University of Alberta

Web Graph Algebra for Interactive Ad-hoc Mining

by

Jiyang Chen ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2004

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-612-95720-9
Our file Notre référence
ISBN: 0-612-95720-9

Giving one a fish is only for a meal, but teaching one to fish can benefit one for a life.
– Chinese Proverb

Acknowledgements

First and foremost, I'd like to thank my supervisors Dr. Osmar Zaïane and Dr. Randy Goebel for their patience, advice and guidance. Next, I'd like to thank my family, my dear father, mother and sister for their support in all phases of my life. Also, thanks to my colleagues Lisheng Sun and Tong Zheng for designing and improving the WEBKVDS system and WEBFRAME project. Finally, I would like to thank all the people who helped me in my master study.

Contents

List of Figures

Chapter 1

Introduction

As of 1999 [35], the publicly indexable World Wide Web(WWW) contains about 800 million pages, encompassing 6 terabytes of text data on about 3 million servers, one in every 269 IP addresses. And it is still expanding at an increasing speed (the web only contained about 320 million pages in 1997 [35]). Additionally, the Internet is becoming the primary global distribution centre for all kinds of information including advertisement, news, music, videos, etc. It is therefore very important to understand the demands of millions of users on the web, and serve them better. Our first choice to facilitate this process is to provide visualization techniques to information publishers and resource administrators.

In fact, it is not necessary for us to represent or visualize the WWW in its entirety after considering the underlying motivation for web visualization. Web information visualization is of interest to two major groups of people [52]. On one hand, web users and information seekers need information visualization tools to facilitate their browsing experience. On the other hand, information publishers and website administrators, who would mostly be concerned with a site's effectiveness, need tools for their maintenance, publishing or analysis work. Administrators would ask questions such as "Are people entering my site in the way I had expected?", "Where are people leaving my site?" or "What is the average viewing time of a particular document?", etc. However, while web users are assisted by many navigation recommendation systems, web site administrators are still lacking convenient visualization tools to help them discover and explain the interesting navigational patterns within their web site usage data.

Why do administrators need information visualization? There are three factors that make it necessary.

- A good visualization system greatly facilitates the website administrators' work to improve the website's quality to keep a convenient environment for user navigation.

- The results generated by web mining system are usually difficult to explain directly by human beings. For example, a medium sized web site normally contains 20,000 web pages [46]. In such a complicated and unorganized structure, it would be very difficult to understand patterns such as association rules without the assistance from visualization systems.

- People are better at understanding colours, lines, shapes or images rather than only text and numbers, which are inherently abstract [30].

In the following, we present the basic concepts underlying the process of creating information visualizations for WWW administrators.

1.1 Understand the Web

First of all, we need to understand the raw data on the web and extract useful knowledge before any information can be visualized. Web mining is the research field that concentrates on how to understand and interpret the web, its connectivity and its usage.

With the huge amount of available online data, the WWW turns into a fertile field for data mining research. The objective of web mining is to explain the web users' behaviour and discover valuable patterns. Generally speaking, web mining is the use of data mining techniques to automatically discover and extract information from Web documents and services [23]. It is at the crossroads of several research fields, such as Artificial Intelligence, Information Retrieval and Database, Machine Learning and Natural Language Processing. Although the research breadth is large today partly due to the interests of various research communities (e,g, interests in e-commerce [33]), the research can be classified into several subtasks:

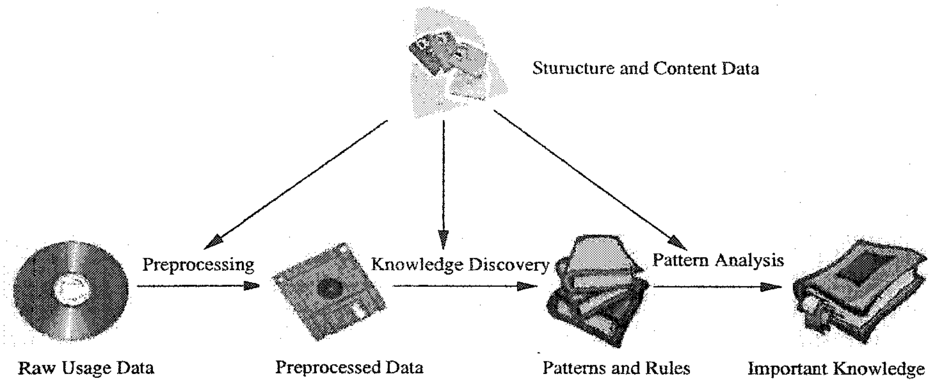


Figure 1.1: Web Usage Mining Procedure

- **Information Search**: Find specified Web documents.
- **Data Selection and Pre-processing**: Pre-process the data and select useful information from retrieved web resources.
- **Pattern Personalization and Generalization**: Create and discover knowledge patterns for individual users as well as multiple sites.
- **Analysis**: Validate and explain the patterns.

Web mining is categorized into three areas of interests based on which part of the Web to mine [9, 38]: *Web content mining*, *Web structure mining*, and *Web Usage mining*. *Web Content Mining* describes the discovery of useful information from the content of the Web text documents or data in other format. *Web Structure Mining* tries to determine the models underlying the link structures of the Web, based on the topology of the hyperlinks with their description [14]. While the above two utilize the real web data, *Web Usage Mining* mines the secondary data generated by the navigation of the users while interacting with the web [22]. The possible source data includes server logs, user profiles, user sessions and transactions, cookies, mouse click sequence and any other similar data. A typical web usage mining process is shown in Figure 1.1.

In this research, we focus on the conjunction of *Web Structure* and *Usage Mining*, to provide a better view and explanation of the navigation patterns and user behaviours with the assistance from information visualization techniques.

1.2 Visualize the Knowledge

The old saying, "a picture is worth a thousand words", is proved by the experiments like [24], showing that people are better at understanding pictures than texts. Therefore, visualizing the extracted pattern and knowledge from web mining efforts would greatly help people to understand and analyze them. In fact, visualization is the process of transforming data, information and knowledge from abstract information structures into a visual form, in order to make use of human beings' natural perceptual capabilities [25]. It can help people solve problems like:

- Disorientation: graphical representations aim to help users maintain a sense of context within an information space [21].
- Cognitive overhead: graphical representations can provide an external representation of the user's memory of the navigation session [21].

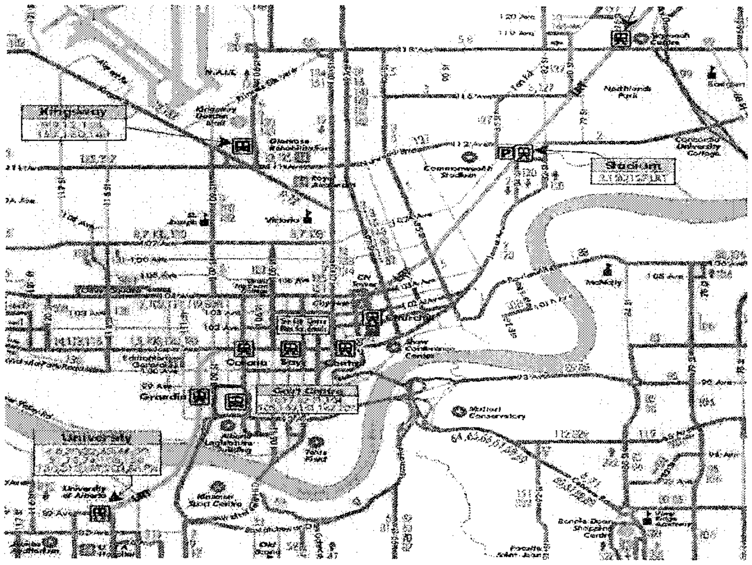


Figure 1.2: City Bus Traffic Map [1]

For example, Figure 1.2 is the map of a city bus traffic system. Just looking at the map results in the viewer forming his or her decision by some understanding of the traffic routes and their regular experience. Usually, attention focuses on the departure and destination stations and a viable route between them. With this map, the task facilitated by visualization is planning a journey. In a simple sense this common traffic map is a visualization tool.

1.3 Visual Web Mining: My Proposed Thesis

While visualizing the web mining results can help, visualization in its own is not enough in this process. Interactive visualization can help evaluate data mining results in a more precise way. **Therefore, we believe that visualization can and should be a means for data mining so that users can distribute the visualization cues to represent web information, manipulate and operate on these visual cues to discover useful knowledge in an ad hoc manner.** We call this process *Visual Web Mining*, which we believe is the key to help users manipulate the visualization data, to observe and discover useful navigational patterns, e.g, web user traffic and recent interests.

In this research, we concentrate on designing a new algebra to operate on web graph objects, visualization objects that represent the structure and usage of the web, to highlight the interesting characteristics of web data, and consequently help discover new patterns and useful implicit navigational facts in web data. A tool FootPath, which is a part of the framework but not one of my contribution, is implemented to visualize the operation's results and thus help web site administrators explain user activity and improve their web site design. The general system helps web researchers better understand the usage patterns and web mining knowledge, and facilitates improved web information retrieval.

1.4 Contributions

There are two important contributions of this work.

- A visual web mining framework is proposed and the web visual mining terminology scheme is presented in order to allow users to interactively explore the web usage data together with its structure. The basic concepts of the framework are:

  - *Web Image* represents the structure and connectivity of the web.
  - *Information Layer* represents the usage data and useful patterns.
  - *Web Graph* assembles together image and layers to represent navigational data and knowledge discoveries in their web structure context.

- More specifically, the algebra for interactive web data manipulation is proposed and explained. Ad hoc visual web mining is then possible by applying the operators of the algebra.

Contrary to most of the previous visualization systems that usually only visualize web data, the system we present is unique in that several operations are designed to mine the rendered visualizations to achieve more valuable patterns. By laying different information as layers on the web structure representation, we are able to select, merge and separate correlated data or patterns. We list the major functionalities the system provides in the following:

- Web Site Structure Mapping
  A radial tree algorithm is used to display the web site topology [53].

- Preprocessed Web Log Visualization
  The system shows the web usage information according to the preprocessed web log files. Multiple kinds of information can be shown at the same time.

- Visual Web Mining
  - Statistics Mining
    Shows the statistics such as web page hits, traversal paths visits, also manipulate them to show other "hot points".
  - Structure-Usage Correlation Mining
    Reveal the hidden relationship between the website structure and usage, for example, the user-oriented information structure.
  - Time Comparison Mining
    Compare the visualization between different time slices to show the usage difference.

The reminder of this dissertation is organized as follows: Chapter 2 discusses the web visualization related work. Chapter 3 introduces the framework with the web visualization object terminology. Chapter 4 explains the web algebra for visual data mining. Chapter 5 describes how to use the algebra. Finally, Chapter 6 concludes the work and suggests future research.

Chapter 2

Web Information Visualization

The term "Information Visualization" was first introduced by Stuart Card et al. [13]. It was originally applied to database and information retrieval systems. However, in the computer science domain, information visualization is more than a method of computing. It is a process enabling the user to observe, digest and make sense of the information [25]. There are many situations where data are available in very large quantities, and where human insight are required to understand them. From the ancient world maps to today's GPS navigation, from the AT&T telephone network to quantitative data interpretation, visualization techniques have been long used by human beings to help them represent, understand and solve complex problems.

The World Wide Web has an enormous size and complexity, but is often held up as a prime example of anarchy [52]. Because of the information explosion, it has more than 50,000,000 web addresses, 100,000 web sites in 1996, a large majority of them are connected by from one to ten other pages and the number is still increasing at a tremendous speed [52]. Based on these statistics, it is clear that if all the implicit connections in the Web were to be represented by a graph, it would be too huge for human being to understand it. Thus, the WWW is obviously an application domain in urgent need of good visualization techniques and systems.

The first web data analysis visualization tool, WEBVIZ [51], shows the access frequency of web pages classically by the colour of the nodes, representing pages, and the thickness of the lines, representing hyperlinks. Although the system is designed to help the WWW database maintainers and designers with a graphical view of their local database and traversal patterns, it is inappropriate for large website database because its node layout is randomly distributed on the screen. However, from then on, more and more researchers have focused on this area and many web data information visualization systems have been created.

In the following, we first introduce several web structure visualization systems, then web mining and navigational pattern visualization systems. After that, we review some existing web information visualization systems that visualize the web usage data together with their structure context. A visualization operator framework is also presented. Finally, the map algebra concepts are introduced.

2.1 Web Structure Visualization

The major challenging issue in visualizing the web structure is the object location distribution. The website is made up of web pages and hyperlinks. Although they do have relationships online, they don't have any preferred positions in the physical world. The layout algorithm for setting the page and link position on the screen is paramount to reveal the web structure and avoid object occlusions and graph clutter so that the system can provide a concise overview together with as many details as possible.

Currently, there are two available web structure models, the hierarchical tree structure and the net structure. The first model treats the home page of a web site as the root of the tree and all other pages connected from it are treated as the son of the root, and so on. On the contrary, there is no root node in the latter model so that all the nodes are treated equal. A good example for the net structure modeling is the Narcissus system [26]. It treats all web pages equally and decides the positions of the corresponding nodes according to the hyperlinks they have or the usage of these hyperlinks.

However, according to [36], website designers "generally followed a process of progressive refinement of their designs from less detail to greater detail, and simultaneously from coarse granularity to finer granularity", which means that most websites were born in a hierarchical way. Therefore, the hierarchical model is more popular in recent visualization research and most of the web information visualization systems adopt this method [18, 34, 43, 44, 47, 50]. There are two ways to implement the model. One is to use a web crawler and an search algorithm, e.g, breadth first search or depth first search, to "walk through" the entire web site following the hyperlinks from the pages to generate the structure [18, 34, 47, 50]. Another is to analyze the structure using the file system directory where the documents are located [43, 44]. The accuracy and the quality of the two methods' construction results vary in different situations, depending on the actual website structure or how the document files are stored. We introduce several representative systems as illustrations in the following.

- **Inxight**

Inxight [34] was developed to solve the "lost in hyperspace" problem [48], and provide better usability to move around in large web structures. It applies the Distorted Display idea, which displays the detailed information in the "Focus" area, which refers to the local area of a graph that shows more details than the rest of the areas, without eliminating the information in the "Context" area, which is the rest of the areas of the graph. The major advantage of the focus+context technique is that it can magnify the detail in chosen area and also provide some neighboring data information.

Inxight was the first to apply the hyperbolic view technique to this area, which lays out a graph in a hyperbolic geometry and then projects it onto a Euclidean plane or space to generate either 2D or 3D graphs, in a 2D plane [27]. The system uses a web crawler and a breadth-first search algorithm to generate the web site structure. An example is shown in Figure 2.1. Any page can be easily moved into focus area by clicking and dragging the corresponding node square to the center. The selected nodes are enlarged and other nodes are pushed to the periphery and shrunk. Double clicking the square opens a new web browser window to display the web page. The user can quickly explore the whole web site structure by keeping dragging nodes. Another impressive functionality of *Inxight* is that it can show the search result on the graph by locating the matched page in the focus area.

The advantage of the system is quite obvious. Users can navigate the website with sufficient context information, even before they choose a hyperlink to continue

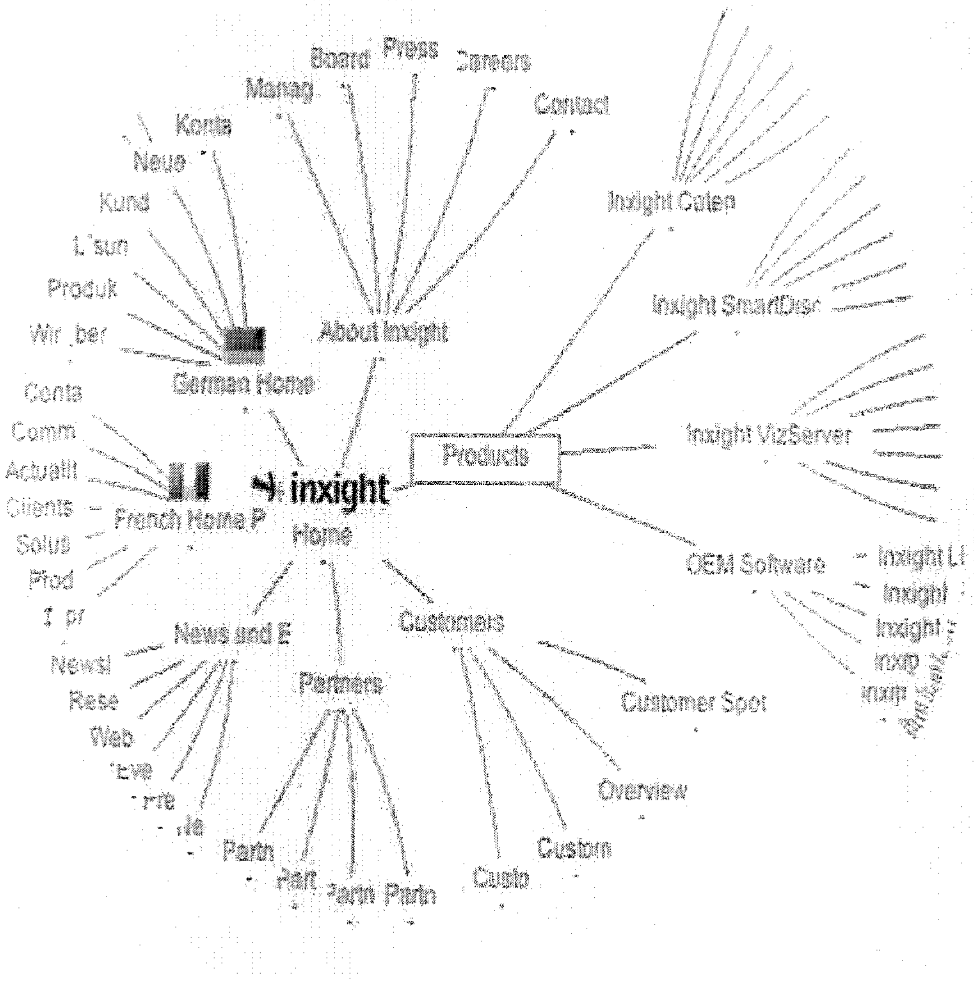


Figure 2.1: Inxight Web Structure Visualization[2]

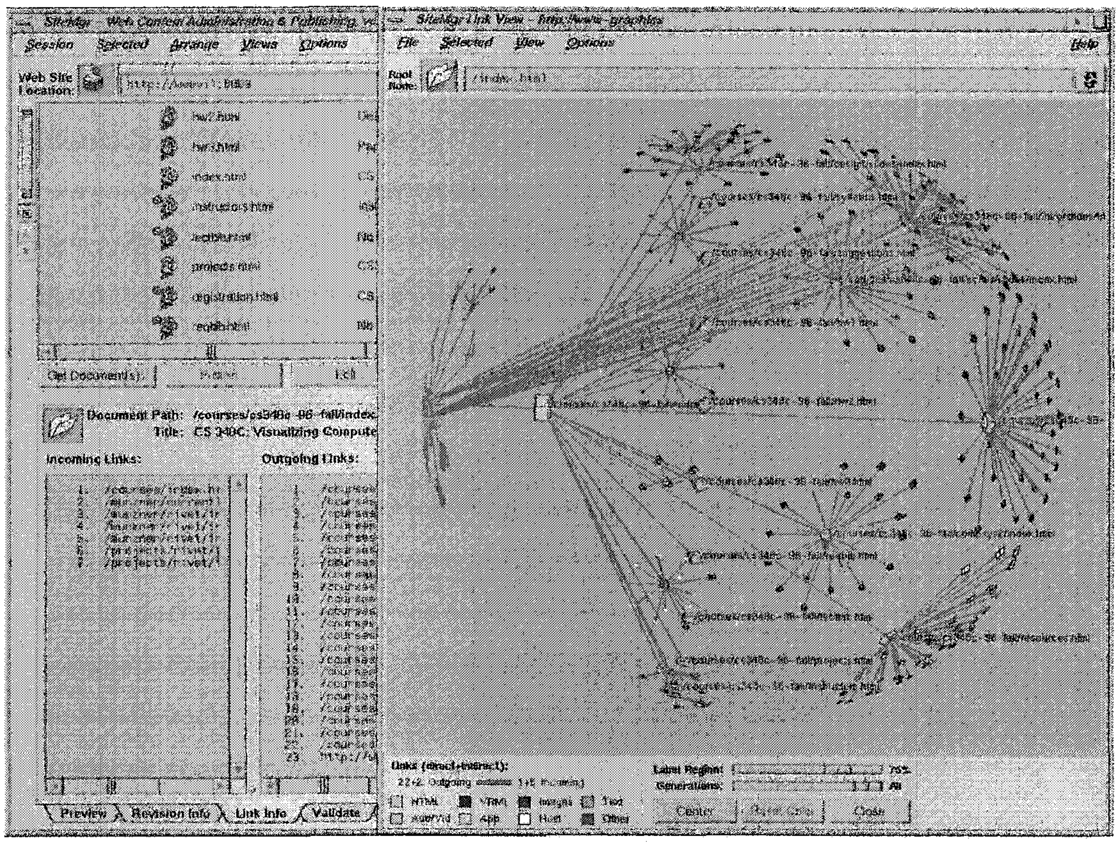
surfing. However, the system is not suitable for data analysis or pattern extraction. Global patterns are difficult to observe because a complete overview cannot be given with some areas distorted. Moreover, an infinite number of objects can be put into a space and the distant information from the "Focus" areas becomes so small that viewers can barely see it [42].

