Web Site Topic-Hierarchy Generation Based on Link Structure

ARTICLE *in* JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE AND TECHNOLOGY · MARCH 2009

Impact Factor: 1.85 · DOI: 10.1002/asi.20990 · Source: DBLP

2 AUTHORS, INCLUDING:

Christopher C. Yang
Drexel University
218 PUBLICATIONS **1,678** CITATIONS

Web Site Topic-Hierarchy Generation Based on Link Structure

Christopher C. Yang
College of Information Science and Technology, Drexel University, 3141 Chestnut Street, Philadelphia, PA, 19104. E-mail: chris.yang@ischool.drexel.edu

Nan Liu
Digital Library Laboratory, William M. W. Mong Engineering Building, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong. E-mail: nliu@se.cuhk.edu.hk

Navigating through hyperlinks within a Web site to look for information from one of its Web pages without the support of a site map can be inefficient and ineffective. Although the content of a Web site is usually organized with an inherent structure like a topic hierarchy, which is a directed tree rooted at a Web site's homepage whose vertices and edges correspond to Web pages and hyperlinks, such a topic hierarchy is not always available to the user. In this work, we studied the problem of automatic generation of Web sites' topic hierarchies. We modeled a Web site's link structure as a weighted directed graph and proposed methods for estimating edge weights based on eight types of features and three learning algorithms, namely decision trees, naïve Bayes classifiers, and logistic regression. Three graph algorithms, namely breadth-first search, shortest-path search, and directed minimum-spanning tree, were adapted to generate the topic hierarchy based on the graph model. We have tested the model and algorithms on real Web sites. It is found that the directed minimum-spanning tree algorithm with the decision tree as the weight learning algorithm achieves the highest performance with an average accuracy of 91.9%.

Introduction

Users looking for information on the Web frequently need to navigate within a Web site to locate particular Web pages satisfying their needs. However, effective Web site exploration is not an easy task. Users may follow hyperlinks to navigate back and forth on the Web site, which is a tedious and time-consuming process. From a user's perspective, such problems can be greatly eased if a table of contents of the Web site is available since a user can quickly narrow down his search scope to particular group(s) of pages about topics relevant to the user's need. Some well-designed Web sites provide sitemaps as a kind of navigational support. A sitemap provides an overview of a Web site's content structure by listing a few of the most important pages that correspond to the major topics of the Web site (Figure 1). By examining the sitemap, a user can quickly locate relevant information without having to visit many pages. Despite the usefulness of sitemaps, they are not provided by most Web sites. Moreover, most existing sitemaps cover only a limited number of pages, which account for only a very small proportion of the Web site. This is because most sitemaps are constructed manually, and thus are very difficult to expand to include hundreds or even thousands of Web pages or to maintain when new Web pages are created. As a result, manually generated sitemaps can only help users to locate Web pages about very broad topics. To look for more specific information, the user can only use the broad topics on the sitemap as an entry point and continue exploring by following hyperlinks.

A Web site's content is usually divided into a set of different topics. Each topic may in turn be further divided into more subtopics. A Web designer uses many different approaches to organize the large amount of information, such as connecting related content through hyperlinks and grouping related files through folders. The goal of this work is to investigate how the underlying content organization of a general Web site can be automatically discovered. In particular, we propose algorithms to automatically generate the topic hierarchy for a given Web site. A topic hierarchy models the topic/subtopic relationships between Web pages.

The hierarchical model is frequently used for the organization of complex bodies of information on Web sites for its simplicity and clarity (Lynch & Horton, 2002). A large Web site usually consists of a number of major topics, each of which may be divided into a number of subtopics. For example, the major topics on the Stanford Infolab's Web site (http://infolab.stanford.edu/) include "Members," "Projects,"

Received March 6, 2008; revised September 26, 2008; accepted September 26, 2008

 DOI: 10.1002/asi.20990

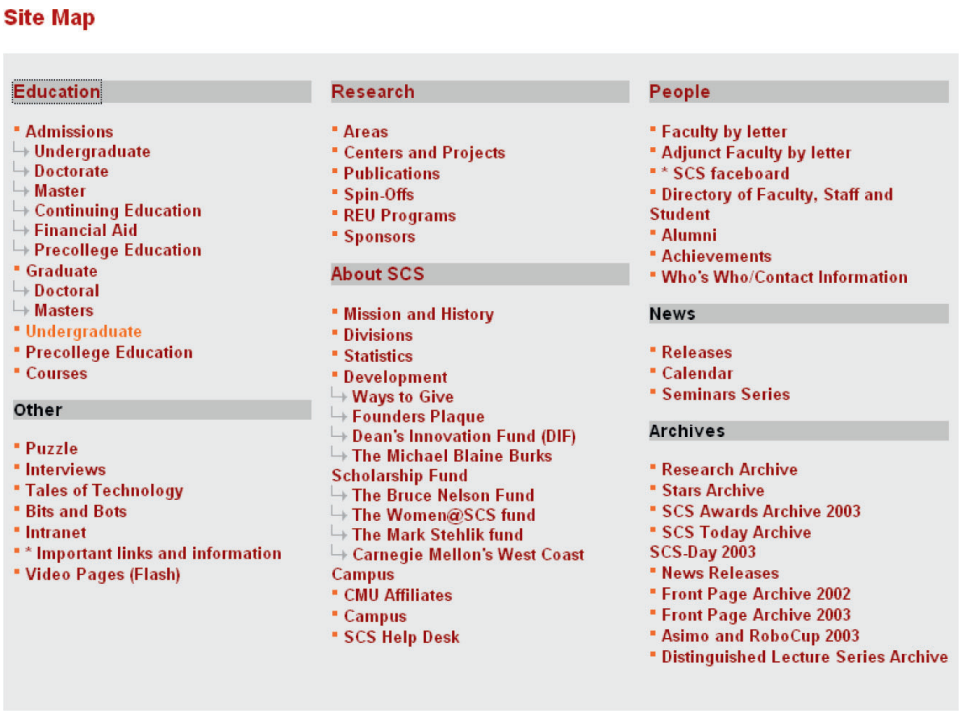

FIG. 1. Sitemap example.

"Classes," and so forth. Within the topic "Projects", each individual project such as "Data Stream" is one particular subtopic. Similarly, subtopics can be further divided into sub-subtopics. In a topic hierarchy, the Web pages are organized into a tree structure so that the topic-subtopic relations between the Web pages are reflected by the parent-child relations in the tree.

Related Work

The problem of generating topic hierarchies is part of more general Web-mining research. Most existing Web-mining research focused on models and techniques for dealing with either the entire World Wide Web or individual Web pages, which can be partially attributed to the success of general Web search engines such as Google. Web-structure mining and Web site mining are two related topics in Web mining.

In Web-structure mining, one discovers knowledge from the connectivity of the Web. Web-structure mining is similar in nature to social-network analysis and citation-network analysis. By analyzing the incoming and outgoing links of Web pages, the importance of Web pages and other information such as authorities and hubs, can be identified (Kleinberg, 1998). Authority pages are Web pages containing a large amount of information relevant to a given user query, while hub pages are the ones providing links to authorities. Identifying the most authoritative pages for a user query is one of the research topics in Web-structure mining (Chakrabarti, 2000). The most notable algorithms for Web-structure mining are the hyperlink induced topic search (HITS; Kleinberg, 1998) and PageRank (Brin & Page, 1998).

A Web site is an organized collection of pages on a specific topic maintained by a single person or group. Web site mining focuses on handling a particular Web site instead of the entire Web or an individual Web page. The link structure of a Web site is quite distinct from the Web. Within one Web site, hyperlinks are mainly created to enable navigation, rather than to recommend pages.

Web site classification is a popular research problem in Web site mining. It classifies Web sites by content, link, and context information. There are three major approaches in Web site classification, namely the superpage approach (Ester, Kriegel, & Schubert, 2002), the keyword vector approach (Ester et al., 2002), and the tree-based approach (Amitay, Carmel, Darlow, Lempel, & Soffer, 2003; Lindemann and Littig, 2006, 2007; Tian, Huang, Gao, Cheng, & Kang, 2003).

