A SURVEY OF EIGENVECTOR METHODS FOR WEB INFORMATION RETRIEVAL*

AMY N. LANGVILLE† AND CARL D. MEYER‡

Abstract. Web information retrieval is significantly more challenging than traditional well-controlled, small document collection information retrieval. One main difference between traditional information retrieval and Web information retrieval is the Web's hyperlink structure. This structure has been exploited by several of today's leading Web search engines, particularly Google and Teoma. In this survey paper, we focus on Web information retrieval methods that use eigenvector computations, presenting the three popular methods of HITS, PageRank, and SALSA.

Key words. Eigenvector, Markov chain, information retrieval, HITS, PageRank, SALSA

AMS subject classifications. 65F15, 65F10, 68P20, 15A18, 15A51, 65C40

1. Introduction. The Information Age ushers in unparalleled possibilities and convenience to information retrieval (IR) users along with unparalleled challenges for IR researchers and developers. For example, many prognosticators predict no halt to the information explosion in the near future. Yet, while users theoretically have a vast amount of information at their fingertips, the accuracy of some IR systems leaves some users frustrated and the IR researchers forever working to improve their systems. We begin this survey of modern linear algebra-based IR methods from a historical perspective.

There are various IR methods in use today, ranging from Boolean to probabilistic to vector space models. Of these, vector space models incorporate the most linear algebra. The prevailing vector space method for small document collections is Latent Semantic Indexing (LSI) [24]. The 1999 *SIAM Review* article by Berry et al. gives an excellent introduction and survey of vector space models, especially LSI [4]. LSI uses the singular value decomposition (SVD) of the term-by-document matrix to capture latent semantic associations. LSI became famous for its ability to effectively handle generally troublesome query terms involving polysems and synonyms. The SVD enables LSI methods to inherently (and almost magically) cluster documents and terms into concepts. For example, synonyms *car*, *automobile* and *vehicle* may be grouped into the same cluster, while the polysem *bank* may be divided according to its various meanings (financial institution, steep slope, billiards shot, etc.). However, the discriminating power of LSI, derived from the SVD, is the reason for its limitation to smaller document collections. The computation and storage of the SVD of the term-by-document matrix is costly. Consider that this term-by-document matrix has as many columns as there are documents in a particular collection. A huge collection like the WWW's webpages is dramatically out of LSI's scope.

It is not just the enormity of the Web that leaves traditional well-tested and successful methods, like LSI behind, it is the Web's other peculiarities that also make it an especially challenging document collection to analyze. The documents on the Web are not subjected to an editorial review process. Therefore, the Web contains

*This work was supported in part by the National Science Foundation under NSF grants CCR-0318575, CCR-ITR-0113121, and DMS-0209695.

†Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205, USA, anlangvi@eos.ncsu.edu.

‡Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205, USA, meyer@ncsu.edu.

redundant documents, broken links, and some very poor quality documents. The Web also is subject to frequent updates as pages are modified and/or added to or deleted from the Web on an almost continual basis. The Web's volatility leaves IR researchers with two choices: either incorporate updates and downdates on a frequent, regular basis or make updates infrequently, trading accuracy for simplicity. The Web is also an interesting document collection in that some of its users, aiming to exploit the mercantile potential of the Web, intentionally try to deceive IR systems [42]. For example, there are papers instructing webpage authors on the methods for "increasing one's ranking" on various IR systems [54]. Ideally, an IR system should be impervious to such spamming. The tendencies of Web users also present additional challenges to Web IR systems. Web users generally input very short queries, rarely make use of feedback to revise the search, seldom use any advanced features of the system and almost always view only the top 10-20 retrieved documents [5, 31]. Such user tendencies put high priority on the speed and accuracy of the IR system. The final feature, and most important for this paper, is the Web's unique hyperlink structure. This inherent structure provides extra, and as will become clear later, very valuable information that can be exploited to improve IR methods.

This hyperlink structure is exploited by three of the most frequently cited Web IR methods: HITS (Hypertext Induced Topic Search) [36], PageRank [14, 15] and SALSA [41]. HITS was developed in 1997 by Jon Kleinberg. Soon after Sergey Brin and Larry Page developed their now famous PageRank method. SALSA was developed in 2000 in reaction to the pros and cons of HITS and PageRank.

This paper is organized as follows. Sections 2-4 cover HITS and PageRank and their connections, followed by SALSA in section 5. Other Web IR methods are briefly overviewed in section 6 and section 7 contains predictions for the future of Web IR.

2. Two Theses for exploiting the Hyperlink Structure of the Web. Each page/document on the Web is represented as a node in a very large graph. The directed arcs connecting these nodes represent the hyperlinks between the documents. The graph of a sample Web is depicted in Figure 2.1.

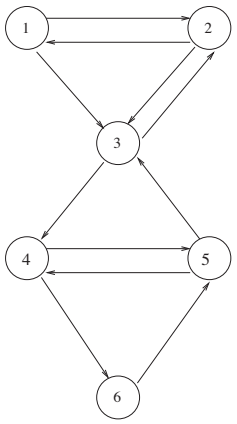

FIG. 2.1. *Hyperlink Structure of a 6-node sample Web*

The HITS IR method defines *authorities* and *hubs*. An authority is a document with several inlinks, while a hub has several outlinks. See Figure 2.2.

The HITS thesis is that good hubs point to good authorities and good authorities are pointed to by good hubs. HITS assigns both a hub score and authority score to

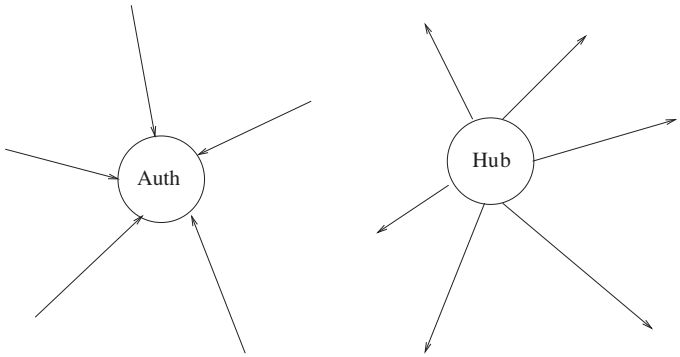

FIG. 2.2. *An authority node and a hub node*

each webpage. For example, the higher the authority score of a particular page, the more authoritative that document is.

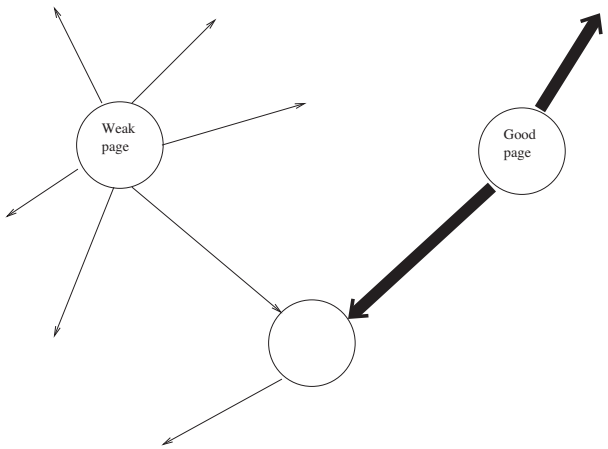

FIG. 2.3. *The PageRank thesis (The bold lines show the extra weight given to links from important pages.)*

On the other hand, PageRank uses the hyperlink structure of the Web to view inlinks into a page as a recommendation of that page from the author of the inlinking page. However, inlinks from good pages (highly revered authors) should carry much more weight than inlinks from marginal pages. Each webpage can be assigned an appropriate PageRank score, which measures the importance of that page. Figure 2.3 depicts the PageRank thesis. The bold lines show the extra weight given to links from important pages.

These are two very similar and related, yet distinct, ideas for ranking the usefulness of webpages. In the next few sections, we analyze these two IR methods in turn.

3. HITS. We repeat the HITS thesis: good authorities are pointed to by good hubs and good hubs point to good authorities. Page i has both an authority score x_i

and a hub score y_i. Let E be the set of all directed edges in the Web graph and let e_{ij} represent the directed edge from node i to node j. Given that each page has somehow been assigned an initial authority score $x_i^{(0)}$ and hub score $y_i^{(0)}$, HITS successively refines these scores by computing

$$x_i^{(k)} = \sum_{j:e_{ji}\in E} y_j^{(k-1)} \quad \text{and} \quad y_i^{(k)} = \sum_{j:e_{ij}\in E} x_j^{(k)} \quad \text{for } k = 1, 2, 3, \ldots. \tag{3.1}$$

These equations can be written in matrix form with the help of the adjacency matrix $\mathbf{L}$ of the directed Web graph.

$$\mathbf{L}_{ij} = \begin{cases} 1, & \text{if there exists an edge from node } i \text{ to node } j, \\ 0, & \text{otherwise.} \end{cases}$$

For example, consider the small graph in Figure 3.1 with corresponding adjacency

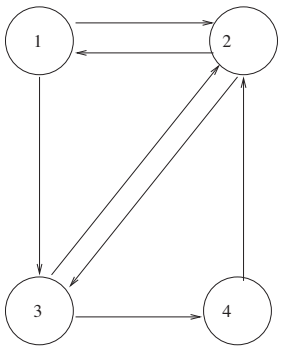

FIG. 3.1. *Node-link graph for small 4-node Web*

matrix $\mathbf{L}$.

$$\mathbf{L} = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}.$$

In matrix notation, the equations in (3.1) assume the form

$$\mathbf{x}^{(k)} = \mathbf{L}^T\mathbf{y}^{(k-1)} \quad \text{and} \quad \mathbf{y}^{(k)} = \mathbf{L}\mathbf{x}^{(k)}.$$

This leads to the following iterative algorithm for computing the ultimate authority and hub scores $\mathbf{x}$ and $\mathbf{y}$.

