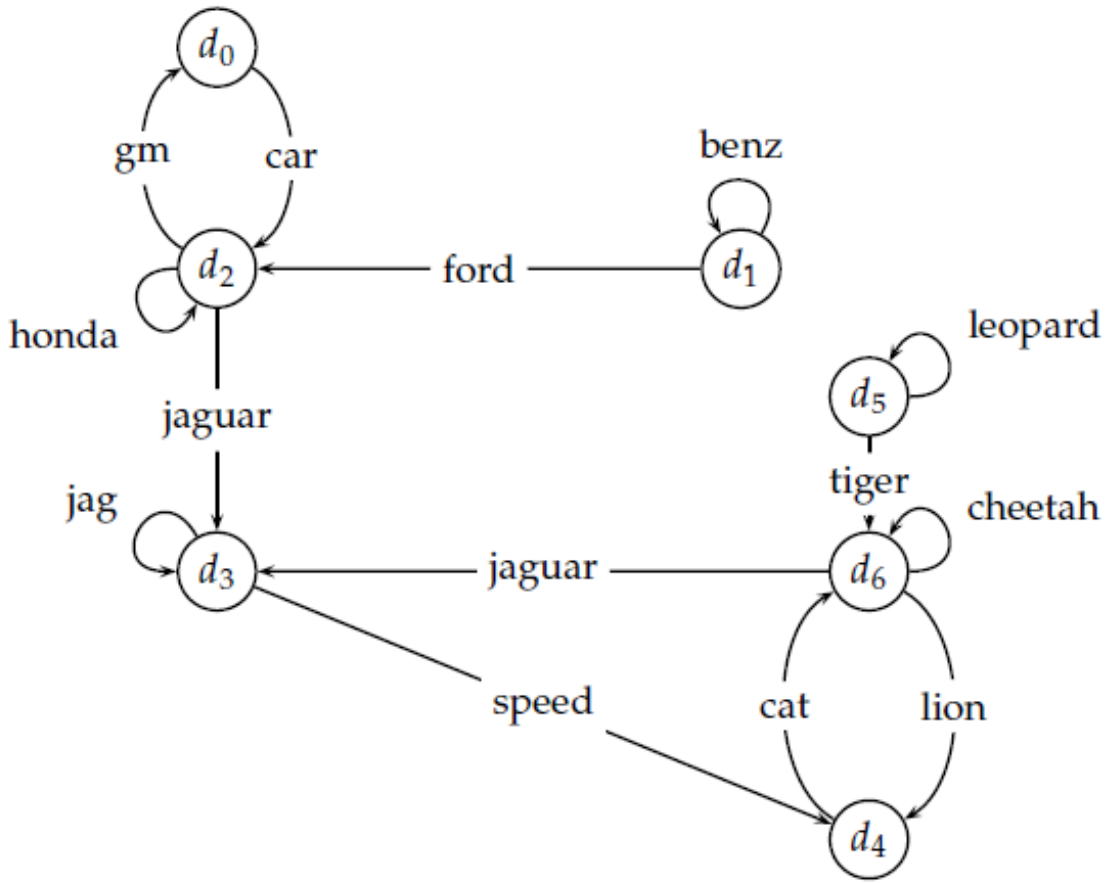
Imagine that the surfer starts in state 1, corresponding to the initial probability distribution vector $\vec{x}_0 = (1\ 0\ 0)$. Then, after one step the distribution is

$$\vec{x}_0 P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \end{pmatrix} = \vec{x}_1. \tag{21.4}$$

After two steps it is

$$\vec{x}_1 P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \end{pmatrix} \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix} = \begin{pmatrix} 1/3 & 1/3 & 1/3 \end{pmatrix} = \vec{x}_2. \tag{21.5}$$

Continuing for several steps, we see that the distribution converges to the steady state of $\vec{x} = (5/18 \quad 4/9 \quad 5/18)$. In this simple example, we may directly calculate this steady-state probability distribution by observing the symmetry of the Markov chain: states 1 and 3 are symmetric, as evident from the fact that the first and third rows of the transition probability matrix in Equation (21.3) are identical. Postulating, then, that they both have the same steady-state probability and denoting this probability by p, we know that the steady-state distribution is of the form $\vec{\pi} = (p \quad 1-2p \quad p)$. Now, using the identity $\vec{\pi} = \vec{\pi}P$, we solve a simple linear equation to obtain $p = 5/18$ and consequently, $\vec{\pi} = (5/18\ 4/9\ 5/18)$.



► **Figure 21.4** A small web graph. Arcs are annotated with the word that occurs in the anchor text of the corresponding link.

