WEB DOCUMENT CLUSTERING USING HYPERLINK STRUCTURES

XIAOFENG HE[*†], HONGYUAN ZHA[*], CHRIS H.Q. DING[†] AND HORST D. SIMON[†]

Abstract. With the exponential growth of information on the World Wide Web, there is great demand for developing efficient and effective methods for organizing and retrieving the information available. Document clustering plays an important role in information retrieval and taxonomy management for the World Wide Web and remains an interesting and challenging problem in the field of web computing. In this paper we consider document clustering methods exploring textual information, hyperlink structure and co-citation relations. In particular, we apply the *normalized-cut* clustering method developed in computer vision to the task of hyperdocument clustering. We also explore some theoretical connections of the normalized-cut method to K-means method. We then experiment with normalized-cut method in the context of clustering query result sets for web search engines.

Keywords. World Wide Web, graph partitioning, cheeger constant, clustering method, K-means method, normalized cut method, eigenvalue decomposition, power method.

1. Introduction.

Currently the World Wide Web contains billions of documents and it is still growing rapidly. Finding the relevant documents to satisfy a user's information need is a very important and challenging task. Many commercial search engines have been developed and used by millions of people all over the world. However, the relevancy of documents returned in search engine result sets is still lacking, and further research and development is needed to really make search engines a ubiquitous information-seeking tool. The World Wide Web has a rich structure: it contains both textual web documents and the hyperlinks that connect them. The web documents and hyperlinks between them form a *directed* graph in which the web documents can be viewed as vertices and the hyperlinks as directed edges. Algorithms have been developed utilizing this directed graph to extract information contained in a collection of hyperlinked web documents. Kleinberg proposed HITS algorithm based purely on hyperlink information to retrieve the most relevant information: *authority* and *hub* documents for a user query [20]. However, if the hypertext collection consists of several topics, authority and hub documents may only cover the most popular topics and leave out the less popular ones. One way to remedy this situation is to first partition the hypertext collection into topical groups, and present the search results as a list of topics to the user. This leads to the need to cluster web documents based on both the textual and hyperlink information.

There exists a large literature on clustering methods and algorithms [13, 19]. Generally speaking, the purpose of *cluster analysis* is to organize the data into meaningful groups: the data objects in the same group are highly similar and those in different groups are dissimilar. Judging the effectiveness of a clustering algorithm is difficult and usually application-dependent. In this paper, we apply a similarity-based clustering method to the problem of clustering web documents. It utilizes a graph-theoretic criterion called *normalized cut* which has its root in the study of graph isoperimetric

[*] Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802, {xhe,zha}@cse.psu.edu. This work was supported in part by NSF grant CCR-9901986.

[†] NERSC Division, Lawrence Berkeley National Laboratory, University of California, Berkeley, CA 94720, {xfhe,chqding,hdsimon}@lbl.gov. Supported by Department of Energy through an LBL LDRD fund.

problems [7, 8]. This method was proposed by Shi and Malik and has been successfully used in image segmentation [28].

It is well-known that clustering web documents based entirely on the textual contents of the documents is not very effective in finding distinct topics [29]. In the context of clustering web documents, in addition to the textual contents of documents, many other sources of information can be effectively used to enhance clustering effectiveness, for example, hyperlinks between documents, and co-citation (co-reference) patterns among documents. In our web document clustering approach, we comprehensively incorporate information from hyperlink structure, co-citation patterns and textual contents of documents to construct a novel similarity metric for measuring the topical homogeneity of web documents. Specifically, the hyperlink structure is used as the dominant factor in the similarity metric, and the textual contents are used to modulate the strength of each hyperlink. The similarity metric is used to construct a weighted graph which is then fed to the clustering method based on normalized cut. This combination gives rise to a powerful method which effectively organizes the retrieved information against a user query and presents the organized information in a more accessible form for the user.

The rest of the paper is organized as follows: In section 2, we give some background materials about search engines and explain why it is important to cluster the search engine result sets effectively. In section 3, we discuss the normalized-cut graph-partitioning criterion and its relationship to the Cheeger constant and spectral graph theory. We also outline the spectral clustering algorithm based on normalized cut. In section 4, we explore the connection between normalized-cut based clustering method and the K-means method in the context of similarity-based clustering. In section 5, we explain how the textual contents, the hyperlink structure and co-citation patterns can be combined to construct a weighted graph for partitioning. Section 6 presents our experiments with the normalized-cut clustering algorithm. We give details of our data preparation process, and analyze the structures of clusters obtained for several broad-topic queries.

2. Organizing query result sets for search engines. The World Wide Web is a rich source of information. It has grown to more than one billion unique web documents [9] and continues to grow roughly at a rate of one million documents per day [4]. While the users are enjoying the large amount of information available on the World Wide Web, it poses a problem to both the users and the search engines. The problem lies in the sheer quantity of the web documents. When users submit a query, the search engine returns the retrieved web documents as the search result set. The query which the users tend to submit to the search engine typically has 1-3 words with little query formulation [1, 10]. Sometimes the users are uncertain about their information need when submitting the query [12], and for the ordinary users, it is difficult to come up with the query terms that accurately specify their information needs. These situations often result in broad-topic queries. Since many search engines retrieve the web documents based on the text similarity and link structures, after a broad-topic query is submitted, it's likely for the search engine to form a search result set with thousands, even up to millions of web documents. A particular issue arises: How to effectively organize such a large number of retrieved web documents for a broad-topic user query on a search engine according to their relevance, and provide the users with the information they are looking for? It is often the case that for some broad-topic queries, the documents for the popular topic tend to dominate the result set. Under this circumstance, web documents ranking algorithms such as HITS are unable to solve this problem. The authoritative documents returned to users by HITS may represent only the most popular topic. Documents related to under-represented topics have less chance to be returned to the users while perhaps they are what the users are expecting. To overcome this problem, before ranking the web documents and presenting them to the user, it's necessary to group the retrieved web documents first, then return to the users the ranked documents for each group according to their relevance.

The hyperlink structure of the World Wide Web provides us with rich information on web communities. It contains a lot of latent human annotation of the web society. This motivates us to cluster the web documents by partitioning the web link graph. The web documents retrieved by the search engine are a mixture of high and low qualities written by individuals of different backgrounds. To reduce the influence of the low quality web documents and modulate other documents based on their relevance during the partitioning, the text information needs to be incorporated into the link graph as a factor of edge weight. The co-citation is another relevance measure between two web documents based on how many other web documents create hyperlinks to both of them. To further improve the quality of the clustering results, combining the co-citation information into the link graph is also necessary.

3. Normalized cut, Cheeger constant and spectral graph partitioning.

In graph theory, a set of edges that separates the graph into two disconnected parts is called an edge-cut of the graph. The edge-cut alone may cause unbalanced partition: it tends to cut a portion of a graph with small number of vertices. In the context of graph clustering, this is in general not desirable. Various graph partitioning criteria have been proposed to measure the quality of the partitioning. In this section, we concentrate on *Cheeger constant* and *normalized cut*, and we will also discuss their relationship.

3.1. Cheeger constant and Normalized cut.

To avoid partitioning out a small part of a graph G by using edge-cut alone, many criteria utilize various normalized forms of edge-cut which are generally obtained by dividing the edge-cut by some measure of the size of the partitions. Given a graph $G = (V, E)$ with edge weight matrix W, let S be a subset of V and $\bar{S} = V - S$, the complement of S in V. One normalized form of edge-cut measuring the quality of the graph partition is the *Cheeger constant*. It was studied by J. Cheeger in early 1970s [7].