1. Initialize: $\mathbf{y}^{(0)} = \mathbf{e}$, where $\mathbf{e}$ is a column vector of all ones. Other positive starting vectors may be used. (See section 3.2.)
2. Until convergence, do

$$\begin{aligned} \mathbf{x}^{(k)} &= \mathbf{L}^T\mathbf{y}^{(k-1)} \\ \mathbf{y}^{(k)} &= \mathbf{L}\mathbf{x}^{(k)} \\ k &= k+1 \\ &\text{Normalize } \mathbf{x}^{(k)} \text{ and } \mathbf{y}^{(k)} \text{ (see section 3.2).} \end{aligned}$$

Note that in step 2 of this algorithm, the two equations

$$\mathbf{x}^{(k)} = \mathbf{L}^T\mathbf{y}^{(k-1)}$$
$$\mathbf{y}^{(k)} = \mathbf{L}\mathbf{x}^{(k)}$$

can be simplified by substitution to

$$\mathbf{x}^{(k)} = \mathbf{L}^T\mathbf{L}\mathbf{x}^{(k-1)}$$
$$\mathbf{y}^{(k)} = \mathbf{L}\mathbf{L}^T\mathbf{y}^{(k-1)}.$$

These two new equations define the iterative power method for computing the dominant eigenvector for the matrices $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$. Since the matrix $\mathbf{L}^T\mathbf{L}$ determines the authority scores, it is called the *authority matrix* and $\mathbf{L}\mathbf{L}^T$ is known as the *hub matrix*. $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$ are symmetric positive semidefinite matrices. Computing the authority vector $\mathbf{x}$ and the hub vector $\mathbf{y}$ can be viewed as finding dominant right-hand eigenvectors of $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$, respectively.

3.1. HITS Implementation. The implementation of HITS involves two main steps. First, a *neighborhood graph* N related to the query terms is built. Second, the authority and hub scores for each document in N are computed and two ranked lists of the most authoritative documents and most "hubby" documents are presented to the IR user. Since the second step was described in the previous section, we focus on the first step. All documents containing references to the query terms are put into the neighborhood graph N. There are various ways to determine these documents. One simple method consults the inverted term-document file. This file might look like:

- aardvark: term 1 - doc 3, doc 117, doc 3961
- ⋮
- aztec: term 10 - doc 15, doc 3, doc 101, doc 19, doc 1199, doc 673
- baby: term 11 - doc 56, doc 94, doc 31, doc 3
- ⋮
- zymurgy: term m - doc 223

For each term, the documents mentioning that term are stored in list form. Thus, a query on terms 10 and 11 would pull documents 15, 3, 101, 19, 1199, 673, 56, 94, and 31 into N. Next, the graph around the subset of nodes in N is expanded by adding nodes that either point to nodes in N or are pointed to by nodes in N. This expansion allows some latent semantic associations to be made. That is, for the query term *car*, with the expansion about documents containing *car*, some documents containing *automobile* may now be added to N, hopefully resolving the problem of synonyms. However, the set N can become very large due to the expansion process; a document containing the query terms may possess a huge indegree or outdegree. Thus, in practice, the maximum number of inlinking nodes and outlinking nodes to add for a particular node in N is fixed, at say 100. For example, only the first 100 outlinking nodes of a document containing a query term are added to N. The process of building the neighborhood graph is strongly related to building level sets in information filtering, which reduces a sparse matrix to a much smaller more query-relevant matrix [63].

Once the set N is built, the adjacency matrix $\mathbf{L}$ corresponding to the nodes in N is formed. The order of $\mathbf{L}$ is much smaller than the total number of nodes/documents on the Web. Therefore, computing authority and hub scores using the dominant

eigenvectors of $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$ incurs a small cost, small in comparison to computing authority and hub scores when all documents on the Web are placed in N.

An additional cost reduction exists. Only one document eigenvector needs to be computed, that of either $\mathbf{L}^T\mathbf{L}$ or $\mathbf{L}\mathbf{L}^T$, but not both. For example, the authority vector $\mathbf{x}$ can be obtained by computing the dominant eigenvector of $\mathbf{L}^T\mathbf{L}$, then the hub vector $\mathbf{y}$ can be obtained from the equation, $\mathbf{y} = \mathbf{L}\mathbf{x}$. A similar statement applies if the hub vector is computed first from the eigenvector problem.

3.2. HITS Convergence. The iterative algorithm for computing HITS vectors is actually the power method applied to $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$. For a diagonalizable matrix $\mathbf{B}_{n\times n}$ whose distinct eigenvalues are $\{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ such that $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \cdots \geq |\lambda_k|$, the power method takes an initial vector $\mathbf{x}^{(0)}$ and iteratively computes

$$\mathbf{x}^{(k)} = \mathbf{B}\,\mathbf{x}^{(k-1)}, \quad \mathbf{x}^{(k)} \longleftarrow \frac{\mathbf{x}^{(k)}}{m(\mathbf{x}^{(k)})},$$

where $m(\mathbf{x}^{(k)})$ is a normalizing scalar derived from $\mathbf{x}^{(k)}$. For example, it is common to take $m(\mathbf{x}^{(k)})$ to be the (signed) component of maximal magnitude (use the first if there are more than one), in which case $m(\mathbf{x}^{(k)})$ converges to the dominant eigenvalue λ_1, and $\mathbf{x}^{(k)}$ converges to an associated normalized eigenvector [45]. If only a dominant eigenvector is needed (but not λ_1), then a normalization such as $m(\mathbf{x}^{(k)}) = \|\mathbf{x}^{(k)}\|$ can be used. (If $\lambda_1 < 0$, then $m(\mathbf{x}^{(k)}) = \|\mathbf{x}^{(k)}\|$ can't converge to λ_1, but $\mathbf{x}^{(k)}$ still converges to a normalized eigenvector associated with λ_1.)

The matrices $\mathbf{L}^T\mathbf{L}$ and $\mathbf{L}\mathbf{L}^T$ are symmetric, positive semidefinite, and nonnegative, so their distinct eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ are necessarily real and nonnegative with $\lambda_1 > \lambda_2 > \cdots > \lambda_k \geq 0$. In other words, it is not possible to have multiple eigenvalues on the spectral circle. Consequently, the HITS specialization of the power method avoids most problematic convergence issues—HITS with normalization always converges. However, there can be a problem with the uniqueness of the limiting authority and hub vectors. While $\lambda_1 > \lambda_2$, the structure of $\mathbf{L}$ might allow λ_1 to be a repeated root of the characteristic polynomial, in which case the associated eigenspace would be multi-dimensional. This means that the different limiting authority (and hub) vectors can be produced by different choices of the initial vector.

A simple example from [27] demonstrates this problem. In this example,

$$\mathbf{L} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} \quad \text{and} \quad \mathbf{L}^T\mathbf{L} = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

The authority matrix $\mathbf{L}^T\mathbf{L}$ (and the hub matrix $\mathbf{L}\mathbf{L}^T$) has two distinct eigenvalues, $\lambda_1 = 2$ and $\lambda_2 = 0$, that are each repeated twice. For $\mathbf{x}^{(0)} = (\,1/4 \quad 1/4 \quad 1/4 \quad 1/4\,)^T$, the power method (with normalization by the 1-norm) converges to $\mathbf{x}^{(\infty)} = (\,1/3 \quad 1/3 \quad 1/3 \quad 0\,)^T$. Yet for $\mathbf{x}^{(0)} = (\,1/4 \quad 1/8 \quad 1/8 \quad 1/2\,)^T$, the power method converges to $\mathbf{x}^{(\infty)} = (\,1/2 \quad 1/4 \quad 1/4 \quad 0\,)^T$. At the heart of this uniqueness problem is the issue of reducibility.

A square matrix $\mathbf{B}$ is said to be reducible if there exists a permutation matrix $\mathbf{Q}$ such that

$$\mathbf{Q}^T\mathbf{B}\mathbf{Q} = \begin{pmatrix} \mathbf{X} & \mathbf{Y} \\ \mathbf{0} & \mathbf{Z} \end{pmatrix}, \quad \text{where } \mathbf{X} \text{ and } \mathbf{Z} \text{ are both square.}$$

Otherwise, the matrix is irreducible. The Perron-Frobenius theorem [45] ensures that an irreducible nonnegative matrix possesses a unique normalized positive dominant eigenvector, called the Perron vector. Consequently, it's the reducibility of $\mathbf{L}^T\mathbf{L}$ that causes the HITS algorithm to converge to nonunique solutions. PageRank encounters the same uniqueness problem, but the Google founders suggest a way to cheat and alter the matrix, forcing irreducibility and hence guaranteeing existence and uniqueness of the ranking vector—see section 4.3. A modification similar to the Google trick can also be applied to HITS.

One final caveat regarding the power method concerns the starting vector $\mathbf{x}^{(0)}$. In general, regardless of whether the dominant eigenvalue λ_1 of the iteration matrix $\mathbf{B}$ is simple or repeated, convergence to a nonzero vector depends on the initial vector $\mathbf{x}^{(0)}$ not being in the range of $(\mathbf{B} - \lambda_1\mathbf{I})$. If $\mathbf{x}^{(0)}$ is randomly generated, then almost certainly this condition will hold, so in practice this is rarely an issue [32]. Miller et al. [27] have developed a modification to HITS, called exponentiated input to HITS, that remedies some of the convergence issues mentioned above.

3.3. HITS Example. We present a very small example to demonstrate the implementation of the HITS algorithm. First, a user presents query terms to the HITS IR system. There are several schemes that can be used to determine which nodes "contain" query terms. For instance, one could take nodes using at least one query term. Or to create a smaller sparse graph, one could take only nodes using all query terms. For our example, suppose the subset of nodes containing the query terms is $\{1, 6\}$. Thus, documents 1 and 6 contain the query terms. Next, we build the neighborhood graph about nodes 1 and 6. Suppose this produces the following graph N, shown in Figure 3.2. From N, the adjacency matrix $\mathbf{L}$ is formed.