- **Site Manager**

The hyperbolic technique was also used in a 3D visualization system in [44, 45, 47]. The result system, *Site Manager*, is basically designed for web administrators and designers to create a dynamic and optional view of the web site structures. Similar to *Inxight*, it provides the ability to explore the web site structure by moving expected web pages into or away from the "Focus" area. The authors first use an appropriate spanning tree as the backbone for further fast layout and ordered drawing, then design an adaptive drawing algorithm with a guaranteed frame rate to generate the image. In Figure 2.2, the hyperlink structure of the Stanford Graphics Group web site [45] is drawn as a 3D graph in a hyperbolic manner. On the left, the directory structure of the web site files is displayed in a traditional browsing way. The snapshot shows the main spanning tree from a course home page, whose ancestors appear on the left and the descendants are visualized on the right.

Site Manager excels in many aspects of visualization. It achieves a good balance between the information density and clutter. The used adaptive drawing algorithm provides a fluid interactive experience for the user by maintaining a guaranteed frame rate and the 3D graphs are easy for human beings to perceive. However, it has some occlusion problems, as many other 3D visualization systems do, e.g, link crossings are seen in many locations in Figure 2.2. Also, an overall image is also difficult for the system to visualize because of the hyperbolic characteristics.

While the hierarchical model becomes so popular in website visualization community and has achieved big success, we note that web sites, by strict definition, are not trees as the model assumes, but directed cyclic graphs, e.g, a common son page of the home page usually has a "Back to Home" link which connects to its parent. Since a tree structure is only an acyclic graph, we lose information in such a transformation,



Web Site Location: http://www-graphics...

Root Node: /index.html

Document Path: /courses/cs348c-96-fall/index.

Title: CS 348C: Visualizing Compute

22+2 Outgoing 1+5 Incoming

Label Region: 75%

Generations: All

Figure 2.2: Site Manager[44]

which is a trade-off for clearer visualization and better presentation. In fact, the information loss and clutter can possibly be reduced to minimum by applying the new proposed usage-based layout algorithm, which collects and visualizes the most important link for each node, those who are visited the most, in the transformation [53].

2.2 Web Mining Visualization

Although knowledge patterns and rules can be extracted from web data sets by many web mining systems, the fact that the number of these patterns and rules can be huge [37] and their characteristics, usually described by mathematical equations and numbers, make it difficult for human beings to interpret and understand them [29]. However, visualization systems can again help people to understand these patterns and even find more useful and valuable patterns visually. Most of these systems focus on visualizing the association rules [29], the clustering [11] or the web navigational sequence [5, 6] and patterns [3, 4, 20, 28, 51, 59]. We introduce some representatives in the following.

- **Association Rule Visualization**

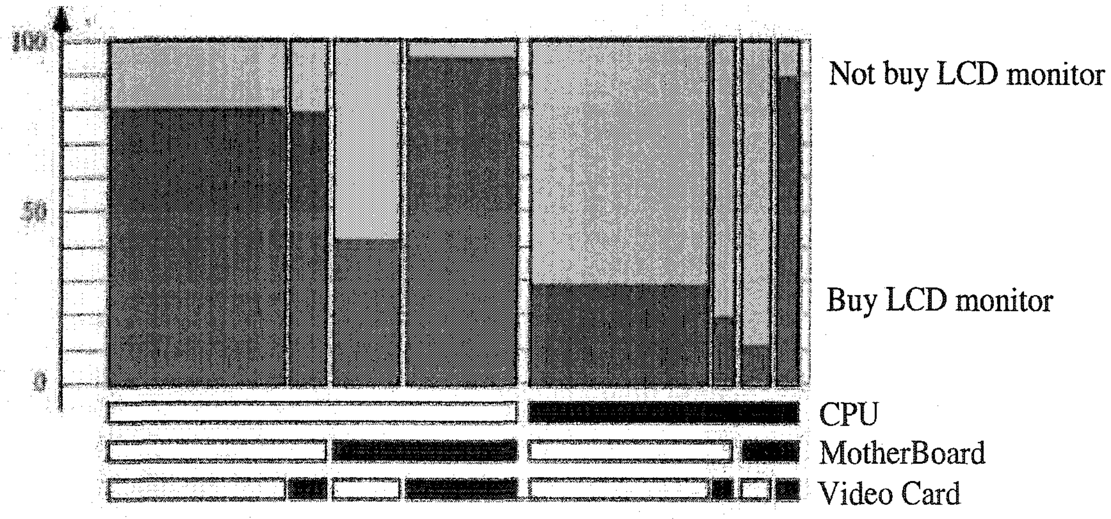


Figure 2.3: Association Rule Visualization

In [29], researchers use a mosaic plots technique to visualize the association rules. As shown in Figure 2.3 for a computer hardware online shop, the vertical axis (the height of the rectangular) represents the rule confidence while the horizontal axis (the width of the rectangular) represents the rule support. Each row of the bars in bottom represents one product, thus there are three rows for buying *CPU*, *Mother Board* and *Video Card* respectively. The colours, white or black, of the bar indicate whether the corresponding product is included in the rule or not. For example, the first rectangular from the right together with the three bar rows in the bottom tells the association rule: "*CPU & Mother Board & Video Card* $->$ *LCD Monitor*" and the fifth rectangular from the right describes the rule: "*Mother Board & Video Card* $->$ *LCD Monitor*". In the graph, we can see that both rules have high confidence (rectangular height) but the latter one has a much higher support (bigger rectangular width) than the other. Therefore, the second rule is better.

The visualization attempt described above can help people understand the individual association rules and make comparison. However, it lacks further functionality to explain and manipulate the rules, such as filtering out the best rules from a large set of possible association rules or choosing two specific rules for comparison, etc.

- **Clustering Pattern Visualization**

An adaptable web site can provide information based on user interests so that information can be delivered more efficiently [41]. To make adaptive web sites possible, a visualization tool called WebCANVAS (Web Clustering ANalysis and VisuAlization of Sequences) is designed in [11] to visualize the user clustering pattern, which gathers the common user surfing behaviour together and display them in groups.

As shown in Figure 2.4, the pages of web site *http://www.msnbc.com* are classified into several categories, which are represented by different colours. Each window represents a different user cluster, and each row in the window represents the entire traversal session of a user. Individual coloured squares represent individual page requests. The colour tells the specific category that the visited web pages belong to. For example, Cluster 1 represents users that are mostly interested in *weather* (dark yellow colour represents weather as the category legend shows at the lower right of

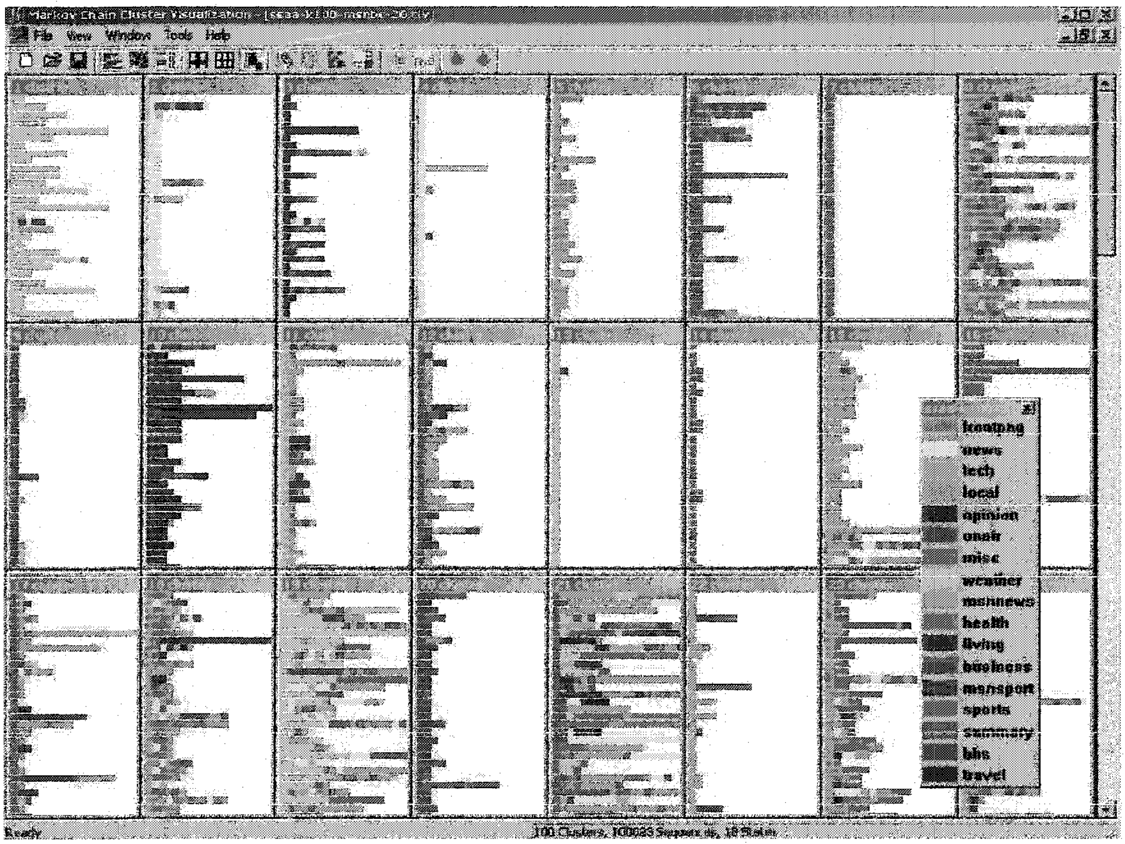


Figure 2.4: WebCanvas Visualization[11]

the figure), while Cluster 10 represents users that are interested in *msnsports* and *sports*. More interestingly, Cluster 12 shows that most users in this group who visit the *"onair"* pages will afterward go to *"local news"* in the same sequence. Thus, the common user surfing patterns can be easily observed and extracted from clustering information with the aid of WebCANVAS visualization. However, the screen space is not used efficiently by the system, we can see a lot of empty space in the example figure. Since user sessions of one cluster can greatly vary in length and the clusters can also vary in size, they should not be treated equally in visualization space distribution. Additionally, one cluster usually contains thousands of users, (Figure 2.4 contains 100 clusters, 100,023 user traversal sequences and 18 states) but each cluster window can only display dozens of user sessions simultaneously. Comparison without scrolling back and forth is impossible.

Besides, decision trees have been long used for the task of classification. Techniques for visualizing decision trees to provide deep insights into the process of decision tree construction is proposed by Mihael Ankerst et al. in [39]. However, the purpose of the visualization effort is to find a state-of-the art algorithm to construct the decision trees to support a better cooperation between human and computer in the construction process, rather than understand the usage data which are used to build the trees.

- **Navigational Sequence and Pattern Visualization**

To understand the behaviour of web users accessing on-line resources, visualization is paramount. However, it is not clear how to analyze and visualize web usage data involving long sequences of on-line activities without losing the big picture [6]. While different kinds of visualization of web usage have been proposed to emphasize different aspects of the web usage, Berendt uses *stratograms*, first introduced in [5], as a way of combining these visualization approaches. Concept hierarchies are used as a basic method of aggregating web pages. Stratogram is then proposed as a method for representing sequences and further mining web usage at different levels of abstraction. The author used the data from agent-supported shopping of an online store as a case study. Customers are encouraged to answer up to 56 questions related to the product they intend to buy, pages are then classified into several categories (shown in the left of Figure 2.5) based on the answers.

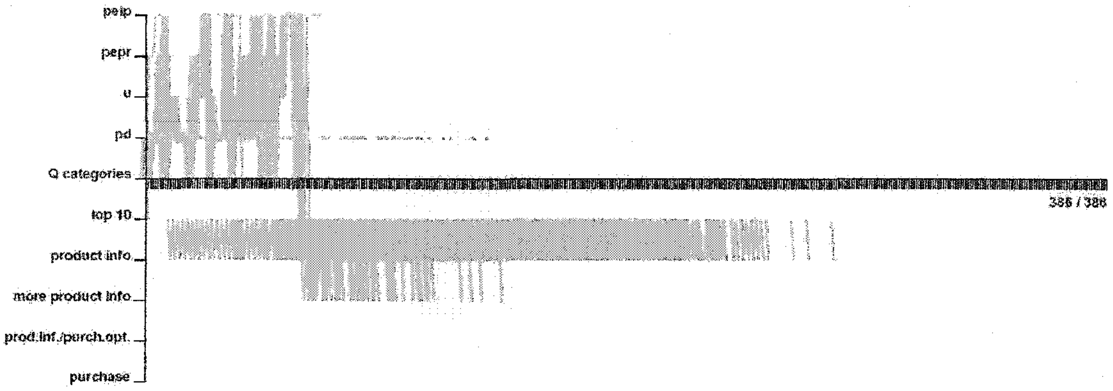


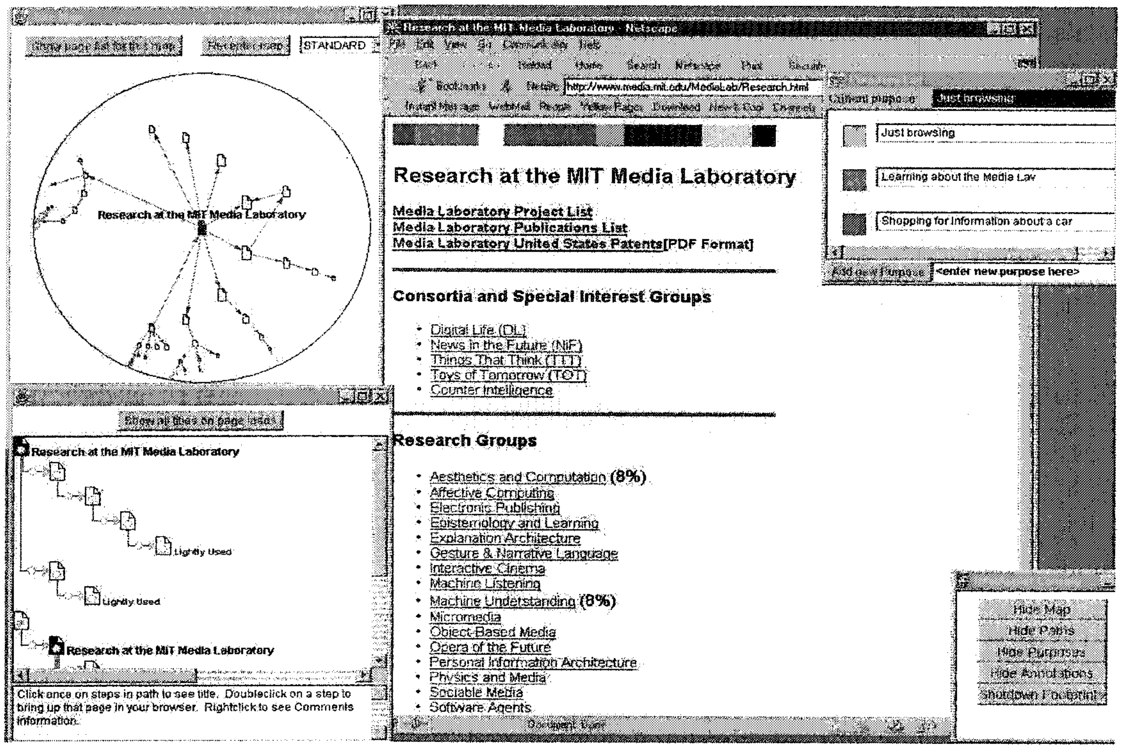
Figure 2.5: Usage Sequence Visualization [6]

Figure 2.5 shows stratograms aggregating the paths taken by 152 camera shoppers through the store. The "Q categories" line separates the upper "Communication Phase" and the "Information Phase". Each x axis segment represents one step in the original logs. Each line in the diagram connects two points, representing two URLs. The thicker the line, the higher the proportion of shoppers that traversed between these two URLs. This figure shows camera shoppers followed the sequence of the questions and request "more product info" pages often in the information phase, which lasted longer than the other phase. The basic stratogram diagrams can be used to compare to each other to find interesting or even unexpected patterns. The author further proposed the interval-based coarsening technique to visualize the usage at different degrees of detail.

The fact that this work is based on user feedback of an online store makes it accurate and, however, limited since the universality and reliability of the usage data may not be good enough. The fixed concept hierarchies are also too simple to describe the dynamic usage behaviour. Moreover, close lines can create a visual block, as shown in the bottom middle of the diagram, which prevents further accurate observation.

In order to deal with the "lost in hyperspace" problem [48], visualizing the navigational pattern and providing necessary assistance, another visualization tool *Footprints* is proposed in [57, 58, 59], focusing more on the user experience in a web site.

As its name may imply, the system gathers visiting patterns previous users left and provides them to new web surfers as references, assuming that people know what they want but may need help in finding their way to the information and understanding what they have found [59], so that the interaction history of the web site can help future users to navigate in a complex web space.


STANDARD
Research at the MIT Media Laboratory
Research at the MIT Media Laboratory
Lightly Used
Lightly Used
Research at the MIT Media Laboratory
Research at the MIT Media Laboratory - Netscape
http://www.media.mit.edu/MediaLab/Research.html
Current purpose
Just browsing
Add new Purpose
Hide Map
Hide Paths
Hide Purposes
Hide Annotations
Shutdown Footprints

Click once on steps in path to see title. Doubleclick on a step to bring up that page in your browser. Rightclick to see Comments information.

Research at the MIT Media Laboratory

Media Laboratory Project List
Media Laboratory Publications List
Media Laboratory United States Patents[PDF Format]

Consortia and Special Interest Groups

- Digital Life (DL)
- News in the Future (NiF)
- Things That Think (TTT)
- Toys of Tomorrow (TOT)
- Counter Intelligence

Research Groups

- Aesthetics and Computation (8%)
- Affective Computing
- Electronic Publishing
- Epistemology and Learning
- Explanation Architecture
- Gesture & Narrative Language
- Interactive Cinema
- Machine Listening
- Machine Undetstanding (8%)
- Micromedia
- Object-Based Media
- Opera of the Future
- Personal Information Architecture
- Physics and Media
- Sociable Media
- Software Agents

Just browsing

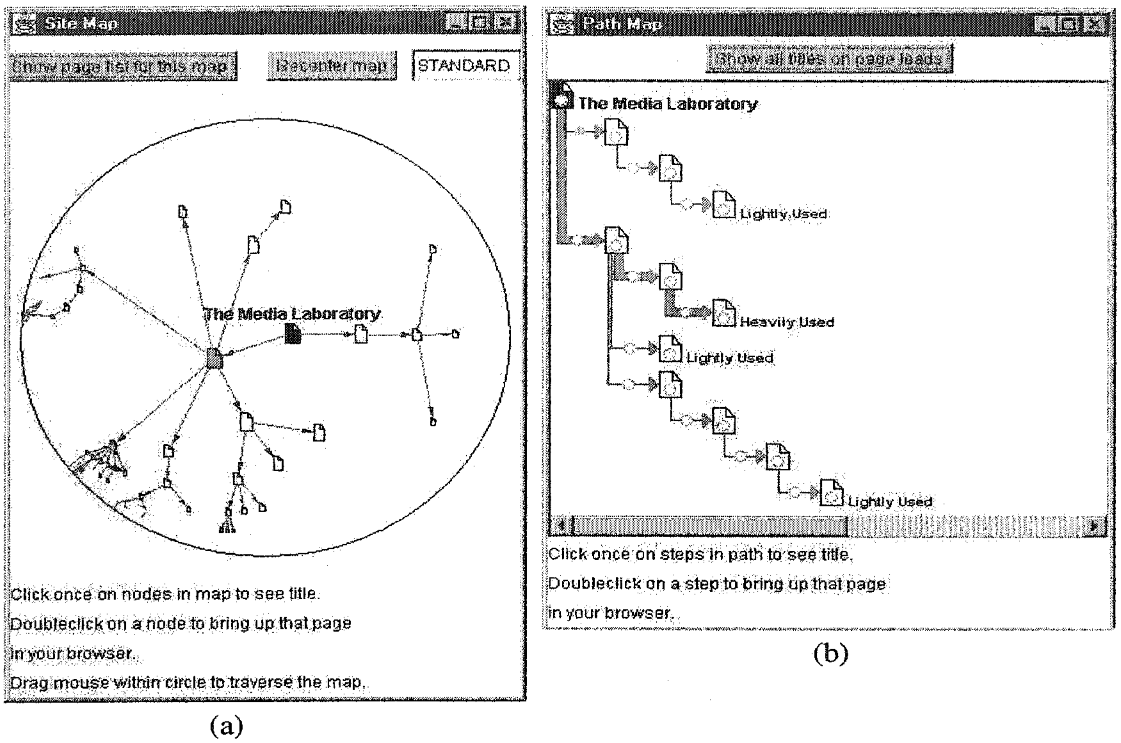
Learning about the Media Lav

Shopping for information about a car

<enter new purpose here>

Figure 2.6: FootPrints Visualization [59]

Footprints provides three tools, maps, paths and signposts, to contextualize web pages using history information. Each tool visualizes the interaction record differently, but they are coordinated and are active aids to navigation rather than static ones. Figure 2.6 shows a screen shot of a user visiting the MIT Media Lab Research Web page with all three tools turned on. Figure 2.7(a) shows the map tools, which use the hyperbolic tree technique to show the traffic through a Web site. Nodes are documents and lines are transitions between them. Only the ones that people have actually visited or used are visualized so that this is typically a fraction of the actual site content. Additionally, all transitions made by the user are tracked, whether they



Click once on nodes in map to see title.
Doubleclick on a node to bring up that page
in your browser.
Drag mouse within circle to traverse the map.