The work in Web site mining that is most closely related to our Web site topic-hierarchy generation problem are (a) Web unit mining, logical-domain extraction, and hierarchical topic segmentation, (b) Web thesaurus construction, and (c) navigation trail and sitemap construction.

Mining within a Web site: Web Unit Mining, Logical-Domain Extraction, and Hierarchical Topic Segmentation

Sun and Lim (2003) showed that information about a particular concept is usually not presented on a single Web page but in a set of semantically related pages connected by hyperlinks within a Web site. For example, a professor's homepage would contain links to pages describing his research, curriculum vitae, and education. These pages together represent a professor instance. This leads to the definition of *Web unit*,

DOI: 10.1002/asi

which is a set of Web pages representing the same concept instance. Web-unit mining determines the set of Web pages constituting Web units and classifies these Web units into different concepts in a given ontology. The iterative Web unit mining (iWUM) algorithm carries out Web-unit construction and Web-unit classification iteratively (Sun & Lim). Initially, a set of Web units is generated using the Web site's directory structure. In each iteration, the Web units are classified into a set of domain-specific concept labels and then recombined to form larger Web units. This process continues until there is no difference between the constructed Web units and their labels in the current iteration and the previous iteration. To apply this method to a Web site of a new domain, one needs to have a domain-specific ontology defining the different concepts covered on the site, and also classifiers for classifying Web units into each concept.

The logical-domain extraction problem (Li, Kolak, Vu, & Takano, 2000) is very similar to Web-unit mining. A logical domain is a group of pages that have a specific semantic relation and are related to a syntactic structure in a Web site. A logical domain is essentially a minisite within a large Web site, such as a professor's personal Web site within his or her department's Web site.

Li et al. (2000) assume that there exists an entry page for each logical domain, which is supposed to be the first page to be visited by users navigating the logical domain. Once the entry page was identified, the other pages in the domain can be extracted based on the link structure and folders. The key problem in logical-domain extraction is the identification of the entry pages (Li et al.). Li et al. developed a rule-based approach for identifying entry pages based on Web page metadata including titles, URL, anchors, contents, link structure, and citations. The technique was able to identify logical domain entry pages with high precision. However, because the method relies on very specific rules, the recall was rather low with many entry pages being left out. In addition, the extracted logical domains tend to be small and correspond to very specific topics such as a professor, a class, and so forth. The result is therefore a very long list of different types of logical domains, which is very hard for a user to browse.

Kumar, Punera, and Tomkins (2006) proposed the hierarchical topic segmentation of a Web site into topically cohesive regions that respect the hierarchical structure of the site. The segments should be sufficiently distinct from each other, and the Web pages within a segment should be reasonably pure in topic. A set of cost measures characterizing the benefit accrued by introducing a segmentation of the site based on the topic labels was developed. The cost measures include cohesiveness costs and node-selection costs. A dynamic program was utilized to optimize the costs in order to identify segments. A precision of 94% and a recall of 94% are obtained.

Web Thesaurus Construction

An important feature of the Web's link structure is topic locality (Davison, 2000). *Topic locality* means that the Web pages connected by hyperlinks are more likely to be about the same topic than those that are unconnected. Chen, Liu, Liu, Pu, and Ma (2003) suggest that by replacing each Web page with their anchor texts, a Web site's link structure can be treated as a semantic network, in which words appearing in the anchor text are nodes and semantic relations are edges. Hence, it is possible to construct a thesaurus by using this semantic network information.

Chen et al. (2003) pointed out that there are two functions for a hyperlink in a Web site: one is navigation convenience, and the other is connecting semantically related Web pages together. To use the link structure as a semantic network, the navigational links should be removed while the semantic links should be retained. A link is classified as a navigation link if one of the following occurs:

1. The target page is in a parent folder of the source page.
2. The link is in a navigation bar and the target page is not in a subdirectory of the source page.
3. The link occurs in many Web pages.

Experimental results showed that the proposed rules can identify navigation links with high precision. However, the recall wasn't measured, so the quality of the cleansed link structure is not assured. Moreover, this simple approach doesn't consider the topology of the resulting link structure. After the removal of recognized navigation links, the link structure remains a complicated graph instead of a well-defined hierarchy.

Navigation Trail and Sitemap Construction

Levene and Wheeldon (2001) proposed a Web site navigation engine to build trails of information as are sequences of linked pages. Such sequences of Web pages are relevant to the user query. The best trail algorithm was developed by Wheeldon and Levene (2003) to achieve this purpose. These trails are constructed to satisfy the user information needs that are represented by user queries.

Toolan and Kusmerick (2002) proposed techniques based on Web usage mining to construct personalized sitemaps that are specialized to the interests of individuals. Toolan and Kusmerick developed the mined-path algorithm to construct personalized sitemaps, and benchmarked this algorithm with two other algorithms, the shortest-path algorithm and the popular-path algorithm. The experimental result showed that the mined-path algorithm achieved higher coverage then two benchmarked algorithms. It achieved close to 30% coverage. Since the Web-usage-mining approach relies on the navigation paths of users, many Web pages within a Web site may not have sufficient log data to be included in the sitemap during the construction process.

Comparison of Our Work With the Related Work

In this work, we do not classify Web sites into functional classes. We analyze a Web site and construct a Web site topic

DOI: 10.1002/asi

hierarchy, which is also known as a sitemap. Each node in a Web site topic hierarchy corresponds to a Web page rather than a Web unit, logical domain, or topic segment as investigated previously (Li et al., 2000, Kumar et al., 2006; Sun & Lim, 2003). As a result, users are able to navigate through the Web site topic hierarchy to explore a Web site instead of identifying a number of major units that consists of numerous Web pages. The Web site topic hierarchy is constructed based on the structure of a Web site but does not rely on user queries or user log data. As a result, it will build a general Web site topic hierarchy based on the original development and structure of a Web site rather than constructing an individual trail or sitemap for each user based on their navigation behavior.

Web Site Topic-Hierarchy Generation

Our approach for generating a topic hierarchy is based on analyzing the Web site's link structure, which is typically a densely connected graph as shown in the left part of Figure 2. To formally define topic hierarchy, we first define two types of hyperlinks based on their purpose. When designing a Web site, there are always hyperlinks pointing from a topic to its subtopics, as these are essential links for browsing a Web site. We refer to the hyperlinks connecting a topic to its subtopics as *aggregation links*, which are indicated by the solid arrows in Figure 2. There exist a large number of hyperlinks not used to connect topics to subtopics, but created to provide a quick way of navigating from page to page, which we refer to as *shortcut links*. These are indicated by the dashed arrows in Figure 2. For example, professors' pages may link to classes they teach or projects they are involved in. Also, when developing large sites, a common practice is to use templates for creating groups of similar Web pages, which often contain a navigation bar linking to a few of the most important pages. The existence of these shortcuts poses a challenge in generating a topic hierarchy based on the link structure, which is to distinguish aggregation links from shortcut links. Figure 2 illustrates the original link structure with both aggregation and shortcut links, as well as the topic hierarchy formed by the aggregation links only. It is interesting to notice that there are 10 shortcut links versus only 5 aggregation links in the original link structure.

The topic hierarchy for a Web site is formally defined as a directed tree with all of the following properties:

- It is rooted at the homepage of the Web site.
- Its vertices include all the pages reachable from the Web site's homepage by following a sequence of hyperlinks.
- Each edge in the tree corresponds to an aggregation link pointing from the parent node (topic) to the child node (subtopic).

Graph-Based Algorithms

Our algorithms for generating a topic hierarchy take a Web site's link structure, which is a general directed graph, as input, and extract a subgraph from it. The extracted subgraph has to be a tree structure in order to represent the Web site topic hierarchy. We have investigated several graph algorithms, including breadth-first search, shortest-path search and directed minimum-spanning tree, to generate the initial tree structure. All three algorithms are capable of extracting a tree from an input graph, but they are different in terms of the input graph representation and criteria used to evaluate edges for constructing the tree. Breadth-first traversal works on unweighted graphs and minimizes the number of hops from the root to other nodes. Shortest-path search and directed minimum-spanning tree handle weighted graphs and minimize the weight of the paths from the root to other nodes and the total weight of all the edges respectively. In the next three sections, we describe these three algorithms. In the subsequent section, we shall present the edge-weight function that is employed in the shortest-path search and directed minimum-spanning tree search algorithms.