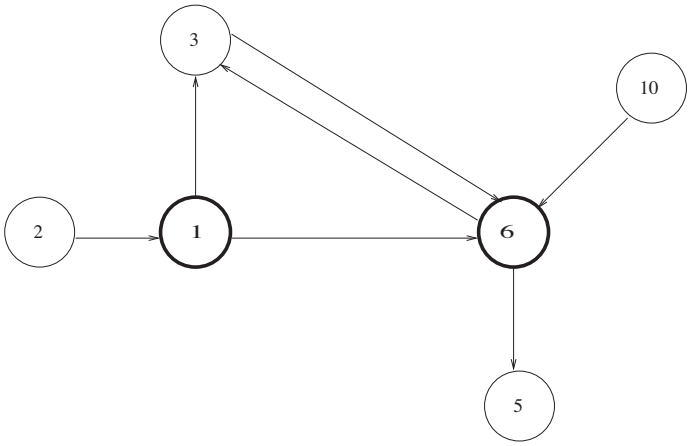

FIG. 3.2. *N: Neighborhood graph about documents 1 and 6*

$$\mathbf{L} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}.$$

The authority and hub matrices are, respectively:

$$\mathbf{L}^T\mathbf{L} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}, \quad \mathbf{L}\mathbf{L}^T = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 2 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array}.$$

The normalized principal eigenvectors with the authority scores $\mathbf{x}$ and hub scores $\mathbf{y}$ are:

$$\mathbf{x}^T = (\,0 \quad 0 \quad .3660 \quad .1340 \quad .5 \quad 0\,),$$
$$\mathbf{y}^T = (\,.3660 \quad 0 \quad .2113 \quad 0 \quad .2113 \quad .2113\,).$$

For the much larger matrices that occur in practice, the existence of identical values in the dominant eigenvectors is not likely. Nevertheless, ties may occur and can be broken by any tie-breaking strategy. Using a "first come, first serve" tie-breaking strategy, the authority and hub scores are sorted in decreasing order and the document id numbers are presented.

$$\text{Auth. ranking} = (\,6 \quad 3 \quad 5 \quad 1 \quad 2 \quad 10\,),$$
$$\text{Hub ranking} = (\,1 \quad 3 \quad 6 \quad 10 \quad 2 \quad 5\,).$$

This means that document 6 is the most authoritative document for the query while document 1 is the best hub for this query.

3.4. Strengths and Weaknesses of HITS. One advantage of the HITS algorithm for IR is its dual rankings. HITS presents two ranked lists to the user: one with the most authoritative documents related to the query and the other with the most "hubby" documents. A user may be more interested in one ranked list than another depending on the application. HITS also casts the overall Web IR problem as a small problem, finding the dominant eigenvectors of small matrices. The size of these matrices is very small relative to the total number of documents on the Web.

However, there are some clear disadvantages to the HITS IR algorithm. Most troublesome is HITS' query-dependence. At query time, a neighborhood graph must be built and at least one matrix eigenvector problem solved. This must be done for each query. HITS' susceptibility to spamming creates a second strong disadvantage. By adding links to and from his webpage, a user can slightly influence the authority and hub scores of his page. A slight change in these scores might be enough to move that webpage a few notches up the ranked lists returned to the IR user. With banner advertisements and financial opportunities, webpage owners have clear incentive to improve their standings in the ranked lists. This is especially important as IR users generally view only the top 20 pages returned in a ranked list. From the perspective of a webpage owner, adding outlinks from a page is much easier than adding inlinks to that page. So influencing one's hub score is not difficult. Yet since hub scores and authority scores share an interdependence and are computed interdependently, an authority score will increase as a hub score increases. Also, since the neighborhood graph is small in comparison to the entire Web, local changes to the link structure will appear more drastic. However, Henzinger and Bharat have proposed a modification to HITS that mitigates the problem of spam by using an L1 normalization step [6].

A final disadvantage of HITS is the problem of topic drift. In building the neighborhood graph N for a query it is possible that a very authoritative yet off-topic document be linked to a document containing the query terms. This very authoritative document can carry so much weight that it and its neighboring documents dominate the relevant ranked list returned to the user, skewing the results towards off-topic documents. Henzinger and Bharat suggest a solution to the problem of topic drift, weighting the authority and hub scores of the nodes in N by a measure of relevancy to the query [6]. In fact, to measure relevance of a node in N (i.e., a document) to the query, they use the same cosine similarity measure that is often used by vector space methods such as LSI [4, 24].

3.5. HITS' relationship to Bibliometrics. The HITS IR algorithm has strong connections to the bibliometrics research. Bibliometrics is the study of written documents and their citation structure. Such research uses the citation structure of a body of documents to produce numerical measures of the importance and impact of papers. Ding et al. have noted the underlying connection between HITS and two common bibliometrics concepts, *co-citation* and *co-reference* [23, 22].

In bibliometrics, co-citation occurs when two documents are both cited by the same third document. Co-reference occurs when two documents both refer to the same third document. In IR, co-citation occurs when two nodes share a common inlinking node, while co-reference means two nodes share a common outlinking node. Ding et al. have shown that the authority matrix $\mathbf{L}^T\mathbf{L}$ of HITS has a direct relationship to the concept of co-citation, while the hub matrix $\mathbf{L}\mathbf{L}^T$ is related to co-reference [22, 23]. Suppose the small hyperlink graph of Figure 3.1 is studied again. The adjacency matrix is

$$\mathbf{L} = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & d_3 & d_4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}.$$

Then the authority and hub matrices are:

$$\mathbf{L}^T\mathbf{L} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 3 & 1 & 1 \\ 1 & 1 & 2 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} = \mathbf{D}_{in} + \mathbf{C}_{cit},$$

$$\mathbf{L}\mathbf{L}^T = \begin{pmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 0 & 0 \\ 1 & 0 & 2 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} = \mathbf{D}_{out} + \mathbf{C}_{ref}.$$

Ding et al. [22, 23] show that $\mathbf{L}^T\mathbf{L} = \mathbf{D}_{in} + \mathbf{C}_{cit}$, where $\mathbf{D}_{in}$ is a diagonal matrix with the indegree of each node along the diagonal and $\mathbf{C}_{cit}$ is the co-citation matrix. For example, the $(3,3)$-element of $\mathbf{L}^T\mathbf{L}$ means that node 3 has an indegree of 2. The $(1,3)$-element of $\mathbf{L}^T\mathbf{L}$ means that node 1 and node 3 share only 1 common inlinking node, node 2 as is apparent from Figure 3.1. The $(4,3)$-element of $\mathbf{L}^T\mathbf{L}$ implies that nodes 3 and 4 do not share a common inlinking node, again, as is apparent from Figure 3.1. Similarly, the hub matrix is actually $\mathbf{D}_{out} + \mathbf{C}_{ref}$, where $\mathbf{D}_{out}$ is the diagonal matrix of outdegrees and $\mathbf{C}_{ref}$ is the co-reference matrix. The $(1,2)$-element of $\mathbf{L}\mathbf{L}^T$

means that nodes 1 and 2 share a common outlink node, node 3. The $(4,2)$-element implies that nodes 4 and 2 do not share a common outlink node. Ding et al. use these relationships between authority and co-citation and hubs and co-reference to claim that simple inlink ranking provides a decent approximation to the HITS authority score and simple outlink ranking provides a decent approximation to hub ranking [23, 22].

We close this section by noting that HITS has been incorporated into the CLEVER project at IBM Almaden Research Center [1]. HITS is also part of the underlying ranking technology used by the search engine Teoma.

4. PageRank. In 1998 (the same year that HITS was born), Google founders Larry Page and Sergey Brin formulated their PageRank concept and made it the basis for their search engine [15]. As stated on the Google web site, "The heart of our software is PageRank$^{\text{TM}} \cdots$ [it] provides the basis for all of our web search tools." By avoiding the inherent weaknesses of HITS, PageRank has been responsible for elevating Google to the position of the world's preeminent search engine.

After web pages retrieved by robot crawlers are indexed and cataloged, PageRank values are assigned prior to query time according to perceived importance so that at query time a ranked list of pages related to the query terms can be presented to the user almost instantaneously. PageRank importance is determined by "votes" in the form of links from other pages on the web. The idea is that votes (links) from important sites should carry more weight than votes (links) from less important sites, and the significance of a vote (link) from any source should be tempered (or scaled) by the number of sites the source is voting for (linking to). These notions are encapsulated by defining the rank $r(P)$ of a given page P to be

$$r(P) = \sum_{Q \in \mathcal{B}_P} \frac{r(Q)}{|Q|}, \quad \text{where} \quad \begin{array}{l} \mathcal{B}_P = \{\text{all pages pointing to } P\}, \\ |Q| = \text{number of out links from } Q. \end{array}$$

This is a recursive definition, so computation necessarily requires iteration. If there are n pages, $P_1, P_2, \ldots, P_n$, arbitrarily assign each page an initial ranking, say, $r_0(P_i) = 1/n$, then successively refine the ranking by computing

$$r_j(P_i) = \sum_{Q \in \mathcal{B}_{P_i}} \frac{r_{j-1}(Q)}{|Q|}, \quad \text{for } \; j = 1, 2, 3, \ldots.$$

This is accomplished by setting $\boldsymbol{\pi}_j^T = \big(r_j(P_1),\, r_j(P_2),\, \cdots,\, r_j(P_n)\big)$, and iteratively computing

$$(4.1)\quad \boldsymbol{\pi}_j^T = \boldsymbol{\pi}_{j-1}^T \mathbf{P}, \text{ where } \mathbf{P} \text{ is the matrix with } p_{ij} = \begin{cases} 1/|P_i| & \text{if } P_i \text{ links to } P_j, \\ 0 & \text{otherwise.} \end{cases}$$

The notation $|P_i|$ is the number of outlinks from page P_i. Again, this is the power method. If the limit exists, the *PageRank vector* is defined to be $\boldsymbol{\pi}^T = \lim_{j\to\infty} \boldsymbol{\pi}_j^T$, and the ith component π_i is the PageRank of P_i. This is the raw idea, but for both theoretical and practical reasons (e.g., ensuring convergence, customizing rankings, and adjusting convergence rates), the matrix $\mathbf{P}$ must be adjusted—how adjustments are made is described in §4.3.