Define

$$h_G(S) = \frac{\mathrm{cut}(S, \bar{S})}{\min(\mathrm{asso}(S, V), \mathrm{asso}(\bar{S}, V))}$$

where $\mathrm{cut}(S, \bar{S}) = \sum_{i \in S, j \in \bar{S}} W_{ij}, \mathrm{asso}(S, V) = \sum_{i \in S, j \in V} W_{ij}$. The *Cheeger constant* h_G of graph G is defined as:

$$h_G = \min_S h_G(S)$$

Assuming $\mathrm{asso}(S, V) \leq \mathrm{asso}(\bar{S}, V)$, then

$$\begin{aligned} \mathrm{cut}(S, V) &= h_G(S) \cdot \min(\mathrm{asso}(S, V), \mathrm{asso}(\bar{S}, V)) \\ &\geq h_G \cdot \mathrm{asso}(S, V). \end{aligned}$$

The problem of determining the Cheeger constant is equivalent to solving the above graph partition problem.

Slightly modifying the definition of Cheeger constant leads to another measure of partitioning quality: *Normalized cut* whose solution can be easily obtained by solving a generalized eigensystem problem.

The normalized cut is defined as:

$$\mathrm{Ncut}(S,\bar{S}) = \frac{\mathrm{cut}(S,\bar{S})}{\mathrm{asso}(S,V)} + \frac{\mathrm{cut}(S,\bar{S})}{\mathrm{asso}(\bar{S},V)}$$

The partitioning problem is to find the partitions that minimize the normalized cut $\mathrm{Ncut}(S,\bar{S})$. If we let D be the degree matrix of W:

$$D_{ij} = \begin{cases} \sum_{k=1}^{n} W_{ik} & \text{if} \quad i = j \\ 0 & \text{otherwise} \end{cases}$$

the minimization problem associated with normalized cut has been shown to equivalent to the following discrete minimization problem [28]:

$$\frac{y^T(D-W)y}{y^T D y} \tag{1}$$

subject to the constraint that $y_i \in \{1, -b\}$ and $y^T D e = 0$ where e is a vector with all elements equal to 1, and b is positive. Expression (1) is in the form of a Rayleigh quotient. Relaxing the condition $y_i \in \{1, -b\}$ and allowing the elements of y to take any real values, we can easily see that (1) can be minimized by the second smallest eigenvector of the generalized eigensystem:

$$(D - W)y = \lambda D y \tag{2}$$

With the introduction of the normalized-cut criterion, we can easily apply it to measure web graph partitioning problems. It effectively avoids the problem of cutting a small part of the graph.

Since

$$\begin{aligned} h_G(S) &= \frac{\mathrm{cut}(S,\bar{S})}{\min(\mathrm{asso}(S,V), \mathrm{asso}(\bar{S},V))} \\ &\geq \frac{1}{2}(\frac{\mathrm{cut}(S,\mathrm{S})}{\mathrm{asso}(S,V)} + \frac{\mathrm{cut}(S,\mathrm{S})}{\mathrm{asso}(\bar{S},V)}) \\ &= \mathrm{Ncut}(S,\bar{S})/2 \\ \min_S h_G(S) &\geq \frac{1}{2}\min_S \mathrm{Ncut}(S,\bar{S}) \\ h_G &\geq \frac{\mathrm{Ncut}}{2} \end{aligned}$$

On the other hand,

$$\begin{aligned} \mathrm{Ncut}(S,\bar{S}) &= \frac{\mathrm{cut}(S,V)}{\mathrm{asso}(S,V)} + \frac{\mathrm{cut}(S,V)}{\mathrm{asso}(\bar{S},V)} \\ &\geq \frac{\mathrm{cut}(S,V)}{\min(\mathrm{asso}(S,V), \mathrm{asso}(\bar{S},V))} \\ &= h_G(S) \end{aligned}$$

We derived:

$$\mathrm{Ncut} \geq h_G \geq \frac{\mathrm{Ncut}}{2} \tag{3}$$

where $Ncut = \min_S Ncut(S,\bar{S})$.

3.2. Fiedler spectral method. In the past decade, spectral graph partitioning has gotten popularized with the work of Pothen, Simon, and Liou [26]. It's also been studied by many other researchers [2, 3, 11]. This approach is based on technique introduced by Czechoslovakian mathematician Miroslav Fiedler in the 1970s [14, 15]. It uses the eigenvector corresponding to the second smallest eigenvalue of the Laplacian matrix $D - W$ to partition the graph. The second smallest eigenvector is called the *Fiedler Vector* in recognition of Fiedler's pioneering work.

In Section 3.1, we actually use the second smallest eigenvector of the generalized eigensystem

$$(D - W)x = \lambda D x \tag{4}$$

to partition the graph. Here we call the eigenvector corresponding to the second smallest eigenvalue of equation (4) as the scaled Fiedler vector. One interesting property of the scaled Fiedler vector is its algebraic connectivity:[1]

THEOREM 3.1. *Given connected graph $G(V, E)$. Let f be the eigenvector corresponding to the second smallest eigenvalue λ_2 of the generalized eigensystem (4). Define $V_1 = \{v \in V : f_v \leq 0\}$, then the subgraph induced by V_1 is connected. Similarly, define $V_2 = \{v \in V : f_v \geq 0\}$, then the subgraph induced by V_2 is also connected.*

REMARK. The above theorem implies that if the original graph is connected, after partitioning using the scaled Fiedler vector, the subgraphs obtained are also connected. But in general this is not true for K-means method which we will discuss in the next section.

Notice that equation (4) can be transformed to

$$\begin{aligned} D^{1/2}(I - D^{-1/2}WD^{-1/2})D^{1/2}x &= \lambda D x \\ (I - D^{-1/2}WD^{-1/2})D^{1/2}x &= \lambda D^{1/2}x \end{aligned}$$

Let $y = D^{1/2}x$, the above system becomes

$$(I - D^{-1/2}WD^{-1/2})y = \lambda y$$

Therefore, we just need to consider the second largest eigenvector of $D^{-1/2}WD^{-1/2}$.

Here we give an illustrative example of using the scaled Fiedler vector to partition the graph. Figure 1(a) is a block matrix generated by matlab. The size of the first diagonal block is 100-by-100. The second one is 200-by-200. Both have density 0.05. The off-diagonal blocks have density 0.002. This block matrix is symmetric with diagonal elements set to be zero. It's the same as the graph matrix. Figure 1(b) shows the plot of the scaled Fiedler vector of the generalized eigensystem corresponding to this matrix. From the plot, we can easily see that the scaled Fiedler vector cuts the nodes of the matrix into two distinct parts, if we choose zero as the cutting point.

3.3. An clustering algorithm based on normalized-cut. In this algorithm, we partition the weighted graph with the scaled Fiedler vector and normalized cut. After the scaled Fiedler vector f of the generalized eigensystem (2) is available, we check N equally spaced splitting points of f for the best partitioning which we define as the one with the smallest normalized cut. We then decide whether to accept or reject the partitioning based on the smallest normalized cut and predefined threshold. If the smallest normalized cut computed is below the threshold, we accept the

[1] The proof of the theorem follows similar arguments as in Theorem inn [26].

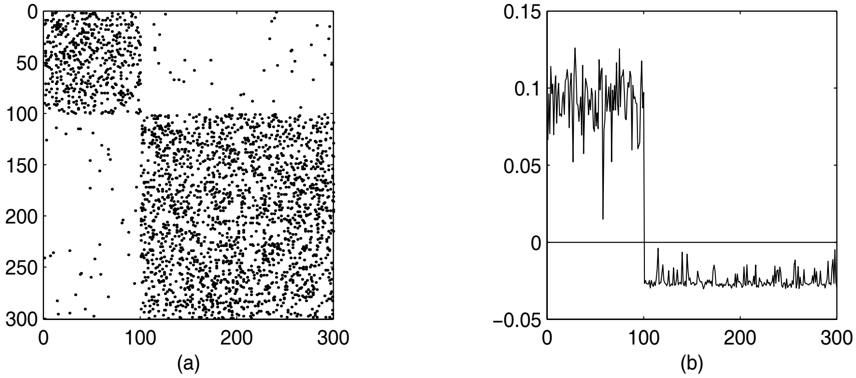


Fig. 1. *The block matrix and the scaled Fiedler vector of corresponding generalized eigensystem. (a)The diagonal blocks have density 0.05. The first diagonal block is 100-by-100 and the second one is 200-by-200. The off-diagonal blocks have density 0.002. (b) is the plot of the scaled Fiedler vector. We can see that the scaled Fiedler vector partitions the nodes of the matrix into two parts at splitting point 0.*

partitioning. If the smallest normalized cut is above the threshold, it means there are more connections between these two subgraphs. We reject the partitioning and consider them as one cluster. This algorithm is first proposed by Shi and Malik in [28] to solve the image segmentation problem.

