Constructing Multi-Granular and Topic-Focused Web Site Maps

Wen-Syan Li Necip Fazıl Ayan[‡] Okan Kolak[‡] Quoc Vu[†]

C&C Research Laboratories, NEC USA, Inc.
110 Rio Robles, M/S SJ100, San Jose, CA 95134, USA
Email:wen,nfa,okan,qvu@ccrl.sj.nec.com
Tel:1-408-943-3008 Fax:1-408-943-3099

Hajime Takano Hisashi Shimamura

Human Media Research Laboratories, NEC Corporation
4-1-1, Miyazaki, Miyamae-ku, Kawasaki, Kanagawa 216, Japan
Email:gen,simamura@hml.cl.nec.co.jp
Tel:81-44-856-2258 Fax:81-44-856-2388

ABSTRACT

Site maps are essential to assist users in navigating a Web site. Most of the site maps are constructed manually and are static. However, different users may have different preferences and purposes for using a Web site. For example, a user may want to see a more detailed map while another user prefers a more abstract map. Two users looking for different topics at a large portal site would benefit more from two site maps with different focuses than a single map. In this paper, we present a technique for automatically constructing multi-granular and topic-focused site maps by utilizing directory paths, page contents, and link structures. In these site maps, the Web site topology is preserved and document importance, indicated by citation and semantic relevancy to user's topics of interest, is used for prioritizing the presentation of pages and directories. Experiments on real Web data have been conducted to validate the usefulness of the technique.

Keywords

Site map, logical domain, multi-granularity, decision tree algorithm, topic distillation

1. INTRODUCTION

Web site maps are essential to assist users in navigating a Web site. They provide users an overview of the contents and link structure (i.e., topology) of the Web site they represent. Many Web-masters presume that users of their sites may have different hardware capabilities, network bandwidths, and preferences for interacting with the site. To support more friendly and desirable Web surfing experience, they support several variations of presentation, such as text mode, graphic mode, with or without frames, and Java scripts. Although users with different hardware/bandwidth capabilities are supported, the fact that different users may have different topics of interest is usually overlooked. For instance, most of the site maps are static, which assumes that one map is suitable for all users who visit the Web site for various purposes. This becomes more evident at big portal sites that present a huge amount of information on many diverse subjects.

We observe that different users may have different preferences and purposes for using a Web site. For example, a user may be looking for a particular piece of information, while another user is simply surfing the Web for enjoyment without any well-defined target in mind. Obviously, expectations of these two users from a site map would be different. The former may want to see a more detailed map to navigate to a specific directory containing the information he/she is looking for while the latter may prefer a more abstract map to get a general idea on the contents of the Web site. Therefore it is desirable to have site maps that support multiple levels of granularity.

We also observe that different users may have different topics of interest. Consider an online super-store for instance, a user looking for "hardware tools" and another user interested in "Pokemon" would like to have a more detailed map on these subjects and would not care much about the rest of the site and hence the rest of the map. Hence, the site map needs to be flexible to adjust to the users with different topics of interest. For each user, the area in the site

WWW10, May 1-5, 2001, Hong Kong.
ACM 1-58113-348-0/01/0005.

[†]This work was performed when the author was with NEC USA Inc.

[‡]This work was performed when the author visited NEC USA Inc. The author is currently a Ph.D. student at Department of Computer Science, University of Maryland.

map related to his/her interests should be emphasized and should contain more details.

Based on the above observations, we summarize the requirements for a more desirable and more user-friendly site maps as (1) capable of summarizing the contents of the Web site; (2) capable of visualizing the topology of the Web site, thus supporting navigation; (3) flexible to present the overview of the contents and topology using multiple granularities; (4) content-sensitive to support users with different interests; and (5) possibility to construct such site maps automatically.

In this paper, we present a technique for automatically constructing multi-granular and topic-focused site maps using trained rules based on Web page URLs, contents, and link structure. In these site maps, the Web site topology is preserved, and document importance, indicated by semantic relevancy and citation, is used for prioritizing the presentation of pages and directories. This type of multi-granular site maps can support better interaction for users to scale up and scale down the details. In addition, our technique allows users to specify topics of interest for the system to emphasize the pages/directories relevant to the focused topics. The system will provide more detail on the regions relevant to the focused topic while keeping the rest of the map compact so that the users can visualize their current navigation positions relative to other landmark nodes in the Web site. This functionality is similar to many interactive road maps available on line. The site map construction technique consists of the following steps:

1. Identifying "logical domains" within a Web site: A logical domain is a group of pages that has a specific semantic relation and a syntactic structure that relates them. For example, "a user home page", "a project group Web site", and "an online tutorial on XML" can be viewed as logical domains. The entry pages of these logical domains are candidates for displaying in the site map since they are more informative and usually designed as starting points for navigation of the logical domains. The logical domain entry page identification process described in this paper is based on machine learning techniques.

2. Determining and adjusting page importance based on citation analysis and content relevancy: If a page is relevant to the focused topic, its importance is increased. This step is optional. A set of top ranked pages based on importance are selected to form a site map. The more pages are selected to form a site map, the more detailed the site map gets.

3. Adjusting the boundary and entry pages of each logical domain based on links, directory paths, and importance: We adjust the logical domain boundary so that all the pages in the domain share a common root directory, where the entry page to the domain is located, linked together, and the total importance measurement of all pages in the logical domain reach a given threshold value.

4. Selecting the entry pages of domains with more important pages to present in the site map.

Experiments on real Web data have been conducted to validate the usefulness of the technique. Based on the experimental results and observation of the algorithm behavior, we identify various parameters that have an impact on the site map construction in terms of map granularity and content sensitivity. These parameters are used to construct multi-granular and topic-focused site maps.

The rest of the paper is organized as follows. In Section 2, we define and discuss the characteristics of logical domains. In Section 3, we describe how we construct a set of rules and their scoring functions for identifying logical domain entry pages and methods for defining logical domain boundary. In Section 4, we present experimental results which validate the usefulness of the logical domain extraction process. In Section 5, we describe the site map construction technique by considering page importance and focused topics. In Section 6, we summarize related work and compare with ours. Finally we give our concluding remarks and future work.

2. CHARACTERISTICS OF LOGICAL DOMAIN ENTRY PAGES

In this section, we describe characteristics of a Web page that we consider and use to identify logical domain entry page. Since the logical domain entry pages are later used to form a site map, some criteria need to be set in a way that logical domain entry pages selected are informative, important, and functional. A *physical domain* is defined as a set of pages that are accessed using the same domain. For example, `www.ccrl.com` and `www.ccrl.com/dl99ws/` are hosted by a Web server (or Web servers) of a unique domain name and thus they are in the same physical domain. On the other hand, a *logical domain* is defined as a set of Web pages in a physical domain which, as a whole, provides a particular function or is self-contained as an atomic information unit. The root page of a logical or physical domain is called the *entry page*, which is meant to be the first page to be visited by the users navigating that domain. We identify and summarize some functions of logical domains as follows.

