21 *Link analysis*

The analysis of hyperlinks and the graph structure of the Web has been instrumental in the development of web search. In this chapter we focus on the use of hyperlinks for ranking web search results. Such link analysis is one of many factors considered by web search engines in computing a composite score for a web page on any given query. We begin by reviewing some basics of the Web as a graph in Section 21.1, then proceed to the technical development of the elements of link analysis for ranking.

Link analysis for web search has intellectual antecedents in the field of citation analysis, aspects of which overlap with an area known as bibliometrics. These disciplines seek to quantify the influence of scholarly articles by analyzing the pattern of citations amongst them. Much as citations represent the conferral of authority from a scholarly article to others, link analysis on the Web treats hyperlinks from a web page to another as a conferral of authority. Clearly, not every citation or hyperlink implies such authority conferral; for this reason, simply measuring the quality of a web page by the number of in-links (citations from other pages) is not robust enough. For instance, one may contrive to set up multiple web pages pointing to a target web page, with the intent of artificially boosting the latter's tally of in-links. This phenomenon is referred to as link spam. Nevertheless, the phenomenon of citation is prevalent and dependable enough that it is feasible for web search engines to derive useful signals for ranking from more sophisticated link analysis. Link analysis also proves to be a useful indicator of what page(s) to crawl next while crawling the web; this is done by using link analysis to guide the priority assignment in the front queues of Chapter 20.

Section 21.1 develops the basic ideas underlying the use of the web graph in link analysis. Sections 21.2 and 21.3 then develop two distinct methods for link analysis, PageRank and HITS.

21.1 The Web as a graph

Recall the notion of the web graph from Section 19.2.1 and particularly Figure 19.2. Our study of link analysis builds on two intuitions:

1. The anchor text pointing to page B is a good description of page B.
2. The hyperlink from A to B represents an endorsement of page B, by the creator of page A. This is not always the case; for instance, many links amongst pages within a single website stem from the user of a common template. For instance, most corporate websites have a pointer from every page to a page containing a copyright notice – this is clearly not an endorsement. Accordingly, implementations of link analysis algorithms will typical discount such "internal" links.

21.1.1 Anchor text and the web graph

The following fragment of HTML code from a web page shows a hyperlink pointing to the home page of the Journal of the ACM:

```
<a href="http://www.acm.org/jacm/">Journal of the ACM.</a>
```

In this case, the link points to the page `http://www.acm.org/jacm/` and the anchor text is *Journal of the ACM.* Clearly, in this example the anchor is descriptive of the target page. But then the target page (B = `http://www.acm.org/jacm/`) itself contains the same description as well as considerable additional information on the journal. So what use is the anchor text?

The Web is full of instances where the page B does not provide an accurate description of itself. In many cases this is a matter of how the publishers of page B choose to present themselves; this is especially common with corporate web pages, where a web presence is a marketing statement. For example, at the time of the writing of this book the home page of the IBM corporation (`http://www.ibm.com`) did not contain the term computer anywhere in its HTML code, despite the fact that IBM is widely viewed as the world's largest computer maker. Similarly, the HTML code for the home page of Yahoo! (`http://www.yahoo.com`) does not at this time contain the word portal.

Thus, there is often a gap between the terms in a web page, and how web users would describe that web page. Consequently, web searchers need not use the terms in a page to query for it. In addition, many web pages are rich in graphics and images, and/or embed their text in these images; in such cases, the HTML parsing performed when crawling will not extract text that is useful for indexing these pages. The "standard IR" approach to this would be to use the methods outlined in Chapter 9 and Section 12.4. The insight

behind anchor text is that such methods can be supplanted by anchor text, thereby tapping the power of the community of web page authors.

The fact that the anchors of many hyperlinks pointing to `http://www.ibm.com` include the word computer can be exploited by web search engines. For instance, the anchor text terms can be included as terms under which to index the target web page. Thus, the postings for the term computer would include the document `http://www.ibm.com` and that for the term portal would include the document `http://www.yahoo.com`, using a special indicator to show that these terms occur as anchor (rather than in-page) text. As with in-page terms, anchor text terms are generally weighted based on frequency, with a penalty for terms that occur very often (the most common terms in anchor text across the Web are Click and here, using methods very similar to idf). The actual weighting of terms is determined by machine-learned scoring, as in Section 15.4.1; current web search engines appear to assign a substantial weighting to anchor text terms.