(a)

Click once on steps in path to see title.
Doubleclick on a step to bring up that page
in your browser.

(b)

Figure 2.7: FootPrints Visualization [59]

come from selecting a link on the page, typing in a URL, selecting a bookmark, etc. The result of this is that lines on the map often do not correspond directly to links embedded in the web page. The paths view tool is shown in Figure 2.7(b). Paths are coherent sequences of nodes followed by an individual. The thickness of the line is used to represent the path usage while a text string is used to give the user an approximate idea of the level of path usage. Moreover, signposts tool allows users to enter feedback on the interaction history of both pages and paths they have seen. Users can click on the circles to bring up a simple text window to view or add comments. With these tools, *Footprints* is fine to support undirected web browsing and pattern identification, but its basic character, only visualizing the web pages and hyperlinks that the users have visited, inevitably limits its full-scale usability, because visualizing usage of the "unvisited" pages and links would greatly help people to improve them.

2.3 Web Usage Visualization in Site Context

After recognizing the fact that visualizing the web structure and the extracted patterns does help user surfing and administrative maintenance, some researchers extend the work to visualize the relationship among the web usage data, users' behaviour and the web structure. Since user navigational activities, which generate the usage data, surely tie up with the web information structure, e.g, one hyperlink may be highly used because it connects to a popular page, displaying these information together, usually by mapping the usage information on the web structure objects using visualizing cues such as colour, size, etc, can greatly help people understand and explain the information-rich graph. Four systems using the idea of visualizing usage data in their structure context are listed below for illustration.

- **Crowds Dynamics**

Minar's *Crowds Dynamics* [40] treats a web site as an inhabited, social and active space. A website structure is constructed, then a user navigation animation visualization system is built to visualize the usage information, such as which paths the users often use, or which pages become more popular recently, etc.

Figure 2.8 is a snapshot of the working system. Each rectangle represents a cluster of relevant web pages, which are handmade by the designer. The positions of these icons illustrate the topology of the website. Each dot represents a web user who is browsing the site and its colour is decided based on the user's domain name, red for "edu" domain, green for "com" domain, etc. Whenever a user visits a page, the corresponding dot moves from one rectangle to another, which becomes brighter to show the popularity. Other unvisited rectangles fade into the background. If one user has no navigation information recorded in the web log file after a certain period of time, he or she is assumed to leave the website, and his or her corresponding dot fades out until finally disappearing.

Minar's work is so creative and impressive that he treats the web user behaviour like a documentary movie and replays the real-time user behaviour while most other systems do that in a static way. However, the informative animation system is too simple and there are many possible improvements. First, its structural map is hand-

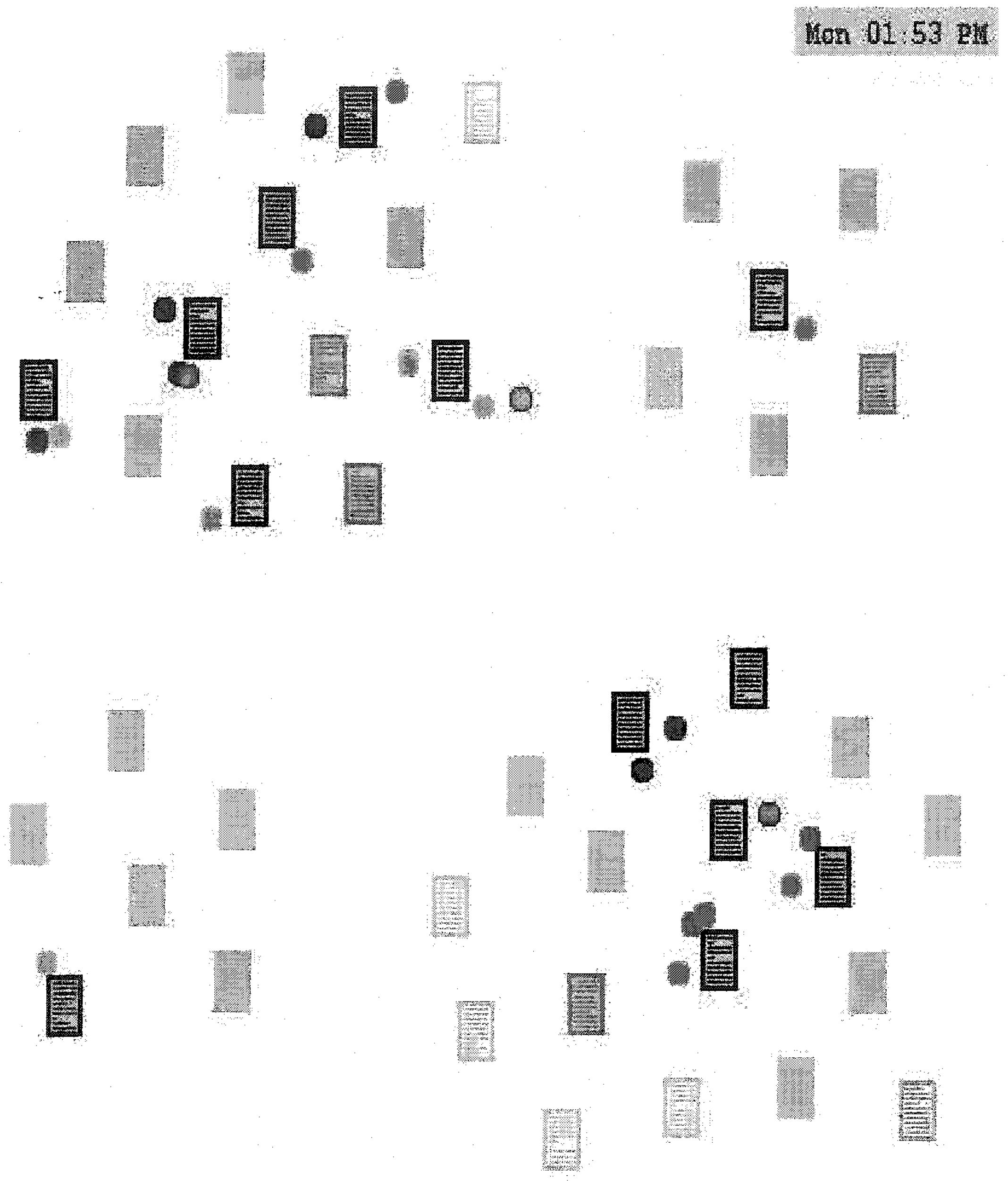


Figure 2.8: Visualization of Crowds at Web Site [40]

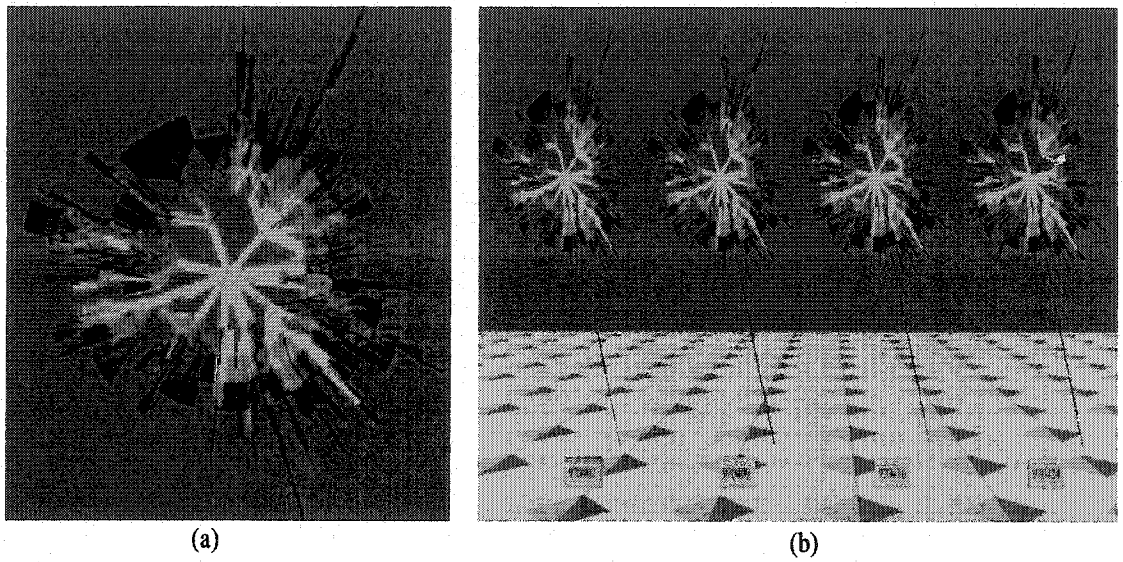
made, site-specific and thus limited. Building a web site map is a large research topic in its own right and is a major challenge. More importantly, rather than showing what web users have done, the system concentrates on what web users are doing, which makes the visualized analysis less revealing.

- **Disktree**

Disktree [16, 18], designed by E.H. Chi et al. at Xerox Palo Alto Research Center, was the first to display the web usage and structure information together by mapping the usage on the structural objects. A disktree is a concentric circular hierarchical layout which has a root node in the middle of the circles. The nodes in the outer circles are children of the inner circle's nodes. This means that the corresponding pages of the outer nodes can be connected from those of the inner nodes. The system uses a breadth-first search algorithm to transform the web graph into a tree by placing a node as closely to the root node as possible [18] to build the disc representations. A disktree example is shown in Figure 2.9(a). Each line displays the information of the nodes to which it connects. The thickness and brightness of a line correspond to the node's visit frequency. The thicker and brighter the line, the more the page is visited. Colour is also used to represent changes in time: red means new added pages and yellow means deleted ones.

To show the evolution of web site trends, the authors proposed the concept of Time Tube Visualizations, a series of visualizations constructed at different time periods but aligned together for comparison. As shown in Figure 2.9(b), the authors placed multiple DiskTrees for different days along a spatial axis representing time, and thus facilitate the usage information comparison through time [18]. Via the Time Tube visualization, viewers can easily observe the popularity trend over time, the hidden correlation between the usage patterns and the new added/deleted pages, etc.

The researchers extend their work based on the Disktree visualization system in [17]. They built a multiple view visualization system "SpreadSheet" to compare the web usage patterns and evolution. Two operators are provided to visualize the useful patterns: colour scale threshold setting and usage data subtraction. Combining these two operations together can powerfully emphasize the difference over different data



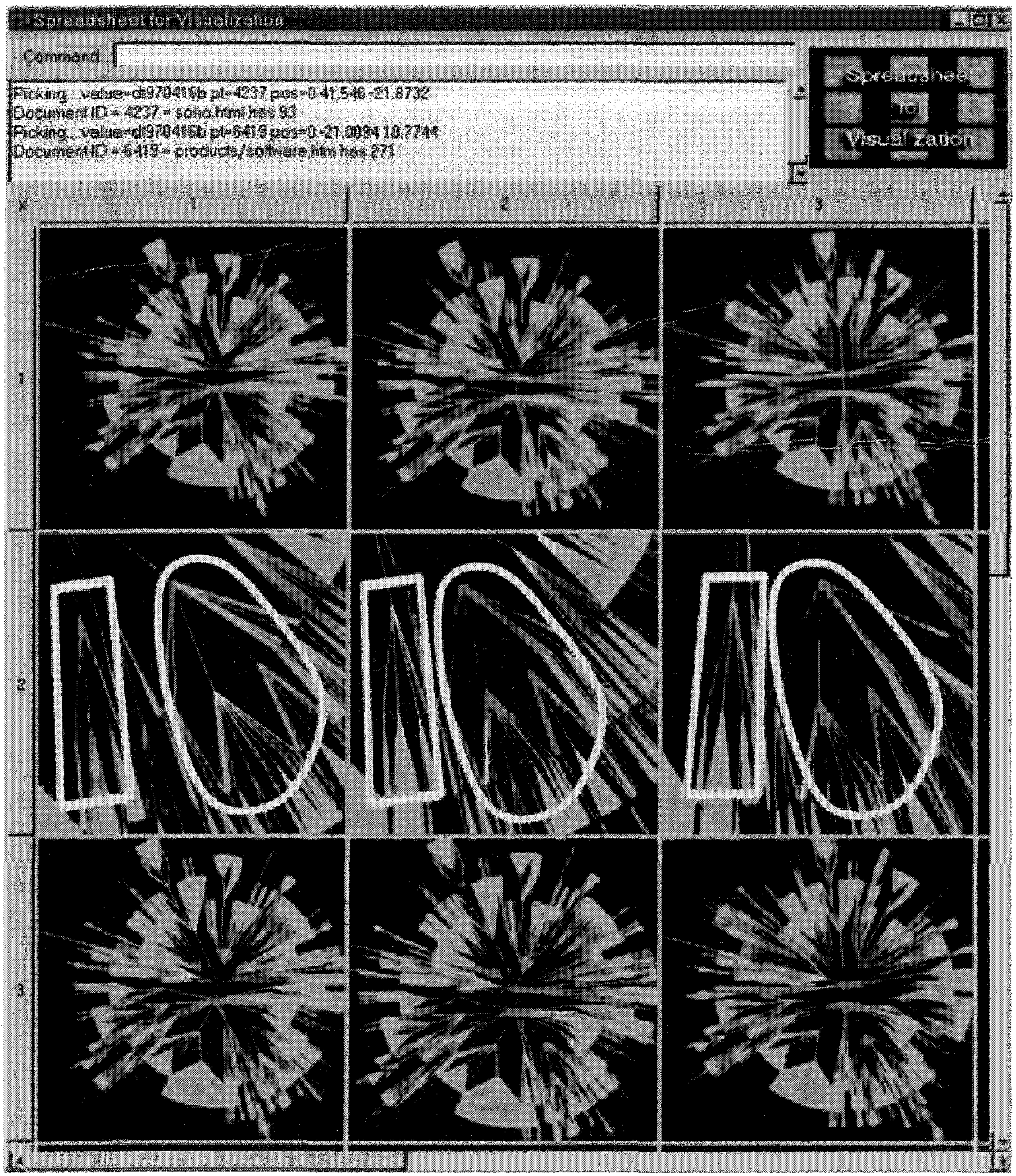
(a) (b)

Figure 2.9: Disktree Time Tube [18]

sets. For example, in Figure 2.10, the first column shows the result of subtracting the data of week one from that of week two, and the second column shows the result of subtracting week three from week two, and so on. The blue colour represents negative values and red represents positive values. Thus, the pages that present increasing popularity and the pages that loose its popularity can be clearly shown.

Disktree has several advantages for web information visualization. First, this tool fits in with the visualization dimension space pretty well, for it does provide insight into large structures, also provide some static images of aggregate navigational behaviour and dynamic graphs of different period comparison [50]. Second, there is no occlusion problem on a 2D plane so that the structure and the pattern are both visualized compactly and recognizably. Finally, it provides several operators for preliminary visualization manipulation, which motivates our research for the web graph algebra. However, the proposed operators have limited functionality and thus are not good enough for further interactive visualization operations. The node and line based DiskTree rendering scheme also prevents the system to visualize significant analysis for user information or navigational path.

- **WEBKIV**



```
Picking...value=dt970416b pt=4237 pos=0 41.546 -21.8732
Document ID = 4237 = soho.html has 93
Picking...value=dt970416b pt=6419 pos=0 -21.0094 18.7744
Document ID = 6419 = products/software.htm has 271
```

Figure 2.10: Disktree SpreadSheet [17]

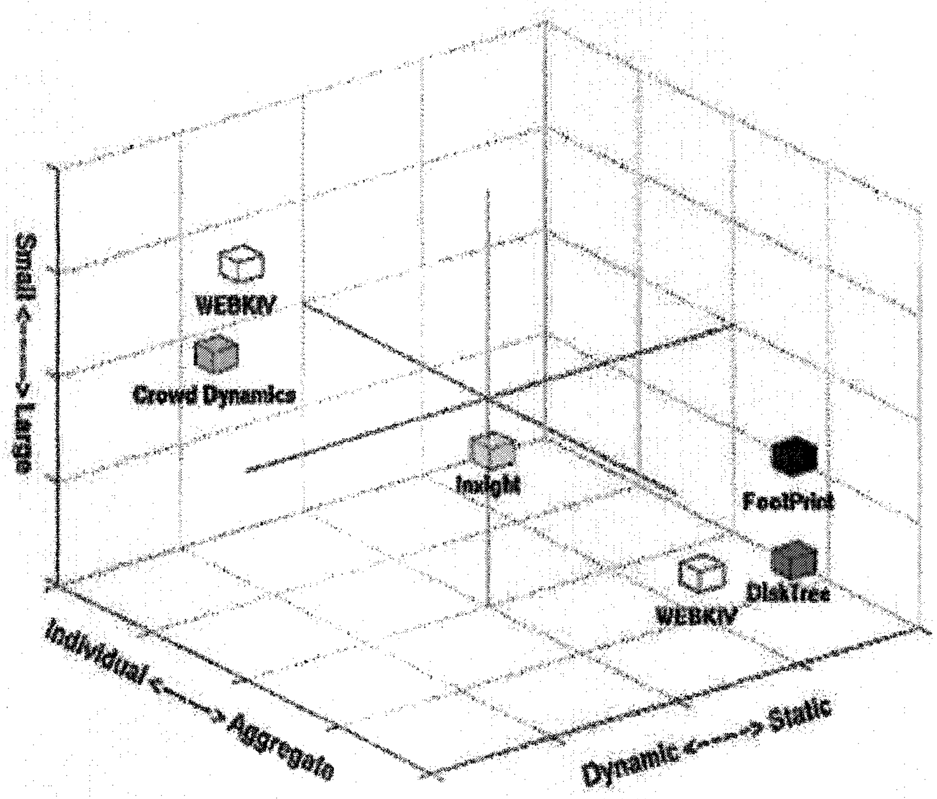
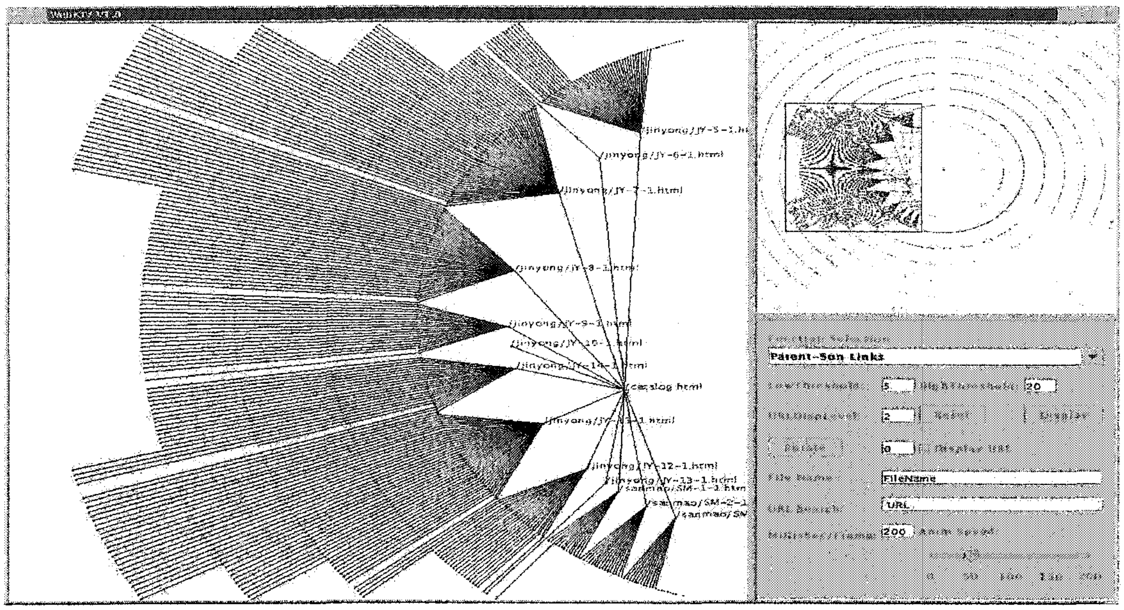