Breadth-first search. Given a graph $G = (V, E)$, where V is a set of nodes and E is a set of edges, and a distinguished source vertex s, *breadth-first search* travels through the edges of G to find every vertex that is reachable from s systematically and constructs a breadth-first tree. Breadth-first search identifies all undiscovered vertices from the latest discovered vertices. The latest discovered vertices are considered as frontiers and have the same distance from s. That means it won't discover any vertices at distance $k + 1$ from s until it has discovered all vertices at distance k. When the algorithm finds a new vertex v from the adjacency list of a discovered vertex u, it adds the edge (u, v) and the vertex v to the breadth-first tree. u becomes

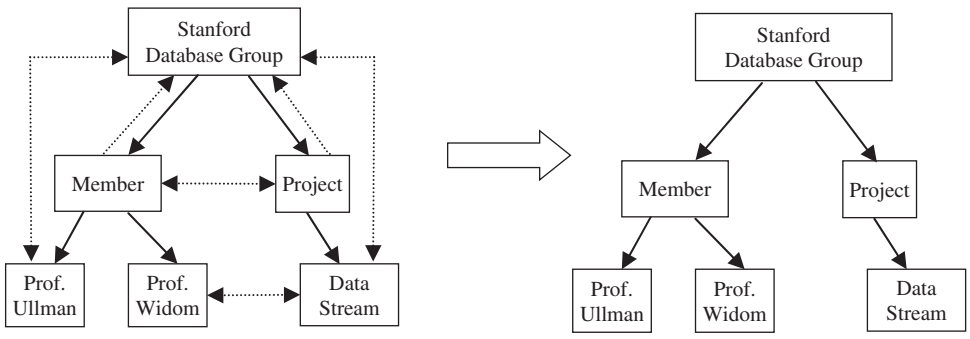

FIG. 2. Link structure versus topic hierarchy.

DOI: 10.1002/asi

the parent of v. The time complexity of breadth-first search is $O(|E|)$.

Shortest-path search. In a *shortest-paths problem*, we are given a weighted directed graph $G = (V, E)$, with weight function w: $E \rightarrow \mathbf{R}$ mapping edges to real-valued weights. The *weight* of path $p = \langle v_0, v_1, \ldots, v_k \rangle$ is the sum of the weights of its constituent edges:

$$w(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i) \quad (1)$$

A *shortest path* from vertex u to vertex v is then defined as the path p with the smallest weight. For generating topic hierarchies, we shall focus on the *single-source shortest-path problem*: Given a graph $G = (V, E)$, we want to find the shortest path from a given source vertex $s \in V$ to each vertex $v \in V$.

The shortest-path search algorithm takes a weighted directed graph $G = (V, E)$ with nonnegative weight function (i.e., $w(u, v) \geq 0$ for each edge $(u, v) \in E$) as input. We employ the Dijkstra's algorithm to maintain a priority queue of vertices whose final shortest path has not been determined. Our algorithm repeatedly selects the vertex u in the queue with the minimum shortest-path estimate and updates the shortest-path estimate for all nodes adjacent to u. When the weight function w is nonnegative, the algorithm is guaranteed to find the shortest paths for all vertices. The subgraph formed by the vertices and edges on the shortest path is a tree rooted at the source vertex (Cormen, Leiserson, Rivest, & Stein, 2003). The time complexity of the algorithm is $O(|V|^2 + |E|) = O(|V|^2)$.

Directed minimum spanning tree. Minimum-spanning tree is a well-known problem in graph theory. For an undirected weighted graph $G(V, E)$ with weight function w, we wish to find an acyclic subset T of E that connects all of the vertices and minimizes the total weight of edges. Since T is acyclic and connects all of the vertices, it must form a tree. The problem of determining the tree T is the *undirected minimum-spanning-tree problem*.

The *directed minimum-spanning tree problem* takes a directed weighted graph $G(V, E)$ and a root vertex s as input and finds a subset T of E whose total weight is minimized such that the root s has only outgoing edges while all the other vertices have only one incoming edge (Chu & Liu, 1965; Edmonds, 1967; Leonidas, 2003). Therefore, there is a unique path from the root to every other vertex in the directed spanning tree.

The topic hierarchy for a Web site is precisely a directed spanning tree rooted at the Web site homepage. Therefore, the topic hierarchy can be generated by finding a directed minimum-spanning tree from the Web site's link graph. In the directed minimum-spanning tree search algorithm, each vertex in the graph selects the incoming edge with lowest weight greedily. If a tree is obtained, it must be the directed minimum-spanning tree. Otherwise, there must be a cycle. The algorithm identifies a cycle and contracts it into a single dummy vertex and recalculates the weights of edges into and out of the cycle. It has been proven that a directed minimum-spanning tree in the contracted graph is equivalent to a minimum-spanning tree in the original graph (Leonidas, 2003). Hence the algorithm can recursively invoke itself on the new graph. The directed minimum-spanning tree search algorithm runs in $O(|V|^3)$ time since each recursive call takes $O(|V|^2)$ to find the lightest incoming edge for each word and to contract the graph. There are at most $O(|V|)$ recursive calls since we cannot contract the graph more than n times.

Discussion. All three graph algorithms described above are able to extract a tree structure from the link graph that could be a potential topic hierarchy. Shortest-path search bears some similarity to breadth-first search. Both algorithms generate the tree by iteratively expanding the frontier of the tree by selecting a node closest to the root so far, but these methods differ in how they evaluate the closeness. The shortest-path distance is based on the real function w, which is more precise than the breadth-first distance, which is simply equal to the number of edges along a path. In a dense graph like a Web site's link graph, it is easy to encounter multiple paths with equal breadth-first distance, which is less likely when using the shortest-path estimate with a properly designed weight function. The shortest-path search thus has more discriminative power. The directed minimum-spanning tree is different from the other two algorithms in the way that it tries to minimize the weight of the incoming edge to the last node on each path, whereas breadth-first search and shortest-path search consider the whole path.

Edge-Weight Function

Both shortest-path search and directed minimum-spanning tree need the edge-weight function w as input; this function plays an important role in selecting edges in the topic hierarchy. Therefore, the effectiveness of these two algorithms depends heavily on the edge-weight function. In the section, we describe the weight function.

There are two types of hyperlinks between Web pages within a Web site for building a Web site topic hierarchy as described earlier: aggregation links, which connect topics and subtopics, and shortcut links, for which there is no hierarchical relationship between the two connected Web pages. Clearly, in topic-hierarchy generation, the aggregation links should be selected to form the tree structure. The graph algorithms described in the previous sections apply different criteria to identify the aggregation links. Breadth-first search does not use edge weight and selects the edges that minimize the number of links connecting the root to each vertex. Shortest-path search and directed minimum-spanning tree consider the weights of edges. The former selects the edges that form a path from the root to a node with the smallest total weight. The latter tries to make the weight on edges into each node as small as possible while maintaining the spanning-tree topology. Both algorithms tend to favor light