Given a weighted graph $G = (V, E)$, its weight matrix is W,

1. $d = We$, where $e = (1, 1, \cdots, 1)$. $D = \text{Diag}(d)$.
2. Solve the generalized eigensystem $(D - W)x = \lambda Dx$ for eigenvector f with the second smallest eigenvalue.
3. Check N equally spaced splitting points of f, find the cutting point with the smallest normalized cut.
4. If the normalized cut is below a certain threshold, accept the partition and recursively partition the sub-graphs. Otherwise, stop.

4. Connection with the K-means method.

In this section, we explore the connection of the normalized-cut based clustering method and the popular K-means clustering method. For ease of discussion, we will only consider the case where the weight matrix W satisfies $W_{ij} \in \{0, 1\}$. Similar conclusions can be drawn for the more general cases.

4.1. K-means method.

The general idea of the K-means method is the following: 1) partition the nodes into two arbitrary clusters; 2) for each node re-assign it to the cluster that the node is in some sense closest to; repeat the above two steps until convergence, i.e., until when the cluster membership of no node will change by running through the two steps. One natural and reasonable heuristics for measuring the closeness of a node to a cluster is the following: for a node k, we examine all the nodes that are adjacent to k, if there are more neighbors of k belonging to cluster one, we then re-assign i to cluster one, otherwise we re-assign k to cluster two. It is easy to see that this approach can be generalized to the multiple cluster case: we re-assign i to the cluster in which i has the biggest number of neighbors. If we denote the cluster assignment of k as $x_k \in \{-1, 1\}$, then

$$\sum_{j=1}^{n} W_{kj} x_j$$

is the difference in the numbers of neighbors of k belonging to the two clusters. Therefore, the re-assignment of k can be computed as

$$x_k^{(i)} = \text{sign}(\sum_{j=1}^{n} W_{kj} x_j^{(i-1)})$$

where i denotes the iteration step. Rewriting in matrix-vector format, we have

$$x^{(i)} = \text{sign}(W x^{(i-1)})$$

As will be discussed in more detail later, the above iteration can be considered as a variation of the power method for computing the largest eigenpair of a symmetric matrix [18], the difference being that a different normalization process is used here. Restricting $x_k \in \{-1, 1\}$ corresponds to hard-clustering: a node either belongs to cluster one or cluster two. For soft-clustering, i.e., we can consider $x_k \in [0, 1]$ as the probability of node k belonging to cluster one, and use normalization

$$y^{(i)} = W x^{(i-1)}, \quad x^{(i)} = y^{(i)} / \|y^{(i)}\|_2$$

This corresponds exactly to the power method for W. In the above discussion we only considered the neighbors of a node k. We can also turn the heuristic argument around: for a node k, we examine all the nodes that are *not* adjacent to k, if there are more of these belonging to cluster one, we then re-assign i to cluster two, otherwise we re-assign k to cluster one. These two arguments can certainly be combined, and we obtain the following re-assignment rule

$$x_k^{(i)} = \text{sign}(\sum_{j \sim k}^{n} x_j^{(i-1)} - \alpha \sum_{j ! \sim k}^{n} x_j^{(i-1)}) \tag{5}$$

where $\alpha > 0$ is a balancing factor. Here we also use $j \sim k$ to denote that node j is adjacent to node k and $j! \sim k$ to denote that node j is not adjacent to node k.

4.2. Connection between normalized cut and K-means method. We now return to the normalized-cut method and consider the following generalized eigenvalue problem

$$(D - W)x = \lambda D x \tag{6}$$

Here D is the diagonal matrix formed from the vector $d = W * e$. e is the vector with all entries equal to 1. As discussed in the previous section, for normalized cut we are interested in computing the second smallest generalized eigenvector of (6). Notice that from

$$We = d = De$$

we obtain

$$D^{-1/2} W D^{-1/2} D^{1/2} e = D^{1/2} e$$

and therefore $D^{1/2}e$ is the eigenvector of $D^{-1/2}WD^{-1/2}$ corresponding to its largest eigenvalue which is equal to 1.[2] For the generalized eigenvalue problem (6) we have

$$D^{-1/2}(D-W)D^{-1/2}D^{1/2}x = \lambda D^{1/2}x, \quad D^{-1/2}WD^{-1/2}D^{1/2}x = (1-\lambda)D^{1/2}x$$

Hence we need to compute the second largest eigenvalue of $D^{-1/2}WD^{-1/2}$. Now we take a closer look at the computation of the second largest eigenpair of $D^{-1/2}WD^{-1/2}$. To this end, we first state a well known result from linear algebra.

Proposition 4.1. *Let A be a symmetric matrix, and $\{u_{\max}, \lambda_{\max}\}$ be its unique largest eigenpair. Then the second largest eigenpair of A is the largest eigenpair of the following rank-one correction of A,*

$$\hat{A} = A - \lambda_{\max} u_{\max} u_{\max}^T / u_{\max}^T u_{\max}$$

So in order to compute the second largest eigenpair of $D^{-1/2}WD^{-1/2}$, we need to compute the largest eigenpair of

$$\hat{W} = D^{-1/2}WD^{-1/2} - \frac{D^{1/2}ee^TD^{1/2}}{e^TDe} \tag{7}$$

One approach for computing the largest eigenpair of a symmetric matrix A is the power method [18]:

```
Start with a unit vector $x^{(0)}$. For $i = 1, 2, \ldots$, until the result converges
    [Matrix-vector multiplication] $y^{(i)} = Ax^{(i-1)}$
    [Normalization]$^3$ $x^{(i)} = y^{(i)}/\|y^{(i)}\|_2$
At the end of convergence, an approximate eigenvalue can be
computed as $\hat{\lambda} = (x^{(i)})^T A x^{(i)}$.
```

Now apply the power method to the matrix in (7), and let the k-th entry of $x^{(i)}$ be $x_k^{(i)}$, we have

$$y_k^{(i)} = \sum_{j=1}^{n} \frac{W_{kj}x_j^{(i)}}{\sqrt{d_k d_j}} - \frac{\sqrt{d_k d_j}x_j^{(i)}}{d_s}$$

where $d_s = e^T d$, the sum of the diagonal elements of D. We can rewrite the above equation as

$$y_k^{(i)} = \sum_{j\sim k} \left(\frac{x_j^{(i)}}{\sqrt{d_k d_j}} - \frac{\sqrt{d_k d_j}x_j^{(i)}}{d_s} \right) - \sum_{j!\sim k} \frac{\sqrt{d_k d_j}x_j^{(i)}}{d_s}$$

Now compare the above with the K-means iteration in (5), we notice the following: for the neighbors of node k, normalized-cut method uses the modified weight

$$\frac{1}{\sqrt{d_k d_j}} - \frac{\sqrt{d_k d_j}}{d_s}$$

and for those nodes not adjacent to k, normalized-cut uses

$$\frac{\sqrt{d_k d_j}}{d_s}$$

as the weight. Therefore, normalized-cut can be considered as a variation of the K-means method.