Figure 2.11: Web Visualization Dimensions[50]

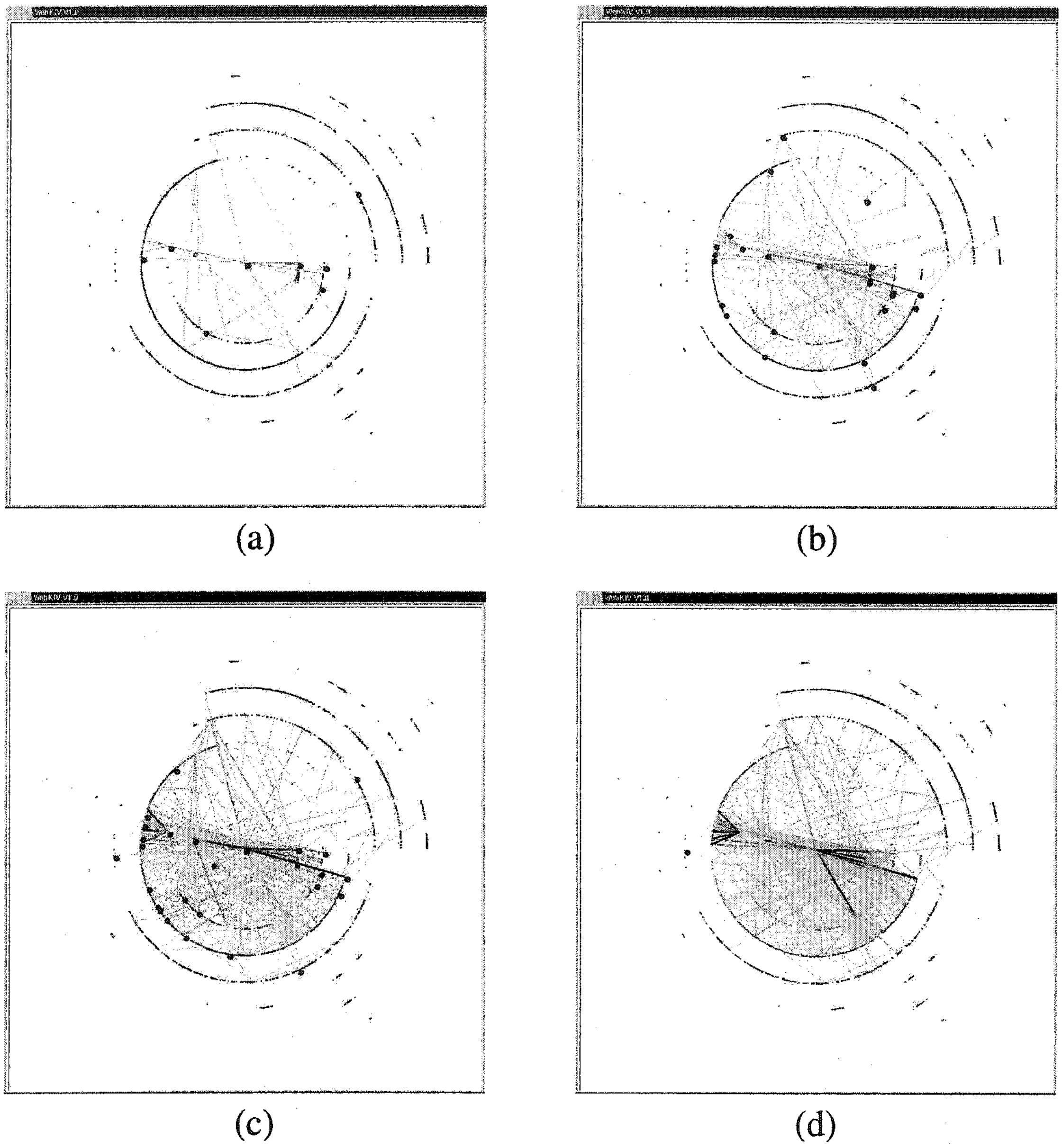


Left: Focus View Window Right: Context Window

Figure 2.12: WEBKIV System Overview [50]

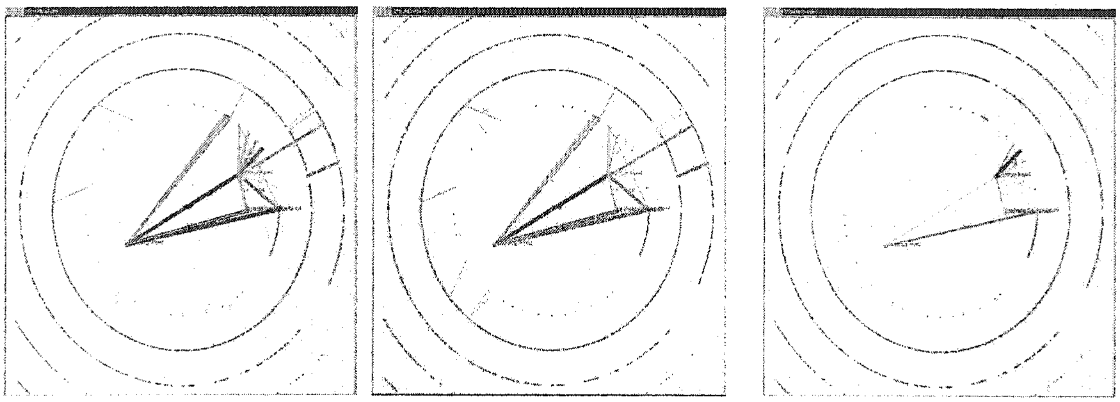
Although it is difficult to classify these various web visualization tools for usage data and mining patterns, researchers begin to understand their characteristics by decomposing the task into the three dimensions: scale of web structure visualization; representation of aggregate and individual navigation usage behaviour; and the comparative display of navigation improvement methods (Figure 2.11) [50]. The authors combine several useful strategies from other web visualization tools and build the *Web Knowledge and Information Visualization* (WEBKIV) [49] system to provide a single method to visualize the web structure, the usage data and the results of web mining on that structure. Functionalities include web structure visualization, web navigation visualization and web mining result comparison (Figure 2.13).

In the WEBKIV system, the DiskTree algorithm [18] is used to construct the web site topology. Zooming and panning techniques are used to view the information in detail if necessary. In order to alleviate the context switching problem inherited from the zooming and panning techniques, WEBKIV provides two display windows (Figure 2.12), a *Context window* is used for the context overview, and a *Focus window* is used for drilling down. Moreover, a rectangle in the *Context window* shows which



(a) (b) (c) (d)

Figure 2.13: WEBKIV Surfing Visualization Example [61]



(a) before applying NCM (b) after applying NCM (c) pattern (b) – (a)

Figure 2.14: WEBKIV NCM Comparison [61]

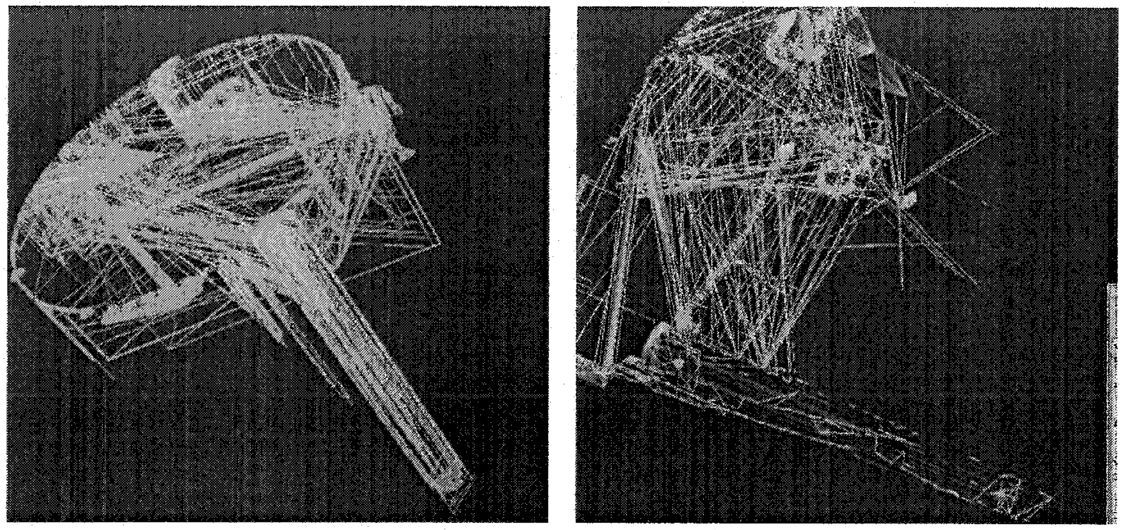
part of the web site is displayed in detail in the *Focus window.*

Figure 2.13 shows an example of the surfing animation and usage aggregation functionality of WEBKIV. The visualized website is a music website introducing musical machines such as drum, synthesizers and recording equipment, etc. With a 10 frames per second frame refresh rate and 50 times faster animation rate than the real time, Figure 2.13(a) shows the user activities at midnight. We can see there were only a few surfers browsing the site and no obvious traversal patterns available. In the morning, the site became quite busy and several heavily used hyperlinks were starting to "stand out", which is shown in Figure 2.13(b). Similarly, Figure 2.13(c) shows the web usage around evening and Figure 2.13(d) concludes the result of one-day trace history the users left.

Another good property of the system is that it can evaluate the effectiveness of Navigational Compressional Models (NCMs) [61] to improve the user navigation quality by a "subtraction" operator, which emphasize the difference between the original web access logs and the NCM generated web logs. As shown in Figure 2.14, part(a) is the visualization of the log files before applying any machine learning method, part(b) visualizes the generated user web log after NCMs have been applied, and part(c) is the result of "subtraction" operator, which illustrates the difference between the original and improved navigational paths.

- **3D Visualization**

A recent preliminary work uses 3D rather than 2D layout techniques to visualize the usage data and map it on the web structure [60]. Figure 2.15(a) shows the 3D visualization of the web usage data together with its web site structure. Figure 2.15(b) presents the same site but uses colour mapping to highlight access paths of user navigation so that popular pages and paths can be observed.



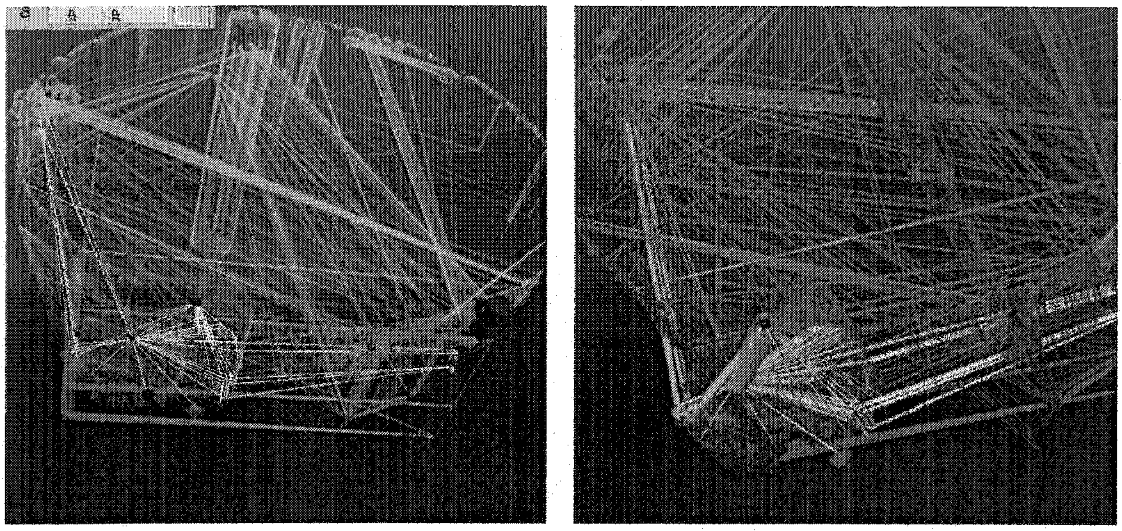
(a) Web Site 3D Visualization

(b) User's Browsing Access Patterns

Figure 2.15: WebSite Visualization in 3D Example [60]

The 3D system also visualizes the web mining patterns of most frequent access as shown in Figure 2.16. The patterns are represented by white edges in (a), and by white tubes in different thickness in (b), which is claimed by the authors to be able to significantly help further analysis of results.

This research implements 3D visualization method to facilitate the maintenance work. However, the work is preoccupied with fashionable 3D visualization issues rather than mining the information itself or making the visualized data more accessible. As a result, the 3D images are information-rich but extremely difficult for an ordinary user to understand. Moreover, there is a significant issue with occlusion, where objects obstruct others and prevent appropriate visualization. Thus, its importance as an information visualization tool, whose objective is to explain and help users understand the information, is very limited.



(a) Frequent Access Patterns

(b) Pattern with Tube Thickness

Figure 2.16: Navigation Pattern 3D Visualization [60]

Although none of the existing visualization tools presented above include the possibility to manipulate web data based on visualization operation, they do provide a general background of where the thesis research sits, prove the feasibility of the usage-on-structure visualization framework, and also motivate our design for the next step of visualization in web domain.

2.4 The Framework of Visualization Operators

In the past several years, researchers have made great advances in information visualization to understand the web space. One promising possible extension of the current information visualization work is to develop the interactivity between human and computers in order to enrich the dialog communication of information. Some theoretical work has been done by Ed Chi and John Riedl in [19]. They propose a novel operator and user interaction model to unify the data analysis process and the complex relationship between view and value to characterize the interactive and non-interactive operations in a visualization system [19].

In order to build a framework for visualization operators, the authors introduce some fundamental properties of operators based on their visualization experiences in [16, 17, 18]. There are two important properties, one is whether an operator is operationally or functionally similar, the other is whether it is a view or value operator. They are important because the similarity deals with an operator's degree of applicability and the view or value property seriously affect the semantics of the operator.

- Operationally Similar

The operators, whose underlying implementations are exactly the same from application to application, e.g, rotations, re-centering, scaling and translation etc. , are operationally similar across applications [19].

- Functionally Similar

Operators are functionally similar if they are semantically similar across applications, but the underlying implementation are actually different for various types of data sets [19], e.g, domain specific data filtering, adding and subtracting.

- View

A view operator changes the visualization method or preference without changing the underlying data set, e.g, rotation, zooming, translation.

- Value

A value operator, on the other hand, modifies the data source by adding or subtracting different data set , filtering or deleting the raw data and other possible operations.

Figure 2.17 is the preliminary visualization operator model presented in [19], where each node represents a certain data state describing the data status, and each directed edge is an possible operator transforming the data from one state to the next, representing the operator application modifying the data. A general visualization pipeline is shown on the right of the figure and can be used to classify the operators. With this state model, the authors build an operator mapping analysis for various visualizations [19]. It can also be used to compare different visualization interaction attempts.

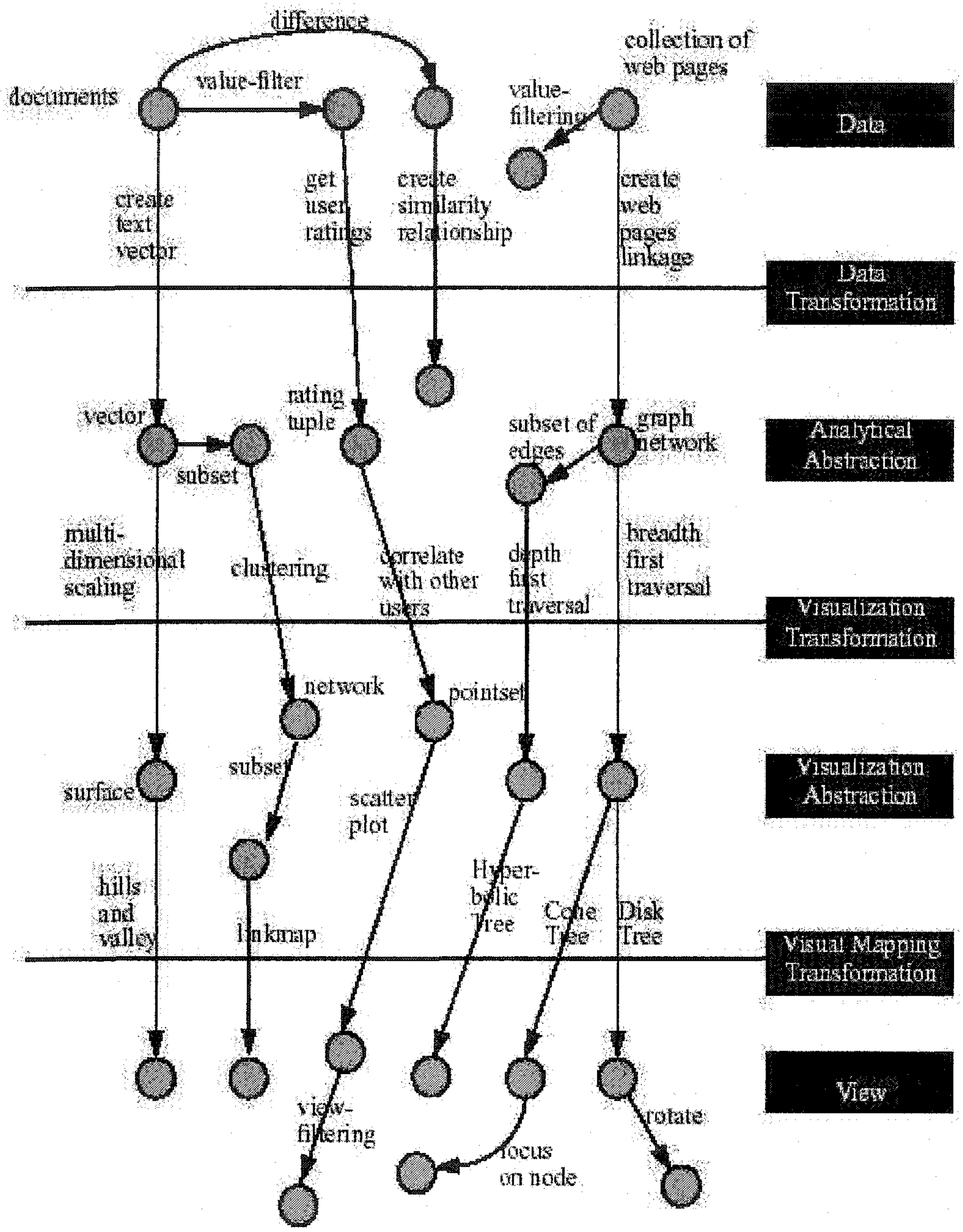


Figure 2.17: Visualization Operator Model[19]

The framework proposed in [19] theoretically facilitates a new way of exploring the visualization interaction space. Many of its propositions and suggestions are adopted together with our own ideas in this thesis research of the web graph algebra and operators. However, our definitions of the "operator" are different. While Chi describes the operators as all the steps from pure data to the final viewing image, we define operators as the possible interactive manipulations of the visualized graph only. We do not label our operators with these properties in this thesis, but some of these properties do correctly describe our operators.

2.5 Map Algebra

Another relevant work worth mentioning here is map-algebra [54], of which the basic idea is background of our web graph algebra. Generally speaking, it pertains to geographical information systems and provides operators to combine information layered on geographical maps.

The *Map Algebra*, a general set of conventions, capabilities, and techniques that have been widely adopted to use with the geographic information system (GIS), and its related cartographic modeling research [7, 8, 54] were first proposed by Dana Tomlin and Joseph Berry in the early 1980s. In order to provide a high-level computation language to describe the geographic data processing, the algebra and modeling is based on the concept of map layers, operators and procedures. The purpose of the language is to create new map layers using existing map layers and operators that are sequenced in procedures [12]. It takes a location-oriented perspective or "worm's eye" view by computing new values for locations based on the location itself, its neighborhood, a related zone, or the entire dataset [54].