- *Entry page for navigation* : a page with the name `index.html` is the default entry page of a directory for most Web servers (e.g. `www.ccrl.com/index.html` is the entry point for `www.ccrl.com`). It usually contains a site map or links to assist users in navigating the site and thus this type of pages are likely logical domain entry pages.

- *Personal site* : Personal web sites are usually located in a physical domain rather than being physical domains by themselves. `www.ccrl.com/~wen/`, for example, is the entry page for a personal Web site. The personal Web sites by themselves are independent. We view these pages as logical domains.

- *Topic site* : Usually web pages related to a particular topic are grouped together as a *portal*. Such logical domains could be used for class information, seminar announcement, faculty directory, or project Web sites. For example, `www.cs.umd.edu/projects/hcil/` and `www-db.stanford.edu/people/` can be viewed as logical domains by themselves. For a topic site that is a portal, it is expected that it would have a large number of incoming links and outgoing links.

- *Popular site :* Sometimes page in a physical domain may be more popular than the entry page of the do-

main. Such a popular page, indicated by a large number of external incoming links (i.e., citations), may be more appropriate to be used as a logical domain entry page. Example pages of this kind include (1) publication pages of well-known professors, such as `www-db.stanford.edu/~ullman/ullman-papers.html`; (2) hobby pages, such as `~sibel/poetry/poetry.html` in `www.cs.umd.edu`; and (3) tutorial, reference, or direction pages, such as `~pugh/intro-www-tutorial` in `www.cs.umd.edu`.

To observe the characteristics of a logical domain entry page performing the above functionalities, we manually select a set of qualifying pages and examine their URL strings, titles, link structures, and anchor text. We have observed the characteristics of logical domain entry pages as follows:

1: The URL of a user home page tends to end with a user home directory in the form of `~user name/`. Since a user home page is most probably a logical domain entry page, examining URLs to see if they match with this pattern could be effective in extracting logical domains. Note that "~user-name/" and "~user name/index.html" are the same. Before we apply the rules for identifying logical domains, we remove "index.html" from the URLs. Examples which match this observation include `www.ccrl.com/~wen/` whereas `www-db.stanford.edu/~widom/widom.html/` does not conform to this observation.

2: We observe that the URL string of many logical domain entry pages contain words that are associated with the functionalities of logical domains. For example, many logical domains in a university Web site are for projects, classes, seminars, and research groups. Thus, if the URL string contains certain words given in the *topic word list*, such as `people` and `seminar`, and it is not under a user home page, then it is probably a logical domain. The topic word list is domain specific and in our current implementation, a topic word list for the `.edu` domain contains `people`, `users`, `faculty`, `students`, `class`, `seminar`, and `project`. Other topic words include `FAQ` and `Information` for general purpose Web sites, such as `NEC` and `W3C`. URLs matching this rule include `www.cs.umd.edu/users/` while the URL `www.cs.umd.edu/projects/omega/` does not conform to this observation.

3: The URL string of many logical domains end with a "/" since this URL is designed to be an entry page for navigating that directory. Note that for `www.ccrl.com/dl99ws/`, there exists an index page (i.e., index.html) in that directory. `www.cs.umd.edu/projects/omega/` matches with this observation while `/~crespo/publications/meteor/` does not match with this observation. The reason is that we would like to identify `/~crespo/` as an entry page instead of having both URLs as entry pages; which may consequently result in several smaller logical domains. However, we do not eliminate the fact that there can be more than one logical domain within a single user Web site. For example, in `www.cs.umd.edu` we identify `/users/sibel/poetry/` as a possible entry page of a logical domain in addition to `/users/sibel/` because this Turkish poetry portal site is very popular, indicated by a large number of external incoming links.

4: It is unlikely that the logical domain entry page is generated dynamically by the programs. Thus the URL strings containing "cgi" or "?" are automatically eliminated from the candidate set of the logical domain entry pages.

5: If the title of a page contains the phrase "home", "welcome", "homepage", etc., the page tends to be a logical domain entry page. One frequently seen title matching this observation is "Welcome to my homepage".

6: If there is a link pointing to a page with the phrase "home", "go home", "return home", etc. in the anchor, there is a high possibility that the page being pointed to is a logical domain entry page.

7: This is the counterpart of *observation 6*. That is, if a page `A` under `B` points to the page `B` with "home", "go home", "return home" in the anchor, then it is more likely that `B` is an entry page (based on *observation 6*). On the other hand, `A` which is under an entry page `B` is less likely to be an entry page too at the same time.

8: If a page has external incoming links from other physical domains, then this page is more likely to be an entry page. The reason is that people tend to link the entry page of a domain rather than pointing to a specific page. We also observe that the higher the number of external incoming links, the higher the probability of the page being a logical domain entry page. Note that the external incoming link information can be obtained from a system like AltaVista Connectivity Server [1].

9: If the page has a large number of outgoing links, it is very likely that this page is an entry page. Our observation is consistent with the observation and the concept of "fan" proposed by R. Kumar et al. [2]. In [2], only those Web pages with more than 6 outgoing links are considered for topic distillation by assuming that, in general, a good page should not have less than 6 pages. We observe that a page with very few outgoing links is usually a content page rather than an index page (i.e., logical domain entry page).

10: If there is no link from any other page in the same domain to this page, that means the page is designed to be accessed directly, and therefore probably an entry page to a logical domain.

3. EXTRACTION OF LOGICAL DOMAINS

In this section, we describe a set of rules that we constructed based on the observations discussed in Section 2. These rules are intended to be used in identifying logical domain entry page candidates. Unlike the systems based on hand crafted rules or manually assigned weights, such as [3] and many other discussed in the related work section, to partition Web sites for query result organization, we develop a more elegant solution based on a novel decision tree algorithm to achieve higher quality outcomes. The experimental results presented later in Section 4 show that our techniques are effective.