4.1. Markov Model of the Web. The "raw" Google matrix $\mathbf{P}$ is nonnegative with row sums equal to one or zero. Zero row sums correspond to pages that that

have no outlinks—such pages are sometimes referred to as *dangling nodes*. If we are willing to assume for a moment that there are no dangling nodes or that they are accounted for by artificially adding appropriate links to make all row sums equal one, then $\mathbf{P}$ is a row stochastic matrix, which in turn means that the PageRank iteration (4.1) represents the evolution of a Markov chain [45, Chapt. 8]. More precisely, this Markov chain is a random walk on the graph defined by the link structure of the web pages in Google's database.

For example, consider the hyperlink structure of Tiny Web consisting of six webpages linked as in Figure 2.1. The Markov model represents Tiny Web's directed graph as a square transition probability matrix $\mathbf{P}$ whose element p_{ij} is the probability of moving from state i (page i) to state j (page j) in one step (click). For example, assume that, starting from any node (webpage), it is equally likely to follow any of the outgoing links to arrive at another node. Thus,

$$\mathbf{P} = \begin{pmatrix} 0 & .5 & .5 & 0 & 0 & 0 \\ .5 & 0 & .5 & 0 & 0 & 0 \\ 0 & .5 & 0 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & .5 & .5 \\ 0 & 0 & .5 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix},$$

which is Tiny Web's raw (unadjusted) "Google matrix" described in (4.1). However, other suitable probability distributions may be used across the rows. For example, if web usage logs show that users accessing page 2 are twice as likely to jump to page 1 as they are to jump to page 3, then $\mathbf{p}_2^T$ (the second row of $\mathbf{P}$) might alternately be defined as

$$\mathbf{p}_2^T = (\, .6667 \quad 0 \quad .3333 \quad 0 \quad 0 \quad 0\,).$$

Other ideas for filling in the entries of $\mathbf{P}$ have been suggested [62, 52].

In general, the dominant eigenvalue for every stochastic matrix $\mathbf{P}$ is $\lambda = 1$. Consequently, if the PageRank iteration (4.1) converges, it converges to the normalized left-hand eigenvector $\boldsymbol{\pi}^T$ satisfying

$$\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}, \quad \boldsymbol{\pi}^T\mathbf{e} = 1, \qquad (\mathbf{e} \text{ is a column of ones}), \tag{4.2}$$

which is the stationary (or steady-state) distribution of the Markov chain [45, Chapt. 8]. This is why Google intuitively characterizes the PageRank value of each site as the long-run proportion of time spent at that site by a web surfer eternally clicking on links at random.[1]

4.2. Computing PageRank. Since the computation of PageRank boils down to solving either the eigenvector problem (4.2) or, equivalently, solving the homogeneous linear system $\boldsymbol{\pi}^T(\mathbf{I}-\mathbf{P}) = \mathbf{0}$ with $\boldsymbol{\pi}^T\mathbf{e} = 1$, determining PageRank might seem like a rather easy task. But quite to the contrary, the size of the problem (there are currently almost 4,300,000,000 pages in Google's database) severely limits the choice of algorithms that can be effectively used to compute $\boldsymbol{\pi}^T$. In fact, this computation has been called "the world's largest matrix computation" [47]. Direct methods (even those tuned for sparsity) as well as eigensolvers can't handle the overwhelming size, and variants of the power method seem to be the only practical choices. The time required by Google to compute the PageRank vector has been reported to be on the order of several days.

[1] Clicking BACK or entering a URL on the command line is excluded in this model.

4.3. Adjusting P . There are a few problems with strictly using the hyperlink structure of the Web to build a transition probability matrix that will adequately define PageRank. First, as noted earlier, the raw Google matrix $\mathbf{P}$ can fail to be a stochastic matrix because $\mathbf{P}$ has a row of all zeros for each node with zero outdegree. This is easily remedied by replacing each zero row with $\mathbf{e}^T/n$, where n is the order of $\mathbf{P}$. Call this new matrix $\bar{\mathbf{P}}$. But this alone doesn't fix all of the problems.

Another greater difficulty can (and usually does) arise: $\bar{\mathbf{P}}$ may be a reducible matrix because the underlying chain is reducible. *Reducible* chains are those that contain sets of states in which the chain eventually becomes trapped—i.e., by a reordering of states the transition matrix of a reducible chain can be made to have the canonical form

$$\mathbf{P} = \begin{array}{c} \\ S_1 \\ S_2 \end{array} \begin{array}{c} \begin{array}{cc} S_1 & S_2 \end{array} \\ \begin{pmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{0} & \mathbf{T}_{22} \end{pmatrix} \end{array}. \tag{4.3}$$

Once a state in set S_2 has been reached, the chain never returns to the states of S_1. For example, if web page P_i contains only a link to page P_j, and P_j contains only a link to P_i, then Google's random surfer who hits either P_i or P_j is trapped into bouncing between these two pages endlessly, which is the essence of reducibility.

An *irreducible* Markov chain is one in which every state is eventually reachable from every other state. That is, there exists a path from node i to node j for all i, j. Irreducibility is a desirable property because it is precisely the feature that guarantees that a Markov chain possesses a unique (and positive) stationary distribution vector $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$—it's the Perron-Frobenius theorem at work [45, Chapt. 8].

The modification of the raw Google matrix $\mathbf{P}$ leading to $\bar{\mathbf{P}}$ as described earlier produces a stochastic matrix, but the structure of the world wide web is such that $\bar{\mathbf{P}}$ is almost certainly reducible. Hence further adjustment is necessary in order ensure irreducibility. Brin and Page force irreducibility into the picture by making every state directly reachable from every other state. They originally did so by adding a perturbation matrix $\mathbf{E} = \mathbf{e}\mathbf{e}^T/n$ to $\bar{\mathbf{P}}$ to form

$$\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{E},$$

where α is a scalar between 0 and 1. The Google reasoning was that this stochastic matrix models a web surfer's "teleportation" tendency to randomly jump to a new page by entering a URL on the command line, and it assumes that each URL has an equal likelihood of being selected. Later Google adopted a more realistic and less democratic stance by using a better (and more flexible) perturbation matrix

$$\mathbf{E} = \mathbf{e}\mathbf{v}^T,$$

where the "personalization" vector $\mathbf{v}^T > \mathbf{0}$ is a probability vector that allows non-uniform probabilities for teleporting to particular pages. More importantly, at least from a business viewpoint, taking the perturbation to be of the form $\mathbf{E} = \mathbf{e}\mathbf{v}^T$ permits "intervention" by fiddling with $\mathbf{v}^T$ to adjust PageRank values up or down according to commercial considerations [60]. Other perturbation terms may be used as well, but, in any case, $\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{E}$ is a convex combination of two stochastic matrices $\mathbf{P}$ and $\mathbf{E}$ such that $\bar{\bar{\mathbf{P}}}$ is stochastic and irreducible and hence $\bar{\bar{\mathbf{P}}}$ possesses a unique stationary distribution $\boldsymbol{\pi}^T$. It's the matrix $\bar{\bar{\mathbf{P}}}$ that is now generally called "the Google matrix" and its stationary distribution $\boldsymbol{\pi}^T$ is the real PageRank vector.

Forcing the irreducibility by adding direct connections between each node might be overkill. To force irreducibility in a minimal sense, only one nonzero entry needs to be added to the leading position in the $(2,1)$-block of zeros in $\mathbf{P}$ once it has been permuted to canonical form (4.3). In other words, if

$$\bar{\bar{\mathbf{P}}} = \begin{pmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{C} & \mathbf{T}_{22} \end{pmatrix}, \quad \text{where } \mathbf{C} = \begin{pmatrix} \epsilon & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} \end{pmatrix},$$

then $\bar{\bar{\mathbf{P}}}$ is irreducible. Several other ways of forcing irreducibility have been suggested and analyzed [39, 10, 59], but Google seems to favor the $\mathbf{E} = \mathbf{e}\mathbf{v}^T$ approach.

A rather happy accident of Google's approach is that the eigendistribution of $\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{E}$ is affected in an advantageous manner. As pointed out earlier, the asymptotic rate of convergence of the power method is governed by the degree of separation between the dominant and closest subdominant eigenvalues. For the Google matrix it's easy to show that if the respective spectrums are $\sigma(\bar{\mathbf{P}}) = \{1, \mu_2, \ldots, \mu_n\}$ and $\sigma(\bar{\bar{\mathbf{P}}}) = \{1, \lambda_2, \ldots, \lambda_n\}$, then

$$\lambda_k = \alpha\mu_k \quad \text{for} \quad k = 2, 3, \ldots, n, \tag{4.4}$$

regardless of the value of the personalization vector $\mathbf{v}^T$ in $\mathbf{E} = \mathbf{e}\mathbf{v}^T$ [45, pg. 502], [39, 30]. Furthermore, the link structure of the web makes it likely that $\mu_2 = 1$ (or at least $\mu_2 \approx 1$). Therefore, choosing α to be farther from 1 increases the gap between 1 and λ_2 and thus speeds the convergence to PageRank. Google originally reported using $\alpha = .85$, which makes it likely that $\lambda_2 = .85$. Since $(\lambda_2)^{114} = (.85)^{114} < 10^{-8}$, it follows that roughly 114 iterations of the power method gives an accuracy on the order of 10^{-8} for Google's PageRank measures, most likely a higher degree of accuracy than they need in practical situations. However, on the con side, the maximally irreducible approach clearly alters the true nature of the chain much more than the minimally irreducible approach.

We feel experimental comparisons between the minimally irreducible $\bar{\bar{\mathbf{P}}}$ and the maximally irreducible $\bar{\mathbf{P}}$ might yield interesting results about surfer behavior and the sensitivity of PageRank to small additive perturbations. Reference [39] contains some theoretical progress in this area.