[2] It can be readily proved that all the eigenvalues of $D^{-1/2}WD^{-1/2}$ are at most 1.

5. Similarity metric. Clustering objects into subgroups is usually based on a similarity metric between objects, with the goal that objects within a subgroup are very similar, and objects between different subgroups are less similar. In clustering a graph, similarity between nodes is represented as weight for the edge.

In the web documents clustering problem, we incorporate link structure, text information, and co-citation information into the similarity metric, which form the weight matrix W. The link structure is the dominant factor, and the textual similarity is used to modulate the strength of each hyperlink. To further enhance the link structure, co-citation is also incorporated.

5.1. Hyperlink structure. The link information is obtained directly from the link graph. Given a link graph $G = (V, E)$, which is directed, we define matrix A to be:

$$A_{ij} = \begin{cases} 1 & \text{if} \quad (i,j) \in E \quad \text{or} \quad (j,i) \in E \\ 0 & \text{otherwise} \end{cases}$$

A is an adjacency matrix of the graph. Link structure alone provides us with rich information on the topic. By exploring the link structure, we are able to extract useful information from the web [5, 17, 6, 16, 21, 24, 22]. One of the most popular algorithm to retrieve information from the link structure is credited to Jon Kleinberg. We will briefly discuss his HITS algorithm in Appendix A.

5.2. Textual information. It is known that clustering documents based entirely on text information is not effective in finding distinct topics [29], but including textual information can better cluster the web documents. Moreover, unlike printed literature, web documents reference each other more randomly. This is another reason that the text information is incorporated. One approach to incorporate the textual information is to measure the similarity between user query and the anchor text (text between ⟨A HREF=...⟩ and ⟨/A⟩) as in [23]. We experimented with this approach and found it did not work as effective in our data sets as we had expected.

Here we use a new approach that (a) utilizes the entire text of a web document, not just the anchor text; (b) measures the textual similarity between two web documents, instead of between the user query and the web document. This approach in fact is the standard similarity between documents in information retrieval. The similarities between documents form the similarity matrix S.

5.3. Co-citation patterns. co-citation is another metric to measure the relevance of two web documents. If there are many documents pointing to both of them, then these two documents are likely to address the similar issue. The co-citation C_{ij} of documents i and j is the number of web documents pointing to both i and j. The co-citation matrix C is easily obtained from the link graph.

Incorporating the above information into the similarity metric, we form the weight matrix of the graph:

$$W = \alpha \frac{A \otimes S}{\|A \otimes S\|_2} + (1 - \alpha) \frac{C}{\|C\|_2} \tag{8}$$

where $\otimes$ means multiplication element by element, $\|C\|_2 = \sqrt{\sum_{i,j} C_{ij}^2}$, and S is similarity matrix. The meaning of this notation applies to the rest of the paper. α is a real value between 0 and 1. The algorithms we introduced will be applied on matrix W.

6. Experiments. We tested the algorithm on the link graphs of queries *amazon, apple* and *star*. We choose query terms *amazon* and *star* because they both have multiple distinct meanings. *Amazon* has at least three meanings. One is related to amazon.com, one of the largest on-line shopping web sites. Another is the famous rain forest in South America. The third is the name of ancient female warriors from Alecto, a female ruled monarchy. For the query *star*, we can think about a natural luminous body visible in the sky, a movie star, a famous athlete and the movie *Star Wars*. The purpose of choosing query term *apple* is a little different. Mentioned apple and we will associate it with a kind of fruit or the apple computer. By choosing *apple*, we hope our algorithms can find a cluster with a topic different from the previous two.

We then apply our clustering algorithm on the data sets. Since each cluster often has a large number of web documents, we choose only the most important web documents among a cluster. The most important or authoritative web documents are determined using the HITS algorithm introduced in Appendix A.

We also compare the results obtained from our algorithm to the results from a simple K-means based algorithm. It shows the normalized-cut based technique performs better that the K-means based algorithm. We give the K-means based algorithm below:

```
Initialize $x_i = \pm 1$ randomly, $i = 1 \ldots |V|$
$W = \alpha \frac{A \otimes S}{\|A \otimes S\|_2} + (1-\alpha) \frac{C}{\|C\|_2}$
do
    for $i = 1 \ldots |V|$
        $x_i = \sum_j W_{ij} x_j$
        if $(x_i \geq 0)$
            $x_i = 1$
        else
            $x_i = -1$
        end
    end
until certain stop criteria met
```

FIG. 2. *K-means based algorithm to partition the web graph*

The certain stop criteria can be either the maximal iteration number reached, the result is stable, or the cluster size is below a certain threshold.

6.1. Data preparation. Each row of the link graphs has the format:
doc_ID #_inbound_links #_outbound_links inbound_doc_IDs outbound_doc_IDs
where doc_ID denotes the document ID.

To obtain the link graphs, we first provide a text-based search engine *hotbot* with query terms. *Hotbot* returns as the query result a list of URLs with highest ranked web documents. We limit the number of returned URLs to be 120 which form the root set. By limiting the number of returned URLs to be 120, we can keep the overall data set within a reasonable size. Then we expand the root set by including all web documents that point to a web document in the root set and those pointed to by a web document in the root set. This is a level one expansion. The link graph can be viewed as a directed graph. It's easy to convert it to the adjacency matrix of an undirected graph.

After the full list of URLs is available, we run web crawler to get the text information of these web documents. The text of a document is obtained using a web crawler that we write in *Perl*. To accommodate the vast differences in web document lengths, we limit the length of each document to be 500 words. The rest of the document is discarded if it has more than 500 words. Stopwords (such as *I, is, a*, etc.) are discarded using a standard list. Words are stemmed using Porter stemming[25], so both of the words *linking, linked* are stemmed to the same root *link*. After the text information of all documents is available, we represent each web document as a vector in the vector space model of IR(Information Retrieval) [27]. We then compute the *similarity* (or *relevance*) between them. The higher the similarity, the more likely the two documents deal with the same topic. For each element of the vector we use the standard tf.idf weighting: $tf(i,j) * idf(i)$. $tf(i,j)$ is the Term Frequency of word i in document j, representing the number of occurrences of word i in document j. *idf* is the Inverse Document Frequency corresponding to word i, defined as

$$\mathrm{idf}(i) = \log(\frac{\text{no. of total docs}}{\text{no. of docs containing word } i})$$

Some words appear too frequently in many documents. We assume these words are not very useful to identify the documents. Inverse Document Frequency can effectively decrease the influence of these words.

Since the term vector lengths of the documents vary, in our experiment, we use cosine normalization in computing similarity. That is, if x and y are vectors of two documents d_1 and d_2, then the similarity between d_1 and d_2 is:

$$S(d_1, d_2) = S(d_2, d_1) = \frac{\sum_i x_i y_i}{\|x\|_2 \|y\|_2}$$

where $\|x\|_2 = \sqrt{\sum_i x_i^2}$.

At this point, we have the necessary data for the experiment. In our experiment, we set α in expression (8) to be 0.5.

6.2. Clustering results. We apply our algorithm on the data sets of three query terms with the threshold of normalized cut set to 0.06, then run the HITS algorithm on each cluster obtained. The top authorities of each significant clusters are listed. By significant we mean the size of the cluster is not too small.

The order of the clusters has no special meanings. It's merely the sequence by which we process the clusters.

6.2.1. Query term: ***amazon*.** There are a total of 2294 URLs in this data set. Applying our algorithm, we obtain following significant clusters. The clusters with small size are not counted. Some small clusters exist as the result of the algorithm because they form the connected components of the link graph. It shows that our algorithm has the capability to find the cluster of small size but tightly connected.