The use of anchor text has some interesting side-effects. Searching for big blue on most web search engines returns the home page of the IBM corporation as the top hit; this is consistent with the popular nickname that many people use to refer to IBM. On the other hand, there have been (and continue to be) many instances where derogatory anchor text such as evil empire leads to somewhat unexpected results on querying for these terms on web search engines. This phenomenon has been exploited in orchestrated campaigns against specific sites. Such orchestrated anchor text may be a form of spamming, since a website can create misleading anchor text pointing to itself, to boost its ranking on selected query terms. Detecting and combating such systematic abuse of anchor text is another form of spam detection that web search engines perform.

The window of text surrounding anchor text (sometimes referred to as *extended anchor text*) is often usable in the same manner as anchor text itself; consider for instance the fragment of web text `there is good discussion of vedic scripture <a>here</a>`. This has been considered in a number of settings and the useful width of this window has been studied; see Section 21.4 for references.

Exercise 21.1

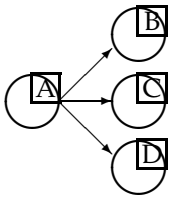
Is it always possible to follow directed edges (hyperlinks) in the web graph from any node (web page) to any other? Why or why not?

Exercise 21.2

Find an instance of misleading anchor-text on the Web.

Exercise 21.3

Given the collection of anchor-text phrases for a web page x, suggest a heuristic for choosing one term or phrase from this collection that is most descriptive of x.



► **Figure 21.1** The random surfer at node A proceeds with probability 1/3 to each of B, C and D.

Exercise 21.4

Does your heuristic in the previous exercise take into account a single domain D repeating anchor text for x from multiple pages in D?

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.

Consider a random surfer who begins at a web page (a node of the web graph) and executes a random walk on the Web as follows. At each time step, the surfer proceeds from his current page A to a randomly chosen web page that A hyperlinks to. Figure 21.1 shows the surfer at a node A, out of which there are three hyperlinks to nodes B, C and D; the surfer proceeds at the next time step to one of these three nodes, with equal probabilities 1/3.

As the surfer proceeds in this random walk from node to node, he visits some nodes more often than others; intuitively, these are nodes with many links coming in from other frequently visited nodes. The idea behind PageRank is that pages visited more often in this walk are more important.

What if the current location of the surfer, the node A, has no out-links? To address this we introduce an additional operation for our random surfer:

TELEPORT

the *teleport* operation. In the teleport operation the surfer jumps from a node to any other node in the web graph. This could happen because he types

an address into the URL bar of his browser. The destination of a teleport operation is modeled as being chosen uniformly at random from all web pages. In other words, if N is the total number of nodes in the web graph[1], the teleport operation takes the surfer to each node with probability $1/N$. The surfer would also teleport to his present position with probability $1/N$.

In assigning a PageRank score to each node of the web graph, we use the teleport operation in two ways: (1) When at a node with no out-links, the surfer invokes the teleport operation. (2) At any node that has outgoing links, the surfer invokes the teleport operation with probability $0 < \alpha < 1$ and the standard random walk (follow an out-link chosen uniformly at random as in Figure 21.1) with probability $1 - \alpha$, where α is a fixed parameter chosen in advance. Typically, α might be 0.1.

In Section 21.2.1, we will use the theory of Markov chains to argue that when the surfer follows this combined process (random walk plus teleport) he visits each node v of the web graph a fixed fraction of the time $\pi(v)$ that depends on (1) the structure of the web graph and (2) the value of α. We call this value $\pi(v)$ the PageRank of v and will show how to compute this value in Section 21.2.2.

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,

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

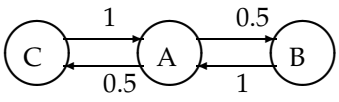
and

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

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.

1. This is consistent with our usage of N for the number of documents in the collection.



► **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.

If a Markov chain is allowed to run for many time steps, each state is visited at a (different) frequency that depends on the structure of the Markov chain. In our running analogy, the surfer visits certain web pages (say, popular news home pages) more often than other pages. We now make this intuition precise, establishing conditions under which such the visit frequency converges to fixed, steady-state quantity. Following this, we set the PageRank of each node v to this steady-state visit frequency and show how it can be computed.

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.

There are many algorithms available for computing left eigenvectors; the references at the end of Chapter 18 and the present chapter are a guide to these. We give here a rather elementary method, sometimes known as *power iteration*. If $\vec{x}$ is the initial distribution over the states, then the distribution at time t is $\vec{x}P^t$. As t grows large, we would expect that the distribution $\vec{x}P^{t2}$ is very similar to the distribution $\vec{x}P^{t+1}$, since for large t we would expect the Markov chain to attain its steady state. By Theorem 21.1 this is independent of the initial distribution $\vec{x}$. The power iteration method simulates the surfer's walk: begin at a state and run the walk for a large number of steps t, keeping track of the visit frequencies for each of the states. After a large number of steps t, these frequencies "settle down" so that the variation in the computed frequencies is below some predetermined threshold. We declare these tabulated frequencies to be the PageRank values.