Map Algebra is composed of variables, expressions and functions. Variables are map layers. Expressions and functions may have parameters and apply to one or more variables. The mechanism is designed to assist layer manipulation and geographic data querying. It models the surface of the earth as a multitude of independent, coincident layers of themes. The layers interact according to mathematical models and are typically based on real world observations [10]. Planners develop layers on development and population while social scientists develop layers on demographics, ethnicity, and economic factors. Applying a Map Algebra model to input layers produces a new layer (Figure 2.18), which may be a physical map sheet, an area vision perceived through a stack of mylars from a plane, or the distribution of the hundreds of power supply centers of the city. These layers allow researchers to explain complex phenomena, predict future trends, or discover useful knowledge patterns.

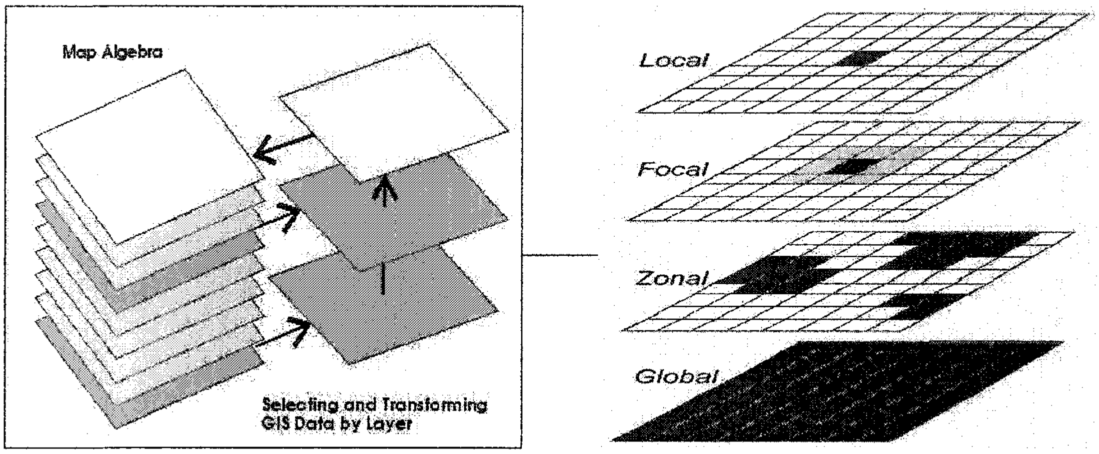
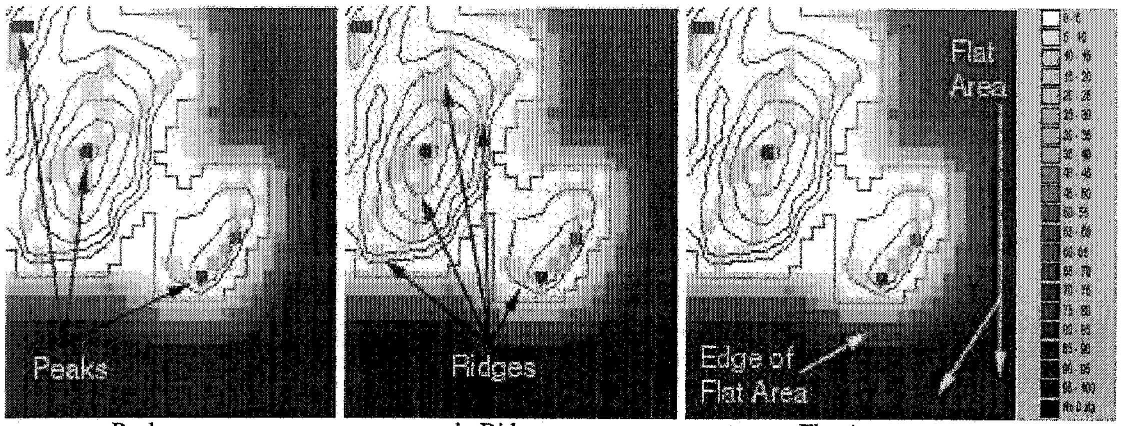


Figure 2.18: Map Algebra Layer Manipulation

As a good illustration for map algebra operators, flag operators focus on *where* conditions are met, rather than *what* meets a condition, to extend spatial analysis capabilities and help reveal the underlying structure of the data being evaluated. Caldwell's paper[12] has introduced some specific examples of flag operators, including the FOCALFLAGMAX operator, which measures local dominance and reveals the underlying structure of the terrain. Figure 2.19 shows the process of revealing the underlying structure of the data. By using the FOCALFLAGMAX operator, it is possible to identify peaks in the data and significant ridges, also isolate and remove the flat areas.

The influence of the map algebra and cartographic modeling have extended beyond the university research community. They are now integral to public domain and commercial software packages and are found in the Academic Map Analysis Package (aMAP), ARC/INFO Grid, Idrisi, MFWorks, MGE Grid Analyst, OSU Map-for-the-PC, PC Raster, and the Professional Map Analysis Package (pMAP). More



a. Peaks b. Ridges c. Flat Areas

Figure 2.19: Underlying Structure Revealed by the FOCALFLAGMAX Operator[12]

information regarding the GIS system and Map Algebra can be found at [55, 56].

In this chapter, several existing visualization systems are summarized. Some of them visualize the website structure only, some concentrates on the web mining and navigational pattern visualization, others visualize the web usage data together with its structure context. The pros and cons are then evaluated. A former proposed operator framework and related research work are also presented. In the end, the map algebra concepts are illustrated. In the next chapter, the WEBKVDS system is introduced. The visualized objects and concepts are discussed first. Then the system architecture is presented.

Chapter 3

The Web Visualization Framework

Previous chapters presented a brief introduction of the research area and a general review of the interesting related work that has been done. In this chapter, we first present our framework with the web visualization object terminology we use, then introduce the architecture of our visualization tool, the WEB Knowledge Visualization and Discovery System [15] (WEBKVDS), in subsequent sections.

3.1 Visualization Objects

As we mentioned, our motivation is to provide a convenient tool for web administrators to interactively manipulate visualization objects in order to discover interesting patterns and interpret the pattern knowledge in an ad hoc visual mining process. To accomplish that, a multi-tier object, the *web graph*, is defined to contain and represent both the web usage data, which are basically statistics extracted from the web log files, and the navigational patterns, discovered by web mining techniques, together with their context, which is the web site structure. The first tier, called *web image*, is a tree representing the structure of a given web site (or part of it), with nodes and lines, representing the web pages and links respectively. Each other tier, called *information layer*, represents usage information in a certain time period about the pages and links in the web image, or navigational patterns that are discovered by web mining processes. The web image is always visualized as the background of the information layers. Each layer is identified separately and can be displayed or inhibited when an object is visualized. Thus, combining these tiers and image together, which generates a web graph as result, is actually putting navigational data on their web structural contexts and localizing the usage information vis-á-vis the web site. We list and summarize the concepts we are using in the following.

3.1.1 The Web Image

A Web Image, also referred to as bare graph, is a tree representation of the structure of a web site or a subset of it. It contains two kinds of objects. These objects exist in web graphs, and can be visualized or inhibited by operators.

- *Vertex*

  A vertex represents a web page. A vertex is displayed as a node in the graph, representing an online HTML document.

- *Edge*

  Edges represent hyperlinks between pages. An edge is displayed as a line in the graph, representing a link that connects two pages.

A web image first has to have a root, which is always located in the centre of the graph. The image is then built by a rendering algorithm based on the selected root, the website topology and the usage information. Since the whole image is usually too big to be displayed clearly on the screen, it has a certain depth, which can be adjusted by the user. When visualized, the tree is displayed as a disc with the root as the centre. The nodes corresponding to other web pages are placed on a circular perimeter according to their depth level, successively away from the root. Figure 3.1 illustrates such discs.

Since the web is basically made up of HTML documents and hyperlinks between them, the edge and vertex objects are good enough to show the web site structure clearly. The rendering issues are discussed later in this chapter.

3.1.2 Information Layers

Similar to the way that scientists classify and manipulate geographic data such as altitude, temperature, etc, as map layers in GIS research, we represent web usage information and knowledge patterns as layers. *An information layer is a coherent*

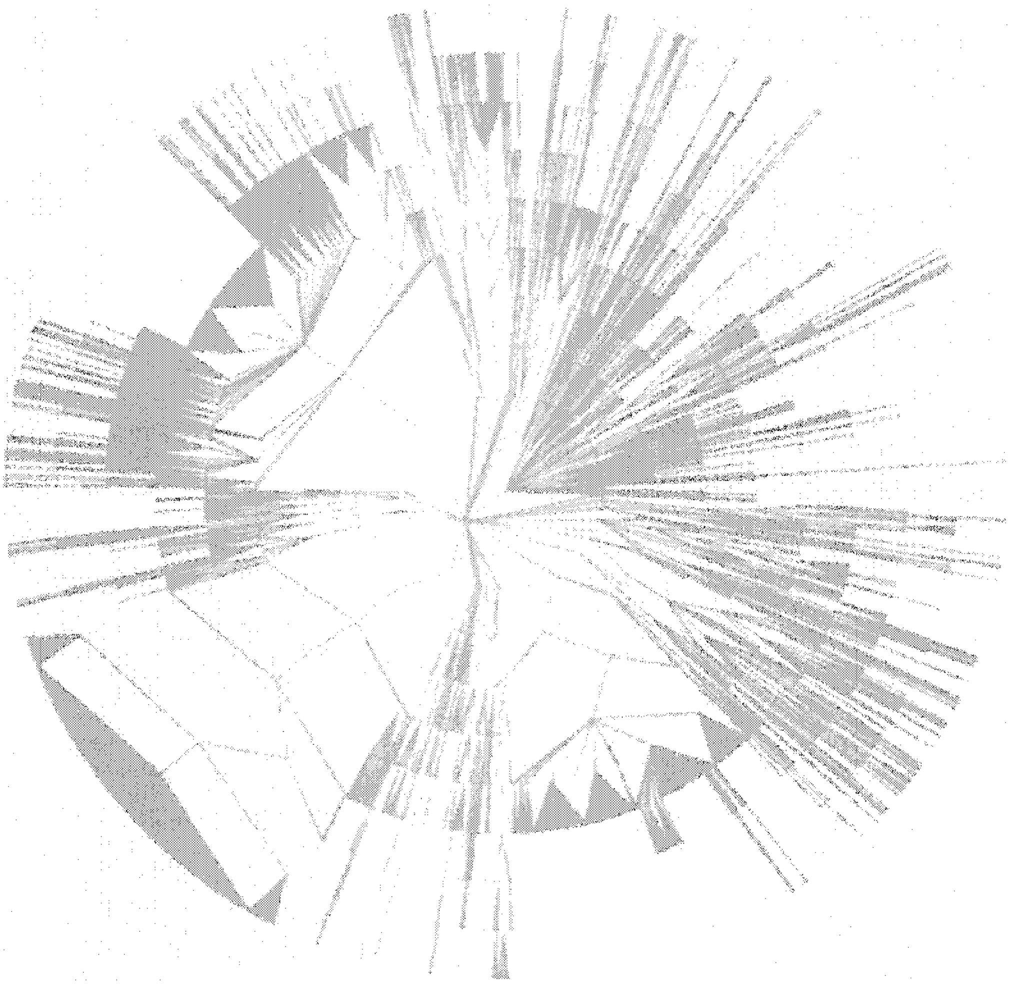


Figure 3.1: The Web Image(Depth=10) of http://www.cs.ualberta.ca/

collection of web data abstractions that can be laid over the web image. The contents of each layer are stored in the web image objects and represented by certain visualization cues of the objects. The user can decide which layer should be inhibited or rendered to show user behaviour and produce information-rich visualization.

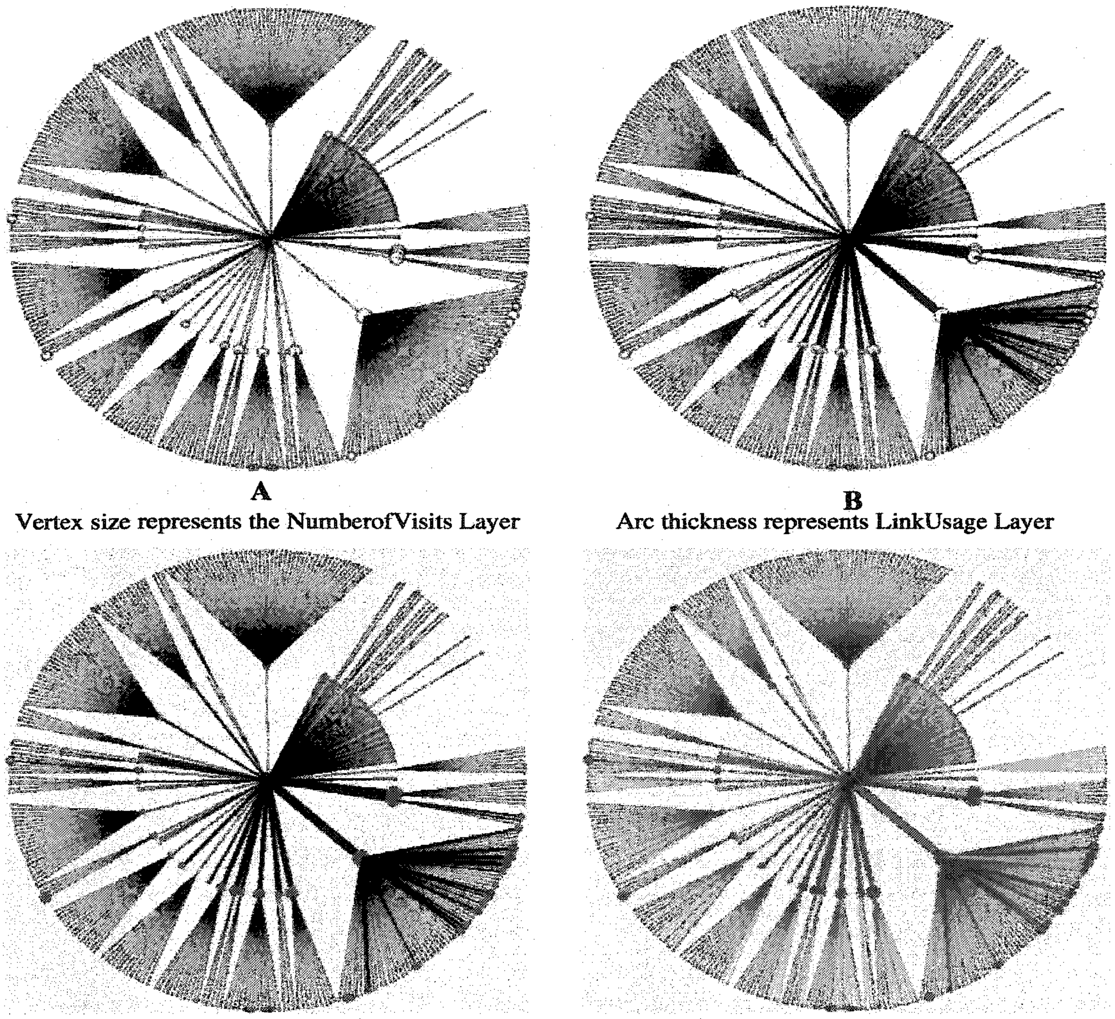
We identify here four basic information layers, which represent some statistics from the pre-processed web log files about web pages and the links that connect them. Users can distribute cues to these layers by themselves. The following provides one possible distribution choice.

- The *NumOfVisit* layer giving the visit statistics per page, represented by the vertex size.
- The *ViewTime* layer containing the average access time per page, represented by the vertex colour.
- The *LinkUsage* layer providing the usage statistics of each link, represented by the edge thickness.
- The *ProbUsage* layer showing the access probability of each link, represented by the edge colour.

Figure 3.2 shows the visualization of these layers overlaid on the web image. Note that visualizing different information layers together is easily achieved by combining different visual cues, which can help people better comprehend the visualized information.

3.1.3 Pattern Layers

While the basic information layers collect statistical data from pre-processed web access log files, *pattern layers hold the navigation patterns discovered by various web mining algorithms such as clustering, association mining and sequence analysis, etc.* We are able to evaluate these pattern layers by visualizing them. At present, we visualize three pattern layers.



A
Vertex size represents the NumberofVisits Layer

B
Arc thickness represents LinkUsage Layer

C
Vertex colour represents the ViewTime Layer

D
Arc colour represents the ProbUsage Layer

Colour Images of all the graphs are available at http://db.cs.ualberta.ca/webkvds/

Figure 3.2: Multiple Layer Visualization

- *Association Rule Layer* statistically showing the hidden relationship between certain pages. While the colour of the arrow head represents possible predefined interestingness measure, the arrow thickness represents the support of a rule and the arrow head size represents the rule confidence. Figure 3.3 shows an example for the Association Rule Layer.

- *Page Clustering Layer* and *Page Classification Layer* providing the page category information. These two layers are generated by different web mining methods, but when visualized, they can both be represented by the colour or the shape of the nodes in the graph, e.g, red can be used to represent department news pages while blue can be used to show faculty home pages.

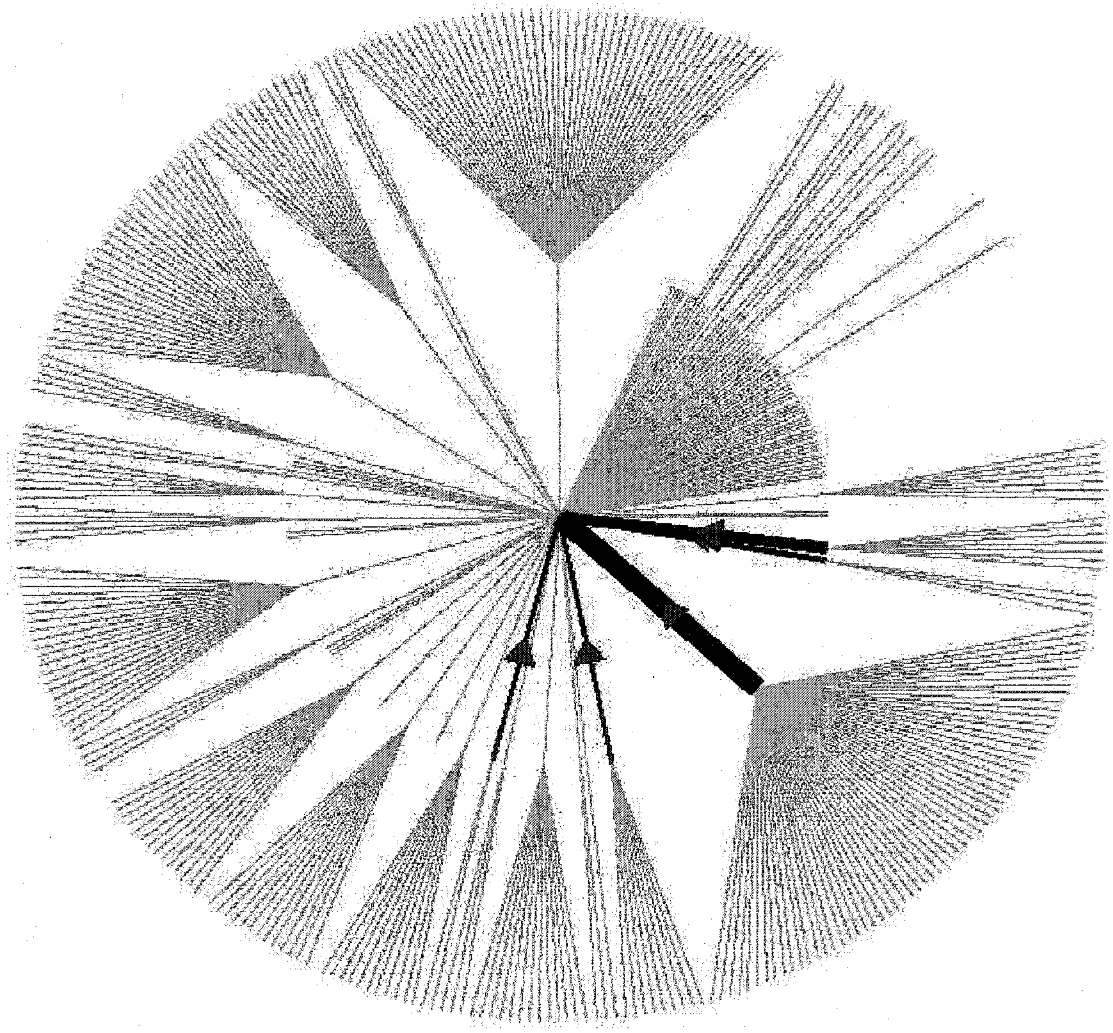
3.1.4 Web Graphs

A web image with one or more information or pattern layers constitutes a web graph, which is a visualization of the web users' former activities, a graphic description of usage information or various patterns as web mining results. The web graph object assembles together data and knowledge discoveries about the navigational behaviour in their web structure context. It can on one hand be rendered, and on the other hand be manipulated for further interpretation and application. When rendered, its web image is always acting as its background, the edge and vertex objects that hold the selected information layers are visualized to display the information. Examples of web graphs are shown in Figure 3.2.