Cluster 1:

- *http://www.amazon.com/*
- *http://www.amazon.co.uk/*
- *http://www.amazon.de*

These three web documents are the home pages of amazon.com, one of the most famous shopping companies in the world. The first website is located in the USA, the second in the UK and third in Germany. It's perfect to cluster them together. We list only three authorities here because the rest documents ranked among the top 10 have very very low authority weights compared with the first three, so we don't list them here.

Cluster 2:

- *http://www.amazoncity.com/*
- *http://www.amazoncityradio.com/*
- *http://www.amazoncity.com/spiderwoman/*
- *http://radio.amazoncity.com/*
- *http://www.wired.com/news/news/culture/story/6751.html*

This cluster is about female issue.

Cluster 3:

- *http://www.amazon.org/*
- *http://www.amazonfembks.com/*
- *http://www.igc.apc.org/women/bookstores/*
- *http://www.teleport.com/ rocky/queer.shtml*
- *http://www.advocate.com/html/gaylinks/resources.html*

The topic of this cluster is on female related issues: bi-sexuality and female books.

Cluster 4:

- *http://sothebys.amazon.com/exec/varzea/tg/special-sales/-/22822/-/8253/ref=gw_m_col_2/*
- *http://sothebys.amazon.com/exec/varzea/subst/home/sothebys.html/ref=gw_m_col_au_2/*
- *http://sothebys.amazon.com/exec/varzea/tg/special-sales/-/22822/-/8253/ref=gw_m_col_au_1/*
- *http://s1.amazon.com/exec/varzea/subst/home/home.html/ref=gw_auc_1/*
- *http://sothebys.amazon.com/exec/varzea/subst/home/sothebys.html/ref=gw_m_ln_br_so_2/*

All five authorities listed here are web documents of a large on-line auction company formed by Sothebys and amazon.com. It's not clustered into Cluster 1 because there are relatively few links between them.

Cluster 5:

- *http://www.swalliance.com/*
- *http://timeline.echostation.com*
- *http://www.echostation.com:8080/˜1*
- *http://downtime.echostation.com*
- *http://rpg.echostation.com/*

The topic of this cluster is about *Star Wars*, surprisingly. But there are a total of 68 documents on this topic, most created by *Star Wars* fans. Some are on-line shopping companies selling goods related to the movie *Star Wars*.

There are two clusters, but for each of them, the web documents are from the same site. These two clusters are:

Cluster 6:

- *http://misc.langenberg.com/*
- *http://cooking.langenberg.com/*
- *http://shipping.langenberg.com/*
- *http://money.langenberg.com/*
- *http://weather.langenberg.com/*

and

Cluster 7:

- *http://www.latingrocer2.com*

- *http://www.latingrocer.com*
- *http://www.latingrocer.com/Pages/customer.html*
- *http://www.latingrocer.com/Pages/privacy.html*
- *http://www.latingrocer.com/Pages/contact.html*

Cluster 8:

- *http://www.internext.com.br/ariau*

This cluster has only one authority. Other web documents in this cluster all point to it. It's a website of a hotel in the Amazon River valley.

After applying the K-means based algorithm on the data, we also get several clusters. Cluster 1 is separated into three clusters, that is, the websites in the three different countries form three clusters under this algorithm. Second, there are more clusters with small size than the result from our algorithm. Many of them should belong to larger clusters, but were partitioned incorrectly.

By adjusting the threshold, the algorithm can group or separate the following web documents:

- *http://www.amazon.com/*
- *http://www.amazon.co.uk/* and
- *http://www.amazon.de*

while the K-means based algorithm can only partition them into different clusters. This verifies that our algorithm is not only more flexible than the other, but also clusters more reasonably.

From the clusters we obtained, we found that, unlike what we had expected, no cluster has a focused topic on the rain forest while Cluster 8 does have something to do with the rain forest in Brazil. Checking the entire data set, we only found a couple of web documents that mention the rain forest. They don't form a cluster with significant size. If we return to *hotbot* and enter the query *amazon*, the web sites presented are dominantly related to amazon.com. The documents about the rain forest are not among the highest-ranked list returned by *hotbot*. It means the number of web documents on amazon the rain forest is small, which makes the web documents about this topic have low rank weight. That's the reason that we can't have a cluster on this topic.

As for the third meaning of amazon, although female warrior dose not directly appear as a distinct topic in any cluster, Cluster 3 focused on female topics, or even the bi-sexual issue. By examining the content of these documents, we are sure that these issues are the extent of the original meaning of amazon as female warriors.

There are clusters with small size that apparently have nothing to do with *amazon*, such as the clusters about websites langenberg.com and latingrocer.com. They are explored as separate clusters because the web documents on the same site point to each other, raising the importance themselves. To avoid such situation, before applying any clustering algorithm, we can coarsen the link graph first, so that the web documents from the same site collapse to one node in the graph.

6.2.2. Query term: *star*. We have 3504 URLs for this query. Setting the threshold to be 0.06, we run our algorithm on the data set of query term *star*. The authorities of each cluster are listed below:

Cluster 1:

- *http://www.starwars.com/*
- *http://www.lucasarts.com/*
- *http://www.sirstevesguide.com/*
- *http://www.jediknight.net/*

- *http://www.surfthe.net/swma/*

This cluster is focused on *star war*.

Cluster 2:

- *http://www.kcstar.com/*
- *http://www.dailystarnews.com/*
- *http://www.kansascity.com/*
- *http://www.starbulletin.com/*
- *http://www.trib.com/*

This cluster includes the web documents of some news media with the word *star* as part of their names.

Cluster 3:

- *http://www.weatherpoint.com/starnews*
- *http://www.starnews.com/digest/sports.html*
- *http://www.starnews.com/digest/citystate.html*
- *http://www.indy.com*
- *http://speednet.starnews.com/*

The top authorities are all web documents of starnews.com in this cluster.

Cluster 4:

- *http://www.state.mn.us/mainmenu.html*
- *http://www.mda.state.mn.us/*
- *http://www.doli.state.mn.us/*
- *http://www.legalethics.com/pa/states/state/mn.htm*
- *http://www.exploreminnesota.com*

This cluster's topic is the state of Minnesota. The reason that Minnesota is a topic of the cluster under query *star* is because the official State of Minnesota web site is called *North Star*, which is named second among all government web sites.

Cluster 5:

- *http://www.star-telegram.com/*
- *http://www.dfw.com/*
- *http://www.virtualtexan.com/*
- *http://marketplace.dfw.com*
- *http://www.star-telegram.com/advertise/vshops/*

A cluster of star-telegram.com located in Texas.

Cluster 6:

- *http://www.starpages.net/*

There is only one authority in this cluster. This authority is a web site introducing preeminent persons in various fields, such as movie star, sport stars, etc. This cluster is what we have expected to be obtained after partitioning.

Cluster 7:

- *http://www.aavso.org/*
- *http://www.astro.wisc.edu/ dolan/constellations/*
- *http://ourworld.compuserve.com/homepages/rawhide_home_page*
- *http://adc.gsfc.nasa.gov/adc/adc_amateurs.html*
- *http://heasarc.gsfc.nasa.gov/docs/www_info/webstars.html*

This cluster talks about space and astronomy. It's also what we have expected. Now All three meanings of *star* are found to be topics of separate clusters.

Clusters 2, 3, 5 are all web documents of news media. They are partitioned to be different clusters since there is no link among the three clusters.

Using the K-means based algorithm, we obtained many more clusters than obtained by our algorithm. The same as the result of the query *amazon* under the K-means based algorithm, many of them are not really clusters with focused topics. On the other hand, Clusters 2 and 4 are grouped together, although there is no link between them. This is because the K-means based algorithm is not global. It's easy to be trapped to a local minimum while the first algorithm avoids this situation by using the scaled Fiedler vector to partition.