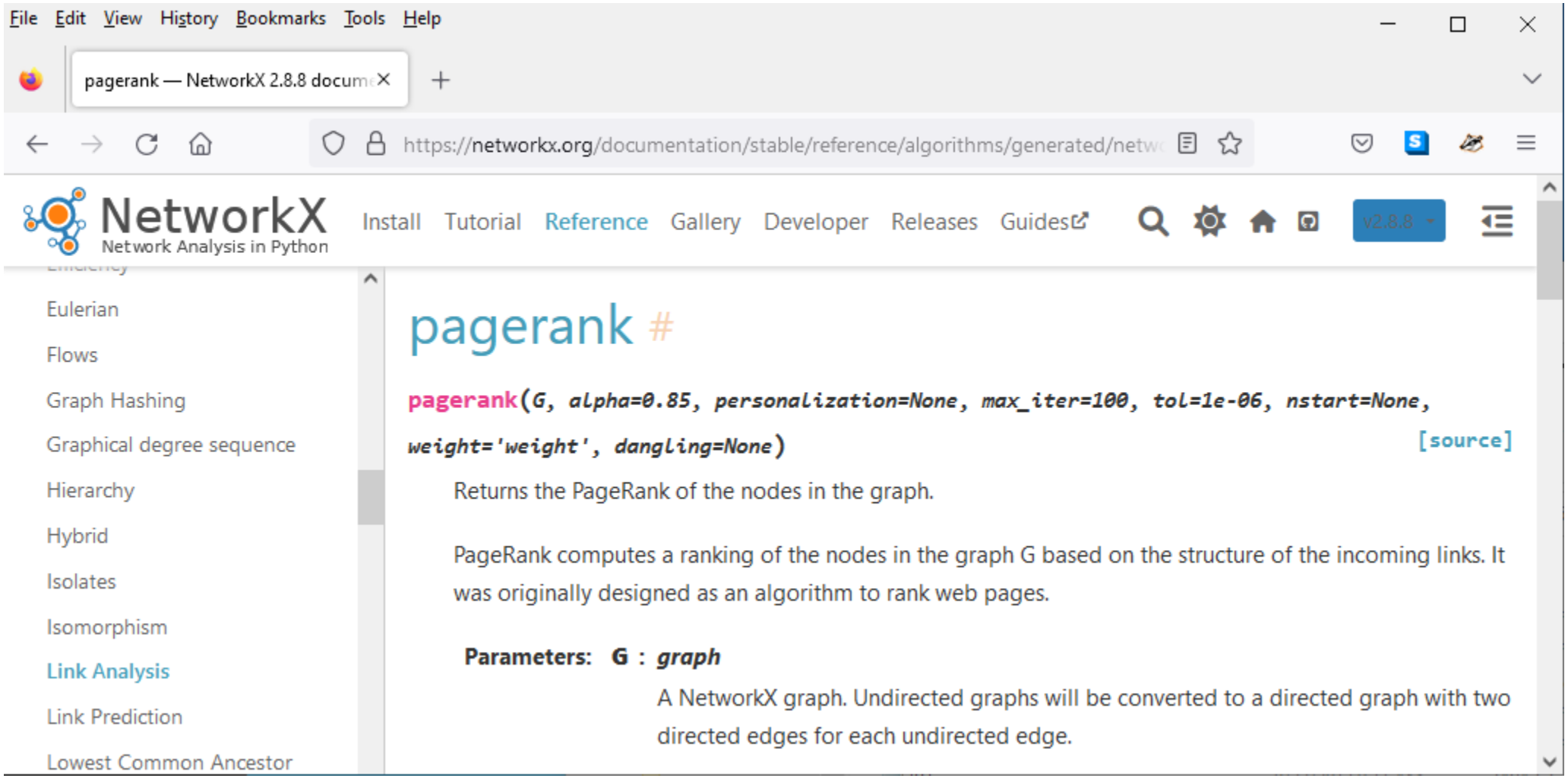
Example 21.1: Consider the graph in Figure 21.4. For a teleportation rate of 0.14 its (stochastic) transition probability matrix is:

$$\begin{array}{ccccccc} 0.02 & 0.02 & 0.88 & 0.02 & 0.02 & 0.02 & 0.02 \\ 0.02 & 0.45 & 0.45 & 0.02 & 0.02 & 0.02 & 0.02 \\ 0.31 & 0.02 & 0.31 & 0.31 & 0.02 & 0.02 & 0.02 \\ 0.02 & 0.02 & 0.02 & 0.45 & 0.45 & 0.02 & 0.02 \\ 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.88 \\ 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.45 & 0.45 \\ 0.02 & 0.02 & 0.02 & 0.31 & 0.31 & 0.02 & 0.31 \end{array}$$

The PageRank vector of this matrix is:

$$\vec{x} = (0.05 \quad 0.04 \quad 0.11 \quad 0.25 \quad 0.21 \quad 0.04 \quad 0.31) \tag{21.6}$$

Observe that in Figure 21.4, q_2, q_3, q_4 and q_6 are the nodes with at least two in-links. Of these, q_2 has the lowest PageRank since the random walk tends to drift out of the top part of the graph – the walker can only return there through teleportation.


File Edit View History Bookmarks Tools Help
pagerank — NetworkX 2.8.8 docum
https://networkx.org/documentation/stable/reference/algorithms/generated/netw
NetworkX
Network Analysis in Python
Install Tutorial Reference Gallery Developer Releases Guides
v2.8.8
Eulerian
Flows
Graph Hashing
Graphical degree sequence
Hierarchy
Hybrid
Isolates
Isomorphism
Link Analysis
Link Prediction
Lowest Common Ancestor
pagerank #
pagerank(G, alpha=0.85, personalization=None, max_iter=100, tol=1e-06, nstart=None, weight='weight', dangling=None)
[source]
Returns the PageRank of the nodes in the graph.
PageRank computes a ranking of the nodes in the graph G based on the structure of the incoming links. It was originally designed as an algorithm to rank web pages.
Parameters: G : graph
A NetworkX graph. Undirected graphs will be converted to a directed graph with two directed edges for each undirected edge.