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

$$\vec{x_0}P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \end{pmatrix} = \vec{x_1}. \tag{21.4}$$

2. Note that P^t represents P raised to the tth power, not the transpose of P which is denoted P^T.

$\vec{x}_0$	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.

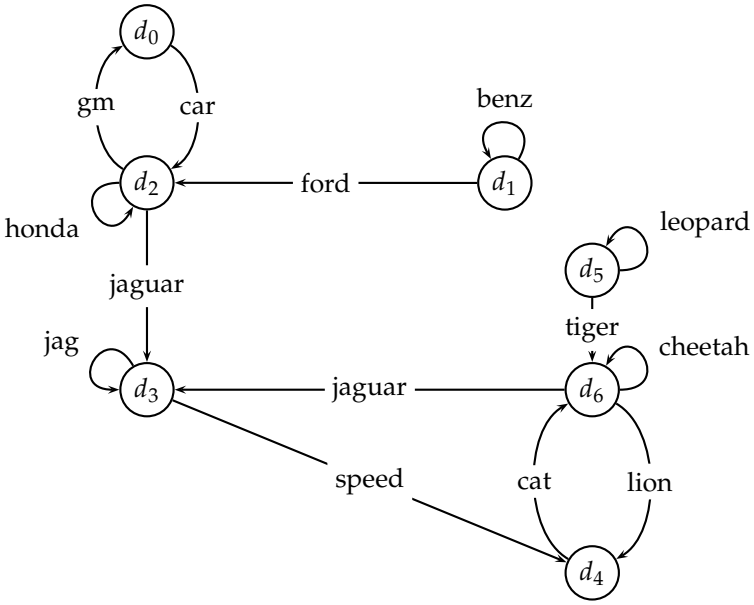
After two steps it is

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

Continuing in this fashion gives a sequence of probability vectors as shown in Figure 21.3.

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, $\vec{\pi} = (5/18\ 4/9\ 5/18)$.

The PageRank values of pages (and the implicit ordering amongst them) are independent of any query a user might pose; PageRank is thus a query-independent measure of the static quality of each web page (recall such static quality measures from Section 7.1.4). On the other hand, the relative ordering of pages should, intuitively, depend on the query being served. For this reason, search engines use static quality measures such as PageRank as just one of many factors in scoring a web page on a query. Indeed, the relative contribution of PageRank to the overall score may again be determined by machine-learned scoring as in Section 15.4.1.



► **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.

21.2.3 Topic-specific PageRank

Thus far we have discussed the PageRank computation with a teleport operation in which the surfer jumps to a random web page chosen uniformly at random. We now consider teleporting to a random web page chosen *non-uniformly*. In doing so, we are able to derive PageRank values tailored to particular interests. For instance, a sports aficionado might wish that pages on sports be ranked higher than non-sports pages. Suppose that web pages on sports are "near" one another in the web graph. Then, a random surfer who frequently finds himself on random sports pages is likely (in the course of the random walk) to spend most of his time at sports pages, so that the steady-state distribution of sports pages is boosted.

Suppose our random surfer, endowed with a teleport operation as before, teleports to *a random web page on the topic of sports* instead of teleporting to a uniformly chosen random web page. We will not focus on how we collect all web pages on the topic of sports; in fact, we only need a non-zero subset S of sports-related web pages, so that the teleport operation is feasible. This may be obtained, for instance, from a manually built directory of sports pages such as the open directory project (http://www.dmoz.org/) or that of Yahoo.

Provided the set S of sports-related pages is non-empty, it follows that there is a non-empty set of web pages $Y \supseteq S$ over which the random walk has a steady-state distribution; let us denote this *sports PageRank* distribution by $\vec{\pi}_s$. For web pages not in Y, we set the PageRank values to zero. We call $\vec{\pi}_s$ the *topic-specific PageRank* for sports.