Generally speaking, a web graph can be created by mapping several selected layers on a web image or by manipulating existing web graphs using a given set of web graph algebra operators, which are introduced and discussed in detail in subsequent chapters of this dissertation.

3.2 The Visualization System

To visualize the objects described above, we designed and implemented a visualization system, named WEB Knowledge Visualization and Discovery System (WEBKVDS), which is mainly composed of two parts:



Support= 0.4%   Confidence = 40%

Figure 3.3: Pattern Layer Visualization

- FootPath [53].

  The web graph object rendering engine for visualizing the web structure with the different data and pattern layers.

- Web Graph Algebra.

  An operator, variable and expression set for manipulating and operating on the web graph objects for visual data mining purpose.

After we present the architecture of our system in the following, the rendering engine is introduced. The algebra is explained in detail in the next chapter.

3.2.1 System Architecture

Figure 3.4 illustrates the WEBKVDS system architecture. Given a website and its web access log database, a web image can be generated using the site structure information and the usage data obtained after pre-processing of the web access log files. The usage data is also used on one hand to generate basic statistic information layers and on the other hand as patterns via web mining modules. The navigational patterns discovered by these web mining modules can be used to generate pattern layers. These different layers can be mapped onto the web image to create web graphs. Finally, the web graph algebra allows an ad hoc visualization manipulation to create new graphs and all these graphs can be visualized by our rendering engine, FootPath. Therefore, the combination of the algebra expressions and the visualization with FootPath constitutes the basis of our visual mining system, WEBKVDS.

To describe the characteristics of WEBKVDS by the web visualization dimensions (Figure 2.11), we can try to locate the system in the eight possible sub-spaces of the model. The system can visualize aggregate or individual usage behaviour by selecting specific information layers and filtering. It can also handle small and large web structure quite well by picking appropriate depth and root of the web image. However, dynamic visualization are not incorporated yet. Therefore, WEBKVDS can be found in all four "static" sub-spaces, but not yet "dynamic" ones.

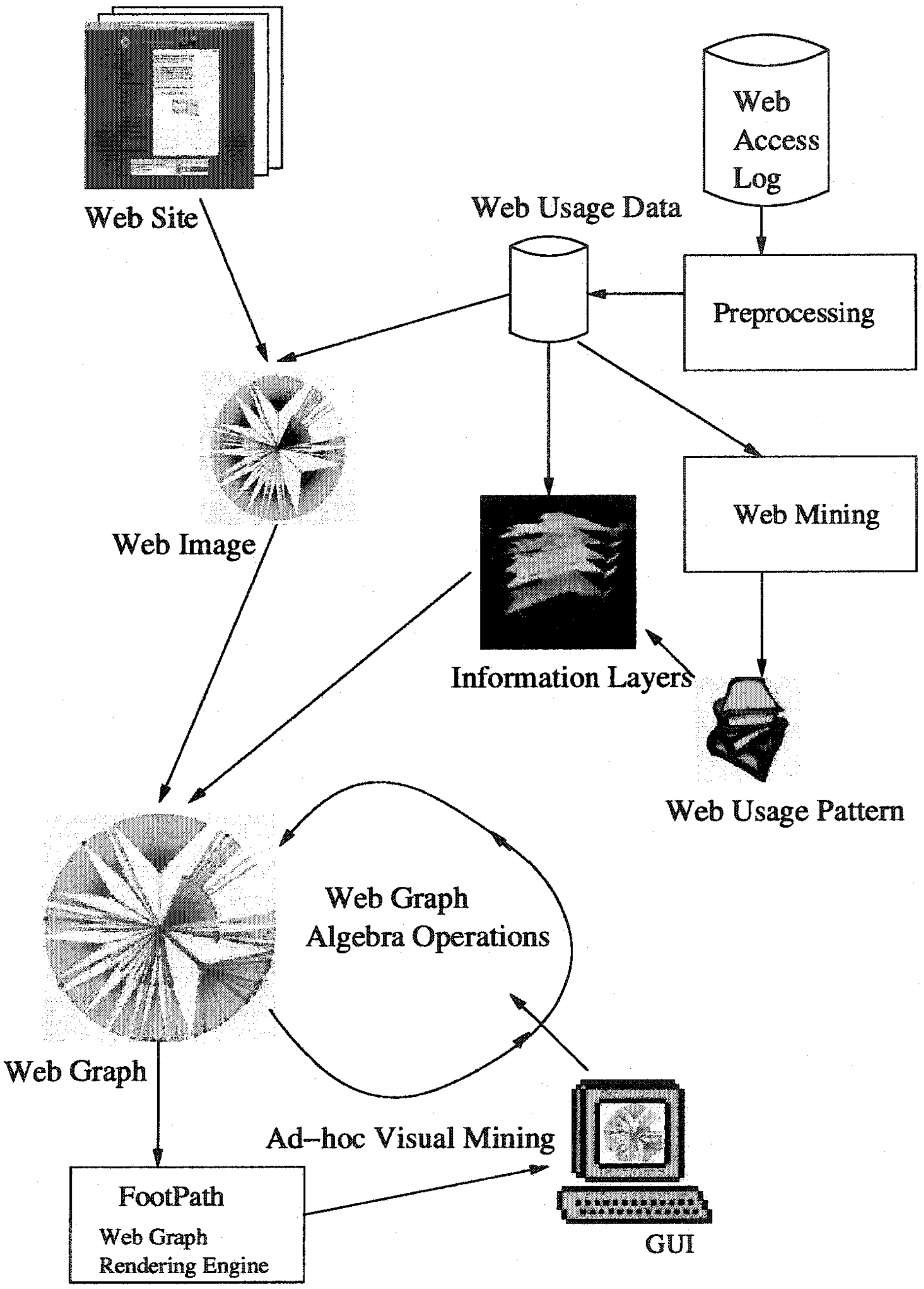


Figure 3.4: The Web Graph Visualization Architecture

3.2.2 FootPath: A Web Graph Rendering Engine

FootPath [53] is the web graph rendering engine for our knowledge visualization and discovery system. The tool is not a contribution of this thesis, however, since it is developed to visualize the system objects, it is briefly introduced in this section. FootPath displays the web graph by first rendering the web image as the graphic background and then distributing visual cues such as colour and size of the nodes, thickness and colour of the edges, etc., to vertices and edges to represent the information layers these objects are holding. To avoid possible serious occlusion problems, we adopted the Disktree idea, first proposed in [18], to visualize the web image topology in a 2D space. In the Disktree representation, a node represents a web page and an edge represents the hyperlink that connects two pages. The root, as we have mentioned in web image introduction part, is a node located in the centre of the rendering image. It is usually the home page of the site or a given page to start the tree-building algorithm. Nodes with different perimeters are classified into levels, which form discs.

The original Disktree algorithm applies the Breadth First Search (BFS) method to convert a connected graph into a tree. For each node, only one incoming link is represented. The strategy is simply to represent the link that first come into the node following the scan of the web site topology. Another alternative method is the Depth First Search (DFS). Although the BFS and DFS method are widely used because of their simplicity, the huge number of the nodes in each level of the disk in their result images makes it really difficult to overlay additional information layers without overlapping other objects and blurring the graph.

To better visualize the web structure and usage data, on one hand, FootPath utilizes the Disktree algorithm applying a new method, called *Usage-Based Method* [53]. Instead of keeping the first incoming link for each page, the Usage-Based method chooses to keep the link with the highest usage count in the web access log files to build the tree. This strategy leads to deeper trees but less nodes per level, without missing any pages. Moreover, the result tree better illustrates the internal relationship between the web usage and structure topology. On the other hand, to avoid occlusion problem due to various edge thicknesses and node sizes, FootPath provides a dynamic layout method to more efficiently use the disc area. More information regarding the rendering and layout issues of FootPath can be found in [53].

Chapter 4

The Web Graph Algebra

After introducing the visualizing system architecture and the FootPath rendering engine, we present our web graph algebra, consisting of a set of operators and rules for visualization objects, to manipulate web graph objects. The operators together with web graph objects are used to form expressions and equations. By executing these expressions, e.g. $WebGraph_1 + WebGraph_2$, we can generate more web graphs for possible web user behaviour investigation. The significance of this algebra is that this scheme makes it possible to do ad hoc visual data mining, which can help people better understand and explain the visualizations from their own points of view. The idea of the web graph algebra is similar to the Map Algebra, which has been introduced in Chapter 2. By analogy, while the researchers in Geographic Information System (GIS) operate the geographic layers on a map, we intend to do web usage mining and data operations on a web graph.

The method we use to develop our web algebra is both top-down and bottom-up. For example, in background work [50, 61], the authors pointed out that by manipulating certain navigational data and hyperlink usage records, entry points of a website can be visualized. This observation has motivated us to find further examples that could be generalized. The idea of a general web graph algebra arises from the desire to have a simple and precise abstract characterization of web visualization objects. This can be done by respecting algebraic properties, in a top-down fashion, and by determining useful objects and operations in a bottom-up fashion.

Furthermore, we can view the bottom-up method as comprising three steps. First, we gather web structure or usage data and their visualizations, defined by terms such as web image, web graph and information layer; Second, we consider possible behaviour analysis and usage investigation questions, which are important or potentially important to understand the online users' activities. Finally, if a visualization manipulation can generate a visualization to answer the analysis questions based on the provided data and visualizations, it is defined as an operator.

The algebra operators can be categorized into two classes: Unary and Binary. Unary operators strip, select specific values from given information layers, or extract sub-parts of the underlying web image. Binary operators combine two web graphs by executing various expressions on the common information layers as well as the vertices and edges constituting the web images. Different operators affect different stages of the web visualization scheme to achieve its functionality. Essentially, the algebra mechanism is designed to assist information layer manipulation and web analysis visualization once the resulting web graph is rendered by the FootPath engine.

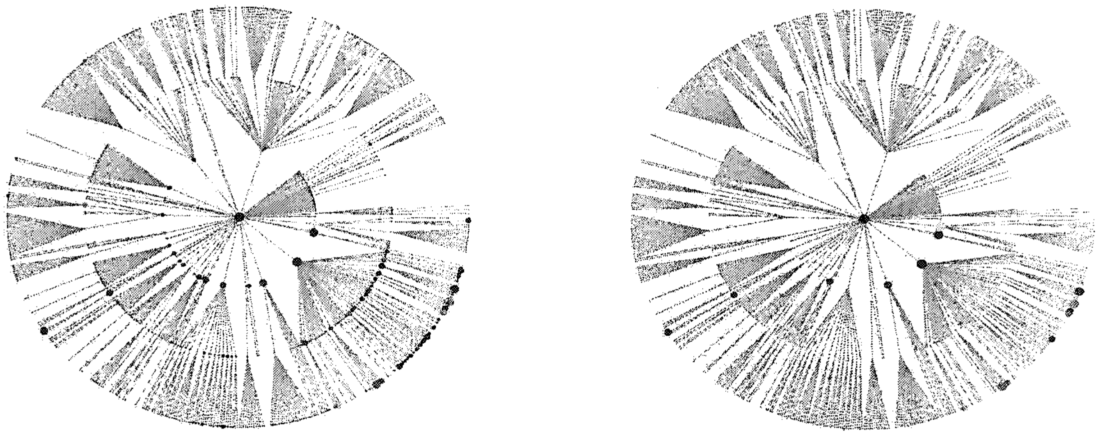
4.1 Unary Operators

Currently, we have seven unary operators, which take only one web graph object as a parameter. Basically, they select objects (i.e. vertices and edges) that satisfy some given conditions, which can be structural or regarding their layer content, and generate a new graph accordingly. We enumerate and summarize them in the following.

The first unary operator *FILTER* selects from web graph G the objects(i.e., vertices and edges), whose layer contents fit within the given threshold parameter range and transfers them into graph G' with no layer content change. The definition is: *For all the objects α in web graph G, we examine its layer content value v with the given layer type L and threshold (t_{down}, t_{up}) and transpose α into G' if it meets the condition $t_{down} < v < t_{up}$.* The considering value here is numbers representing the usage, however, they can also be extracted values from the web content, e.g, meta tags describing the page topic.

$$G' = FLT_{L,(t_{down},t_{up})}(G)$$

This operator is used to refine the graph: if the original graph is showing the complete usage record of the website, the user can apply the *FILTER* to display more specific range of the navigational information, for example, all the pages (vertices) that have a visit statistic value (content of $NumOfVisit$ layer) bigger than 1000 (Figure 4.1). Moreover, the *FILTER* operator acts as a first step for many other operations and further analysis manipulations. These are discussed in Chapter 5.



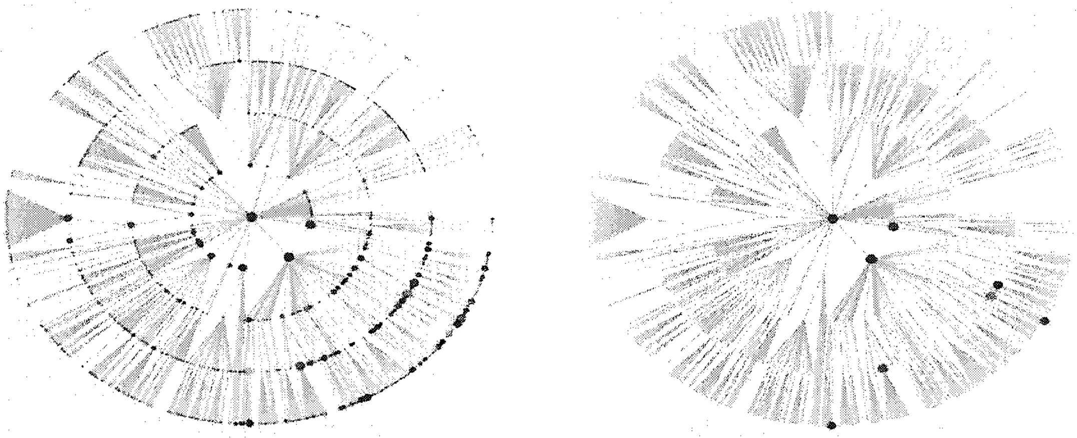
Before FILTER **After FILTER**

Figure 4.1: Operator FILTER: $G' = FLT_{NumofVisit,(1000,\infty)}(G)$

The second unary operator $SELECT$ chooses N biggest (top) or smallest (bottom) objects according to their layer contents from web graph G and transfers them into graph G' with no layer value change. The definition is: *For all the objects α in web graph G, we examine its layer content value v with the given layer type L and transpose α into G' if it meets the condition that α is the top/bottom N objects according to their layer content in G.*

$$G' = SLT_{L,N,TopOrBottom}(G)$$

This operator is designed for object filtering. While operating together with *FILTER*, it can select almost all possible navigational information with specific layer value range setting. Thus, the two operators are usually used first in graph manipulations to filtrate the graph. Figure 4.2 shows an example for *SELECT* 10 top vertices regarding to layer *NumofVisit*. Note that there are only eight visualized nodes in the result graph. The reason is that the depth parameter of that graph is not big enough to include the levels of the missing nodes.



Before SELECT

After SELECT

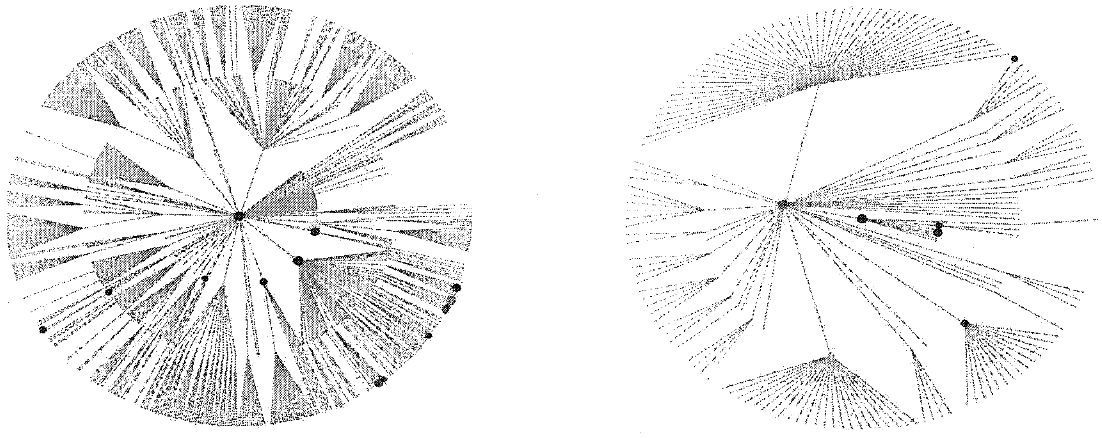
Figure 4.2: Operator SELECT: $G' = SLT_{NumofVisit,10,TOP}(G)$

The third unary operator *CENTER* reconstructs the rendered image with the given new root, includes all the original objects in web graph G, and transposes them into graph G' with no layer content change. The definition is: *For web graph G, we select a node R that exists in its web image and rebuild the whole graph with R as the new root.*

$$G' = CTR_R(G)$$

This operator makes it possible for the user to maximize and view the usage information of any parts of the web. Usually, some information are unexpectedly ignored due to the fact that the monitor screen area is limited and the peripheral vertices and edges are too close to each other. With this operator, the user can locate the information from any parts of the website on the center of the screen and have the best view for observation. An example is shown in Figure 4.3.

Because of the fact that the web image has a depth parameter when visualized, the new root may or may not physically appear in the original visualization. This phenomena brings two special cases of the operator *CENTER*: "*CONFINE*" and "*EXPAND*", concerning the visualized object appearance. *CONFINE* means every object that is visualized in G' was formerly in G and *EXPAND* means G' not only contains the objects that have appeared in G but also includes other objects, which were inhibited and undisplayed before.



Before CENTER

After CENTER

Figure 4.3: Operator CENTER: $G' = CTR_R(G)$

The fourth unary operator *ROTATE* rebuilds the information graph with a new angle towards a given direction. It includes all the objects in web graph G, and transfers them into graph G' with no layer content change. The definition is: *For web graph G, we rotate the graph for an angle A in direction D (clockwise or anticlockwise) and transfer the graph with no layer content change.*

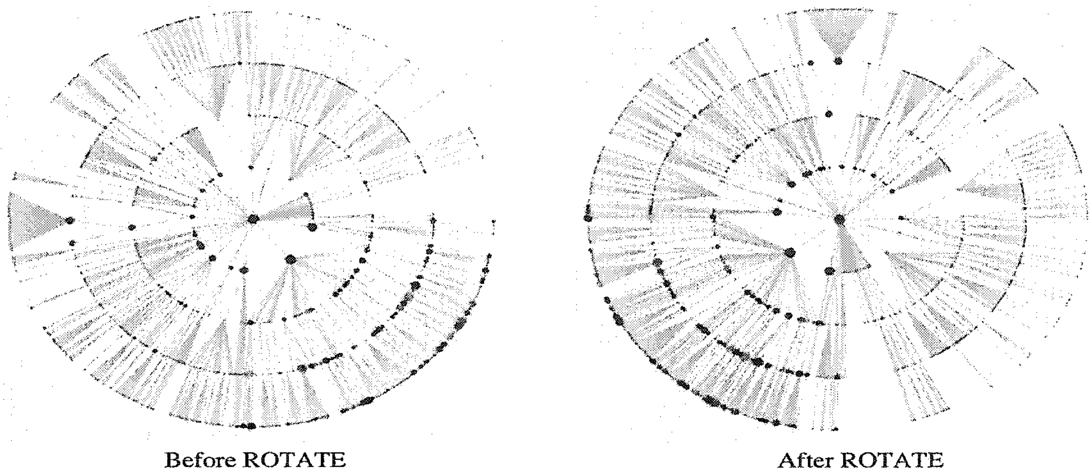
$$G' = ROT_{A,D}(G)$$

The major usage for this operator is to compare the graphs with different degrees to make it possible for human visual sense to discover the knowledge patterns For example, the whole graph is simply rotated in Figure 4.4.