3.1 Constructing Rules and Determining Their Scoring Functions

We have developed a set of rules for identifying logical domain entry pages based on the available Web page metadata, such as title, URL string, anchor text, link structures, and popularity indicated by citations. The rules in regular expression and their scoring functions are summarized in Figure 1 and these ten rules are corresponding to the ten observations discussed in Section 2. Each rule has an associated scoring function. When a rule matches with a page, the scoring function of the rule is applied to the page by increasing or decreasing its score by a number. Every page is assigned with a score. The higher the score of a page, the

```
Rule#1   url                              : "/~[^/]*/?$"                  : +105
Rule#2   url  : "^[^~]*/(people|users?|class(es)?|projects?|seminars?)/$": +11
Rule#3   url                              : "^[^~]*/$"                    : +70
Rule#4   url                              : "/cgi\-bin/"                  : 0
Rule#5a  title                            : "\bhome\b"                    : +45
     5b  title                            : "\bweb\b.*\bpage\b"           : 0
     5c  title                            : "\bwelcome\b"                 : +8
Rule#6a  incoming link anchor text        : "^home$"                      : +22
     6b  incoming link anchor text        : "\bgo\b.*\bhome\b"            : +8
     6c  incoming link anchor text        : "\breturn\b.*\bhome\b"        : 0
Rule#7a  outgoing link anchor text        : "^home$"                      : +37
     7b  outgoing link anchor text        : "\bgo\b.*\bhome\b"            : 0
     7c  outgoing link anchor text        : "\breturn\b.*\bhome\b"        : 0
Rule#8a  external incoming link count     : >0                            : +363
     8b  external incoming link count     : >1                            : +404
     8c  external incoming link count     : >6                            : +332
     8d  external incoming link count     : >25                           : +376
     8e  external incoming link count     : >95                           : +262
     8f  external incoming link count     : >142                          : +321
Rule#9a  outgoing link count              : >17                           : +117
     9b  outgoing link count              : >122                          : +61
     9c  outgoing link count              : >257                          : +413
     9d  outgoing link count              : >313                          : +811
     9e  outgoing link count              : >431                          : +251
Rule#10a internal incoming link count     : <8                            : +125
     10b internal incoming link count     : == 0                          : +37
```

Figure 1: Rules and scoring functions for identifying logical domain entry pages

more likely that a page is a logical domain entry page.

In [3], the score functions are manually specified and adjusted. Here we present a mechanism for automated scoring function assignment using machine learning techniques. The procedure to determine scoring functions consists of three steps. We now explain the details of each step in the following subsections.

3.1.1 Transformation of the Score Assignment Problem

We transform the problem of assigning scoring functions to each rule to the problem of assigning weights to the features (attributes). Weight assignment to the attributes is a common problem in machine learning where several algorithms have been proposed in the literature [4, 5]. For computing the weights of attributes in a data set, we need a training data set which consists of a sufficient number of Web pages (i.e., *examples*) in the form of the attribute value vector.

3.1.2 Preparation of the Training Data

After the Web pages are transformed into vectors of attribute values, human input that provides the likelihood of a page being a logical domain entry page (through assignment of classes) is needed. This information is later used to guide the process of assigning weights to the attributes. The assigned classes will affect the results of the classification, so we attempted to be as discreet as possible while assigning classes to the pages. To increase the accuracy of the classification, we assigned one of ten classes to the page among 10 different values instead of just classifying them as logical domain or not. The motivation behind this approach is the fact that lots of pages can not be classified as logical domains or otherwise. Thus, assigning some degree of "being logical domain" to each page seems more appropriate than rejecting them as logical domains. It is impractical to assign classes to all the pages in a certain domain. We select a subset of the HTML pages which is sufficient to create the training data. The pages which we could not decide their classes are dropped from the training data.

3.1.3 Assigning Weights to the Attributes

Given that all attributes are not of the same importance during the classification process, we need to measure the relative importance of each attribute. One immediate thought is using neural networks. Neural network algorithms would generate a "black box" (i.e., a matrix of connected nodes and weights for their links) which captures the hidden mapping between the input (i.e., trainning data set) and output (i.e., human feedback represented as classes). However, we are also interested in the importance of each rule in addition to a trained neural network classifier. In this paper, we introduce an approach to assigning weights to the attributes using the information gain of each attribute in the *ID3* classifier algorithm, proposed by Quinlan [6], as its weight. This approach provides us not only classification capability, but also the option to examine the weight of each rule to study its relative importance in logical domain extraction process. Furthermore, information gain for each attribute can be computed more efficiently in general with respect to the training time required for neural network algorithms.

In [7], Quinlan uses the information theory that underpins the criterion to construct the best decision tree for classifying objects as follows:

> "The information conveyed by a message depends on its probability and can be measured in bits as minus the logarithm to base 2 of that probability."

Let S be the set of n instances and let C be the set of k classes. Let $P(C_i, S)$ be the fraction of the examples in S that have class C_i. Then, the expected information from

this class membership is as follows:

$$Info(S) = -\sum_{i=1}^{k} P(C_i, S) \times log(P(C_i, S))$$

If a particular attribute A has v distinct values, the expected information required for the decision tree with A as the root is then the weighted sum of expected information of the subsets of A according to distinct values. Let S_i be the set of instances whose value of attribute A is A_i.

$$Info_A(S) = \sum_{i=1}^{v} \frac{|S_i|}{|S|} \times Info(S_i)$$

Then, the difference between $Info(S)$ and $Info_A(S)$ gives the information gained by partitioning S according to testing A.

$$Gain(A) = Info(S) - Info_A(S) \quad (1)$$

We provide only a summary of information gain in *ID3* here. For more detailed description, please see chapters 1, 2, and 3 in [7]. We will now discuss how to handle *continuous valued attributes*.

Calculating the information gain is easier for nominal attributes since there is a finite number of distinct values of each nominal attribute, and each instance is associated with one of those values. In our case, each categorical attribute takes one of two values (either 0 or 1) and the number of classes is set to be 10 ($v = 2$ and $k = 10$ in the formulas presented in the previous section).

On the other hand, it is not easy to compute the information gain for continuous valued (linear) attributes. To overcome the problem of dealing with continuous valued attributes by subsequent binary discretization which maximizes the information gain for that attribute, we computed information gain for each cutoff value where each distinct value of that attribute is treated as a cutoff point. Let an attribute A have v distinct values $A_1, A_2, \ldots, A_v$. For a specific attribute value A_t, let S_1^t be the set of instances whose attribute value is less than or equal to A_t and S_2^t be the rest of the instances. Note that to maximize the information gain obtained by the attribute A, we have to find the minimum value of $Info_A$. Then,

$$S_1^t = \{example\ e \mid e[A] \leq A_t\}$$

$$S_2^t = \{example\ e \mid e[A] > A_t\}$$

$$Info_A^t(S) = \sum_{i=1}^{2} \frac{|S_i^t|}{|S|} \times Info(S_i^t)$$

$$Info_A(S) = min(Info_A^t(S))\ for\ all\ t \in \{1, \ldots, v\}$$

By subsequent applications of this process, we find successive cutoff values which maximize the information gain for this attribute. Each cutoff value is used as an additional rule for the external incoming links. So, the rule about external incoming links (i.e., observation 8) is transformed into five rules. The specific numbers of the level of discretization are chosen so that the information gain is maximized.