4.4. PageRank Implementation. Note that PageRank actually gives an importance score for each webpage, not a relevancy score. PageRank is just one part of Google's ranking system. In fact, PageRank is combined with other scores to give an overall ranking [11]. However, to simplify the examples, we present a basic model for use of PageRank. In this model, the implementation of the PageRank IR system involves two primary steps. In the first step, a full document scan determines the subset of nodes containing the query terms. This subset is called the relevancy set for the query. This is analogous to the first step of HITS, in which the neighborhood graph is formed. In the second step, the relevancy set is sorted according to the PageRank scores of each document in the set. Thus, PageRank does not depend on the query. In fact, each document has a PageRank score that is independent of all queries. It has been reported that Google computes PageRank once every few weeks for all documents in its Web collection [55]. As mentioned earlier, the computation of PageRank is a costly, time-consuming effort that involves finding the stationary vector of an irreducible stochastic matrix whose size is on the order of billions, and the power method seems to have been Google's method of choice [28, 15]. The algorithm to compute the PageRank vector $\boldsymbol{\pi}^T$ for the Google matrix $\bar{\bar{\mathbf{P}}} = \alpha\bar{\mathbf{P}} + (1-\alpha)\mathbf{E}$ is

simply stated. After specifying a value for the tuning parameter α, set $\boldsymbol{\pi}_0^T = \mathbf{e}^T/n$, where n is the size of $\mathbf{P}$, and iterate

$$\boldsymbol{\pi}_{k+1}^T = \alpha \boldsymbol{\pi}_k^T \bar{\mathbf{P}} + (1-\alpha)\mathbf{v}^T \tag{4.5}$$

until the desired degree of convergence is attained.

This implementation (4.5) has a couple of things going for it. First, the method does not destroy the extreme sparsity that is inherent. Second, the main vector-matrix multiplication of $\boldsymbol{\pi}_k^T \bar{\mathbf{P}}$ requires only sparse inner products, and these sparse inner products are easily implemented in parallel. Using parallelism is imperative for a problem of this size. More advanced iterative system/eigensolver methods do speed theoretical convergence, but in practice the fail to deliver due to immense storage requirements and increased computational complexity. Brin and Page report rather quick convergence with the simple power method—they claim useful results are obtained with only 50 power iterations on a $\mathbf{P}$ of order $n = 322,000,000$.

There have been several recent advances in PageRank computation and implementation, proposed largely by researchers at Stanford. Arasu et al. [2] suggest using the Gauss-Seidel method [56] in place of the simple power method. On one test example, they report faster convergence, most especially in the beginning of the iteration history. Another group of researchers at Stanford, Kamvar et al., have developed several modifications to the power method that accelerate convergence. One technique uses quadratic extrapolation, similar to Aitken's Δ^2 method, to speed convergence to the PageRank vector. The results show speedups of 50-300% over the basic power method, with minimal additional overhead [35]. The same group of researchers have also developed a BlockRank algorithm that uses aggregation methods to empirically achieve speedups of a factor of 2 [34]. One final algorithm uses an adaptive method to monitor the convergence of individual elements of the PageRank vector. As soon as elements of this vector have converged, they are no longer computed. Empirical evidence shows speedier convergence by 17% as the work per iteration decreases throughout the iteration history [33]. Very recent work partitions the $\mathbf{P}$ matrix into two groups according to dangling nodes (with no outlinks) and nondangling nodes [40, 39, 37, 38]. Then using exact aggregation the problem is reduced by a factor of 4 by lumping all the dangling nodes into one state. (Dangling nodes account for one-fourth of the web's nodes.) The most exciting feature of all these algorithms is that they do not compete with one another. In fact, it is possible to combine several algorithms to achieve even greater speedups.

4.4.1. PageRank Convergence. Since the power method seems to be about the only game in town as far as PageRank is concerned, a couple more comments are in order. Because the iteration matrix $\bar{\mathbf{P}}$ is a stochastic matrix, the spectral radius $\rho(\bar{\mathbf{P}}) = 1$. If this stochastic matrix is reducible, it may have several eigenvalues on the unit circle, causing convergence problems for the power method. One such problem was identified by Brin and Page as a rank sink, a node with no outlinks that keeps accumulating more and more PageRank at each iteration. This rank sink is actually an absorbing state of the Markov chain. More generally, a reducible matrix may contain an absorbing class that eventually sucks all the PageRank into states in its class. The web graph may contain several such classes and the long-run probabilities of the chain then depend greatly on the starting vector. Some states and classes may have 0 rank in the long-run, giving an undesirable solution and interpretation for the PageRank problem. However, the situation is much nicer and the convergence much cleaner for an irreducible matrix.

For an irreducible stochastic matrix, there is only one eigenvalue on the unit circle, all other eigenvalues have modulus strictly less than one [45]. This means that the power method applied to an irreducible stochastic matrix $\mathbf{P}$ is guaranteed to converge to the unique dominant eigenvector—the stationary vector $\boldsymbol{\pi}^T$ for the Markov matrix and the PageRank vector for the Google matrix. This is one reason why Brin and Page added the fudge factor matrix forcing irreducibility. Unlike HITS, there are now no issues with uniqueness of the ranking vector, and any positive probability vector can be used to start the iteration.

Even though the power method applied to the irreducible stochastic matrix $\bar{\bar{\mathbf{P}}}$ converges to a unique PageRank vector, the rate of convergence is a crucial issue, especially considering the scope of the matrix-vector multiplications—it's on the order of billions, since unlike HITS, PageRank operates on Google's version of the full web. As alluded to earlier, the asymptotic rate of convergence of (4.5) is governed by the rate at which $\lambda_2^k \to 0$, so, in light of (4.4), the asymptotic rate of convergence is the rate at which $(\alpha\mu_2)^k \to 0$, regardless of the value of the personalization vector $\mathbf{v}^T$ in $\mathbf{E} = \mathbf{e}\mathbf{v}^T$. The structure of the web forces $\mu_2 = 1$ (or at least $\mu_2 \approx 1$) with very high probability, so the rate of convergence of (4.5) boils down to how fast $\alpha^k \to 0$. In other words, Google engineers can dictate the rate of convergence according to how small they choose α to be.

Consequently, Google engineers are forced to perform a delicate balancing act. The smaller α is, the faster the convergence, but the smaller α is, the less the true hyperlink structure of the web is used to determine webpage importance. And slightly different values for α can produce very different PageRanks. Moreover, as $\alpha \to 1$, not only does convergence slow drastically, but sensitivity issues begin to surface as well [39].

4.4.2. PageRank Accuracy. Another implementation issue is the accuracy of PageRank computations. We do not know the accuracy with which Google works, but it at least has to be high enough to differentiate between the often large list of ranked pages that Google commonly returns. Since $\boldsymbol{\pi}^T$ is a probability vector, each π_i will be between 0 and 1. Suppose $\boldsymbol{\pi}^T$ is a 1 by 4 billion vector. Since the PageRank vector is known to follow a power law or Zipfian distribution [3, 50, 25][2], it is possible that a small section of the tail of this vector, ranked in decreasing order, might look like:

$$\boldsymbol{\pi}^T = (\,\cdots \quad .000001532 \quad .0000015316 \quad .0000015312 \quad .0000015210 \quad \cdots\,).$$

Accuracy *at least* on the order of 10^{-9} is needed to distinguish among the elements of this ranked subvector. However, comparisons are made only among a subset of elements of this ranked vector. While the elements of the entire global PageRank vector may be tightly packed in some sections of the (0,1) interval, elements of the subset related to a particular query are much less densely packed. Therefore, extreme accuracy on the order of 10^{-12} is most likely unnecessary for this application.

The fact that Brin and Page report reasonable estimates for $\boldsymbol{\pi}^T$ after only 50 iterations of the power method on a matrix of order $322,000,000$ has one of two implications: either (1) their estimates of $\boldsymbol{\pi}^T$ are not very accurate, or (2) the subdominant eigenvalue of the iteration matrix is far removed from $\lambda_1 = 1$. The first statement is a claim that outsiders not privy to inside information can never verify,

[2]Kamvar et al. have implemented an adaptive power method that exploits the power law structure of the PageRank vector to reduce work per iteration and convergence times [33].

as Google has never published information about their convergence tests. The implication of the second statement is that the "fudge factor" matrix $\mathbf{E} = \mathbf{e}\mathbf{e}^T/n$ (or more generally, $\mathbf{E} = \mathbf{e}\mathbf{v}^T$) must carry a good deal of weight and perhaps α is lowered to .8 in order to increase the eigengap and speed convergence. By decreasing α and simultaneously increasing the weight of the fudge factor, the transition probability matrix moves further from the Web's original hyperlink structure. Observations (admittedly limited by Google standards) suggest that the web's natural link structure tends to produces a nearly completely decomposable[3] (NCD) Markov chain or a Markov chain with NCD subgraphs [45, 56]. For example, there is considerable evidence that portions of the Web, like Africa's Web, are indeed NCD [12].

NCDness may be masked by the increasing weight given to the fudge factor. If NCDness is discovered, then a new avenue for IR research and implementation will be opened because significant work has already been done on computing the stationary vector for NCD systems [56, 58, 18, 46, 57]. References [2, 16, 12, 25, 50] contain information about the graph structure of the Web and some suggestions for exploiting this structure.

4.4.3. PageRank Updating. Updating PageRank is a significant concern. Since the computation of $\boldsymbol{\pi}^T$ is such an expensive endeavor, Google has reported that PageRank is only updated once every few weeks. In the interim however, significant changes to the Web's structure occur, and this presents another dilemma to Google engineers—how long can the huge effort of a complete update be put off before Google's output becomes too stale. The updating bottleneck is exacerbated by the fact that the PageRank vector from a prior period is nearly useless for the purpose of initializing or somehow gaining an advantage to run the power method for the next period, so, at last report, Google starts more or less from scratch each time the PageRank vector is updated.

Some research has recently been done on updating PageRank [17, 38, 49, 48] in which the goal is to use the old PageRanks from the prior period plus the changes to the Web's structure to estimate the new PageRanks without total recomputation. However, most of this work addresses only link updates and not node updates. A link modification occurs when hyperlinks are added, deleted or changed. A node modification occurs when a webpage is added to or deleted from the Web. Node updates affect the size of $\mathbf{P}$ and thus create a much more difficult problem.

Updating Markov chains is an old problem, and there are several exact algorithms for doing so [38], but they don't work well for the PageRank problem. Classical updating techniques applied to PageRank generally require more effort and time than total recomputation using the power method. Approximate updating algorithms that require less effort than exact methods but produce reasonable estimates of PageRank exist [17, 38].