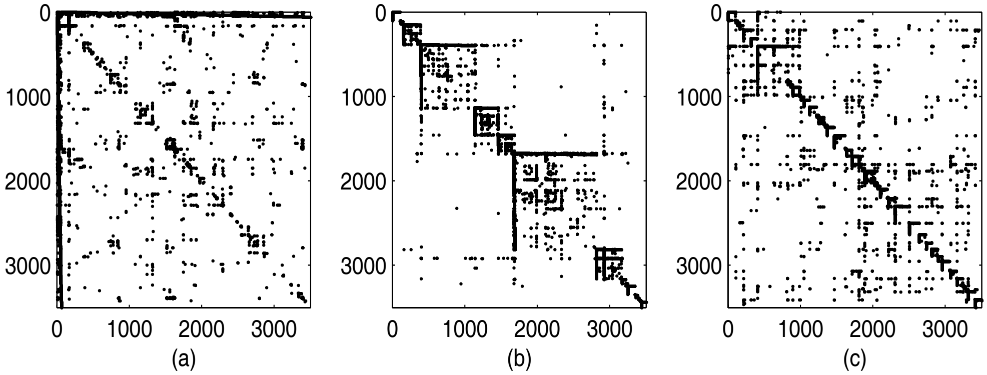


FIG. 3. *Clustering result of query star. (a)Weighted link graph of star. (b)Clustering result using our algorithm. The threshold is 0.06. (c)Clustering result using K-means based algorithm.*

In Figure 3, graph (a) is the original weighted link graph of data for the query *star*. (b) is the re-ordered graph after clustering using our algorithm. (c) is the re-ordered graph after clustering using the K-means based algorithm. In graph (b) and (c), each diagonal block corresponds to a resulting cluster. Comparing graphs (b) and (c), we can see that, in graph (b), the off-diagonal blocks are much sparser than off-diagonal blocks in graph (c). This means our algorithm creates clusters with much fewer connections between them than the K-means based algorithm does.

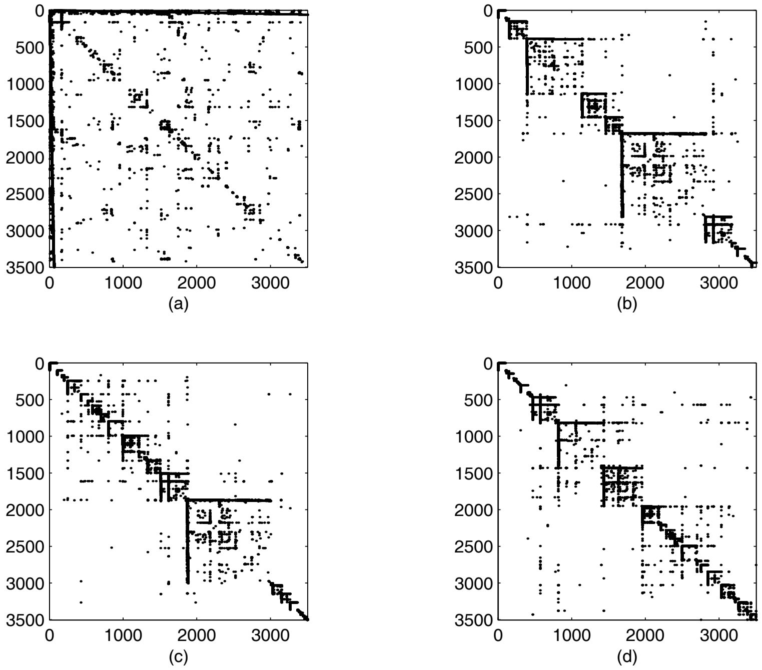
In Figure 4, graph (a) is the original weighted link graph of data for the query *star*. (b) is the re-ordered graph after clustering using our algorithm with the threshold equal to 0.06. (c) is the re-ordered graph after clustering using our algorithm with the threshold equal to 0.1. (d) is the re-ordered graph after clustering using our algorithm with the threshold equal to 0.2. The graph clearly shows that by increasing the threshold, we can obtain more clusters with the size of each cluster smaller. In this way, we can control the clustering result by changing the value of the threshold in the algorithm. It makes this algorithm more flexible than the K-means based one.

6.2.3. Query term: *apple.* In this data set, there are 2757 URLs returned by the search engine. The threshold is still 0.06. The authorities of each cluster are listed below after running the algorithm on the data of the query term *apple*:

Cluster 1:

- *http://www.apple.com/*
- *http://www.apple.com/support/*
- *http://www.apple.com/education/*
- *http://www.apple.com/quicktime/*
- *http://www.apple.com/hotnews/*

Here all top authorities are from the same web site: *www.apple.com.* This cluster is dominant in this query. Most URLs belong to it. After running the HITS algorithm



(a)

(b)

(c)

(d)

Fig. 4. *Clustering result of query star using our algorithm. (a)Weighted link graph of star. (b)Clustering result with threshold = 0.06 (c)Clustering result with threshold = 0.1 (d)Clustering result with threshold = 0.2*

with the site information considered, that is, the URLs from the same web site are collapsed to one node, we obtain different top authorities:

- *http://www.apple.com/powermac/server/*
- *http://www.claris.com/*
- *http://www.apple.ru/hardware/displays*
- *http://www.cs.brandeis.edu/~xray/oldmac.html*
- *http://www.next.com/*

The second URL in this cluster is the website of a computer software company. It produces software used for the Macintosh. The other four URLs are all about the apple computer. This list of authorities provides more useful information than the previous one does.

Cluster 2:

- *http://www.yabloko.ru/*
- *http://www.cityline.ru/politika/*
- *http://www.russ.ru/*
- *http://www.forum.msk.ru/*
- *http://www.novayagazeta.ru*

All web documents in this cluster are written in Russian.

Cluster 3:

- *http://www.michiganapples.com/*

This cluster has only one authority. It's related to apple, the fruit.

The following cluster is formed in this query simply because its name happens to contain the query term *apple*:

Cluster 4:

- *http://www.ci.apple-valley.mn.us/*

which is the website of the city of Apple Valley, MN.

Cluster 5:

- *http://www.valleyweb.com/*

This is the website of Annapolis Valley in Canada where there is the Apple Blossom Festival in the spring celebrating the traditions and agricultural heritage.

There are not many clusters formed by the algorithm, nor do we find interesting topics which are beyond our expectation.

The performance of the K-means algorithm is mixed. On one hand it doesn't form a cluster as large as Cluster 1. For example, it forms a new cluster around *http://www.france.euro.apple.com/* which belongs to Cluster 1 obtained with our algorithm. On the other hand, it groups totally different web documents together. For instance, the top authorities of one cluster are:

- *http://www.jokewallpaper.com/*
- *http://the-tech.mit.edu/Macmade/*
- *http://www.geocities.com/SiliconValley/Vista/7184/guitool.html*

But *http://www.jokewallpaper.com/* and *http://the-tech.mit.edu/Macmade/* mention totally different things.

7. Further discussion.

7.1. Importance of text information. Intuitively, The links in the webgraph can not be regarded as equally important, especially many web links are created randomly. The text similarity between two web documents provide us with a useful metric to address this issue. Without the similarity as the measure of the link strength, that is, if we form the weight matrix as:

$$W = \alpha \frac{A}{\|A\|_2} + (1-\alpha)\frac{C}{\|C\|_2},$$

then we unduly raise the strength of some links which connect two web documents with little in common. When applied to the data set *amazon*, our algorithm groups Clusters 1, 3 and 4 into one single cluster, using the same threshold as the stopping criterion, namely 0.06 in our previous experiments. Cluster 3 addresses the female issues and Clusters 1 and 4 are related to *amazon.com*. Apparently this clustering result is not a good one. This result justifies our choice of incorporating the text information into the weight metric.