After computation, we use the weights of the attributes as the scoring functions of the corresponding rules. For presentation purposes, we mapped the information gains computed (which is a number between 0 and 1) to integer numbers by multiplying them by 1,000. The rules with the automatically assigned scoring functions are presented in Figure 1.

3.2 Defining Boundary of a Logical Domain

After all pages are scored, a certain percentage or number of pages with higher scores are chosen as entry page candidates to form logical domains. The boundaries of logical domains are identified using directory path information and link structure. The boundary definition tasks start with using directory path information to assign all pages under certain entry pages by following the intuition that the pages in a logical domain would be in a directory, where the entry page is at the top level. A page must be under the same directory as the entry page or in a subdirectory under the entry page because this is how usually people organize HTML files in the directories.

With consideration of directory path information alone, we observe that some logical domains may have isolated subdomains. That is, we may not be able to navigate from a logical domain entry page to all pages in the same logical domain. In Figure 2, three logical domains are created. The pages in the shadowed area in D_3 are connected by following the links from a page in D_1 and there is no link from the pages in the other part of D_3 to that area. Note that we can only consider the links within D_3 because otherwise D_3 is not a connected subgraph by itself. For example, we can not consider the link from P_3 to P_4 in D_1 and then pointing to the shadowed area as shown In Figure 2. Therefore, we reassign the pages in the shadowed area to other domain; otherwise the shadowed area of D_3 can not be accessed by its logical domain entry page P_3. With these considerations, we need to check accessibility through links in addition to directory path relationship.

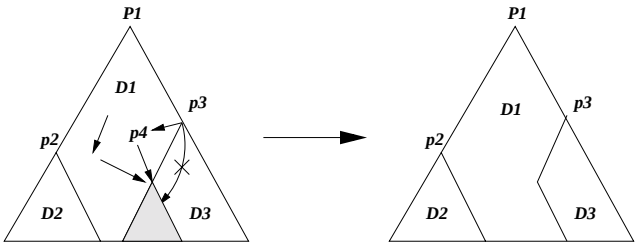

Figure 2: The issue of logical domains with isolated sub-domains

The detailed algorithm of this approach is described below. It takes the results from Section 3 (i.e. all n pages and their scores) and two parameters, the initial number of entry page candidates, k, and the level of links to follow for verifying accessibility, *radius*.

Step 1: Select k pages, $P_1 \ldots P_k$ with the highest score as entry page candidates.

Step 2: Build $Parent\_Children\_List$ for $P_1 \ldots P_k$ based on the directory path. P_i is the parent of P_j if $P_i.hostdir$ is the longest prefix of $P_j.hostdir$. $P_i.hostdir$ = URL of P_i without the file name at the end.

Step 3: Assign $P_{k+1} \ldots P_n$ to be under one of the entry pages $P_1 \ldots P_k$ to form logical domains $D_1 \ldots D_k$. P_j is assigned to be under P_i if and only if $P_i.hostdir$ is

Term	Description
SF_{auto}	Rules with the automated assigned scoring functions (Figure 1)
R_{auto}	Pages returned as logical domain entry pages using the rules with automated scoring functions
TD	Training data (pages that are assigned classes by human beings)
LD_{deg}	Degree of being logical domain

Table 1: Summarization of terms and their descriptions

the longest prefix of $P_j.hostdir$. P_i is the entry page of the logical domain D_i and P_j can be reached from P_i by following r hyperlinks within the union of D_i, P_j, and D_j, where r is the radius specified for checking link accessibility.

Step 4: Output all logical domain entry pages, P_i and their corresponding domains, D_i.

Note that *Step 3* implies that the assignment of pages to logical domains is performed in the bottom up fashion recursively until all conditions are satisfied. For the complete study of various logical domain extraction algorithms, please see [3].

4. EVALUATION EXPERIMENTS ON LOGICAL DOMAIN EXTRACTION

In this section, we present the experimental results on evaluating the proposed logical domain extraction technique. We will use abbreviations for ease of presentation. Table 1 summarizes the terms and descriptions used in this section.

4.1 Training Data

To compute the information gain for each attribute, we created training data using the 3,047 pages collected from `www-db.stanford.edu`. Nearly one-third of the pages in `www-db.stanford.edu` are selected for manual inspection to assign sample pages to a specific class according to whether or not a page can be used or viewed as a logical domain entry page. This class represents a degree for being logical domain. A class 0 refers to the pages that we are certainly sure that the page is not a logical domain, and a class 9 refers to those pages that are certainly believed to be logical domains. If we can not make a decision on a page, we dropped these pages from the training data set. As a result of this manual classification, the training data we created consists of 439 pages, which have 10 different classes describing the degree of logical domain. The distribution of classes in the training data are given in Table 2. Note that in the manual classification, those pages which are ambiguous to classify are dropped.

4.2 Interpretation of Rules and Scoring Functions

The automatic scoring mechanism presents the feedback to validate (or invalidate) our observations discussed in Section 2. As a result of our experiments, some of our intuitions tend to be incorrect as many rules are assigned a scoring function of 0, indicating these rules have no effect in logical domain identification process. Note that we initially thought that URL strings, titles, and anchor text are as important as the links. However, the experimental results have showed

Class	0	1	2	3	4	5	6	7	8	9
Number	148	123	29	7	10	24	30	21	35	12

Table 2: Number of training examples for each class

TOP n pages	Min. Class	Total # of Pages	# of Same Docs	Precision	Max Rank
200	9	12	12	100%	22
200	8	47	47	100%	197
200	7	68	68	100%	197
200	6	98	93	94%	197
200	5	122	113	92%	198

Table 3: Comparison of the results using SF_{auto} with TD

that the URL strings are still relatively important while titles and link anchor text are proven not significant. Our interpretation is that URL and links are applied to all Web pages. However, the specific words used in these rules can be applied to only a subset of the Web pages. For example, the rule examines whether or not the title of a page contains "project", "seminar", etc., can not be applied to personal home pages. Also note that the rule 7 is supposed to be assigned a scoring function of ≤ 0 based on our initial intuition but a scoring function of 37 is assigned after the automatic scoring process. This indicates that our observation 7 is not correct.

4.3 Experiments on Logical Domain Extraction from Real Web Sites

We have conducted experiments on `www-db.stanford.edu` , `www.cs.umd.edu`, and `www.w3c.org`. The pages collected from these domains were crawled from the root pages by following the links within the same domains. 3047, 18872, and 13356 pages were collected from these three sites, respectively.

The first experiment is to evaluate the effectiveness of rules. The results are satisfactory. The scores of the top 30 pages extracted from `www-db.stanford.edu` are given in Figure 3. As we can see, most home pages of professors and projects are selected; which is a desirable outcome. Note that four pages are selected from the Web site under the home page of Professor Ullman while not all home pages of the database group members are selected. This observation shows a good property of our rule selection - more popular, hence, presumably more important pages are more likely to be selected. Thus, we see that logical domains are very desirable for Web site map construction.