DOI: 10.1002/asi

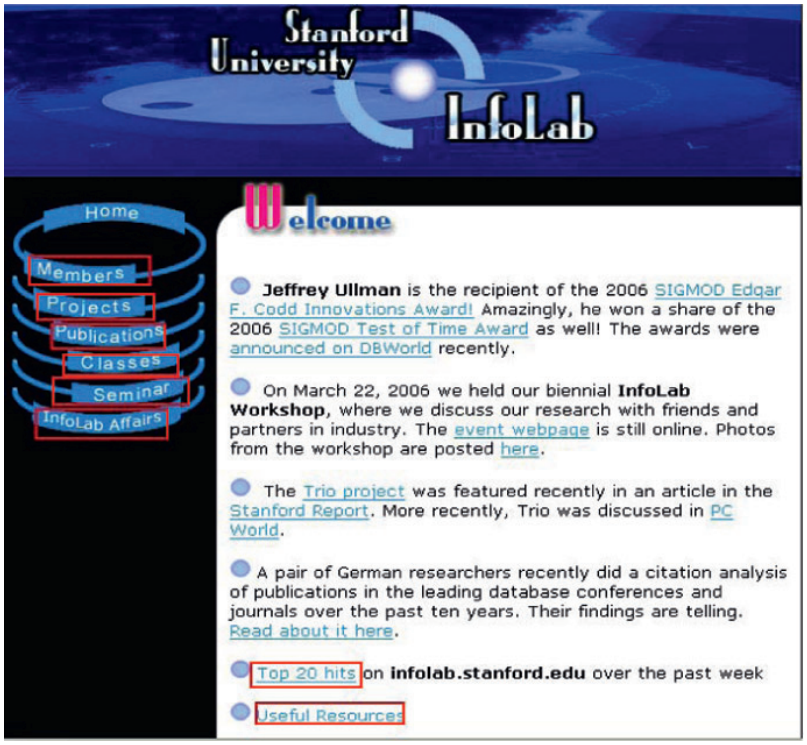

FIG. 3. A Web page with aggregation being links highlighted.

edges over heavy edges. Therefore, a proper edge-weight function should assign a smaller weight to aggregation links and a larger weight to shortcut links.

Our approach is to treat the edge-weighting problem as a classification problem, in which we attempt to classify each link into one of two classes: aggregation links versus shortcut links.

A Web page contains more structural information than a normal text document. The HTML markups on a Web page could reveal a variety of information such as the presentation and layout of different elements in the document. These markups are processed by a browser to produce a visual presentation of the Web page to the user. A Web designer usually utilizes different visual hints on a Web page to facilitate navigation. Figure 3 shows the Stanford Database Group's homepage, on which we have highlighted the aggregation links. Using only this example, we notice several patterns about aggregation links and shortcut links:

1. Links in a navigation bar are more likely to be aggregation links.
2. Aggregation links are usually associated with an anchor image or short anchor text.
3. Links that occurs in the middle of a sentence are less likely to be aggregation links.

All these patterns are associated with the presentation and layout of the Web page. By examining a large number of typical Web pages, we identified a set of features, based on the content, path, presentation, and layout of the Web pages, that are useful for distinguishing aggregation and shortcut links. The details of these features are described in the next subsection.

Features for link classification. Using the Web page's content, path, and markups, we propose the following features for a link (u, v) in link classification.

• **Path relationship.** This feature is based on the URL of u and v. A URL string, such as www.cs.cmu.edu/people/index.html consists of a domain name www.cs.cmu.edu and a path /people/index.html. As we are dealing Web pages of particular Web site, the host name portion of their URLs must be the same. The URL's path portion reveals two useful features: One is the location of the folder where the page is stored and the other is the name of the file corresponding to this page. In some cases, a URL does not include the file name but ends with the symbol /, in which case it generally refers to the "index.html" file within the folder.

Folders are commonly used for organizing a large number of documents. Related files are usually grouped within the same folder and may be further divided into subfolders. Given the paths of all the pages, we can easily build a directory tree to represent the directory structure of the Web site, in which the leaf nodes correspond to individual files and the internal nodes represent folders, as shown in the partial directory tree in Figure 4.

The *path relationship* feature is based on the locations of u and v in the Web site's directory. However, we decide to make the path relationship a categorical feature instead of a numeric feature. The reason for this is that the directory distance is

DOI: 10.1002/asi

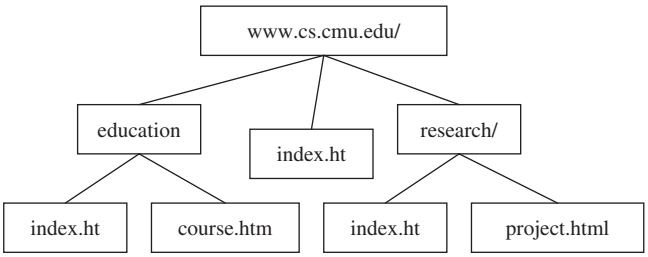

FIG. 4. Partial directory tree for www.cs.cmu.edu.

TABLE 1. Types of path relationships.

Type	Description
I	*u* and *v* are in the same folder. e.g. *u*: /ullman/index.html, *v* : /ullman/publication.html
II	*u* is in the parent folder of the *v* e.g. *u*: /index.html, *v* : /db_pages/members.html
III	*u* is in the *v*'s grandparent or higher folder e.g. *u*: /index.html, *v* : /cs104/proj/proj1.html
IV	*u* is in some subfolder within *v*'s folder e.g. *u*: /cs104/proj/proj1.html, *v* : /cs104/index.html
V	Otherwise e.g. *u*: /cs104/index.html, *v* : /cs202/index.html

a symmetric measure, so it will be the same regardless of whether *u* is in a parent folder or subfolder of *v*. However, a link is more likely to be an aggregation link when *u* is in a parent folder of *v* than in a subfolder of *v* (Chen et al., 2003). We defined five types of relationships for this feature, which are described in Table 1.

• **Explicit entry page.** Each subtree within a topic hierarchy corresponds to a particular topic of a Web site. We define the page sitting at the root of a sub tree as the *entry page* for the topic of that subtree. It is quite often that the entry page of a particular topic is named "index.html," for instance, http://infolab.stanford.edu/~ullman/index.html. However, an entry page is not necessarily named "index.html"; for instance, http://infolab.stanford.edu/db_pages/members.html, which serves as the entry page for the topic "Group Members" and uses a meaningful word "members" as its name. We refer to Web pages named "index.html" as explicit entry pages, as they are easily recognizable by the file name.

For a link (*u*, *v*), if *u* is an explicit entry page, it is more likely to be an aggregation link than if *u* is not. Hence, we consider the explicit entry page to be a categorical feature with two values, "yes" and "no," corresponding to whether page *u* is an explicit entry page.

• **Content relevance.** The content relevance reflects the similarity between the texts on *u* and *v*. To compare the textual content of Web pages, we first preprocess the Web pages by removing all HTML markups. The remaining text is then used to build a vector representation of the content, $[w_{1,k}, w_{2,k}, \ldots w_{t,k}]^{\mathrm{T}}$, where $w_{i,k}$, the weight of each term *i* in document *k*, is its *tf* **idf* value, where *tf* is the frequency of the term *i* in document *k*, and *idf* is its inverted document frequency of term *i* (Yates & Neno, 1999). The content relevance is calculated by the cosine similarity between the two document vectors:

$$r_{content}(u, v) = \frac{\sum_{i=1}^{t} w_{i,u} \cdot w_{i,v}}{\sqrt{\sum_{i=1}^{t} w_{i,u}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,v}^2}} \qquad (2)$$

• **Navigation bar feature.** A designer uses navigation bars to highlight lists of hyperlinks in a contiguous area on Web pages such as the top, side, and bottom. For Web sites of enormous size, navigation bars enable users to move around in the Web sites more efficiently. Usually, links in the navigation bar point to subtopics of the current page. Therefore, whether a link is located inside a navigation bar can be a useful feature for telling if it is an aggregation link.

Our method for identifying the navigation bars on a Web page relies on analyzing the document structure of the HTML file for the Web page. In order to analyze the HTML document, we pass Web pages through an open source HTML parser, openXML (http://www.openxml.org), which corrects the markup, so we do not need to worry about error resilience, and create a document object model (DOM) tree. The document object model (www.w3c.org/DOM) is a standard for creating and manipulating in-memory representations of HTML (and XML) documents. The different elements of a HTML document (e.g., table, table rows, table cells, paragraphs, links, etc.) are organized hierarchically in a DOM tree, which reveals useful information about the Web page's layout, such as which element is contained within which element (see Figure 5).

```
<TABLE>
<TBODY>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</TBODY>
</TABLE>
```

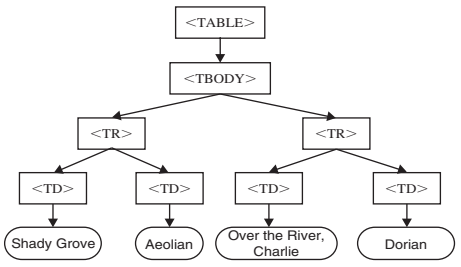

FIG. 5. A fragment of HTML for simple table and the corresponding graphical DOM tree representation.