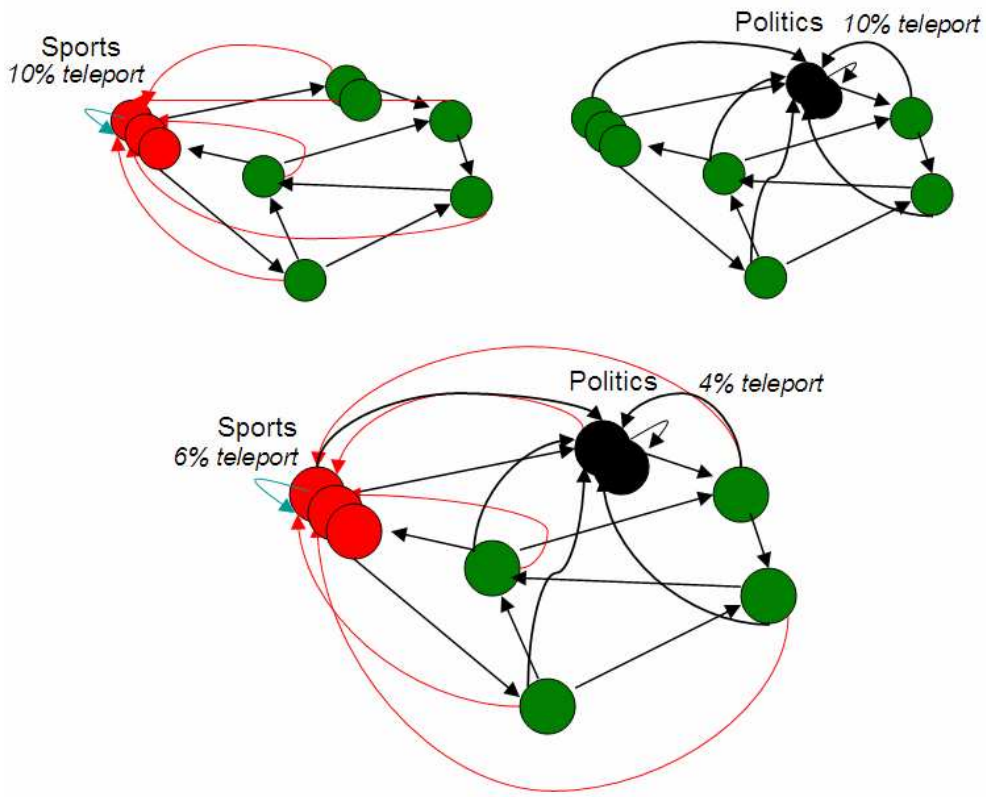
TOPIC-SPECIFIC PAGERANK

We do not demand that teleporting takes the random surfer to a uniformly chosen sports page; the distribution over teleporting targets S could in fact be arbitrary.

In like manner we can envision topic-specific PageRank distributions for each of several topics such as science, religion, politics and so on. Each of these distributions assigns to each web page a PageRank value in the interval $[0, 1)$. For a user interested in only a single topic from among these topics, we may invoke the corresponding PageRank distribution when scoring and ranking search results. This gives us the potential of considering settings in which the search engine knows what topic a user is interested in. This may happen because users either explicitly register their interests, or because the system learns by observing each user's behavior over time.

But what if a user is known to have a mixture of interests from multiple topics? For instance, a user may have an interest mixture (or *profile*) that is 60% sports and 40% politics; can we compute a *personalized PageRank* for this user? At first glance, this appears daunting: how could we possibly compute a different PageRank distribution for each user profile (with, potentially, infinitely many possible profiles)? We can in fact address this provided we assume that an individual's interests can be well-approximated as a linear

PERSONALIZED PAGERANK



► **Figure 21.5** Topic-specific PageRank. In this example we consider a user whose interests are 60% sports and 40% politics. If the teleportation probability is 10%, this user is modeled as teleporting 6% to sports pages and 4% to politics pages.

combination of a small number of topic page distributions. A user with this mixture of interests could teleport as follows: determine first whether to teleport to the set S of known sports pages, or to the set of known politics pages. This choice is made at random, choosing sports pages 60% of the time and politics pages 40% of the time. Once we choose that a particular teleport step is to (say) a random sports page, we choose a web page in S uniformly at random to teleport to. This in turn leads to an ergodic Markov chain with a steady-state distribution that is personalized to this user's preferences over topics (see Exercise 21.16).

While this idea has intuitive appeal, its implementation appears cumbersome: it seems to demand that for each user, we compute a transition prob-

ability matrix and compute its steady-state distribution. We are rescued by the fact that the evolution of the probability distribution over the states of a Markov chain can be viewed as a linear system. In Exercise 21.16 we will show that it is not necessary to compute a PageRank vector for every distinct combination of user interests over topics; the personalized PageRank vector for any user can be expressed as a linear combination of the underlying topic-specific PageRanks. For instance, the personalized PageRank vector for the user whose interests are 60% sports and 40% politics can be computed as

$$0.6\vec{\pi}_s + 0.4\vec{\pi}_p, \tag{21.7}$$

where $\vec{\pi}_s$ and $\vec{\pi}_p$ are the topic-specific PageRank vectors for sports and for politics, respectively.