Updating PageRank is an active area of research and recent breakthroughs have been made to simultaneously accommodate both link and node updates. The most promising algorithms along these lines seem to be an iterative aggregation technique [38] and an adaptive acceleration [33] of the power method. Each of these two approaches seems to decrease overall effort and cost of computing PageRank by a factor as great as 10. And preliminary studies indicate that these two processes can be

[3] A Markov chain is described as NCD if the state space can be partitioned into disjoint subsets, with strong interactions among the states of a subset but with weak interactions among the subsets themselves. The transition probability matrix of an NCD chain can be reordered to have a dense block diagonal form with sparse off-diagonal blocks.

married to produce a reduction of effort on even a more dramatic scale—details are forthcoming.

The PageRank implementation issues of convergence, accuracy, and updating, are just a few of the areas that numerical analysts and IR researchers have been studying recently. Other active areas (but not reported on here) are clustering and parallelism.

4.5. PageRank Example. In this section, we present a very small and simple example to solidify the PageRank ideas discussed above. Consider the tiny 6-node Web of Figure 4.1.

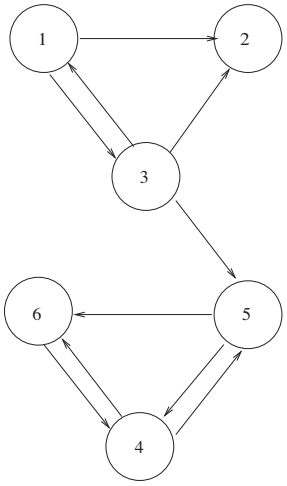

FIG. 4.1. *6-node Web for PageRank example*

First create the *raw* Google Matrix as shown below.

$$\mathbf{P} = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{array} \begin{array}{c} \begin{array}{cccccc} d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \end{array} \\ \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}.$$

The second row of $\mathbf{P}$ is entirely zero because there are no outlinks from the second page, and consequently $\mathbf{P}$ is not a stochastic matrix. (Brin and Page refer to node 2 as a rank sink.) Add 1/6 to each entry in the second row to remedy this problem—the result is

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

This matrix is now stochastic, but it's reducible, so it cannot have a unique positive

stationary distribution. To force irreducibility. choose $\alpha = .9$ and form

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

This matrix is both stochastic and irreducible, and its stationary vector (and the PageRank vector) is

$$\boldsymbol{\pi}^T = (\, .03721 \quad .05396 \quad .04151 \quad .3751 \quad .206 \quad .2862 \,)$$

(to four significant places). These PageRanks are query-independent. Suppose a query is entered containing terms 1 and 2. The inverted term-document file below is accessed.[4]

Inverted file storage of document content

$$\begin{array}{ll} \text{term 1} & \rightarrow \quad \text{doc 1, doc 4, doc 6} \\ \text{term 2} & \rightarrow \quad \text{doc 1, doc 3} \\ \vdots & \end{array}$$

Thus, the relevancy set for a query on terms 1 and 2 is {1, 3, 4, 6}. The PageRanks of these 4 documents are now compared to determine which of these 4 relevant documents is most important by sorting their associated page ranks $\pi_1, \pi_3, \pi_4, \pi_6$ in decreasing order, which gives

$$\begin{aligned} \pi_4 &= .3751 \\ \pi_6 &= .2862 \\ \pi_3 &= .04151 \\ \pi_1 &= .03721. \end{aligned}$$

Consequently, document 4 is the most important of the relevant documents, followed by documents 6, 3 and 1. Should another query be entered, the relevancy set is quickly determined by consulting the inverted term-document file and those documents in the relevancy set are quickly sorted by their PageRanks. The power of PageRank's query-independence cannot be understated.

4.6. PageRank's connection to HITS. Ding et al. have shown a very elegant connection between PageRank and HITS [23, 22]. They point out PageRank's similarity to HITS' authority score. Ding et al. have even extended this connection to create a "hub score" for PageRank. In the paper, the authors claim that the number of inlinks into a document gives a good approximation to the PageRank of that document. However, we and Raghavan et al. have found this approximation to be very rough [50]. A simple inlink count ignores the PageRank thesis that nodes are important if they are linked to by other *important* nodes. The inlink count measures only the *quantity* not the *quality* of these inlinks.

[4]Inverted file storage is similar to the index in the back of a book—it's a table containing a row for every term in a collection's dictionary. Next to each term is a list of all documents that use that term (or are related to that term).

4.7. Strengths and Weaknesses of PageRank. We have already mentioned one weakness of PageRank—the topic drift problem due to the importance of determining an accurate relevancy score. Much work, thought and heuristics must be applied by Google engineers to determine the relevancy score, otherwise, no matter how good PageRank is, the ranked list returned to the user is of little value if the pages are off-topic. This invites the question: why does *importance* serve as such a good proxy to *relevance*? Or does it? By emphasizing the importance of documents, are lesser-known, obscure, yet highly relevant documents being missed? Bharat et al. succinctly state this weakness of PageRank in [8]. "Since PageRank is query-independent, it cannot by itself distinguish between pages that are authoritative in general and pages that are authoritative on the query topic." Some of these questions may be unanswerable due to the proprietary nature of Google, but they are still worthy of consideration.

On the other hand, the use of importance, rather than relevance, is the key to Google's success and the source of its strength. By measuring importance, query-dependence, the main weakness of HITS, becomes a non-issue. Instead, the PageRank measure of importance is a query-independent measure. At query time, only a quick lookup into an inverted file storage is required to determine the relevancy set, which is then sorted by the precomputed PageRanks.

Another strength of PageRank is its virtual imperviousness to spamming. As mentioned during the HITS discussion, it is very hard for a webpage owner to add inlinks into his page from other important pages. Chien et al. have proven that if the owner succeeds in doing this, the PageRank is guaranteed to increase. However, this increase will likely be inconsequential since PageRank is a global measure [9, 48]. In contrast, HITS' authority and hub scores are derived from a local neighborhood graph and slight increases in the number of inlinks or outlinks will have a greater relative impact. Thus, in summary, webpage owners have very little ability to and likelihood of affecting their PageRank scores. Nevertheless, there have been some papers describing ways to both influence PageRank and recognize spam attempts [10, 61].

And finally there is the flexibility of the "personalization" (or "intervention") vector $\mathbf{v}^T$ that Google is free to choose in defining the fudge-factor term $\mathbf{E} = \mathbf{e}\mathbf{v}^T$. The choice of $\mathbf{v}^T$ affects neither mathematical nor computational aspects, but it does alter the ranks in a predictable manner. This can be a terrific advantage if Google wants to intervene to push a site's PageRank down or up, perhaps to punish a suspected "link farmer" or to reward a favored client. The outside world is not privy to the extent to which Google actually does this, but it is known that Google is quite vigilant and sensitive concerning people who try to manipulate PageRank [60].

5. SALSA. We now move on to the third eigenvector Web IR method, Stochastic Approach for Link Structure Analysis (SALSA). SALSA, developed by Lempel and Moran in 2000, was spawned by the combination of ideas from both HITS and PageRank [41]. Like HITS, both hub and authority scores for webpages are created, and like PageRank, they are created through the use of Markov chains. In this introduction to SALSA, we first walk through an example of the algorithm, then assess its strengths and weaknesses.

5.1. Example. In a manner similar to HITS, the neighborhood graph N associated with a particular query is formed. We use the same neighborhood graph N from figure 3.2. SALSA differs from HITS in the next step. Rather than forming an adjacency matrix $\mathbf{L}$ for the neighborhood graph N, a bipartite undirected graph,

denoted G, is built. G is defined by three sets: V_h, V_a, E, where V_h is the set of hub nodes (all nodes in N with outdegree > 0), V_a is the set of authority nodes (all nodes in N with indegree > 0) and E is the set of directed edges in N. Note that a node in N may be in both V_h and V_a. For the example with neighborhood graph given by Figure 3.2,

$$V_h = \{1, 2, 3, 6, 10\},$$
$$V_a = \{1, 3, 5, 6\}.$$

The bipartite undirected graph G has a "hub side" and an "authority side". Nodes in V_h are listed on the hub side and nodes in V_a are on the authority side. Every directed edge in E is represented by an undirected edge in G. Next, two Markov

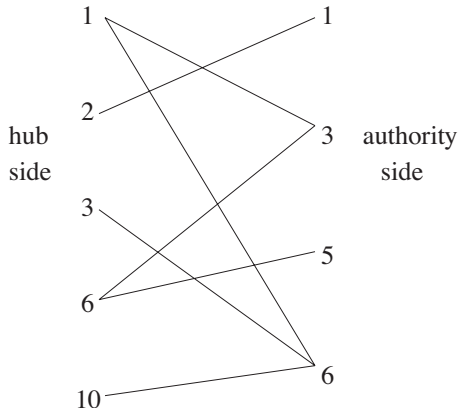

FIG. 5.1. *G: bipartite graph for SALSA*

chains are formed from G, a hub Markov chain with transition probability matrix $\mathbf{H}$, and an authority Markov chain with matrix $\mathbf{A}$. Reference [41] contains a formula for computing the elements of $\mathbf{H}$ and $\mathbf{A}$, but we feel a more instructive approach to building $\mathbf{H}$ and $\mathbf{A}$ clearly reveals SALSA's connection to both HITS and PageRank. Recall that $\mathbf{L}$ is the adjacency matrix of N used by HITS. HITS computes authority and hub scores using the unweighted matrix $\mathbf{L}$, while PageRank computes a measure analogous to an authority score using a row-weighted matrix $\mathbf{P}$. SALSA uses both row and column weighting to compute its hub and authority scores. Let $\mathbf{L}_r$ be $\mathbf{L}$ with each nonzero row divided by its row sum and $\mathbf{L}_c$ be $\mathbf{L}$ with each nonzero column divided by its column sum. Then $\mathbf{H}$, SALSA's hub matrix consists of the nonzero rows and columns of $\mathbf{L}_r\mathbf{L}_c^T$ and $\mathbf{A}$ is the nonzero rows and columns of $\mathbf{L}_c^T\mathbf{L}_r$. For our running example (from Figure 3.2),

$$\mathbf{L} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}, \qquad \mathbf{L}_r = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array},$$

and

$$\mathbf{L}_c = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{3} & 0 \end{pmatrix} \end{array}.$$