Our technique aims at considering both link structures and contents. Although we do not examine the document contents directly, our rules examine the titles and anchor text to see if they contain some "topic words" that are typically used in logical domain entry pages. The rules also examine the number of external incoming links and use that information as an indicator to judge the importance of each page. This concept is similar to and is consistent with the concept of "topic distillation" for organizing Web query results proposed by J. Kleinberg [8]. In the experiment results, many popular pages on particular topics out-score most personal and project Web sites. Some logical domains are identified mainly by their popularity.

One of more "scientific" ways to evaluate the effectiveness of decision tree algorithm in automating attribute weight adjustment is to compare training data with the experimental

```
Score           Document
==== ============================================
2879 gio/1994/vocabulary.html
2420 people/
2304 index.html
2282 ~ullman/
2088 tsimmis/
1932 warehousing/
1885 ~testbed/python/manual.1.3/lib/Function-Index.html
1869 ~widom/
1789 pub/gio/
1789 dbseminar/
1719 warehousing/publications.html
1705 ~junyang/seven/
1699 ~ullman/fcdb.html
1670 lore/
1639 people/widom.html
1634 ~testbed/python/manual.1.3/lib/Concept-Index.html
1600 pub/keller/gates-map.html
1594 people/gio.html
1512 tsimmis/tsimmis.html
1493 ~widom/widom.html
1485 ~cho/
1475 people/hector.html
1475 warehousing/warehouse.html
1448 ~widom/pubs.html
1448 ~sergey/
1448 ~ullman/ullman-papers.html
1448 ~zhuge/
1448 ~ullman/ullman-books.html
1448 ~zhuge/zhuge.html
1413 SKC/
==== ============================================
```

Figure 3: Logical domain entry page identification results for `www-db.stanford.edu` using SF_{auto}

results. We conducted an experiment to compare the logical domain entry pages that are returned by the rules with the training data. We checked the top 200 pages in R_{auto} and counted how many of the pages in TD have a LD_{deg} above a certain threshold. We tried five different thresholds for the value of LD_{deg}. In Table 3, we present the number and percentage of the pages in TD that are found in top 200 pages in R_{auto}. In this table, "Min. Class" refers to the minimum level of degree of logical domain that we search for, "Total # of Pages" refers to the number of pages which have a degree of logical domain greater than or equal to minimum class, "# of Same Docs" refers to the number of pages found in R_{auto} which have a degree of logical domain greater than or equal to minimum class, "Precision" refers to "# of Same Docs" divided by "Total # of Pages", and "Max. Rank" refers to the maximum rank of the page (the order of the last page) which have a degree of logical domain greater than or equal to min.class found in R_{auto}. The results indicate that top 200 pages contain 92% of the pages which have a LD_{deg} 5 or more. This indicates that the automatic scoring functions perform very effectively. Later we will also perform evaluation on applying these scoring functions to pages in other physical domains that are not used for training and their characteristics are somehow different. The experimental results also show the effectiveness of our approach.

The set of rules with the automated scoring functions performed very well on `www-db.stanford.edu` domain. In order to show that the rules are effective in other domains, we ran the process of identifying logical domain entry pages on other domains such as `www.cs.umd.edu` and `www.w3.org`.

Description	Number
User home pages	36
Project pages	26
Hobby pages	10
Index Pages	26

Table 4: Top 100 pages in `www.cs.umd.edu` using SF_{auto}

In the experimental results on `www.cs.umd.edu`, the logical domains identified are mostly Web sites for people, projects, and classes. The results indicate that most pages in these two Web sites are organized in such ways for users to navigate. We evaluated the top 100 pages returned by the application of rules by looking each of them one by one. 95% of them turned out to have a LD_{deg} of more than 4. We will not give the complete list of the logical domain entry pages identified but the distribution of the top 100 pages. Table 4 presents the number of pages that are user home pages, project pages, pages related to hobbies, and index pages of the users or tutorials or publications, etc.

On the other hand, `www.w3.org` behaves more like a single entity. We observed that the logical domains in `www.w3.org` are defined based on *subjects* rather than "entities" as we observed in two other university Web sites. Some representative logical domains identified are as follows:

```
www.w3.org/MarkUp/        www.w3.org/Protocols/
www.w3.org/XML/           www.w3.org/People/
www.w3.org/TR/            www.w3.org/Provider/
www.w3.org/Tools/         www.w3.org/RDF/
```

The results obtained show that automated scoring functions perform well also for other domains. Although the numbers occurring in the scoring functions are specific for only `www-db.stanford.edu` domain, they produce good results for other domains, too. This supports our claim about the power of rules and the automated scoring functions to extract logical domain entry pages.

5. CONSTRUCTION OF SITE MAPS

One option to construct a site map is simply taking the output results from the previous section and produce a map. This option is probably desirable when applying to the area of query result organization. However, when we apply the logical domain definition technique to site map construction we need to consider the additional requirements identified in Section 1. In the next three sub-sections, we present methods for extraction of logical domains for the specific purpose of constructing dynamic site maps.

5.1 Constructing More Informative Site Maps

When we identify a logical domain entry page in a physical domain, whether or not it enables easy navigation and accessibility is an important factor. However, when we design a site map, the most important factor is that the nodes in the map (selected from the entry pages) must be informative and representative. Therefore, the first step of constructing a site map is to determine the importance of each logical domain entry page, which is based on the total importance of all pages in the logical domain it represents. A logical domain consisting of few important pages, indicated by citation, may be more important and informative than a

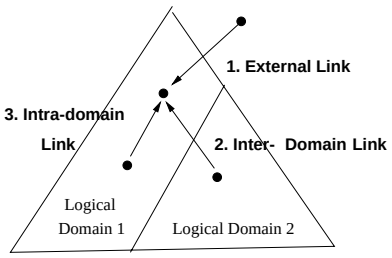

Figure 4: Types of links with respect to the definition of logical domains

logical domain consisting of a large number of unimportant pages.

Once logical domains are defined, we classify the types of incoming links as follows and shown in Figure 4:

- External links: Direct incoming links from pages in different physical domains from where the linked page is in.
- Inter-logical domain links: Direct incoming links from pages in different logical domains from the one the linked page is in. Note that the linking page and the linked page must be in the same physical domain.
- Intra-logical domain links: Direct incoming links from pages in the same logical domain where the linked page is in.

We believe the importance of citation implied by these three types of links should be in the order of external links, inter-logical domain links, and intra-logical domain links. These intuitions can be justified with the analogy; "an internationally renowned article is in general more important than a domestically renowned article. And a domestically renowned article is in general more important than a statewide renowned article." We define the importance of a page p, $Importance(p)$, as follows:

$W_{ext} \times number\_of\_external\_link +$
$W_{inter} \times number\_of\_inter\_logical\_domain\_link +$
$W_{intra} \times number\_of\_intra\_logical\_domain\_link$

where W is the weight assigned to the importance of citation implied by each type of links and $W_{ext} \geq W_{inter} \geq W_{intra}$.