Recall that our weight matrix is formed as:

$$W = \alpha \frac{A \otimes S}{\|A \otimes S\|_2} + (1-\alpha)\frac{C}{\|C\|_2}$$

If the value of α changes, the weight matrix will change accordingly. Then we can set different thresholds in the algorithm to have three authoritative web documents in Cluster 1 of the data set *amazon* separated into different clusters. Table 1 lists the α value and the corresponding threshold T such that these three web documents are separated by the algorithm. From the (α, T) pairs in the table, we see clearly that when the value of α increases, to partition these three web documents into different clusters, the threshold decreases monotonically.

α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
T	1.1	1.0	0.8	0.7	0.6	0.5	0.5	0.4	0.3	0.3

TABLE 1
(α, T) pairs that separate three authoritative web documents in Cluster 1 of query **amazon**

7.2. Scaled Fiedler vector. Now we use a concrete example to show that partitioning a graph with the scaled Fiedler vector and normalized cut is feasible. Here we use the subgraph which includes URLs in Clusters 1, 3 and 4 of the data set *amazon*. There are a total of 1839 web documents in this subgraph. From the discussion above, we know that this subgraph contains multiple topics and can be produced by using the weight matrix without the text information. Figure 5(top) is the sorted scaled Fiedler vector of the subgraph. The horizontal axis is the node index and the vertical axis is the corresponding entry of the scaled Fiedler vector. Figure 5(bottom) is the node index and corresponding normalized cut value. We see that the sorted scaled Fiedler vector behaves like step function. Each segment takes nearly constant value. The jumping points can be used as the cutting points because they correspond to small normalized cut values. See Figure 5(bottom) for their values. After we have determined the threshold, we can obtain the cutting points, and partition the graph into clusters. If we set the threshold to be 0.06, we obtain the following cutting points:

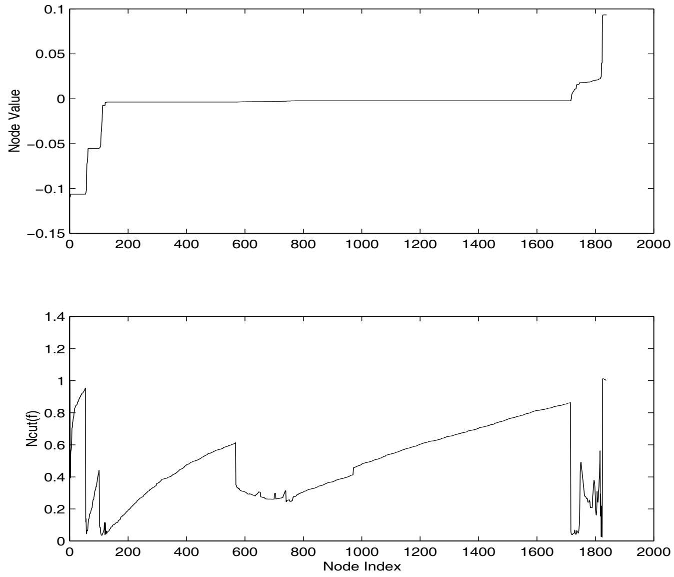


FIG. 5. *The sorted scaled Fiedler vector value(top) and corresponding normalized cut(bottom)*

(57 58)
(106 107 108 109 110 111 112 113 121 124 125 127 128 129 130 131)
(1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728 1731 1732 1733 1734 1735 1739 1740 1741 1742)
(1820 1823)
The numbers in the same pair of () are almost consecutive, we can choose any one of

them as the cutting point. This won't affect the resulting clusters significantly since these nodes correspond to URLs which are not important, i.e., not good authorities nor good hubs.

If we enlarge Figure 5 to include only the first 200 nodes, we can see the result more clearly Figure 7.2.

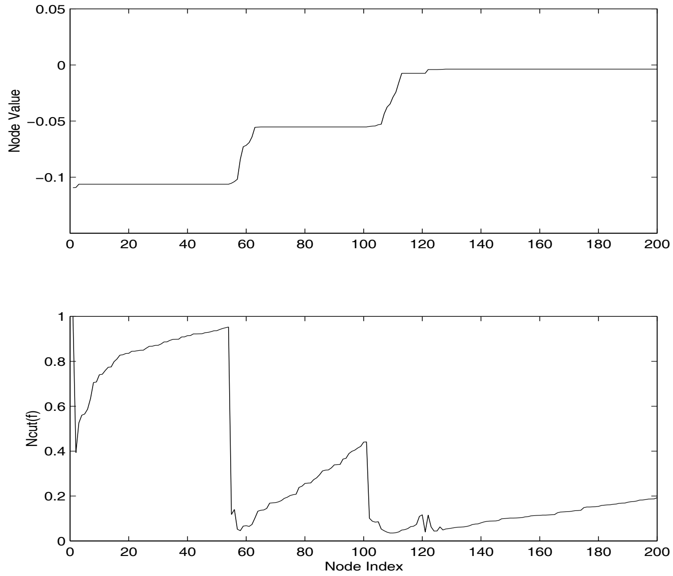


FIG. 6. *The Sorted scaled Fiedler vector value(top) and corresponding normalized cut(bottom) for the first 200 nodes*

For convenience, we choose the cutting points which correspond to the smallest normalized cut in each group: 58 110 1721 1820. This gives us 5 groups:

1. 1 – 57: with top authority *amazon.org*
2. 58 – 109: with top authority *fembks.com*
3. 110 – 1720: corresponds to Cluster 1 of *amazon*
4. 1721 – 1819: corresponds to Cluster 4 of *amazon*
5. 1820 – 1839: with top authority *ethnobotany.org*

Groups 1 and 2 correspond to Cluster 3. They are separated here because we compute the normalized cut value at point 58 with respect to the whole subgraph of 1839 nodes. The value is below the threshold 0.06. While in our algorithm we first partition this subgraph and obtain a smaller subgraph with nodes from 1 to 109, then continue to partition this subgraph. At this point, the smallest normalized cut value is still obtained at node 58, but with the value of 0.0715 which is above the threshold. The partition is rejected. So groups 1 and 2 form one cluster in our algorithm. The result here indicates that our algorithm may be improved by checking the points with the normalized cut values below the threshold and partition the graph into several subgraphs at the same time.

Our algorithm dose find the cluster corresponding to group 5. Since its size is too small, we didn't list it in our results.

7.3. Robustness. Figure 7 is the plot of the α value in Expression (8) and the corresponding average normalized cut value. The graph shows that for the query term *amazon*(top), the smallest average normalized cut value corresponds to $\alpha = 0.7$ while

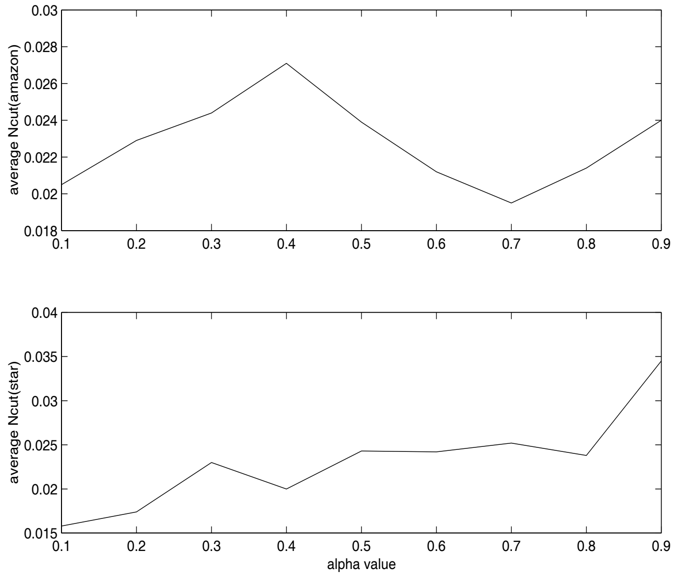


FIG. 7. *α value and corresponding average normalized cut (top):amazon, (bottom):star*

for the query term *star*(bottom), $\alpha = 0.1$ leads to the smallest average normalized cut value. From this figure, We know that we cannot choose an α value or a range of α values that can minimize the average normalized cut in all cases.