Exercise 21.5

Write down the transition probability matrix for the example in Figure 21.2.

Exercise 21.6

Consider a web graph with three nodes 1, 2 and 3. The links are as follows: $1 \rightarrow 2, 3 \rightarrow 2, 2 \rightarrow 1, 2 \rightarrow 3$. 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) $\alpha = 1$.

Exercise 21.7

A user of a browser can, in addition to clicking a hyperlink on the page x he is currently browsing, use the *back button* to go back to the page from which he arrived at x. Can such a user of back buttons be modeled as a Markov chain? How would we model repeated invocations of the back button?

Exercise 21.8

Consider a Markov chain with three states A, B and C, and transition probabilities as follows. From state A, the next state is B with probability 1. From B, the next state is either A with probability p_A, or state C with probability $1 - p_A$. From C the next state is A with probability 1. For what values of $p_A \in [0, 1]$ is this Markov chain ergodic?

Exercise 21.9

Show that for any directed graph, the Markov chain induced by a random walk with the teleport operation is ergodic.

Exercise 21.10

Show that the PageRank of every page is at least α/N. What does this imply about the difference in PageRank values (over the various pages) as α becomes close to 1?

Exercise 21.11

For the data in Example 21.1, write a small routine or use a scientific calculator to compute the PageRank values stated in Equation (21.6).

Exercise 21.12

Suppose that the web graph is stored on disk as an adjacency list, in such a way that you may only query for the out-neighbors of pages in the order in which they are stored. You cannot load the graph in main memory but you may do multiple reads over the full graph. Write the algorithm for computing the PageRank in this setting.

Exercise 21.13

Recall the sets S and Y introduced near the beginning of Section 21.2.3. How does the set Y relate to S?

Exercise 21.14

Is the set Y always the set of all web pages? Why or why not?

Exercise 21.15 [⋆ ⋆ ⋆]

Is the sports PageRank of any page in S at least as large as its PageRank?

Exercise 21.16 [⋆ ⋆ ⋆]

Consider a setting where we have two topic-specific PageRank values for each web page: a sports PageRank $\vec{\pi}_s$, and a politics PageRank $\vec{\pi}_p$. Let α be the (common) teleportation probability used in computing both sets of topic-specific PageRanks. For $q \in [0, 1]$, consider a user whose interest profile is divided between a fraction q in sports and a fraction $1 - q$ in politics. Show that the user's personalized PageRank is the steady-state distribution of a random walk in which – on a teleport step – the walk teleports to a sports page with probability q and to a politics page with probability $1 - q$.

Exercise 21.17

Show that the Markov chain corresponding to the walk in Exercise 21.16 is ergodic and hence the user's personalized PageRank can be obtained by computing the steady-state distribution of this Markov chain.

Exercise 21.18

Show that in the steady-state distribution of Exercise 21.17, the steady-state probability for any web page i equals $q\pi_s(i) + (1 - q)\pi_p(i)$.

21.3 Hubs and Authorities

HUB SCORE
AUTHORITY SCORE

We now develop a scheme in which, given a query, every web page is assigned *two* scores. One is called its *hub score* and the other its *authority score*. For any query, we compute two ranked lists of results rather than one. The ranking of one list is induced by the hub scores and that of the other by the authority scores.

This approach stems from a particular insight into the creation of web pages, that there are two primary kinds of web pages useful as results for *broad-topic searches*. By a broad topic search we mean an informational query such as "I wish to learn about leukemia". There are authoritative sources of information on the topic; in this case, the National Cancer Institute's page on

leukemia would be such a page. We will call such pages *authorities*; in the computation we are about to describe, they are the pages that will emerge with high authority scores.

On the other hand, there are many pages on the Web that are hand-compiled lists of links to authoritative web pages on a specific topic. These *hub* pages are not in themselves authoritative sources of topic-specific information, but rather compilations that someone with an interest in the topic has spent time putting together. The approach we will take, then, is to use these hub pages to discover the authority pages. In the computation we now develop, these hub pages are the pages that will emerge with high hub scores.

A good hub page is one that points to many good authorities; a good authority page is one that is pointed to by many good hub pages. We thus appear to have a circular definition of hubs and authorities; we will turn this into an iterative computation. Suppose that we have a subset of the web containing good hub and authority pages, together with the hyperlinks amongst them. We will iteratively compute a hub score and an authority score for every web page in this subset, deferring the discussion of how we pick this subset until Section 21.3.1.