To select more important and informative entry pages and domains for constructing a site map, a new parameter *min_importance* is introduced for specifying the minimum required importance of a logical domain. That is, a logical domain formed must have a certain level of importance as a whole so that its entry page appearing in the site map is representative and informative. The detailed algorithm is as follows (steps 1 to 3 are for identifying initial logical domains as described in the previous section):

Step 1: Select k pages, $P_1 \ldots P_k$ with the highest score as entry page candidates.

Step 2: Build $Parent\_Children\_List$ for $P_1 \ldots P_k$ based on the path. P_i is the parent of P_j if $P_i.hostdir$ is the longest prefix of $P_j.hostdir$. $P_i.hostdir$ = URL of P_i without the file name at the end.

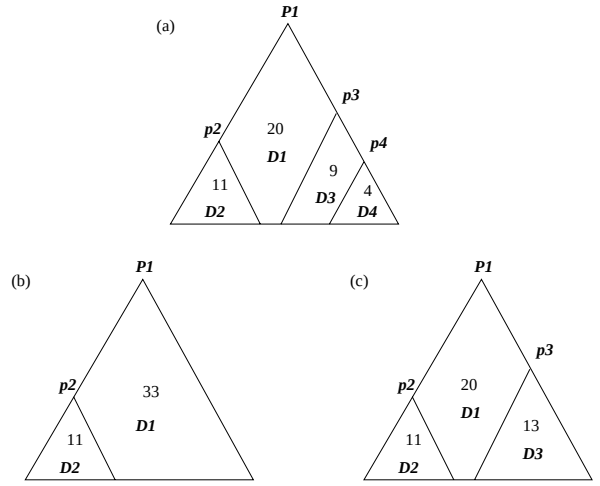

Figure 5: (a) Initial assignment; (b) adjustment by checking min. domain importance without dynamic page reassignment; and (c) adjustment by checking min. domain importance with dynamic page reassignment

Step 3: Assign $P_{k+1} \ldots P_n$ to be under one of the entry pages $P_1 \ldots P_k$ to form logical domains $D_1 \ldots D_k$. P_j is assigned to be under P_i if only if $P_i.hostdir$ is the longest prefix of $P_j.hostdir$ and P_j can be reached from P_i by following r hyperlinks within the union of D_i, P_j, and D_j, where r is the radius specified for checking link accessibility.

Step 4: Calculate *importance* for every page, $P_1 \ldots P_n$.

Step 5: Merge the pages in D_j and P_j which can be reached from P_i by following r hyperlinks within D_i with D_i recursively from the bottom to the top if the summation of importance for all pages in D_j and P_j is less than *min_importance*, where P_i is the immediate parent of P_j. Recalculate *importance*() for every page in D_i and D_j, P_i, and P_j if D_i and D_j are merged.

Step 6: Output all logical domain entry pages, P_i, and their corresponding domains, D_i.

Note that we must perform the entry page definition task in a bottom up fashion in step 5. Let's use Figure 5(a) as an example for illustration. P_1, P_2, P_3, and P_4 are the entry pages for the logical domains D_1, D_2, D_3, and D_4, respectively. The numbers indicate the total importance score in each initial logical domain. We identify the $Parent\_children\_List$ as (P_1, P_2), (P_1, P_3), and (P_3, P_4). In step 5, we perform adjustments by reassigning those domains with importance less than the minimal required importance. For example, we would like to consider only the domains with importance greater than 10. One way is to just remove all entry pages whose importance is less than 10. However, one drawback is that we will remove both D_3 and D_4 as shown in Figure 5(b). With this scheme, the domains closer to the root page will gather a lot of released pages from the lower domains. However, with dynamic page reassignment in our algorithm, D_4 will be merged into D_3.

The reason we need to recalculate the importance of certain pages while domain mergers occur (in Step 5) is that

Figure 6: (a) A site map for www-db.stanford.edu (b) The same site map with focused topic *ullman*

Experiment #	Initial Number of Entry Pages	Min. total Importance	Link Radius	Logical Domains Identified	Avg Logical Domain Size	Site Map Depth
1	200	10	3	111	27.39	5
2	200	30	3	58	52.41	4
3	200	50	3	42	72.38	4
4	200	70	3	34	89.41	4
5	200	90	3	30	101.33	4

Table 5: Results on `www-db.stanford.edu` using different minimum total importance scores

some inter-logical-domain links may be "down-graded" to intra-logical-domain links. Thus, the importance of some pages needs to be adjusted.

By considering importance of pages, each domain is restricted to have minimum required importance. The algorithm is tunable. By increasing the value of *min_importance*, we can drop those less significant domains and pages. In Figure 6(a), we show a site map for `www-db.stanford.edu` by setting *radius* to 3 and *min_importance* to 30. In this map, 58 logical domain entry pages are selected to construct the site map for representing 3047 pages in www-db.stanford.edu.

5.2 Topic-Focused Site Maps

A site map needs to be adaptive to the subject of interests of a user. For example, the site maps for `www.nec.com` with respect to "computer" and "multimedia" should be different. We modify our algorithm for considering the following two requirements

- The logical domains that have more pages related to focused topic should be emphasized more and have more detailed information.
- Although the logical domains related to focused topic are emphasized, the topology of the whole physical Web site and other logical documents are still needed for visualizing broader topic space and guiding users to navigate the Web site.

In this section, we present a variation of the site map construction algorithm in which the contents of pages are considered. We can define a relevance measurement function $Relevance(p, topic)$ which returns a relevance score for the page p with respect to *topic*. In this paper, we use the scores returned by `HotBot` using *topic* to query and limiting the search within a given domain. For example, we use the keyword `ullman` to query `HotBot` by searching only `www-db.stanford.edu` in our experiments. In the previous subsection, we define a function for measuring the importance of a page p, $Importance(p)$, based on external links, inter-logical domain links, and intra-logical domain links. Now we modify the function of the importance of a page p with respect to a given topic, $Importance\_topic(p)$, as

$$(1 + 3 \times Relevance(p, topic)) \times Importance(p)$$

In this formula, the importance of a page irrelevant to the focused topic will remain the same while the importance of pages relevant to the focused topic can be increased up to 300%. By adjusting this percentage value, we can control the degree of a map being topic-focused. To consider a focused topic, two new variables *topic* and $importance_{topic}$ and a new parameter $min\_importance_{topic}$ are introduced for specifying the minimum required importance of all logical domains with respect to a topic. The topic-focused site map construction algorithm is similar to the algorithm described in the previous sub-section. The only difference is to use $Importance_{topic}()$ instead of $Importance()$ in step 5. Thus we skip the details of the algorithm for brevity.