Hence

$$\mathbf{L}_r\mathbf{L}_c^T = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} \frac{5}{12} & 0 & \frac{2}{12} & 0 & \frac{3}{12} & \frac{2}{12} \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & 1 & 0 & \frac{3}{4} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \end{pmatrix} \end{array}, \quad \mathbf{L}_c^T\mathbf{L}_r = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 5 & 6 & 10 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{6} & 0 & \frac{5}{6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

Taking just the nonzero rows and columns of $\mathbf{L}_r\mathbf{L}_c^T$ to form $\mathbf{H}$ gives

$$\mathbf{H} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 6 \\ 10 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 6 & 10 \end{array} \\ \begin{pmatrix} \frac{5}{12} & 0 & \frac{2}{12} & \frac{3}{12} & \frac{2}{12} \\ 0 & 1 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & 0 & 0 & \frac{3}{4} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \end{pmatrix} \end{array}.$$

Similarly,

$$\mathbf{A} = \begin{array}{c} \\ 1 \\ 3 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 3 & 5 & 6 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{6} & 0 & \frac{5}{6} \end{pmatrix} \end{array}.$$

If G is connected, then $\mathbf{H}$ and $\mathbf{A}$ are both irreducible Markov chains and $\boldsymbol{\pi}_h^T$, the stationary vector of $\mathbf{H}$, gives the hub scores for the query with neighborhood graph N and $\boldsymbol{\pi}_a^T$ gives the authority scores. If G is not connected, then $\mathbf{H}$ and $\mathbf{A}$ contain multiple irreducible components. In this case, the global hub and authority scores must be pasted together from the stationary vectors for each individual irreducible component. (Reference [41] contains the justification for the above two if-then statements.)

From the structure of the $\mathbf{H}$ and $\mathbf{A}$ matrices for our example, it is easy to see that G is not connected. Thus, $\mathbf{H}$ and $\mathbf{A}$ contain multiple connected components. $\mathbf{H}$ contains two connected components, $C = \{2\}$ and $D = \{1, 3, 6, 10\}$, while $\mathbf{A}$'s connected components are $E = \{1\}$ and $F = \{3, 5, 6\}$. Also clear from the structure of $\mathbf{H}$ and $\mathbf{A}$ is the periodicity of the Markov chains. All irreducible components of $\mathbf{H}$ and $\mathbf{A}$ contain self-loops, implying that the chains are aperiodic. The stationary vectors for the two irreducible components of $\mathbf{H}$ are

$$\boldsymbol{\pi}_h^T(C) = \begin{array}{c} 2 \\ \end{array} \begin{pmatrix} 1 \end{pmatrix}, \quad \boldsymbol{\pi}_h^T(D) = \begin{array}{cccc} 1 & 3 & 6 & 10 \end{array} \begin{pmatrix} \frac{1}{3} & \frac{1}{6} & \frac{1}{3} & \frac{1}{6} \end{pmatrix},$$

while the stationary vectors for the two irreducible components of $\mathbf{A}$ are

$$\begin{array}{rcc} & 1 & \\ \boldsymbol{\pi}_a^T(E) = & \big(1\big), & \end{array} \quad \begin{array}{rccc} & 3 & 5 & 6 \\ \boldsymbol{\pi}_a^T(F) = & \big(\frac{1}{3} & \frac{1}{6} & \frac{1}{2}\big). \end{array}$$

Proposition 6 of [41] contains the method for pasting the hub and authority scores for the individual components into global scoring vectors. Their suggestion is simple and intuitive. Since the hub component C only contains 1 of the 5 total hub nodes, its stationary hub vector should be weighted by 1/5, while D, containing 4 of the 5 hub nodes, has its stationary vector weighted by 4/5. Thus the global hub vector is

$$\begin{array}{rccccc} & 1 & 2 & 3 & 6 & 10 \\ \boldsymbol{\pi}_h^T = & \big(\frac{4}{5}\cdot\frac{1}{3} & \frac{1}{5}\cdot 1 & \frac{4}{5}\cdot\frac{1}{6} & \frac{4}{5}\cdot\frac{1}{3} & \frac{4}{5}\cdot\frac{1}{6}\big) \\ & 1 & 2 & 3 & 6 & 10 \\ = & \big(.2667 & .2 & .1333 & .2667 & .1333\big). \end{array}$$

With similar weighting for authority nodes, the global authority vector can be constructed from the individual authority vectors as

$$\begin{array}{rcccc} & 1 & 3 & 5 & 6 \\ \boldsymbol{\pi}_h^T = & \big(\frac{1}{4}\cdot 1 & \frac{3}{4}\cdot\frac{1}{3} & \frac{3}{4}\cdot\frac{1}{6} & \frac{3}{4}\cdot\frac{1}{2}\big) \\ & 1 & 3 & 5 & 6 \\ = & \big(.25 & .25 & .125 & .375\big). \end{array}$$

Compare the SALSA hub and authority vectors with those of HITS in section 3.3. They are quite different. Notice that the presence of multiple connected components (which occurs when G is not connected) is actually a very good thing computationally, because the Markov chains to be solved are much smaller. This can be contrasted with PageRank's correction for a disconnected web graph, whereby irreducibility is forced by adding direct connections between all nodes. Also, note that other weighting schemes can be applied to paste the individual component scores together to create global scores.

5.2. Strengths and Weaknesses of SALSA. As SALSA was developed by combining some of the best features of both HITS and PageRank, it has many strengths. Unlike HITS, SALSA is not victimized by the topic drift problem, related to the "TKC" problem referenced by [41]. Recall that another problem of HITS was its susceptibility to spamming due to the interdependence of hub and authority scores. SALSA is less susceptible to spamming since the coupling between hub and authority scores is much less strict. However, neither HITS nor SALSA are as impervious to spamming as PageRank. SALSA, like HITS, also gives dual rankings, something which PageRank does not supply. The presence of multiple connected components in SALSA's bipartite graph G, a common occurrence in practice, is computationally welcomed. However, one major drawback to the widespread use of SALSA is its query-dependence. At query time, the neighborhood graph N for the query must be formed and two Markov chains must be solved. Another problematic issue for SALSA is convergence. The convergence of SALSA is similar to that of HITS. Because, both HITS and SALSA, unlike PageRank, do not force irreducibility onto the graph, the resulting vectors produced by their algorithms may not be unique (and may depend on the starting vector) if the neighborhood graph is reducible. Nevertheless, a simple

solution is to adopt the PageRank fix and force irreducibility by altering the graph in some small way.

Before moving onto to other Web IR methods, we would like to refer the reader to an excellent article by Farahat et al. on the existence and uniqueness of the eigenvectors used by these three Web IR methods: HITS, PageRank, and SALSA [27].

6. Other Methods and Hybrids. This section contains suggestions for new and hybrid methods that are related to the three ranking systems of HITS, PageRank, and SALSA. Recall that the major drawback of HITS is its query-dependence. At query time, authority and hub scores must be computed for those documents in the neighborhood graph N. One way to make HITS query-independent is to compute the authority and hub scores once using the entire graph. This requires the dominant eigenvector computation of the matrices $\mathbf{L}\mathbf{L}^T$ and $\mathbf{L}\mathbf{L}^T$, where this time the order of $\mathbf{L}$ is equal to the total number of documents in the collection. However, similar to PageRank, this is a one-time computation, requiring recomputation on a monthly basis to accommodate link changes. The question is whether these authority and hub scores would give much more or different information than the PageRank scores.

In an opposite fashion, researchers have experimented with forcing PageRank to be query-dependent, thus relegating the problem of topic drift [53, 6, 29]. Henzinger and Bharat have successfully resolved topic drift by incorporating both content information and hyperlink information into the HITS matrices [6]. Since Henzinger was once head of Google's research division, one might guess that this is how Google's $\mathbf{P}$ matrix is created as well. Ding et al. have created some HITS-PageRank hybrid algorithms [23]. They compare the original PageRank, the original HITS, hub PageRank and two other hybrid algorithms, using several experimental datasets. They conclude that with respect to hub scores especially, the algorithms select different sets of top ranked hubs. Thus, in the future, perhaps medleys of IR algorithms will provide the most relevant and precise documents to user queries [44]. Reference [13] contains an excellent summary of other IR methods built from HITS, PageRank and/or SALSA. These methods include PHITS [19], SALSA [41], pSALSA [13], exponentiated HITS [26], and Randomized HITS [49].

7. Future. HITS, PageRank, and SALSA all make extensive use of the Web's hyperlink structure to return relevant documents to IR users. Older methods like LSI, instead focus on the content of the documents. We feel that the coming methods will combine both the hyperlink structure and content analysis as well as make accommodations for latent semantic associations [21]. We also predict that more work will be done on tailoring search engines to individual users [20, 51]. Some studies examining the structure of the Web [13] have encouraged others to exploit properties of the Web's structure to partition the Web, in hopes of speeding matrix computations [2]. Finally, other trends in IR lean toward creating additions to current, popular search engines, like Google, Yahoo! or Altavista. For example, Mendelzon and Rafiei's TOPIC system [43] as well as Bharat et. al's topic classification system [7] annotate popular search engines with information on page topics. Such additions also aim at user adaptability.

8. Conclusion. Web IR is a significantly different problem than traditional IR, due in part to the Web's massive scale and its unique hyperlink structure. Traditional IR methods like LSI fall short when applied to Web IR problems. Instead, new methods, such as HITS, PageRank, and SALSA, have emerged. One common-

ality among these three Web IR methods is their use of and reliance on eigenvector computation. Such computation opens new research doors for numerical analysts and applied mathematicians in a field that has historically been dominated by computer scientists and database engineers. Now is certainly an exciting time to be studying Web IR methods and theory as the very nature and conduct of science and research is rapidly evolving and moving online. Addressing enormous information problems, like Web IR, will have both immediate impact and long-standing influence on the future of science.

REFERENCES

[1] Clever—IBM corporation almaden research center. online at `http://www.almaden.ibm.com/cs/k53/clever.html`. Accessed October 11, 2004.