For a web page v in our subset of the web, we use $h(v)$ to denote its hub score and $a(v)$ its authority score. Initially, we set $h(v) = a(v) = 1$ for all nodes v. We also denote by $v \mapsto y$ the existence of a hyperlink from v to y. The core of the iterative algorithm is a pair of updates to the hub and authority scores of all pages given by Equation 21.8, which capture the intuitive notions that good hubs point to good authorities and that good authorities are pointed to by good hubs.

$$\begin{aligned} h(v) &\leftarrow \sum_{v \mapsto y} a(y) \\ a(v) &\leftarrow \sum_{y \mapsto v} h(y). \end{aligned} \tag{21.8}$$

Thus, the first line of Equation (21.8) sets the hub score of page v to the sum of the authority scores of the pages it links to. In other words, if v links to pages with high authority scores, its hub score increases. The second line plays the reverse role; if page v is linked to by good hubs, its authority score increases.

What happens as we perform these updates iteratively, recomputing hub scores, then new authority scores based on the recomputed hub scores, and so on? Let us recast the equations Equation (21.8) into matrix-vector form. Let $\vec{h}$ and $\vec{a}$ denote the vectors of all hub and all authority scores respectively, for the pages in our subset of the web graph. Let A denote the adjacency matrix of the subset of the web graph that we are dealing with: A is a square matrix with one row and one column for each page in the subset. The entry

A_{ij} is 1 if there is a hyperlink from page i to page j, and 0 otherwise. Then, we may write Equation (21.8)

$$\begin{aligned} \vec{h} &\leftarrow A\vec{a} \\ \vec{a} &\leftarrow A^T\vec{h}, \end{aligned} \tag{21.9}$$

where A^T denotes the transpose of the matrix A. Now the right hand side of each line of Equation (21.9) is a vector that is the left hand side of the other line of Equation (21.9). Substituting these into one another, we may rewrite Equation (21.9) as

$$\begin{aligned} \vec{h} &\leftarrow AA^T\vec{h} \\ \vec{a} &\leftarrow A^TA\vec{a}. \end{aligned} \tag{21.10}$$

Now, Equation (21.10) bears an uncanny resemblance to a pair of eigenvector equations (Section 18.1); indeed, if we replace the $\leftarrow$ symbols by $=$ symbols and introduce the (unknown) eigenvalue, the first line of Equation (21.10) becomes the equation for the eigenvectors of AA^T, while the second becomes the equation for the eigenvectors of A^TA:

$$\begin{aligned} \vec{h} &= (1/\lambda_h)AA^T\vec{h} \\ \vec{a} &= (1/\lambda_a)A^TA\vec{a}. \end{aligned} \tag{21.11}$$

Here we have used λ_h to denote the eigenvalue of AA^T and λ_a to denote the eigenvalue of A^TA.

This leads to some key consequences:

1. The iterative updates in Equation (21.8) (or equivalently, Equation (21.9)), if scaled by the appropriate eigenvalues, are equivalent to the power iteration method for computing the eigenvectors of AA^T and A^TA. Provided that the principal eigenvalue of AA^T is unique, the iteratively computed entries of $\vec{h}$ and $\vec{a}$ settle into unique steady-state values determined by the entries of A and hence the link structure of the graph.

2. In computing these eigenvector entries, we are not restricted to using the power iteration method; indeed, we could use any fast method for computing the principal eigenvector of a stochastic matrix.

The resulting computation thus takes the following form:

1. Assemble the target subset of web pages, form the graph induced by their hyperlinks and compute AA^T and A^TA.

2. Compute the principal eigenvectors of AA^T and A^TA to form the vector of hub scores $\vec{h}$ and authority scores $\vec{a}$.

3. Output the top-scoring hubs and the top-scoring authorities.

HITS This method of link analysis is known as *HITS*, which is an acronym for *Hyperlink-Induced Topic Search*.

Example 21.2: Assuming the query jaguar and double-weighting of links whose anchors contain the query word, the matrix A for Figure 21.4 is as follows:

$$\begin{array}{ccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 2 & 1 & 0 & 1 \end{array}$$

The hub and authority vectors are:

$$\vec{h} = (0.03 \quad 0.04 \quad 0.33 \quad 0.18 \quad 0.04 \quad 0.04 \quad 0.35)$$

$$\vec{a} = (0.10 \quad 0.01 \quad 0.12 \quad 0.47 \quad 0.16 \quad 0.01 \quad 0.13)$$

Here, q_3 is the main authority – two hubs (q_2 and q_6) are pointing to it via highly weighted jaguar links.