By considering the page relevant to the focused topic, logical domains with relevant contents are more likely to be selected. Thus, there are more nodes in the map which are relevant to the topic. In Figure 6(b), we show a site map for `www-db.stanford.edu` with focused topic of `ullman` by setting *radius* to 3 and *min_importance* to 30. In this map, 3 additional logical domain entry pages are selected,

Figure 7: An abstract map for `www-db.stanford.edu`

as highlighted, in `~sergey/` and `~ullman/`. The additional logical domains provide more information to the users who are interested in `ullman`. Note that the logical domain list in the rest of the map is not changed because there are not a lot of pages relevant to the focused topic. Our technique successfully emphasizes the areas in the Web site which are relevant to the focused topic. Note that additional improvements can be made by better user interface design and by highlighting the areas relevant to the focused topic to guide users for quick navigation.

5.3 Site Maps with Multi-Granularity of Details

In Table 5 we present a set of experimental results to show the behavior of the algorithms with respect to selection of $min_{importance}$ values. One desirable property is that the number of logical domains in the site map is a function of $min_{importance}$. By selecting different values for the minimum total importance scores, we can create site maps at different levels of granularity in terms of detail. Note that in the more detailed map, the summarized site topology may have both a larger breadth and a greater depth. Table 5 shows that the depth of the most detailed site map is 5, which is greater than the depth of all other site maps.

Figure 8: A detailed map for `www-db.stanford.edu`

The importance of logical domains is used to prioritizing the presentation of pages and directories. Based on these adjustments on the parameter values, we can define site maps with different levels of granularity in terms of detail. In Figures 7 and 8 we show two variations of the site map for `www-db.stanford.edu`. The anchor text is based on URLs' titles. If a document does not have a title, then we use the file name in the URL instead. The folders indicate there are sub-directories (sub-folders) under them. When the user clicks on a folder, the documents and sub-folders (if any) under it are shown. The user can click on the anchors in the site map navigator to retrieve the pages of his/her interests.

The trace of user's navigation steps are illustrated by footprints. The fresher footprints indicate the pages just visited, and these footprints fade as the user visits new pages. The map in Figure 8 has more detail (by setting minimum required importance to 10) than the map in Figure 7 (by set-

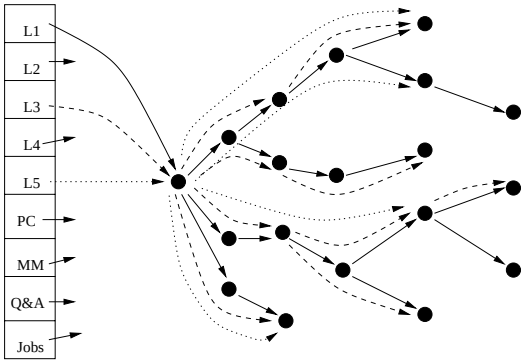

Figure 9: Data structure for supporting multi-granularity of Details

ting minimum required importance to 80). Note that the more abstract site map is not necessarily a subset of the map with more details.

5.4 Implementation Status

To support multi-granular and topic-focused site maps, one way is to create several separate maps. When the user requests a specific map, one of the maps is shown to the user. However, this results in a large overhead due to duplication. In Figure 9, we show one possible data structure that supports the functionalities of multi-granular Web site maps without a large overhead of duplicated data. On the left of the figure, a vector is used for storing the initial entry pointer for different types or granularity of site maps. $L1 \ldots L5$ point to five site maps with different levels of granularity in detail, in which $L1$ points to the most detailed map while $L5$ points to the most abstract map. There are also pointers to site maps for specific topics, such as `PC`, `Multimedia`, `Q&A`, and `Jobs`. If the user requests the most detailed map, the map can be generated dynamically by following the pointers for $L1$. Similarly, if the user requests different maps, which are at different levels of granularity or for specific topics, a map of users' focused topic can be generated by following the corresponding pointers.

The site map construction technique presented here is being integrated with the site map navigator. The site map navigator is a user interface currently used in NEC BigLobe (in Japanese)[9]. Figure 10 shows an operational demonstration. On the left of Figure 10, a site map for BigLobe FunSites is used to assist users in navigating interesting sites in BigLobe, such as horse racing, photo, etc. The site map navigator takes a hierarchical data structure and generates a navigation interface based on Java Script.

The purpose of this site map is to give users an overview of the contents and topology of the Web site. Once the users locate a logical domain of his/her interests, he/she can then use the entry page of that logical domain to start navigating the pages in the domain using hyperlinks in the page.

6. RELATED WORK

The problem of constructing Web site maps has two major issues: how to summarize the Web sites or Web space and how to visualize the abstract information. Our work addresses both issues.

A popular approach for the visualization of abstract information is to use Focus+Context techniques. In this technique the information of interest to the user is shown in detail, smoothly integrated with other context information. By balancing local detail and global context, this technique can simultaneously display information at multiple levels of abstraction. This technique has been applied to WWW, such as [10, 11]. Several systems that integrate data simplification and visualization techniques for the WWW have been developed including *WebCutter* [12] and [13]. Other systems that aggregate similar information together to form a high-level structure for the presentation have been proposed, including *WebBook* [14], [15], [16], and [17].

Many of the above-mentioned work and systems are based on an assumption that the input data is a set of query results. The site maps or overview diagrams they derive are by exploring the neighborhood nodes and links. Compared with the above-mentioned existing work, our work is novel in defining logical domains in a Web site. The Web site summarization unit is the logical domain rather than a single page. With logical domains and link accessibility analysis, our approach can not only preserve the topology of a Web site in the site map but also summarize the contents of the Web site with respect to semantics. Another novelty of our work is that we integrate logical domain extract with distillation. The links in our system are assigned different importance depending on whether they are inter-logical domain links, intra-logical domain links, and inter-physical domain links.

7. CONCLUDING REMARKS

In this paper, we present a technique for constructing multi-granularity and topic-focused site maps. Our technique has the following advantages over other existing work: it is (1) capable of summarizing the contents of the Web site and visualizing the topology of the Web site, thus supports navigation; (2) flexible to present the overview of the contents and topology using multiple granularity; and (3) content-sensitive to support users with different interests. More importantly, our technique can be automated and has many tunable parameters for customization of Web site construction. In addition, the criteria for selecting logical domain entry pages to form site maps are automatically adjusted by machine learning techniques. Future work includes development of techniques for identifying mirror sites and identical documents with symbolic links for further improvement of the experimental results as well as extending this work to a group of Web sites, such as all Web sites within the NEC organization.