DOI: 10.1002/asi

TABLE 2. Types of link positions.

Type	Description
I	The link is not a segment in a sentence.
II	The link is at the beginning of a sentence.
III	The link is in the middle of a sentence.
IV	The link is at the end of a sentence.

A navigation bar is supposed to match some subtrees within the DOM tree of the corresponding documents. To identify such subtrees, we traverse the DOM tree recursively. For each node, we calculate the ratio of the length of linked text (i.e., anchor text for hyperlinks) to the length of all the text under the node. If this ratio exceeds a threshold, which we set as 0.8, the node is classified as a *link node*. The root node for a navigation bar is a node whose children are all link nodes and whose parent is not a link node. Upon detecting all the navigation bars, we can decide the value of the *inside-navigation-bar* feature as either "yes" or "no" according to whether the link is located within a navigation bar in the DOM structure.

• **Coreference.** This feature is the counterpart of the navigation bar feature. The observation that links within a navigation bar are usually subtopics of the page with the navigation bar also implies the relationship between two pages in a navigation bar. If two pages frequently co-occur in the same navigation bar on other pages, they are more likely to be siblings instead of parent and child in the topic hierarchy. The value for the numeric feature *co-reference* is equal to the number of times u and v co-occur in the same navigation bar on other pages in the Web site.

• **Position in text.** Instead of navigation bars, hyperlinks may appear inside the text portion on a Web page; the position of a link (u, v) within the text also affects how likely it is that v is a subtopic of u. We first preprocess the text in a Web page by extracting all the text blocks and splitting each text block into sentences using the Brill Tagger (Brill, 1995). The position of a link in the text is then classified as one of the four types listed in Table 2 examples for each type of link position are shown in Figure 6.

The *position* feature has been widely used in text summarization to identify an important and representative phrase or sentence (Yang & Wang, 2008). We adopt this feature to support link classification.

• **Anchor text length.** We observed that the anchor texts for subtopics are usually described by short phrases, such as "people," "projects," and so forth, which provides a good abstraction of the nature of the subtopics. In contrast, long anchor text like sentences are usually used to highlight certain news or events, such as "IBM scientists discover new way to explore and control atom-scale magnetism," which are often not subtopics of the page. In the *anchor text length* feature, we measure the number of non-stop words inside the anchor text (a list of stop words such as "and," "to," "on," etc. are excluded).

• **Anchor text font size.** HTML allows an author to decorate text using font and colors so as to emphasize certain text and distinguish different types of text. Our assumption is that the font used for anchor text of aggregation links is supposed to emphasize the link, therefore, a larger font size or more distinctive color should be used for the anchor text. However, as the property of "a distinctive color" is quite ambiguous and may involve image-processing techniques, which will lower efficiency, we focus here on the font size associated with the Web page.

Notice that the scales of the font size used on Web pages from different Web sites can be rather different, so we cannot use the font size specified in the <font> tag directly as the value for this feature. We collect all the different font sizes used on a Web page and divide each particular font size by the average of the font sizes used on the Web page to obtain the normalized font size. So a value of 1.0 indicates the font size is normal, while a value greater or less than 1.0 indicates the font is large or small.

Similar to other Web site classification and mining techniques, features are important in knowledge discovery, but feature extraction is usually a time-consuming process especially as the number of features increases. However, a sitemap of a Web site does not change significantly from time to time. We usually monitor if there are any new Web pages added to a Web site and enhance the sitemap based on the features of the added Web pages.

Learning algorithms. In the previous section, we described eight types of features for classifying aggregation and shortcut links. Each instance of hyperlink can be represented by a feature vector $\mathbf{x} = < x_1, x_2 \ldots x_8 >$, where x_i is the value for the ith feature for this link. Let c_1 and c_2 denote the two categories: aggregation link and shortcut links. A classifier is a function $f(\mathbf{x})$ that outputs the corresponding category for an instance. We have investigated three kinds of classifiers for this specific classification problem, which are decision tree, naïve Bayes, and logistic regression.

A decision tree is a natural and intuitive way to classify an instance through answering a sequence of questions, in which the next question asked depends on the answer to the current question. The classification of a particular instance begins at the root node. The questions asked at each node concern a particular feature of the instance. If it is categorical, the question is about which case the value of the feature belongs to. If it is numerical, the question is what range the value of the feature falls in. To determine the category for the instance, we follow the downward links until a terminal or leaf node is reached, which contains the category decisions. In our study, we used the popular C4.5 algorithm for decision-tree learning (Quinlan & Rivest, 1993).

The naïve Bayes classifier (Mitchell, 1997) is a highly practical learning method of the more general Bayesian learning method, which adopts a probabilistic approach to

DOI: 10.1002/asi

FIG. 6. Examples of different types of link positions.

classification. One important feature of the probabilistic learning method is that it provides a quantitative approach to weighting the evidence supporting alternative decisions.

The Bayesian approach to classifying a new instance is to assign the most probable category, c_{MAP}, given the feature vector $<x_1, x_2 \ldots x_n>$ that describes the instance.

$$C_{MAP} = \arg\max_{c_i \in C} P(c_i \mid x_1, x_2 \ldots x_n) \tag{3}$$

Using Bayes' theorem, this expression can be rewritten as

$$\begin{aligned} C_{MAP} &= \arg\max_{c_k \in C} \frac{P(x_1, x_2 \ldots x_n \mid c_k) P(c_k)}{p(x_1, x_2 \ldots x_n)} \\ &= \arg\max_{c_k \in C} P(x_1, x_2 \ldots x_n \mid C_i) P(c_k) \end{aligned} \tag{4}$$

It is easy to estimate each of the $P(c_k)$ by counting the frequency with which each class c_i occurs in the training data. The naïve Bayes classifier is based on the simplifying assumption that the attribute values are conditionally independent given the class. In other words, the assumption is that given the class of the instance, the probability of observing the conjunction $x_1, x_2 \ldots x_n$ is just the product of the probabilities for the individual features: $P(x_1, x_2 \ldots \mid c_i) = \prod P(x_j \mid c_k)$.

Logistic regression (Hastie, Tibshirani, & Friedman, 2001) is another classifier based on probabilistic models. It attempts to learn $P(c \mid x_1, x_2 \ldots x_n)$ by assuming a parametric form for the distribution, which in the case of two classes is

$$\begin{aligned} P(c_0 \mid x_1, x_2 \ldots x_n) &= \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i x_i)} \\ P(c_1 \mid x_1, x_2 \ldots x_n) &= \frac{\exp(w_0 + \sum_{i=1}^{n} w_i x_i)}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i x_i)} \end{aligned} \tag{5}$$

Note that the two probabilities always sum to 1. One highly convenient property of this form is that it leads to a simple linear expression for classification. To classify an instance $X = <x_1, x_2 \ldots x_n>$, we generally want to output the value c_k that maximizes $P(c_k \mid x_1, x_2 \ldots x_n)$. In other words, we output c_0 if the following condition holds:

$$\frac{P(c_0 \mid x_1, x_2 \ldots x_n)}{P(c_1 \mid x_1, x_2 \ldots x_n)} > 1 \tag{6}$$

which is equivalent to the following simple condition:

$$0 < w_0 + \sum_{i=1}^{n} w_i x_i \tag{7}$$

and assigns class c_1 otherwise. The parameter values for logistics regression can be determined by maximizing the conditional data likelihood using some gradient ascent procedure.

Edge weighting through classification. In the previous section, we provide details of three classification algorithms, which generate classifiers for predicting an instance's category based on training examples. When building a topic hierarchy, our aim is to distinguish aggregation links from shortcut links. Using the three algorithms described earlier, we build classifiers that can predict whether a link is an aggregation link or a shortcut link based on the features introduced above.

In most applications, a classifier is used to produce symbolic output of the predicted class label. However, a numeric value is needed to produce the weight of the links. Besides, both the shortest-path tree model and directed minimum-spanning tree model favor edges with smaller weight. A reasonable weighting method should assign larger weight to a shortcut link and a smaller weight to an aggregation link.