The fifth unary operator *CONNECT-TO* selects the vertices that connect to current displayed objects, vertices and edges, in web graph G and transposes them into graph G' with no layer content change. The definition is: *For all the objects α in web graph G, we take the vertex β and transpose it into G' if it meets the condition that, β connects to α.*

$$G' = - > .(G)$$

This operator is usually used to further analyze the property of a set of visualized objects. For example, Figure 4.5 shows a set of pages that with an average visit

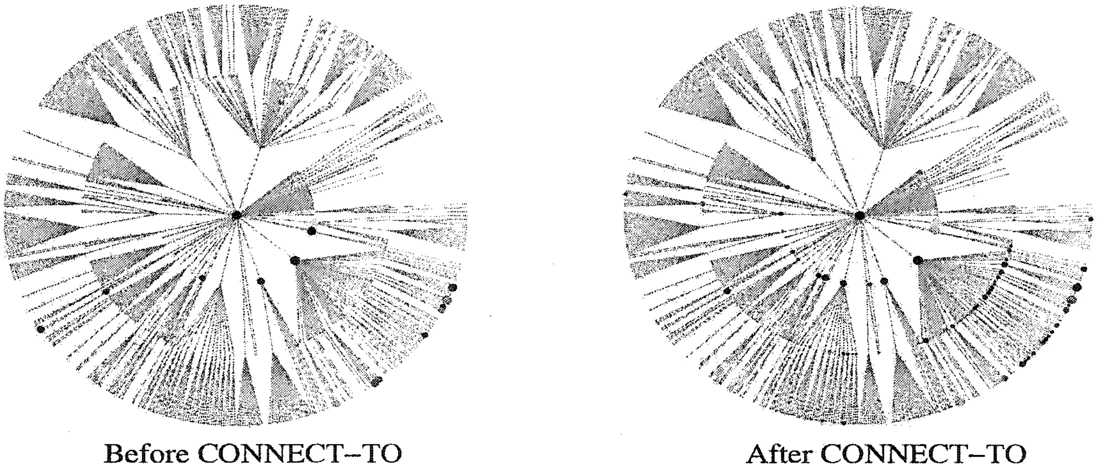


Before ROTATE

After ROTATE

Figure 4.4: Operator ROTATE: $G' = ROT_{90,Clockwise}(G)$

more than 1000 (achieved by *FILTER*), the *CONNECT-TO* operator is then used to visualize the pages that lead the user to these “hot” pages.



Before CONNECT–TO

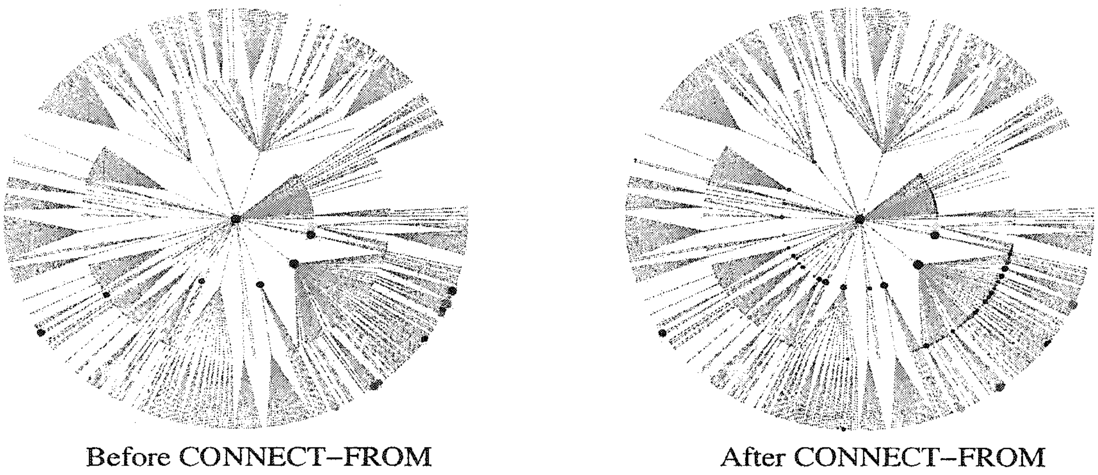
After CONNECT–TO

Figure 4.5: Operator CONNECT-TO: $G' = - > .(G)$

Contrary to the above operator, the sixth unary operator *CONNECT-FROM* selects the vertices that are connected **from** current displayed objects, vertices and edges, in web graph G and transposes them into graph G' with no layer content change. The definition is: *For all the objects α in web graph G, we take the vertex β and transpose it into G' if it meets the condition that, β can be connected from α.*

$$G' = .- > (G)$$

This operator is similarly used for farther analysis of the characteristics of an available web graph. Figure 4.6 shows a set of "hot" pages and the set of pages where those "hot" pages and links lead the users to.



Before CONNECT–FROM

After CONNECT–FROM

Figure 4.6: Operator CONNECT-FROM: $G' = .- > (G)$

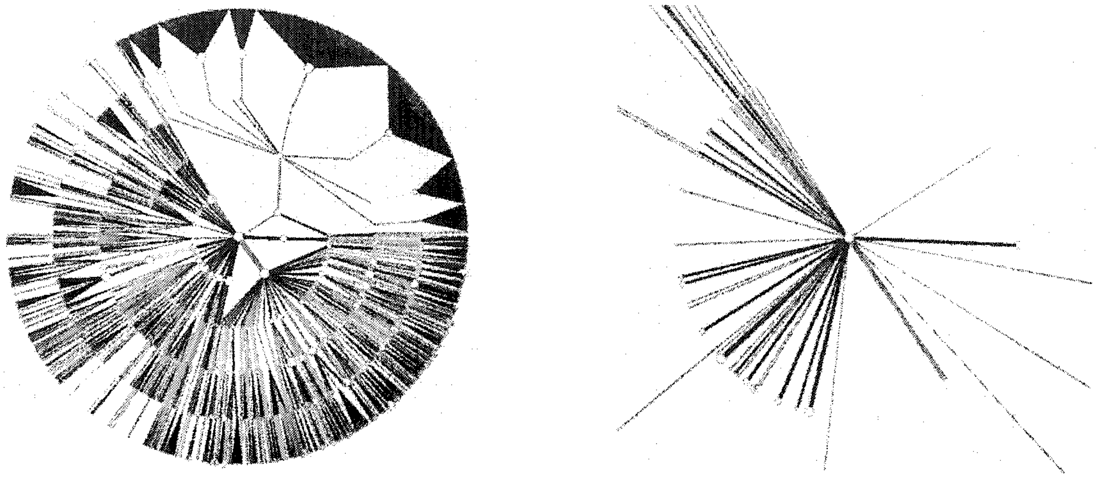
Finally, the FOCUS operator is designed to show how the links on a specific page are being used. The definition is: *For one given vertex object V in web graph G, we transpose V, all the edge objects corresponding to its out-links, and all the vertex objects that are connected from V into G'. V is set as the root for G'.*

$$G' = \underset{V}{\odot}(G)$$

Although most of the web links are not visualized in the original image because of the fact that the usage based algorithm only selects the link with the highest usage to build the topology representative tree, all links from the given node are visualized and all other unrelated objects are inhibited to avoid occlusion in the result graph. Thus, administrators can use the operator to verify whether the users are navigate the web as expected as Figure 4.7 shows.

4.2 Binary Operators

We have six binary operators at present. Different from unary operators, the binary ones take two web graphs as parameters, select objects that satisfy given conditions,



Before FOCUS

After FOCUS

Figure 4.7: Operator FOCUS: $G' = \bigodot_{Root}(G)$

calculate the new layer content and create the result graph accordingly. We enumerate and summarize the binary operators in the following.

The first binary operator *ADD* combines together the usage data and the structure of two different web graphs. The definition is: *For all the objects α exists in both web graph G_1 and G_2, we transpose α into G' with the sum of the respective content from the available layers. For the object β that exist in only one graph, we keep it into G' with no change. Note that each individual information layer sum is calculated separately.*

$$G' = G_1 + G_2$$

This operator makes a navigational summary possible. For example, to see an information visualization of several consecutive months' data together, simply adding the graphs achieves that (Figure 4.8). The result graph represents an aggregation over a time tube as defined in [18].

The second operator *MINUS* combines the structure, computes and visualizes the difference between the usage data of two graphs. The definition is: *For all the objects α exists in both web graph G_1 and G_2, we transpose α into G' with the difference of respective contents from the available layers. For the object β that exists in only one graph, we keep it into G' with a positive value (no change) in G' if it is from G_1 and a negative value if it is from G_2. Note this operator also acts on each separate*

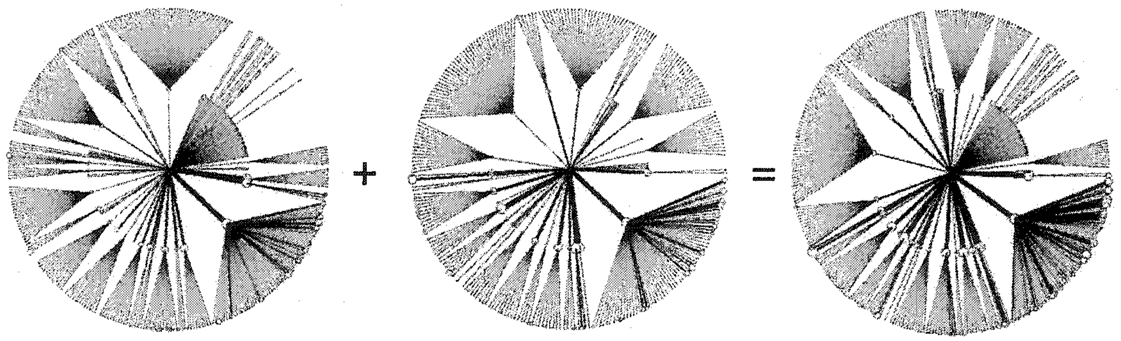


Figure 4.8: Operator ADD: $G_1 + G_2 = G'$

information layers.

$$G' = G_1 - G_2$$

The operator makes the comparison analysis possible. For example, the difference of two months' usage data is displayed by month visualizations, but absolutely not easy to be observed. By applying the MINUS operator, we can generate a understandable graph describing the dissimilarity of two visualizations. The Time Tube analysis defined in [18] actually is using the MINUS operator to compare graphs of different time periods (Figure 4.9).

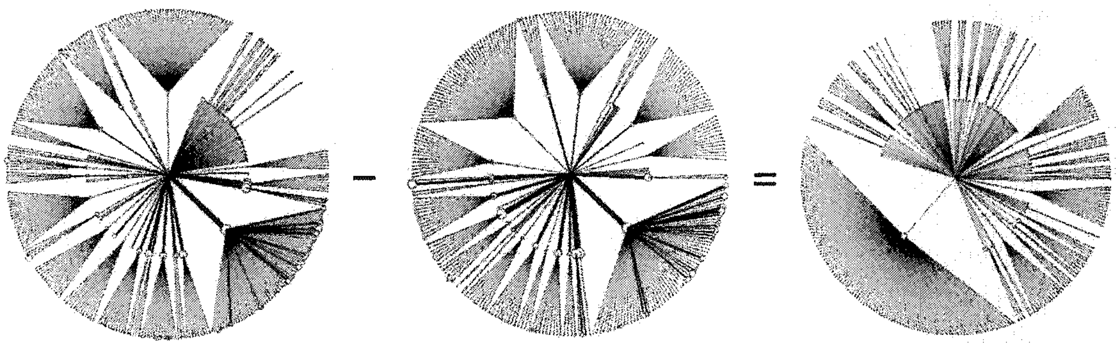


Figure 4.9: Operator MINUS: $G_1 - G_2 = G'$

The third operator *COMMON* combines the structure, selects and visualizes the similarity of the usage data of two web graphs. The definition is: *For all the objects α exists in both web graph G_1 and G_2, we transpose α into G' with the minimum of the respective contents from the available layers. For the object β that exists in only one graph, the value becomes 0, which is also the minimum. The operator acts on* *separate information layers.*

$$G' = G_1 :: G_2$$

As its name may imply, the *COMMON* operator extracts the resemblance and highlights the overlapping part of the two input graphs, similar to the notion of intersection. For example, a vertex with *NumOfVisit* and *ViewTime* layer values respectively 90 and 40 in G_1, 50 and 20 in G_2 are transposed into G' with *NumOfVisit* value 50 and *ViewTime* value 20. Figure 4.10 shows the similarity between the page sets that has more than 1000 average visit of two different months, using COMMON operator.

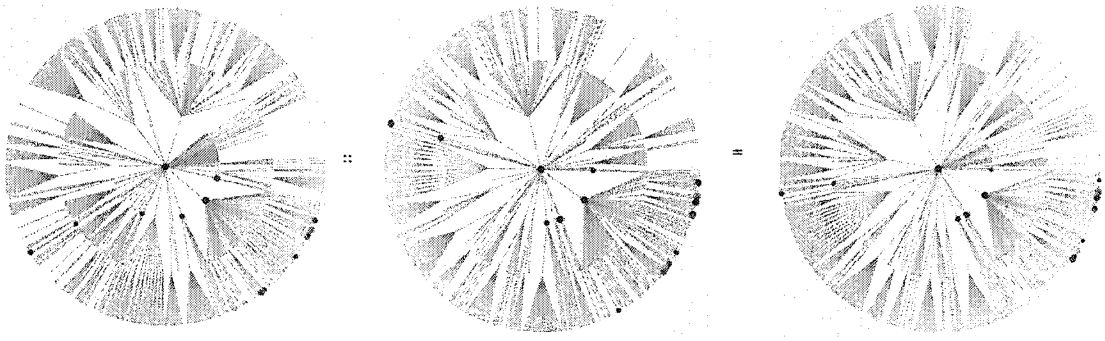


Figure 4.10: Operator COMMON: $G_1 :: G_2 = G'$

The fourth binary operator *EXCEPT* keeps the original structure, but visualizes the visualization difference of two web graphs. The definition is: *For all the objects α visualized in web graph G_1, we transpose α into G' if it is not visualized in G_2, with no changes in the layer content. Note that we keep the web image of G_1 here while the above three operators all rebuild the image since the usage data are changed and some objects are added.*

$$G' = G_1 \ni G_2$$

Contrary to *MINUS*, which shows the graph layer difference, *EXCEPT* intends to reveal the structural difference between any two graphs. Additionally, while the *MINUS* operator makes it possible to show the time trends, the *EXCEPT* operator provides new possibilities for Time Tube comparison, such as identifying the recent popular pages. By applying *EXCEPT* on last month's graph from the current month's and adjusting an appropriate *FILTER* threshold setting, we can show the pages that has just become "hot" recently (Figure 4.11).

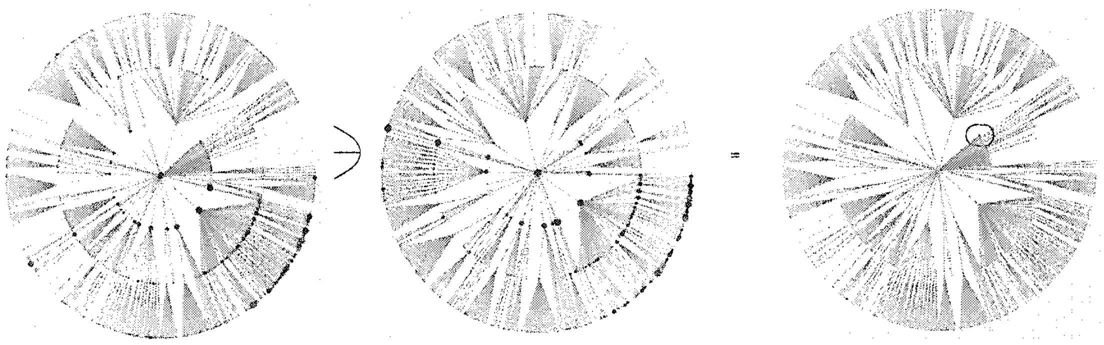


Figure 4.11: Operator EXCEPT: $G_1 \ni G_2 = G'$

The fifth operator *MINUS_IN* subtracts values **across** different information layers. Variable graphs G_1 and G_2 are required to have information layer *NumOfVisit* and *LinkUsage*, respectively. The operation is basically similar to *COMMON* except that the *NumOfVisit* layer of the resulting graph G' is calculated in the following way:

$$G' = G_1 - .G_2$$

The NumOfVisit value of a vertex N in G' is calculated by subtracting the sum of the values in G_2's LinkUsage layer of all links that connecting **to** *N from the NumOfVisit value of N in G_1.* An example is shown in Figure 4.12 and its meaning is explained in the next chapter.

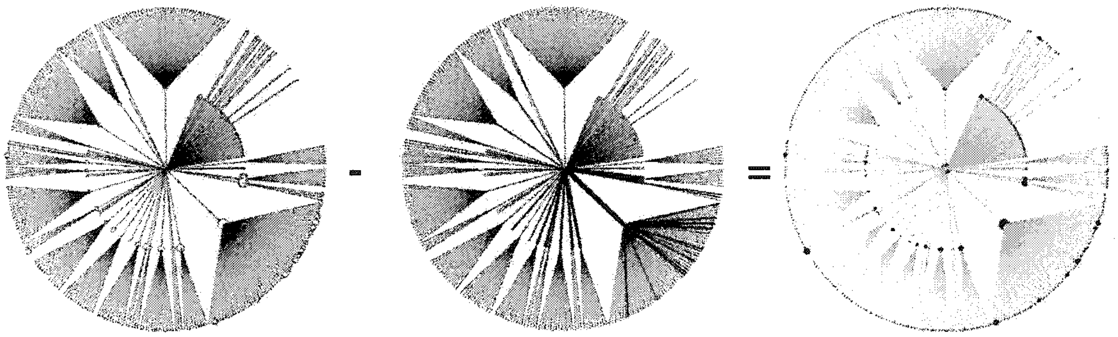


Figure 4.12: Operator MINUS_IN: $G_1 - .G_2 = G'$

The specification of the last binary operator *MINUS_OUT* is exactly like *MINUS_IN* except that the *NumOfVisit* layer value of the resulting graph G' is differently calculated. It is evaluated as the following:

$$G' = G_1. - G_2$$

*The NumOfVisit value of a vertex N in G' is calculated by subtracting the sum of the values in G_2's LinkUsage layer of all links that connecting **from** N from the NumOfVisit value of N in G_1.* An example is shown in Figure 4.13 and its meaning is explained in the next chapter.

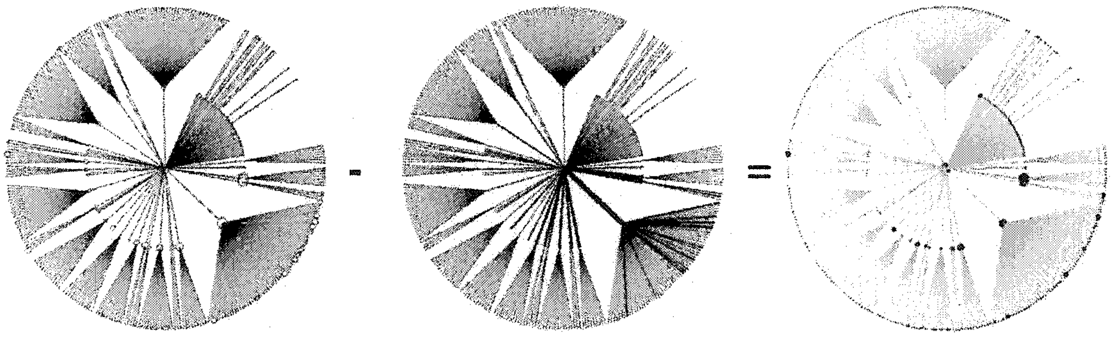


Figure 4.13: Operator MINUS_OUT: $G_1. - G_2 = G'$

These operators look simple but are in fact powerful. By applying them to combine and compare simple web log data visualizations, we can interactively generate many other explainable and information-rich visualizations.

4.3 Visual Effect of Operators

In the above, we present the definition of 13 operators, 7 unary and 6 binary, of our web graph algebra. Although all of them play important roles in our visual web mining process, the approaches to achieve their functionalities, i.e, the way these operators change web graph visualization objects, are diverse. In this section, we discuss the internal relationship between the operators and web graph objects to help us better understand the working scheme of the algebra.