Acknowledgement

The authors would like to express their appreciations to `www-db.stanford.edu`, `www.cs.umd.edu`, and `www.w3c.org` for their data used in our experiments. The experimental results presented in this paper are for the purposes of scientific research only.

8. REFERENCES

[1] Krishna Bharat, Andrei Broder, Monika Henzinger, Puneet Kumar, and Suresh VenKatasubramanian. The Connectivity Server: fast access to linkage information on the Web. *Computer Networks and ISDN Systems*, 30(1-7):469–477, May 1998.

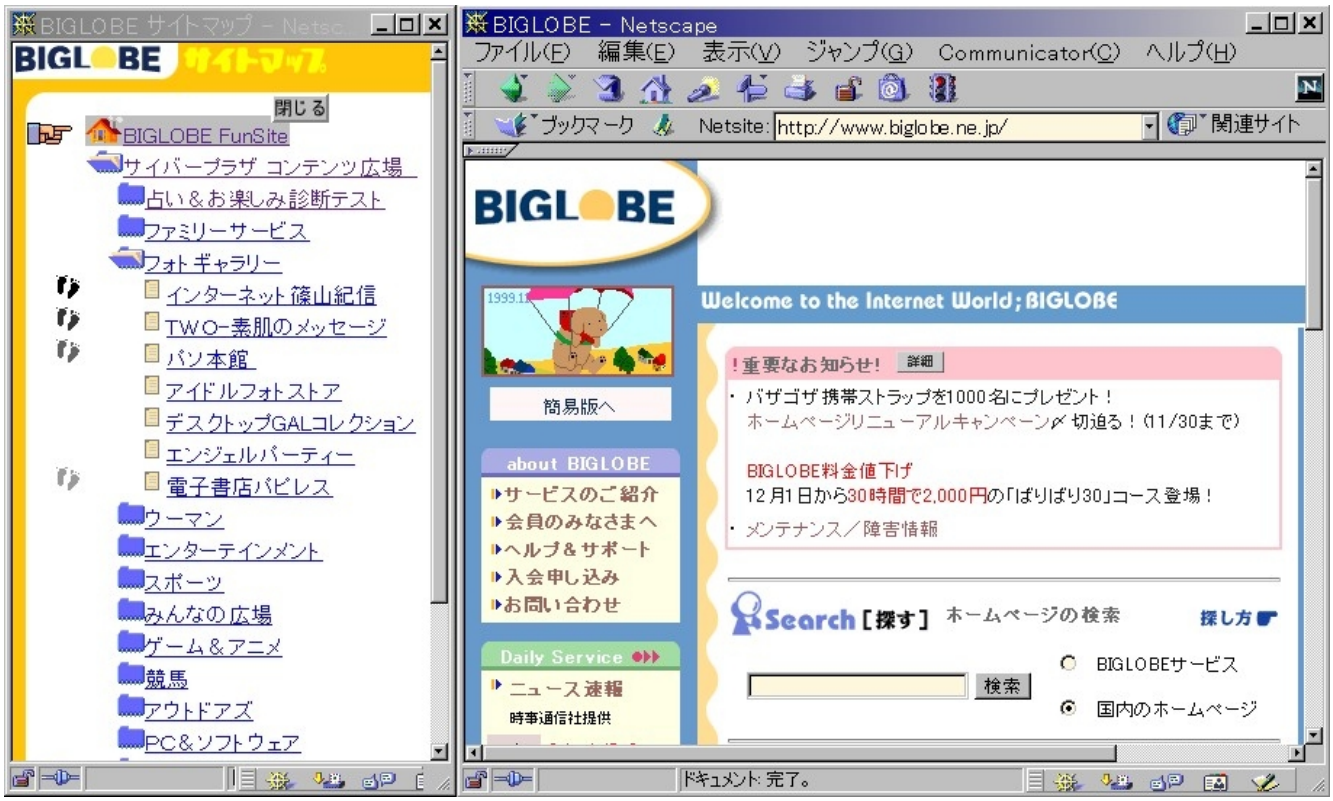

Figure 10: Usage of the site map construction

[2] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Trawling the Web for Emerging Cyber-Communities. In *Proceedings of the 8th World-Wide Web Conference*, Toronto, Canada, May 1999.

[3] Wen-Syan Li, Okan Kolak, Quoc Vu, and Hajime Takano. Defining Logical Domains in a Web Site. In *Proceedings of the 11th ACM Conference on Hypertext*, pages 123–132, San Antonio, TX, USA, May 2000.

[4] J. D. Kelly and L. Davis. A Hybrid Genetic Algorithm for Classification. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 645–650, 1991.

[5] P. Langley and A. L. Blum. Selection of Relevant Features and Examples in Machine Learning. *Special Issue of Artificial Intelligence on Relevance*, 1994.

[6] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1, 1986.

[7] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, California, 1993.

[8] Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, January 1998.

[9] NEC Corporation. *Information available at http://www.biglobe.ne.jp/*.

[10] S. Mukherjea and Y. Hara. Focus+Context Views of World-Wide Web Nodes. In *Proceedings of the 8th ACM Conference on Hypertext*, pages 187–196, Southampton, England, April 1997.

[11] Sougata Mukherjea. WTMS: a system for collecting and analyzing topic-specific Web information. In *Proceedings of the 9th World-Wide Web Conference*, Amsterdam, Netherland, May 2000.

[12] Y. Maarek and I.Z.B. Shaul. WebCutter: A System for Dynamic and Tailorable Site Mapping. In *Proceedings of the Sixth International World-Wide Web Conference*, pages 713–722, Santa Clara, CA, April 1997.

[13] D. Durand and P. Kahn. MAPA: a System for Inducing and Visualizing Hierarchy in Websites. In *Proceedings of the Ninth ACM Conference on Hypertext*, pages 66–76, Pittsburgh, Pa, June 1998.

[14] Stuart K. Card and George G. Robertson and William York. The WebBook and the Web Forager: An Information Workspace for the World-Wide Web. In *Proceedings of the 1996 ACM CHI Conference*, pages 111–117, Vancouver, BC, Canada, April 1996.

[15] C. Chen. Structuring and Visualizing the WWW by Generalised Similarity Analysis. In *Proceedings of the 8th ACM Conference on Hypertext*, pages 177–186, Southampton, England, April 1997.

[16] C. Chen and M. Czerwinski. From Latent Semantics to Spatial Hypertext - An Integrated Approach. In *Proceedings of the Ninth ACM Conference on Hypertext*, pages 77–86, Pittsburgh, Pa, June 1998.

[17] L. Terveen and H. Will. Finding and Visualizing Inter-site Clan Graphs. In *Proceedings of the ACM SIGCHI '98 Conference on Human Factors in Computing Systems*, pages 448–455, Los Angeles, Ca, April 1998.