Let c_0 and c_1 denote aggregation link and shortcut link respectively. Given an edge (u,v) with corresponding feature vector X_{uv}, the weight on the edge, $w(u,v)$, is measured by $P(c_1 \mid X_{uv})$. Links that are more likely to be shortcut links

DOI: 10.1002/asi

have larger weights. Notice that the range of $P(c_1 \mid X_{uv})$ is [0, 1], which makes the differences between the link weights small. Therefore, we use the value $-\log P(c_0 \mid X_{uv})$ as the edge weight. For links that are most probably aggregation links, that is $P(c_0 \mid X_{uv})$ is high, the edge weight would be low because of the negation. Also using the log transformation, the edge weight now ranges from 0 to ∞. Following this edge-weighting scheme, we are able to interpret the shortest-path tree and directed minimum-spanning tree using probabilities. In particular, the shortest-path tree is indeed the tree that maximizes the probability that all the paths from the root to each node consist of aggregation links:

$$\begin{aligned} T &= \arg\min_T \sum_{i \in V} \sum_{(u,v) \in Path_T(i)} w(u, v) \\ &= \arg\min_T \sum_{i \in V} \sum_{(u,v) \in Path_T(i)} -\log(P(c_0 \mid X_{uv})) \\ &= \arg\max_T \prod_{i \in V} \prod_{(u,v) \in Path_T(i)} P(c_0 \mid X_{uv}) \end{aligned} \quad (8)$$

where $Path_T(i)$ is the path to node i in a candidate tree T.

On the other hand, the directed minimum-spanning tree is the tree that maximizes the probability that all edges in the tree correspond to aggregation links:

$$\begin{aligned} T &= \arg\min_T \sum_{(u,v) \in E_T} w(u, v) \\ &= \arg\min_T \sum_{(u,v) \in E_T} P(c_0 \mid X_{uv}) \\ &= \arg\max_T \prod_{(u,v) \in E_T}^{T} P(c_0 \mid X_{uv}) \end{aligned} \quad (9)$$

The major difference between the shortest-path tree and the directed minimum-spanning tree model is that every edge is counted exactly once in a directed minimum-spanning tree while in the shortest-path tree, an edge may be counted multiple times depending on how many times it is on the tree path to other nodes.

The computation of $P(c_0 \mid X_{uv})$ based on the logistic regression and naïve Bayes classifier. In logistic regression, a specific parametric form for $P(c_0 \mid X_{uv})$ is assumed. So after training a logistic regression classifier, we may use Equation 5 to calculate $P(c_0 \mid X_{uv})$ based on the optimized weights W. For naïve Bayes, the probabilities $P(c_i)$ and $P(x_j \mid c_i)$ are available. To obtain $P(X_{uv} \mid c_0)\ P(c_0)$, we may apply Bayes' rule:

$$\begin{aligned} P(c_0 \mid X_{uv}) &= \frac{P(X_{uv} \mid c_0) P(c_0)}{P(X_{uv} \mid c_0) P(c_0) + P(X_{uv} \mid c_1) P(c_1)} \\ &= \frac{\prod_i P(x_i \mid c_0) P(c_0)}{\prod_i P(x_i \mid c_0) P(c_0) + \prod_i P(x_i \mid c_1) P(c_1)} \end{aligned} \quad (10)$$

Unlike the other two classifiers, the decision-tree classifier doesn't take a probabilistic approach to the classification task. To estimate the probability $P(c_0 \mid X_{uv})$, we need to consider the set of training instances that reach each node, using which we may count the number of instances in each class. Suppose there are n_0 out of a total of n instances that are of class c_0 at this node. Let the true probability $P(c_0 \mid X_{uv})$ be q; the n instances can be considered as being generated by a Bernoulli process with parameter q, of which n_0 turn out to be class c_0. Notice that it is inappropriate to use the observed frequency $f = n_0/n$ as the estimate for q, as it doesn't take into account the size of sample used to estimate q. One standard method is to construct a confidence interval for q and use the upper confidence limit as the estimate (Whitten & Frank, 2005). Given a particular confidence level c (the default value used by C4.5 is $c = 25\%$), we find the upper confidence limit z standard normal distribution tables, which for $c = 25\%$ is $z = 0.69$. Using the confidence limit z, we can estimate $P(c_0 \mid X_{uv})$ at a leaf node:

$$P(c_0 \mid X_{uv}) = \frac{f + \frac{z^2}{2n} + z\sqrt{\frac{f}{n} - \frac{f}{n}^2 + \frac{z^2}{4n^2}}}{1 + \frac{z^2}{n}} \quad (11)$$

Experiments

In the Graph-Based Algorithms section of this paper, we have described three graph algorithms for constructing topic hierarchies using the Web site's link structure: the breadth-first tree model on unweighted graphs, and the shortest-path tree and directed minimum-spanning tree on weighted graphs. In the following section, we provided details on estimating edge weights using machine-learning methods. In this section, we describe a set of experiments using real Web data, which tests the effectiveness of the proposed algorithms and techniques.

Data Preparation

We have chosen five Web sites from different domains for evaluating the algorithms; these sites are listed in Table 3. To download the Web pages from these Web sites, we provide the homepages of these Web sites as the seed page to a crawler. The crawler then traverses the Web site following breadth-first order. So Web pages linked from these homepage are downloaded first, followed by the Web pages linked

TABLE 3. Web sites used for evaluation.

URL	Type	Description
www.cs.cmu.edu	Education	School of Computer Science Carnegie Mellon University
infolab.stanford.edu	Education	Database Research Group Stanford University
www.whitehouse.gov	Government	U.S. Government
www.research.ibm.com	Commercial	IBM Research
www.palmsource.com	Commercial	Palm Software and Palm OS

DOI: 10.1002/asi

from these pages and so on. The crawler examines every link it encounters and only follows links to any URL that is in the same domain as the homepage. As the complete Web sites may contain an enormous number of Web pages, we set the maximum crawling depth to 5. Therefore, only pages that can be reached from the homepage by following no more than 5 links are downloaded. For all the downloaded pages, we change any relative paths used by the hyperlinks to absolute paths and append the default file name "index.html" to any path ending with "/" (e.g. /∼ullman/ to /∼ullman/index.html).

For each Web site, we count the number of pages at different breadth-first levels, where Level 1 contains the homepage, Level 2 contains the pages linked from the homepage, and so forth. These statistics are shown in Table 4. We also counted the average degree (i.e., number of in-links plus number of out-links) of the nodes at each level (see Table 5). We observe the following properties about Web sites' link structure:

- The number of pages grows very fast as the depth increases. This is a property of any tree structure.
- The degree of nodes tends to decrease rapidly as a node gets further away from the homepage of the Web site.

To obtain benchmark data on these Web sites, two human judges working in the field of information retrieval are asked to manually produce a partial topic hierarchy for each Web site independently. The judges examined the Web pages in each site following depth-first order starting at the homepage. For each page they visit, the judges first extract the set of links that point to a subtopic of the current page from the set of hyperlinks on the page. They then randomly select a subset of the identified subtopics, which are further explored. Repeating this process for each Web site, we obtain a directed tree, which is a subgraph of the complete topic hierarchy. We refer to this tree as a *partial topic hierarchy* and use it as the benchmark for evaluating a new tree generated by an algorithm. Table 6 presents the sizes of benchmarks for each dataset. We have measured the interrater reliability by the Cohen's kappa test and obtained a result of 0.97. It shows a high reliability between the two judges and demonstrates the quality of the benchmark.

TABLE 4. Number of pages at each level.

Dataset/level	1	2	3	4
www.cs.cmu.edu	1	37	246	336
infolab.stanford.edu	1	9	102	769
www.whitehouse.gov	1	55	2424	20928
www.research.ibm.com	1	51	245	1256
www.palmsource.com	1	26	351	384

TABLE 5. Average degree of nodes at each level.