Since the iterative updates captured the intuition of good hubs and good authorities, the high-scoring pages we output would give us good hubs and authorities from the target subset of web pages. In Section 21.3.1 we describe the remaining detail: how do we gather a target subset of web pages around a topic such as leukemia?

21.3.1 Choosing the subset of the Web

In assembling a subset of web pages around a topic such as leukemia, we must cope with the fact that good authority pages may not contain the specific query term leukemia. This is especially true, as we noted in Section 21.1.1, when an authority page uses its web presence to project a certain marketing image. For instance, many pages on the IBM website are authoritative sources of information on computer hardware, even though these pages may not contain the term computer or hardware. However, a hub compiling computer hardware resources is likely to use these terms and also link to the relevant pages on the IBM website.

Building on these observations, the following procedure has been suggested for compiling the subset of the Web for which to compute hub and authority scores.

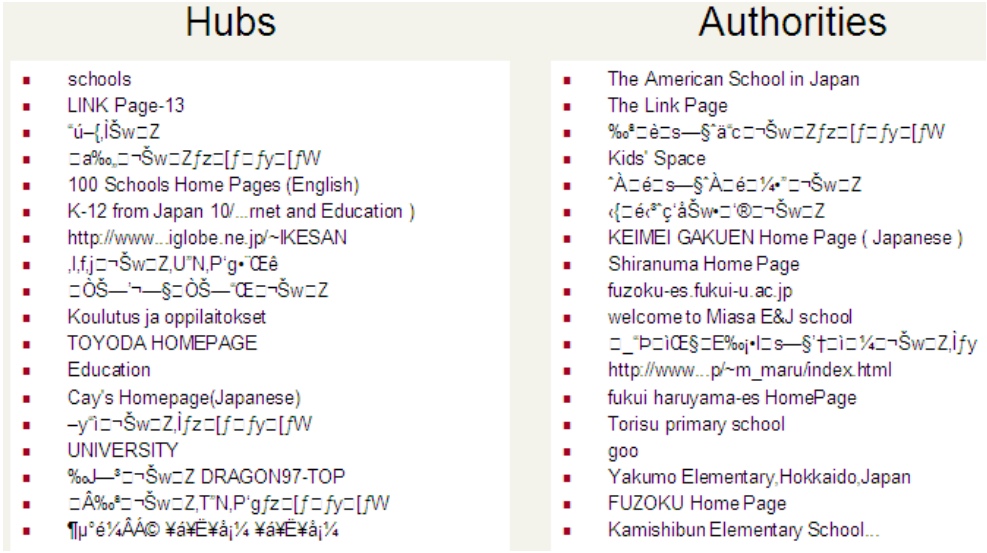
1. Given a query (say leukemia), use a text index to get all pages containing leukemia. Call this the *root set* of pages.
2. Build the *base set* of pages, to include the root set as well as any page that either links to a page in the root set, or is linked to by a page in the root set.

We then use the base set for computing hub and authority scores. The base set is constructed in this manner for three reasons:

1. A good authority page may not contain the query text (such as computer hardware).
2. If the text query manages to capture a good hub page v_h in the root set, then the inclusion of all pages linked to by any page in the root set will capture all the good authorities linked to by v_h in the base set.
3. Conversely, if the text query manages to capture a good authority page v_a in the root set, then the inclusion of pages which point to v_a will bring other good hubs into the base set. In other words, the "expansion" of the root set into the base set enriches the common pool of good hubs and authorities.

Running HITS across a variety of queries reveals some interesting insights about link analysis. Frequently, the documents that emerge as top hubs and authorities include languages other than the language of the query. These pages were presumably drawn into the base set, following the assembly of the root set. Thus, some elements of *cross-language retrieval* (where a query in one language retrieves documents in another) are evident here; interestingly, this cross-language effect resulted purely from link analysis, with no linguistic translation taking place.

We conclude this section with some notes on implementing this algorithm. The root set consists of all pages matching the text query; in fact, implementations (see the references in Section 21.4) suggest that it suffices to use 200 or so web pages for the root set, rather than all pages matching the text query. Any algorithm for computing eigenvectors may be used for computing the hub/authority score vector. In fact, we need not compute the exact values of these scores; it suffices to know the relative values of the scores so that we may identify the top hubs and authorities. To this end, it is possible that a small number of iterations of the power iteration method yields the relative ordering of the top hubs and authorities. Experiments have suggested that in practice, about five iterations of Equation (21.8) yield fairly good results. Moreover, since the link structure of the web graph is fairly sparse (the average web page links to about ten others), we do not perform these as matrix-vector products but rather as additive updates as in Equation (21.8).