[2] Arvind Arasu, Jasmine Novak, Andrew Tomkins, and John Tomlin. PageRank computation and the structure of the Web: experiments and algorithms. In *The Eleventh International WWW Conference*, New York, May 2002. ACM Press.

[3] Albert-Laszlo Barabasi. *Linked: The New Science of Networks*. Plume, 2003.

[4] Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. Matrices, vector spaces and information retrieval. *SIAM Review*, 41:335–362, 1999.

[5] Michael W. Berry, P. Wang, and J. Bownas. Website query analysis: trend and behavior detection. In *Second SIAM Conference on Data Mining*, April 2002.

[6] Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in hyperlinked environments. In *21st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 104–111, 1998.

[7] Krishna Bharat, Farzin Maghoul, and Raymie Stata. The term vector database: fast access to indexing terms for webpages. *Computer Networks*, 33:247–255, 2000.

[8] Krishna Bharat and George A. Mihaila. When experts agree: using non-affiliated experts to rank popular topics. *ACM Transactions on Information Systems*, 20(1):47–58, 2002.

[9] Monica Bianchini, Marco Gori, and Franco Scarselli. PageRank: A circuital analysis. In *The Eleventh International WWW Conference*, May 2002.

[10] Monica Bianchini, Marco Gori, and Franco Scarselli. Inside PageRank. *ACM Transactions on Internet Technology*, 5(1), 2005. To appear.

[11] Nancy Blachman, Eric Fredricksen, and Fritz Schneider. *How to Do Everything with Google*. McGraw-Hill, 2003.

[12] Paolo Boldi, Bruno Codenotti, Massimo Santini, and Sebastiano Vigna. Structural properties of the African web. In *The Eleventh International WWW Conference*, New York, May 2002. ACM Press.

[13] Allan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, and Panayiotis Tsaparas. Finding authorities and hubs from link structures on the World Wide Web. *World Wide Web*, pages 415–429, 2001.

[14] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 33:107–117, 1998.

[15] Sergey Brin, Lawrence Page, R. Motwami, and Terry Winograd. The PageRank citation ranking: bringing order to the Web. Technical Report 1999-0120, Computer Science Department, Stanford University, 1999.

[16] Andrei Broder, Ravi Kumar, and Marzin Maghoul. Graph structure in the Web. In *The Ninth International World Wide Web Conference*, pages 309–320, New York, May 2000. ACM Press.

[17] Steve Chien, Cynthia Dwork, Ravi Kumar, and D. Sivakumar. Towards exploiting link evolution. In *Workshop on algorithms and models for the Web graph*, 2001.

[18] Hwajeong Choi and Daniel B. Szyld. Application of threshold partitioning of sparse matrices to Markov chains. In *IEEE International Computer Performance and Dependability Symposium IPDS'96*, pages 158–165, September 1996.

[19] David Cohn and Huan Chang. Learning to probabilistically identify authoritative documents. In *Proceedings of the 17th International Conference on Machine Learning*, pages 167–174, Stanford, CA, 2000.

[20] David Cohn, Huan Chang, and Andrew McCallum. Creating customized authority lists. In *Proceedings of the 17th International Conference on Machine Learning*, Stanford, CA, 2000.

[21] David Cohn and Thomas Hofmann. The missing link: a probabilistic model of document content and hyperlink connectivity. *Advances in Neural Information Processing Systems*, 13, 2001.

[22] Chris Ding, Xiaofeng He, Parry Husbands, Hongyuan Zha, and Horst Simon. Link analysis: hubs and authorities on the World Wide Web. Technical Report 47847, Lawrence Berkeley National Laboratory, May 2001.

[23] Chris Ding, Xiaofeng He, Hongyuan Zha, and Horst Simon. PageRank, HITS and a unified framework for link analysis. In *Proceedings of the 25th ACM SIGIR Conference*, pages 353–354, Tampere, Finland, August 2002.

[24] Susan T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments and Computers*, 23:229–236, 1991.

[25] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262, 1999.

[26] Ayman Farahat, Thomas Lofaro, Joel C. Miller, Gregory Rae, F. Schaefer, and Lesley A. Ward. Modifications of Kleinberg's HITS algorithm using matrix exponentiation and web log records. In *ACM SIGIR Conference*, pages 444–445, September 2001.

[27] Ayman Farahat, Thomas Lofaro, Joel C. Miller, Gregory Rae, and Lesley A. Ward. Existence and uniqueness of ranking vectors for linear link analysis. *SIAM Journal on Scientific Computing*, 2004. submitted April 2004.

[28] Taher H. Haveliwala. Efficient computation of PageRank. Technical Report 1999-31, Computer Science Department, Stanford University, 1999.

[29] Taher H. Haveliwala. Topic-sensitive PageRank. In *The Eleventh International WWW Conference*, May 2002.

[30] Taher H. Haveliwala and Sepandar D. Kamvar. The second eigenvalue of the Google matrix. Technical Report 2003-20, Stanford University, 2003.

[31] Monika R. Henzinger, Hannes Marais, Michael Moricz, and Craig Silverstein. Analysis of a very large AltaVista query log. Technical Report 1998-014, Digital SRC, October 1998.

[32] Elizabeth R. Jessup and Ilse C.F. Ipsen. Improving the accuracy of inverse iteration. *SIAM J. Sci. Stat. Comput.*, 13:550–571, 1992.

[33] Sepandar D. Kamvar, Taher H. Haveliwala, and Gene H. Golub. Adaptive methods for the computation of PageRank. Technical Report 2003-26, Stanford University, 2003.

[34] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Exploiting the block structure of the Web for computing PageRank. Technical Report 2003-17, Stanford University, 2003.

[35] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Extrapolation methods for accelerating PageRank computations. Twelfth International World Wide Web Conference, 2003.

[36] Jon Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46, 1999.

[37] Amy Langville and Carl Meyer. A reordering for the PageRank problem. Technical Report CRSC Tech Report crsc-tr04-06, Center For Research in Scientific Computation, 2004. submitted to SIAM Journal on Scientific Computing, 2004.

[38] Amy N. Langville and Carl D. Meyer. Updating the stationary vector of an irreducible Markov chain. Technical Report crsc02-tr33, N. C. State, Mathematics Dept., CRSC, 2002.

[39] Amy N. Langville and Carl D. Meyer. Deeper inside PageRank. *Internet Mathematics Journal*, 2004. Accepted in February 2004.

[40] Chris Pan-Chi Lee, Gene H. Golub, and Stefanos A. Zenios. A fast two-stage algorithm for computing PageRank and its extensions. Technical Report SCCM-2003-15, Scientific Computation and Computational Mathematics, Stanford University, 2003.

[41] Ronny Lempel and S. Moran. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. In *The Ninth International WWW Conference*, May 2000.

[42] Massimo Marchiori. The quest for correct information of the web: hyper search engines. In *The Sixth International WWW Conference*, April 1997.

[43] Alberto O. Mendelzon and Davood Rafiei. Topic: measuring webpage reputation. online at `http://www.cs.toronto.edu/db/topic/about.html`. Accessed on September 19, 2002.

[44] Alberto O. Mendelzon and Davood Rafiei. An autonomous page ranking method for metasearch engines. In *The Eleventh International WWW Conference*, May 2002.

[45] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, Philadelphia, 2000.

[46] Violeta Migallon, Jose Penades, and Daniel B. Szyld. Experimental study of parallel iterative solutions of Markov chains with block partitions. In Brigitte Plateau, William J. Stewart, and Manuel Silva, editors, *Numerical Solutions of Markov Chains (NSMC'99)*, pages 96–110. Prensas Universitarias de Zaragoza, 1999.

[47] Cleve Moler. The world's largest matrix computation. *Matlab News and Notes*, pages 12–13, October 2002.

[48] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Link analysis, eigenvectors and stability. In *The Seventh International Joint Conference on Artificial Intelligence*, 2001.

[49] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Stable algorithms for link analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference*. ACM, 2001.

[50] Gopal Pandurangan, Prabhakar Raghavan, and Eli Upfal. Using PageRank to Characterize Web Structure. In *The Eighth Annual International Computing and Combinatorics Conference (COCOON)*, 2002.

[51] Jesus U. Quevedo, Ana G. Covarrubias, and S. H. Huang. Improving retrieval by querying and examining prestige. In *The Eleventh International WWW Conference*, May 2002.

[52] Davood Rafiei and Alberto O. Mendelzon. What is this page known for? computing web-page reputations. In *The Ninth International WWW Conference*, pages 823–835. Elsevier Science, May 2000.

[53] Matthew Richardson and Petro Domingos. The intelligent surfer: probabilistic combination of link and content information in PageRank. *Advances in Neural Information Processing Systems*, 14:1441–1448, 2002.

[54] Chris Ridings. PageRank explained: everything you've always wanted to know about PageRank. online at `http://www.rankwrite.com/`. Accessed on May 22, 2002.

[55] Markus Sobek. Google dance: The index update of the Google search engine. *eFactory: Internet-Agentur KG*. Accessed October 11, 2004.

[56] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[57] William J. Stewart and Tugrul Dayar. Comparison of partitioning techniques for two-level iterative solvers on large, sparse Markov chains. *SIAM Journal on Scientific Computing*, 21(5):1691–1705, 2000.

[58] William J. Stewart and W. Wu. Numerical experiments with iteration and aggregation for Markov chains. *ORSA Journal on Computing*, 4(3):336–350, 1992.

[59] John A. Tomlin. A new paradigm for ranking pages on the World Wide Web. In *The Twelfth International World Wide Web Conference*, 2003.

[60] Michael Totty and Mylene Mangalindan. As google becomes web's gatekeeper, sites fight to get in. *Wall Street Journal*, CCXLI(39), 2003. February 26.

[61] Ah Chung Tsoi, Gianni Morini, Franco Scarselli, and Markus Hagenbuchner. Adaptive ranking of webpages. In *The Twelfth International World Wide Web Conference*, 2003.

[62] Dell Zhang and Yisheng Dong. An efficient algorithm to rank web resources. *Computer Networks*, 33:449–455, 2000.

[63] Xiaoyan Zhang, Michael W. Berry, and Padma Raghavan. Level search schemes for information filtering and retrieval. *Information Processing and Management*, 37:313–334, 2001.