Dataset/level	1	2	3	4
www.cs.cmu.edu	428	292	20	22
infolab.stanford.edu	119	24	15	13
www.whitehouse.gov	2367	1537	130	2
www.research.ibm.com	997	89	28	11
www.palmsource.com	374	250	28	3

TABLE 6. Sizes of benchmarks for each dataset.

Dataset	Benchmark size
www.cs.cmu.edu	124
infolab.stanford.edu	260
www.whitehouse.gov	179
www.research.ibm.com	441
www.palmsource.com	123

To evaluate a generated topic hierarchy, we compare the parent-child relationships between the benchmark topic hierarchy and the generated topic hierarchy. For each page v in the benchmark topic hierarchy, let u be the parent of v and w be the parent of the corresponding node of v in the generated tree. We count it as a *hit* if w matches u, and a *miss* otherwise. This allows us to measure the quality of the generated tree by accuracy, which can be computed by dividing the number of hits by the number of nodes in the benchmark tree minus one. One is subtracted in the denominator to exclude the root node.

$$Accuracy = \frac{\text{number of matches between } w \text{ and } u \text{ for all } v \text{ in the benchmark tree}}{\text{number of nodes in the benchmark tree-1}} \quad (12)$$

where u is the parent of v in a benchmark tree and w is the parent of the corresponding node of v in a generated tree.

If the generated topic hierarchy is identical to the benchmark tree, the accuracy is 1. However, as the number of misplaced Web pages in the generated topic hierarchy increases, the parent-child relationship will not be maintained and the accuracy decreases.

Performances of Breadth-first Search

The breadth-first search algorithm (see above) is the simplest among the described approaches. It involves no adjustable parameters. We use it as the baseline to compare with the shortest-path search algorithm and the directed minimum-spanning tree algorithm. The accuracies of the topic hierarchy generated by breadth-first search for each Web site are shown in Table 7.

TABLE 7. Performance of breadth-first search (BFS).

Dataset	Accuracy
www.cs.cmu.edu	77.2%
www.db.stanford.edu	74.3%
www.whitehouse.gov	79.4%
www.research.ibm.com	71.9%
www.palmsource.com	71.3%

DOI: 10.1002/asi

TABLE 8. Performance of shortest-path search using relevance method for estimating edge weight.

Dataset	Accuracy
www.cs.cmu.edu	94.9%
infolab.stanford.edu	94.6%
www.whitehouse.gov	82.4%
www.research.ibm.com	78.9%
www.palmsource.com	88.9%

TABLE 9. Training data statistics.

Dataset	Number of aggregation links	Number of shortcut links
www.cs.cmu.edu	1240	1023
infolab.stanford.edu	26	658
www.whitehouse.gov	179	1601
www.research.ibm.com	441	2057
www.palmsource.com	123	756

Performances of Shortest-Path Search and Directed Minimum-Spanning Tree

Both shortest-path search and directed minimum-spanning tree algorithms work with weighted graphs, which we can construct using the machine-learning method described above. An earlier algorithm proposed by Liu and Yang (2005a, 2005b) also used the shortest-path search algorithm on weighted graphs in order to extract a tree structure from a Web site's link graph. They proposed the relevance method, which estimates the edge weight by summing two features: the content relevance and path relevance between the end nodes of a link. Notice that the algorithm involves tuning a weighting parameter. For the purpose of comparison, we report only the optimal performance of that algorithm on the five testing Web sites in Table 8.

The machine-learning method described under Web Site Topic-Hierarchy Generation above treats the edge-weighting problem as a classification task. A classifier is used to predict the confidence of a link being an aggregation link based on a set of automatically derived features.

To apply the machine-learning method for edge weighting, we must first obtain the link classifiers by running the learning algorithms on some training data. For the classification of hyperlinks, the training data should consist of instances of both aggregation links and shortcut links. The construction of the training data is based on the manually collected benchmark data for evaluating topic hierarchies. The benchmark data itself consists of aggregation links only. To get the shortcut-links, we apply the following rule: If a link (u,v) is included in the benchmark (i.e., is an aggregation link), then the other links pointing to v are shortcut links. For each Web site, we use the benchmark data to get the aggregation links and find the corresponding shortcut links based on the Web site's link graph. The numbers of aggregation and shortcut links in the training data generated from each Web site are shown in Table 9.

When evaluating the machine-learning method, our objective is to find a classifier that is capable of classifying links in some unseen Web site instead of the Web sites it was trained on. Therefore, we use the leave-one-site-out scheme to conduct the evaluation as follows: For each of the five Web sites, we train a classifier using the data of the other four sites, and then use the classifier to predict the edge weight for the remaining site, which guarantees the data used for training and testing are independent. Therefore, the performance measures achieved can fairly indicate how well the method can be generalized to other new Web sites. For each dataset, we trained a decision tree, a naïve Bayes, and a logistic regression classifier to estimate edge weights for the directed graph, and measured the performance of a shortest-path search and a directed minimum-spanning tree search on the estimation of edge weights for different graphs using different classifiers. The results are summarized in Tables 10 and 11. As can be seen, the decision-tree classifier produced the best results on all but the Stanford dataset, while the performances of naïve Bayes and logistics regression are close to each other and significantly lower than that of decision tree.

TABLE 10. Performance of shortest-path search using different classifiers.

Dataset	Decision tree	Naïve Bayes	Logistic regression
www.cs.cmu.edu	*97%	91.5%	81.4%
infolab.stanford.edu	90.95%	*92.5%	89.8%
www.whitehouse.gov	*87.8%	81.0%	81.6%
www.research.ibm.com	*79.4%	75.3%	76.6%
www.palmsource.com	*88.9%	69.2%	70.1%

*The best performance obtained for each dataset.

TABLE 11. Performance of directed minimum spanning tree using different classifiers.

Dataset	Decision tree	Naïve Bayes	Logistic regression
www.cs.cmu.edu	*98.3%	92.4%	89.0%
infolab.stanford.edu	88.2%	*88.6%	85.5%
www.whitehouse.gov	*94.6%	89.1%	79.6%
www.research.ibm.com	*85.9%	80.2%	80.2%
www.palmsource.com	*92.3%	91.5%	91.5%

*The best performance obtained for each dataset.

Comparison of Different Algorithms

In the previous three subsections, we have presented the experimental results for each individual algorithm for generating a topic hierarchy, which are breadth-first search, shortest-path search, and directed minimum-spanning tree. For shortest-path search and directed minimum-spanning tree, the relevance method and machine-learning method for

DOI: 10.1002/asi

TABLE 12. Performance comparison of different algorithms for topic hierarchy generation.

Dataset/algorithm	Bfs	sps-rel	sps-dt	sps-nb	sps-log	dmst-dt	dmst-nb	dmst-log
CMU	77.2%	94.9%	97.5%	91.5%	81.4%	*98.3%	92.4%	89.0%
Stanford	74.3%	94.6%	90.9%	92.5%	89.8%	88.2%	88.6%	85.5%
Whitehouse	79.4%	82.4%	87.8%	81.0%	81.6%	*94.6%	89.1%	79.6%
IBM	71.9%	78.9%	79.4%	75.3%	76.6%	*85.9%	80.2%	80.2%
Palmsource	71.3%	88.9%	88.9%	69.2%	70.1%	*92.3%	91.5%	91.5%
Average	74.8%	87.9%	88.9%	81.9%	79.9%	*91.9%	88.4%	85.2%

*The best performance obtained for each dataset.

estimating the edge weights are evaluated respectively. The common measure of performance used in all the above evaluations is the accuracy of the generated topic hierarchy. In this section, we summarize and compare the performance across all the proposed algorithms and identify the best approach to generating a topic hierarchy. The performances of these algorithms are summarized in Table 12. The eight algorithms being compared include breadth-first search (bfs); shortest-path search using the relevance method for edge weighting (sps-rel); shortest-path search, using decision tree for edge weighting (sps-dt); directed minimum-spanning tree using decision tree for edge weighting (dmst-dt); shortest-path search, using naïve Bayes for edge weighting (sps-nb); directed minimum-spanning tree using naïve Bayes for edge weighting (dmst-nb); shortest-path search, using logistic regression for edge weighting (sps-log); directed minimum-spanning tree using logistic regression for edge weighting (dmst-log).