► **Figure 21.6** A sample run of HITS on the query japan elementary schools.

Figure 21.6 shows the results of running HITS on the query japan elementary schools. The figure shows the top hubs and authorities; each row lists the `title` tag from the corresponding HTML page. Because the resulting string is not necessarily in Latin characters, the resulting print is (in many cases) a string of gibberish. Each of these corresponds to a web page that does not use Latin characters, in this case very likely pages in Japanese. There also appear to be pages in other non-English languages, which seems surprising given that the query string is in English. In fact, this result is emblematic of the functioning of HITS – following the assembly of the root set, the (English) query string is ignored. The base set is likely to contain pages in other languages, for instance if an English-language hub page links to the Japanese-language home pages of Japanese elementary schools. Because the subsequent computation of the top hubs and authorities is entirely link-based, some of these non-English pages will appear among the top hubs and authorities.

? **Exercise 21.19**

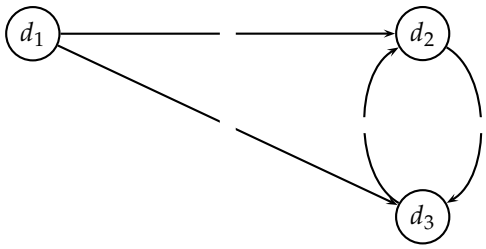
If all the hub and authority scores are initialized to 1, what is the hub/authority score of a node after one iteration?

Exercise 21.20

How would you interpret the entries of the matrices AA^T and A^TA? What is the connection to the co-occurrence matrix CC^T in Chapter 18?

Exercise 21.21

What are the principal eigenvalues of AA^T and A^TA?



► **Figure 21.7** Web graph for Exercise 21.22.

Exercise 21.22

For the web graph in Figure 21.7, compute PageRank, hub and authority scores for each of the three pages. Also give the relative ordering of the 3 nodes for each of these scores, indicating any ties.

PageRank: Assume that at each step of the PageRank random walk, we teleport to a random page with probability 0.1, with a uniform distribution over which particular page we teleport to.

Hubs/Authorities: Normalize the hub (authority) scores so that the maximum hub (authority) score is 1.

Hint 1: Using symmetries to simplify and solving with linear equations might be easier than using iterative methods.

Hint 2: Provide the relative ordering (indicating any ties) of the three nodes for each of the three scoring measures.

21.4 References and further reading

Garfield (1955) is seminal in the science of citation analysis. This was built on by Pinski and Narin (1976) to develop a *journal influence weight*, whose definition is remarkably similar to that of the PageRank measure.

The use of anchor text as an aid to searching and ranking stems from the work of McBryan (1994). Extended anchor-text was implicit in his work, with systematic experiments reported in Chakrabarti et al. (1998).

Kemeny and Snell (1976) is a classic text on Markov chains. The PageRank measure was developed in Brin and Page (1998) and in Page et al. (1998).

A number of methods for the fast computation of PageRank values are surveyed in Berkhin (2005) and in Langville and Meyer (2006); the former also details how the PageRank eigenvector solution may be viewed as solving a linear system, leading to one way of solving Exercise 21.16. The effect of the teleport probability α has been studied by Baeza-Yates et al. (2005) and by Boldi et al. (2005). Topic-specific PageRank and variants were developed in Haveliwala (2002), Haveliwala (2003) and in Jeh and Widom (2003). Berkhin (2006a) develops an alternate view of topic-specific PageRank.

Ng et al. (2001b) suggests that the PageRank score assignment is more robust than HITS in the sense that scores are less sensitive to small changes in graph topology. However, it has also been noted that the teleport operation contributes significantly to PageRank's robustness in this sense. Both PageRank and HITS can be "spammed" by the orchestrated insertion of links into the web graph; indeed, the Web is known to have such *link farms* that collude to increase the score assigned to certain pages by various link analysis algorithms.

LINK FARMS

The HITS algorithm is due to Kleinberg (1999). Chakrabarti et al. (1998) developed variants that weighted links in the iterative computation based on the presence of query terms in the pages being linked and compared these to results from several web search engines. Bharat and Henzinger (1998) further developed these and other heuristics, showing that certain combinations outperformed the basic HITS algorithm. Borodin et al. (2001) provides a systematic study of several variants of the HITS algorithm. Ng et al. (2001b) introduces a notion of *stability* for link analysis, arguing that small changes to link topology should not lead to significant changes in the ranked list of results for a query. Numerous other variants of HITS have been developed by a number of authors, the best know of which is perhaps SALSA (Lempel and Moran 2000).