For the query *amazon*, $\alpha = 0.7$ gives the best average normalized cut. We choose this α value to test the average normalized cut under different threshold. In case of the query *star*, it's $\alpha = 0.1$ that gives the best average normalized cut. But we can't make α too small, since doing so will lower the importance of text similarity. So we choose $\alpha = 0.4$ to test because it's a local minimum. From Figure 8 we see that, as the threshold increases, the average normalized cut value increases, too. The relationship between them are almost linear.

8. Concluding remarks. In this paper, we present an algorithm to solve the web document clustering problem. We treat the problem as graph partitioning, measure the partitioning result using the *normalized cut* criterion which is first proposed in the field of image segmentation. Combining normalized cut and the scaled Fiedler vector together, this approach forms a global, unbiased algorithm which can effectively extract different topics contained in the webgraph. Compared with the K-means based algorithm, it avoids creating clusters with small size by controlling the threshold on the value of normalized cut. The clusters obtained have high similarity within clusters and dissimilarity between clusters. In our experiment, after choosing suitable threshold, we obtain the clusters, each with distinct topics.

Acknowledgment

We thank Daniel Denton of the CITG group, Lawrence Berkeley National Laboratory for the proofreading of this paper.

Appendix A

Here we briefly introduce Kleinberg's HITS algorithm to find the authorities and

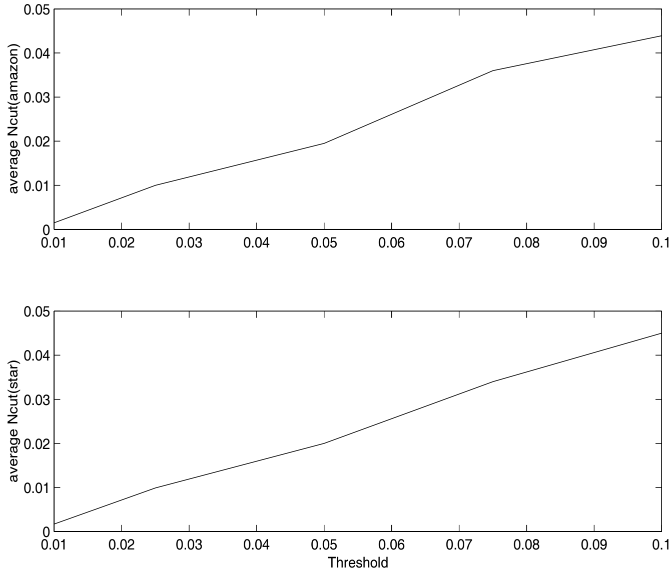


FIG. 8. *Threshold and corresponding average normalized cut (top):amazon* $\alpha = 0.7$*, (bottom):star,*$\alpha = 0.4$

hubs of each cluster we obtained. Kleinberg[20] defines the *authorities* as the most relevant documents for the topic. The *Hubs* are defined as the web documents which link to many related authorities. They implicitly represent an "endorsement" of the authorities they point to. The authority and hub information can be retrieved based entirely on the link structure information. Since a good *authority* is pointed to by many good *hubs* and a good *hub* points to many good authorities, such mutually reinforcing relationship can be represented as:

$$x_p = \sum_{q:(q,p)\in E} y_q \tag{9}$$

$$y_p = \sum_{q:(p,q)\in E} x_q \tag{10}$$

where x_p is the authority weight of web document x and y_p is the hub weight. E is the set of links(edges). Iteratively update the authority and hub weights of every web document, using equations (9) and (10), and sort the web documents in decreasing order according to their authority and hub weights, respectively, we can obtain the authorities and hubs of the topic. Many other issues need to be taken into consideration, such as the hyperlinks from the same web site, etc.

REFERENCES

[1] P. G. Anick. Adapting a full-text information retrieval system to compute the troubleshooting domain. *Proc. of ACM SIGIR '94*, pages 349–358, 1994.

[2] S. T. Barnard, A. Pothen, and H. D. Simon. A spectral algorithm for envelope reduction of sparse matrices. *Proc. Supercomputing '93, IEEE*, pages 493–502, 1993.

[3] M. W. Berry, B. Hendrickson, and P. Raghavan. Sparse matrix reordering schemes for browsing hypertext. *Lectures in Applied Mathematics*, 32, 1996.

[4] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public web search engines. *Proc. of the 7th World-Wide Web Conference (WWW7)*, 1998.

[5] S. Chakrabarti, B. E. Dom, and J. M. Kleinberg. Mining the link structure of the world wide web. Feb 1999.

[6] S. Chakrabarti, B. E. Dom, P. Raghavan, S. Rajagopalan, D. Gibson, and J. Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. *Computer Networks and ISDN Systems*, 30(1-7):65–74, 1998.

[7] J. Cheeger. *A Lowere Bound for the Smallest Eigenvalue of the Laplacian, Problems in Analysis.* Princeton University Press, 1970.

[8] F.R.K. Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.

[9] Inktomi Corporation. *Inktomi webmap press release.* http://www.inktomi.com/webmap, Jan. 2000.

[10] W. B. Croft, R. Cook, and D. Wilder. Providing government information on the internet: Experience with 'thomas'. *Technical Report, University of Mass.*, 95-45.

[11] R. V. Driessche and D. Roose. An improved spectral bisection algorithm and its application to dynamic load balancing. *Parallel Computing*, 21:29–48, 1995.

[12] E. N. Efthimiadis. A user-centered evaluation of ranking algorithms for interactive query expansion. *Proc. of ACM SIGIR '93*, pages 146–159, 1993.

[13] B. Everitt. *Cluster Analysis.* Edward Arnold, 1993.

[14] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Math J.*, 23:298–305, 1973.

[15] M. Fiedler. A property of eigenvectors of non-negative symmetric matrices and its application to graph theory. *Czechoslovak Math Journal*, 25:619–633, 1975.

[16] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. *KDD*, 2000.

[17] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *Proceedings of the 9th ACM Conference on Hypertext and Hypermedia (HYPER-98)*, pages 225–234, New York, June 20–24 1998. ACM Press.

[18] G. Golub and C. V. Loan. *Matrix Computations, 2nd edition.* Johns Hopkins, Baltimore, 1989.

[19] A. D. Gordon. *Classification.* Chapman and Hall, 1981.

[20] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Proc. 9th ACM-SIAM Symposium on Discrete Algorithm*, pages 668–677, 25–27 January 1998.

[21] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Extracting large-scale knowledge bases from the web. *Proc. of the 25th VLDB Conference*, 1999.

[22] R. R. Larson. Bibliometrics of the world wide web: an exploratory analysis of the intellectual structures of cyberspace. *Proc. SIGIR'96*, pages 71–78, 1996.

[23] Yanhong Li. Towards a qualitative search engine. *IEEE Internet Computing*, pages 2–7, July-August 1998.

[24] P. Pirolli, J. Pitkow, and R. Rao. Silk from a sow's ear: Extracting usable structures from the web. *Proc. SIGCHI'96*, pages 118–125, 1996.

[25] M. F. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980.

[26] A. Pothen, H. D. Simon, and K. P. Liou. Partitioning sparse matrices with egenvectors of graph. *SIAM Journal of Matrix Anal. Appl.*, 11:430–452, July 1990.

[27] C. J. V. RijsBergen. *Information Retrieval.* Butterworths, 1979.

[28] J. Shi and J. Malik. Normalized cuts and image segmentation. *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, June 1977.

[29] P. Willett. Recent trends in hierarchical document clustering. *Information Processing and Management*, 24:577–597, 1988.