It is found that the shortest-path search and directed minimum-spanning tree outperform breadth-first search significantly. Although the effectiveness of shortest-path search and directed minimum-spanning tree still depend on the method for edge weighting, their different variations consistently outperform breadth-first search. This confirms that a weighted-graph model is more suitable for representing the link structure of a Web site.

For four out of the five datasets, the most accurate topic hierarchy was generated by the directed minimum-spanning tree. Also, given the same edge-weighting method, the performance of directed minimum-spanning tree was consistently better than that of the shortest-path search. Therefore the directed minimum-spanning tree is the more appropriate model for modeling topic hierarchies of Web site.

Another important question to answer is what the most effective way of estimating edge weights is. For the machine-learning method, the best classifier for this task is clearly the decision tree. Both shortest-path search and directed minimum-spanning tree generate more accurate topic hierarchies when the edges are weighted by decision tree than when they are weighted by the other two classifiers. The optimal performances on four of the five Web sites were also produced on graphs weighted by the decision tree. However, when using the other classifiers to estimate edge weights, the resulting topic hierarchy can be worse than that generated based on the relevance method. This indicates the importance of picking a suitable classifier when applying the machine-learning method.

Conclusions and Future Work

In this work, several graph algorithms have been adapted to extract the topic hierarchy from a Web site's link graph, including breadth-first search, shortest-path search, and directed minimum-spanning tree. We compared the unweighted- and weighted-graph models for representing a Web site's link structure. For the weighted-graph models, we proposed machine-learning methods for computing the edge weights. We have done extensive experiments using real Web site data to study the performance of different algorithms and techniques. The experimental results have shown the clear superiority of the weighted directed-graph model in modeling the hierarchical relationships more effectively, and that the decision-tree classifier is the most effective tool to estimate edge weight. When operating on the weighted-graph model produced using the decision-tree classifier, the directed minimum-spanning algorithm had the best overall performance in terms of the accuracy of the generated topic hierarchies. In this experiment, we have used three classes of Web sites, i.e. education, government, and commercial. It will be interesting to evaluate how the proposed techniques perform in other classes of Web sites such as nonprofit organizations, e-commerce shops, and blogs.

Topic-hierarchy generation for Web sites is a very new research topic. There are many possible improvements to our proposed techniques. One interesting issue to investigate is whether more flexible representations than trees can be used for the topic hierarchy. Using the tree model, a restriction is that each page can have one and only one parent. However, a general ontology is not necessarily a tree but a network, or a directed acyclic graph. Besides, a label or a brief description on each node will be helpful for users to understand a sitemap. In our current work, we have not investigated the technique of labeling a node automatically. This can be considered in future work.

A variety of Web site mining tasks such as Web site classification may also benefit from a more accurate representation of a Web site's content structure. In the future, it is possible to conduct more experiments to investigate the effect of the topic hierarchy on the performance of a Web site mining application. By showing the usefulness of topic hierarchy, we would

DOI: 10.1002/asi

also have a stronger motivation to further improve techniques for topic-hierarchy generation.

References

Amitay, E., Carmel, D., Darlow, A., Lempel, R., & Soffer, A. (2003). The connectivity sonar: Detecting site functionality by structural patterns. In Proceedings of the 14th ACM Conference on Hypertext and Hypermedia (HYPERTEXT '03) (pp. 38–47). New York: ACM Press.

Brill, E. (1995). Transformation-based error-driver learning and natural language processing: A case study in part of speech tagging. Computational Linguistics, 21, 543–565.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. In Proceedings of the Seventh World Wide Web Conference (Article FP11). Retrieved October 13, 2008, from http://www7.scu.edu.au/00/index.htm

Chakrabarti, S. (2000). Data mining for hypertext: A tutorial survey. ACM SIGKDD Explorations, 1(2), 1–11.

Chen, Z., Liu, S., Liu, W., Pu, G. & Ma, W.Y. (2003). Building a Web thesaurus from Web link structure. In Proceedings of the 26th ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 48–55). New York: ACM Press.

Chu, Y.J., & Liu, T.H. (1965). On the shortest arborescence of a directed graph. Science Sinica, 14, 1396–1400.

Cormen, T., Leiserson, C. Rivest, R., & Stein, C. (2003). Algorithm design and analysis (2nd ed.). Cambridge, MA: MIT Press.

Davison, B.D. (2000). Topical locality in the Web. In Proceedings of the 23rd Annual ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 272–279). New York: ACM Press.

Edmonds, J. (1967). Optimum branchings. Journal of Research of the National Bureau of Standards, 71B, 233–240.

Ester, M., Kriegel, H.P., & Schubert, M. (2002). Web site mining: A new way to spot competitors, customers, and suppliers in the World Wide Web. In Proceedings of the Eighth International Conference on Knowledge Discovery and Data Mining (pp. 249–258). New York: ACM Press.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). Elements of statistical learning. Berlin: Springer.

Kleinberg, J.M. (1998). Authoritative sources in a hyperlinked environment. In Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (pp. 668–677). Philadelphia, PA: Society for Industrial and Applied Mathematics.

Kumar, R., Punera, K., & Tomkins, A. (2006). Hierarchical topic segmentation of Web sites. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 257–266). New York: ACM Press.

Leonidas, G. (2003). Arborescene optimization problems solvable by Edmonds' algorithm. Theoretical Computer Science, 301, 427–437.

Levene, M. & Wheeldon, R. (2001). A Web site navigation engine. In Proceedings of the Tenth International World Wide Web Conference. New York: ACM Press.

Li, W.S., Kolak, O., Vu, Q., & Takano, H. (2000). Defining logical domains in a Web site. In Proceedings of the 11th ACM Conference on Hypertext and Hypermedia (pp. 123–132). New York: ACM Press.

Lindemann, C., & Littig, L. (2006). Coarse-grained classification of Web sites by their structural properties. In Proceedings of the Eighth ACM International Workshop on Web Information and Data Management (pp. 35–42). New York: ACM Press.

Lindemann, C., & Littig, L. (2007). Classifying Web sites. In Proceedings of the 16th International World Wide Web Conference (pp. 1143–1144). New York: ACM Press.

Liu, N., & Yang, C,C. (2005a). Mining Web sites' topic hierarchy. In Proceedings of the International World Wide Web Conference. Poster Session (pp. 980–981). New York: ACM Press.

Liu, N., & Yang, C.C. (2005b). Automatic extraction of Web site's content structure from link structure. Proceedings of the International Conference on Information and Knowledge Management (pp. 345–346). New York: ACM Press.

Lynch, P., & Horton, S. (2002). Web style guide: Basic design principles for creating Web sites (2nd ed.). New Haven, CT: Yale University Press.

Mitchell, T.M. (1997). Machine learning. New York: McGraw-Hill.

Quinlan, J.R., & Rivest, R. (1993). C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufmann.

Sun, A., & Lim, E.P. (2003). Web unit mining: Finding and classifying subgraphs of Web pages. In Proceedings of the 12th International Conference on Information and Knowledge Management (pp. 108–115). New York: ACM Press.

Tian, Y., Huang, T., Gao, W., Cheng, J., & Kang, P. (2003). Two-phase Web site classification based on hidden Markov tree models. In Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence (p. 227). Washington, DC: IEEE Computer Society.

Toolan, F., & Kusmerick, N. (2002). Mining Web logs for personalized site maps. In Proceedings of the Third International Conference on Web Information Systems Engineering (Workshops) (pp. 223–237). Washington, DC: IEEE.

Wheeldon, R., & Levene, M. (2003). The best trail algorithm for adaptive navigation in the World Wide Web. In Proceedings of the First Latin American Web Congress. Washington, DC: IEEE.

Whitten, I.H. & Frank, E. (2005). Data Mining: Practical machine learning tools and techniques (2nd ed.), San Francisco: Morgan Kaufman.

Yang, C.C., & Wang, F.L. (2008). Hierarchical summarization of large documents. Journal of the American Society for Information Science and Technology, 59, 887–902.

Yates, R., & Neto, B. (1999). Modern information retrieval. New York: ACM Press.

DOI: 10.1002/asi