The web graph contains the whole web topology and the related usage data, but it is only partly visible for many reasons. Some links are not available because of the tree representation issues, some pages exist but are inhibited by certain operators, and adjusting depth or certain root settings can also change the graph. Figure 4.14

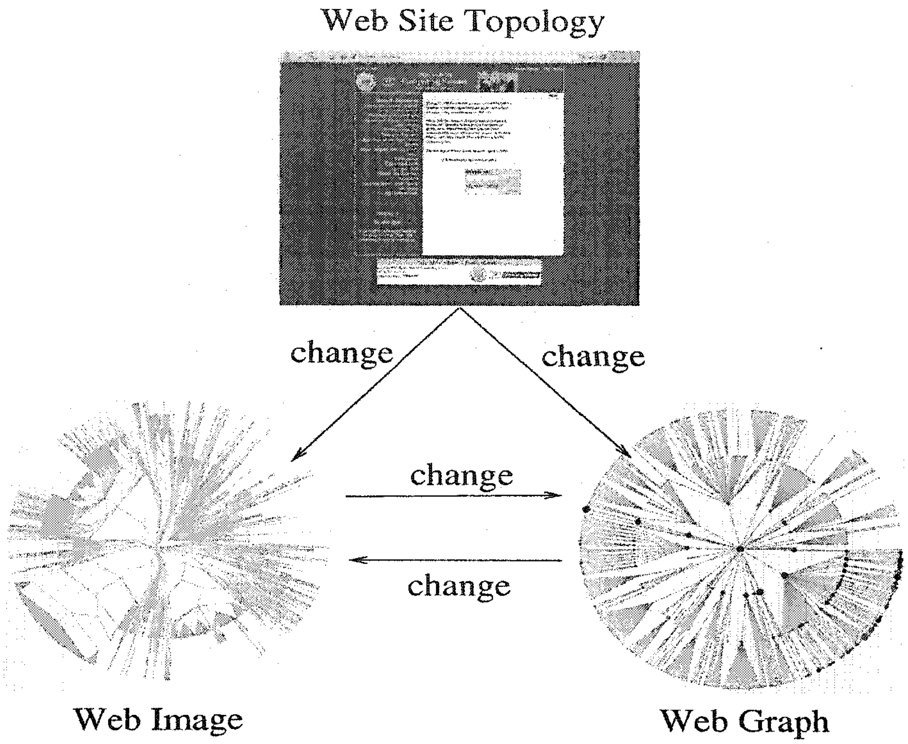


Figure 4.14: Three Functional Stages

shows the visual effects of the operators. There are basically three things operators can affect. Since the web graph is built on a web image, which is extracted from the web site topology, changing the web site topology would result in a reconstruction of all the visualizations; changing the web image would lead to rebuilding the image and the graph; changing the web graph has a side effect on the image, because of the fact that the image is built based on the usage data, which can be changed by some binary operators. Operators can be categorized into three classes based on which they affect.

Affecting Web site topology

Five binary operators, *MINUS, ADD, COMMON, MINUS_IN* and *MINUS_OUT*, can be categorized into this class if they operate on two graphs of different time periods. Since the graphs are built based on usage data in different time, their web site structure may not be the same. To display all the useful information, we include all the vertices or edges that appear in these two graphs. In other words, all the pages and links in the web site history in these two time periods are represented in the result graph. Even though some of them may not exist in the later graph, they can be used to display the formal usage information.

Affecting Web image

Operator *CENTER* and *ROTATE* affect the web image stage directly. While the web topology remains the same, the web image and then the web graph have to be reconstructed after choosing a new root or a new angle. Additionally, operator *FOCUS* changes the web image in a different way, by including some edge objects that were formerly ignored by the layout algorithm to show the complete link usage for a specific page.

Affecting Web graph

In this class, *MINUS, ADD, COMMON, MINUS_IN* and *MINUS_OUT* is only changing the web graph stage if they operate on two graphs with the same web site topology. However, all of these operators modify the usage data stored in the information layers, which might lead to a web image reconstruction. For example, after applying the ADD operator, another link instead of the original one, which is displayed in the former web image, now has the highest usage count and is visualized in the result graph. We call this the *side effect* of the web graph stage to the web image stage. Some other operators that are in this category but wouldn't result in any *side effect* to the web image are *FILTER, SELECT, CONNECT-TO, CONNECT-FROM* and *EXCEPT*, since they don't change any information layer contents.

Chapter 5

Using the Algebra

To help web administrators understand web navigational behaviour, web mining researchers have already developed various analytic packages that can provide basic performance indicators and generate standard reports to assess navigational patterns, evaluate the performance and identify potential problems for possible improvement. Some supplementary analysis are proposed by Kohavi and Parekh in [32]. In fact, each of these analyses, which describe particular aspects of the behaviour of online users, can be combined into one simple investigation question according to the administrators' current information need. Unfortunately, because of the fact that these information needs are usually diverse according to the dynamic surfing behaviour and administrator's personal interests, the static analytic reports might not be good enough to contain and emphasize the requested information. One way to solve the problem is to provide an interactive environment that allows the user to manipulate the web usage data in an ad hoc manner and generate the visualization of the interesting information. This interaction can be achieved by the web graph algebra.

To prove the feasibility and usability of the algebra, we adopt the web usage data from the log files of our department home page *http://www.cs.ualberta.ca* as input to the FootPath and then manipulate the generated visualizations by the possible algebra operation expressions. The details of the manipulation are presented in this chapter. First, the data preprocessing steps are introduced, then the implemented system functionalities are presented. Finally, knowledge discovery based on web visualizations and web graph algebra is illustrated and discussed.

5.1 Data Preparation

In our visualization system, what we use to feed our FootPath engine is the structural and usage data extracted from the pure web log data of our department website, thanks to Tong Zheng's work [61]. The abstraction process details and the used methods are beyond the scope of this dissertation, but can be found in [61].

An example snapshot of the resource data is shown in Figure 5.1. The user navigation information is separated into sessions. Each session has a user ID, identifying the host address from which the access was performed. "IDFrom" defines the method for generating the UserID, e.g., from cookie or from "IP + user agent". The surfing requests are stored after the user navigation path completion. Each request log has an ID of the request page, the transfer size of the requested document (*0* means unknown), the time stamp of the request and the page view time (*-1* means unknown and *-2* means that page request is not in the log and obtained through path completion.) Additionally, before loading the data into our visualization system, we statistically preprocess the data, calculate the total number of visits and the average view time of each page, the total number of access and the probability of the usage of each link, and save all these information in a file with a special suffix. There are two other files providing the *page id - page URL* mapping and page topology information.

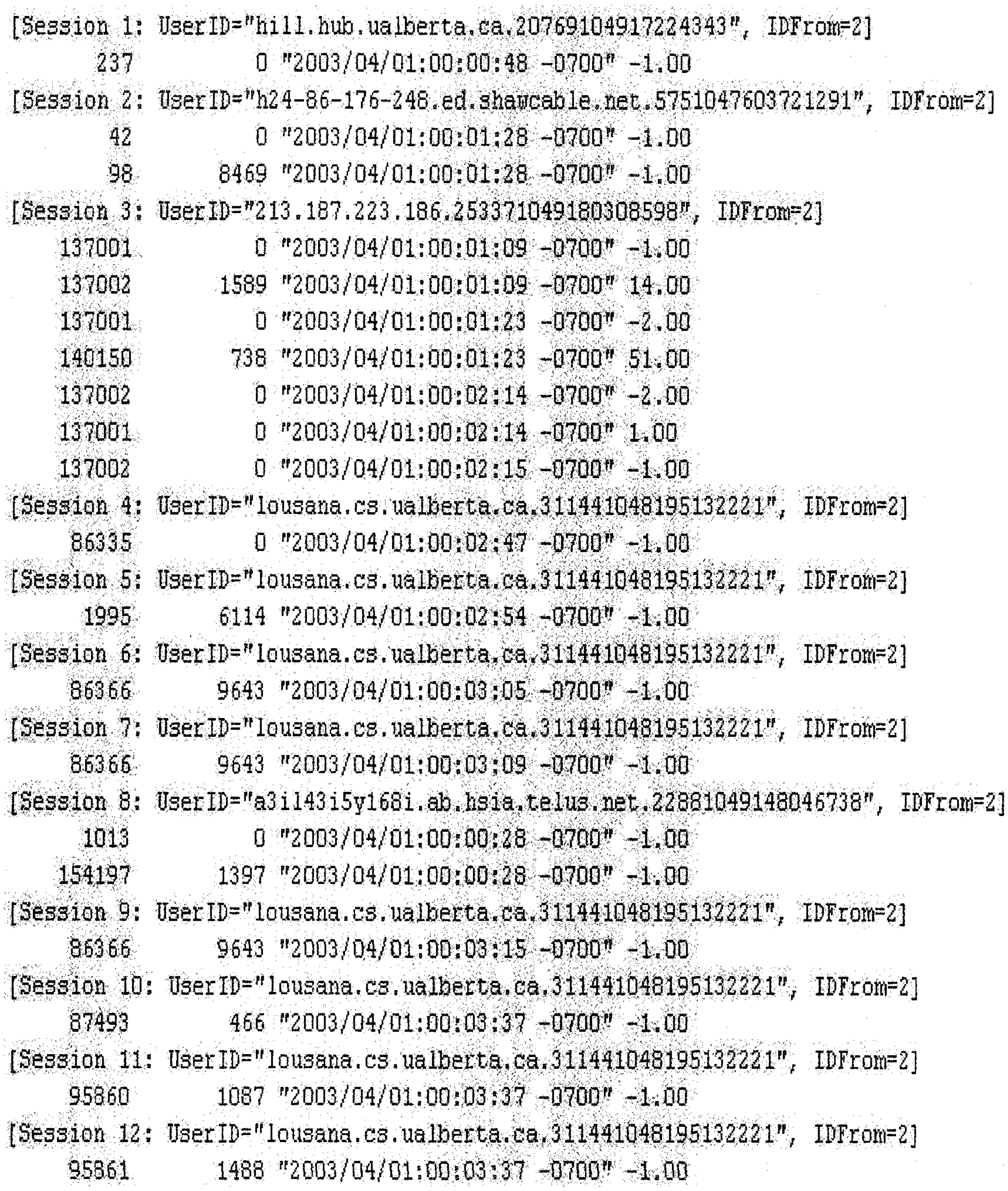
Based on these web usage and structural data, we are able to build the web graph of our department and manipulate them to do visual mining operations.

5.2 System Functionality

The visualizing system provides several functionalities to facilitate the visual mining process. They can be classified into three classes based on their purposes as the following.

- To build a convenient human-computer interface

Although there is only one graph that can be visualized and shown at one time in WEBKVDS, the visualization system provides a multi-panel so that users can conveniently switch between relevant graphs to compare the similarities and differences.



```
[Session 1: UserID="hill.hub.ualberta.ca.20769104917224343", IDFrom=2]
      237          0 "2003/04/01:00:00:48 -0700" -1.00
[Session 2: UserID="h24-86-176-248.ed.shawcable.net.5751047603721291", IDFrom=2]
       42          0 "2003/04/01:00:01:28 -0700" -1.00
       98       8469 "2003/04/01:00:01:28 -0700" -1.00
[Session 3: UserID="213.187.223.186.253371049180308598", IDFrom=2]
   137001          0 "2003/04/01:00:01:09 -0700" -1.00
   137002       1589 "2003/04/01:00:01:09 -0700" 14.00
   137001          0 "2003/04/01:00:01:23 -0700" -2.00
   140150        738 "2003/04/01:00:01:23 -0700" 51.00
   137002          0 "2003/04/01:00:02:14 -0700" -2.00
   137001          0 "2003/04/01:00:02:14 -0700" 1.00
   137002          0 "2003/04/01:00:02:15 -0700" -1.00
[Session 4: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    86335          0 "2003/04/01:00:02:47 -0700" -1.00
[Session 5: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
     1995       6114 "2003/04/01:00:02:54 -0700" -1.00
[Session 6: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    86366       9643 "2003/04/01:00:03:05 -0700" -1.00
[Session 7: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    86366       9643 "2003/04/01:00:03:09 -0700" -1.00
[Session 8: UserID="a3il43i5y168i.ab.hsia.telus.net.22881049148046738", IDFrom=2]
     1013          0 "2003/04/01:00:00:28 -0700" -1.00
   154197       1397 "2003/04/01:00:00:28 -0700" -1.00
[Session 9: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    86366       9643 "2003/04/01:00:03:15 -0700" -1.00
[Session 10: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    87493        466 "2003/04/01:00:03:37 -0700" -1.00
[Session 11: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    95860       1087 "2003/04/01:00:03:37 -0700" -1.00
[Session 12: UserID="lousana.cs.ualberta.ca.311441048195132221", IDFrom=2]
    95861       1488 "2003/04/01:00:03:37 -0700" -1.00
```

Figure 5.1: Abstracted Data from Web Log Files

Hotkeys for almost all possible system operations are designed to facilitate operations for experienced users. During observation, users can click on any nodes to select it as the new root and see the link traversal from the specific node. If a result graph is generated, it can be saved with its important properties, such as the project name, the level depth or the root node of the graph, etc.

- To explain the result graph

To help users explain the visual graph, the system is capable of showing the level information and the page or link information in the status bar, which is located in the bottom of the program window, when the user moves the mouse onto them. The manipulation operation session for the graph is always shown and also recorded when the graph is saved.

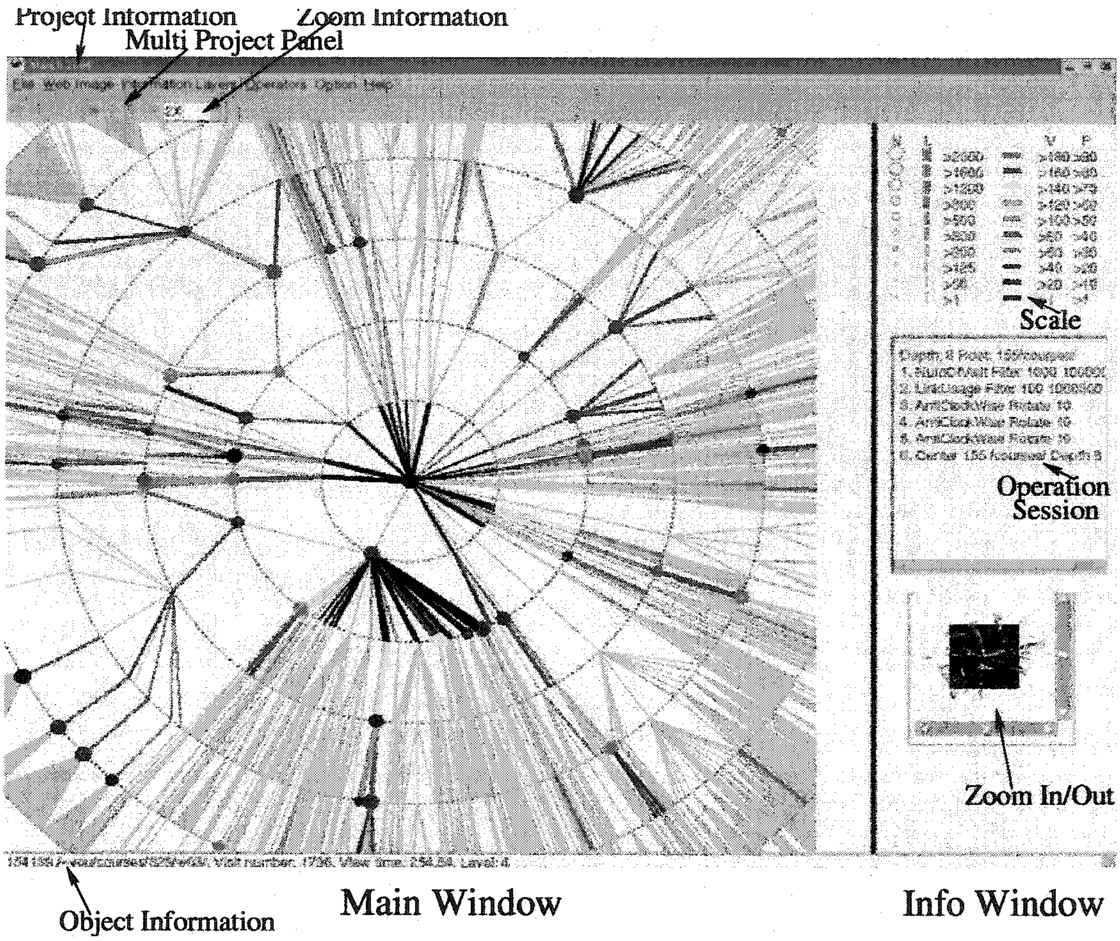
- To show the information clearly or in different perspectives

There are some other system displaying options. “Show Context” indicates the website topology is visualized as background for the graph while ”Show Background” tells the engine only to display the objects that connects to or from the current visualized ones. If none of the two options is selected, the graph is shown without any structural background. Additionally, the decided graph can also be zoomed in and out so that users can check any interesting small or global features they notice.

Figure 5.2 shows what our visualization system, WEBKVDS, looks like. In the left main window, the web graph is visualized. In its top, there is the multi-project switching panel, together with the project and zooming information. The information for the visualized objects is displayed in the bottom if the mouse cursor is moved onto it. In the right info window, the legend and the operation session are always displayed while the zooming context window is visible only when the graph is zoomed in or out.

5.3 Visual Investigation

Generally speaking, the goal of all visualization effects is to better understand user behavior patterns in the web. To achieve that, we have already generated and overlaid



Depth: 8 Root: 155/courses/
1. NumOfVisit Filter: 1000 100000
2. LinkUsage Filter: 100 1000000
3. AntiClockWise Rotate: 10
4. AntiClockWise Rotate: 10
5. AntiClockWise Rotate: 10
6. Center: 155 /courses/ Depth: 8

154 155 /~[illegible]/courses/[illegible], Visit number: 1736, View time: 254.84, Level: 4

Figure 5.2: WEBKVDS System Snapshot

data layers on the web image to get a web graph. In fact, each graph is corresponding to one specific investigation question, e.g, what are the 10 most popular pages in this website or what are these popular pages leading the users to? One objective of visual mining is to generate visual answers to display and explain corresponding usage patterns for these questions, using provided operators. We call these graph-represented investigations *Visual Investigations.* There are two types of visual investigation: *Basic Investigation* and *Time Tube Investigation.* We illustrate each of them in detail in the following.

5.3.1 Basic Investigation

As we mentioned in Chapter 3, there are totally four types of basic information layers, which are generated directly from pre-processed web log files. The *NumOfVisit* layer and *ViewTime* layer show the accessing record of web pages, the *LinkUsage* layer and *ProbUsage* layer demonstrate the navigational statistic of hyper-links. Although one graph with only these layers is ordinary, we can generate graphs for more specific underlying investigations by manipulating the same graph but different layers using provided operators. These investigations, which are generated based on only one original graph, are called *Basic Investigation.* We list some examples in the following. Note that we didn't discover all possibilities that are useful and can be represented by algebra expressions, the following can be just a small part of the whole picture.

Entry Page: From which pages do users enter the website?

The first available basic investigation is "Entry Page", showing pages where visitors typically enter the web site. We emphasize these pages in the web graph by showing their entry importance in the layer represented by the size of the vertex. To achieve that, we use the *MINUS_IN* operator: subtract the sum of the access into the page from any other pages that have a link to it from the number of visit of that page. Therefore, the left value is how often users enter the page directly, e.g., from their favorite folders or by typing the URL in the browser.

The answer shown in Figure 5.3 is capital since if web administrators have this "entry" information, they can redesign the web pages with a high "entry" possibility to introduce recent changes in the website and lead users to where they want users to go. In fact, all web administrators keep doing that for pages that are important from their own points of view, e.g. the site home page, but the "Entry" analysis provides objective evidences that they may need to do that for many other pages as well while the home page is still the biggest entry point in most of the cases.

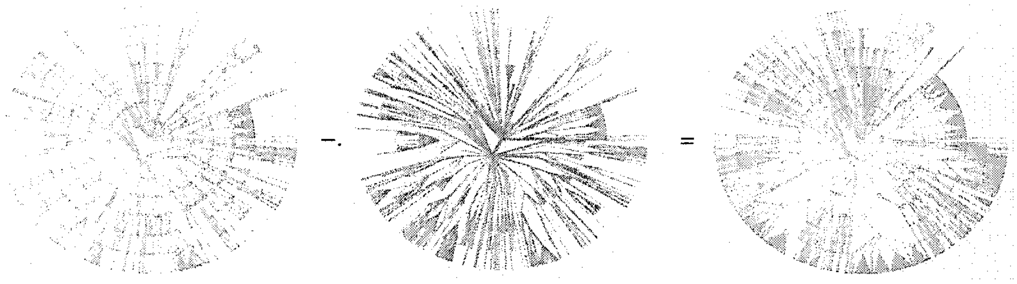


Figure 5.3: Entry Page: $G - .G = G'$

Exit Page: From which pages do users leave the website?

As a companion for the *Entry Page*, *Exit Page*, showing pages where visitors leave the web site, can also be created and represented in the vertex size layer. We use the *MINUS_OUT* operator to subtract the sum of the access from the specific page to other pages from the number of visit of that page. Therefore, the left value is how often a user stops his navigation and leaves the site in that particular page. Figure 5.4 shows exit pages of March, 2003.

The *Exit Page* investigation is also interesting. Why do users always exit in these pages? Are they supposed to exit? For example, if the page contains the last part of a long online article, it is reasonable for people to leave from that point. Otherwise, pages with a "high" exit value might have some designing or content shortcomings, e.g., misleading linking information or boring story-telling tongue, etc., which cause users lose their willing to continue surfing. If a better service is provided there, we may keep the leaving users. Therefore, finding out these pages would be greatly helpful for administrators to understand users' behaviour and improve the site's quality.

Port Page: Which pages begin and end navigation transactions?

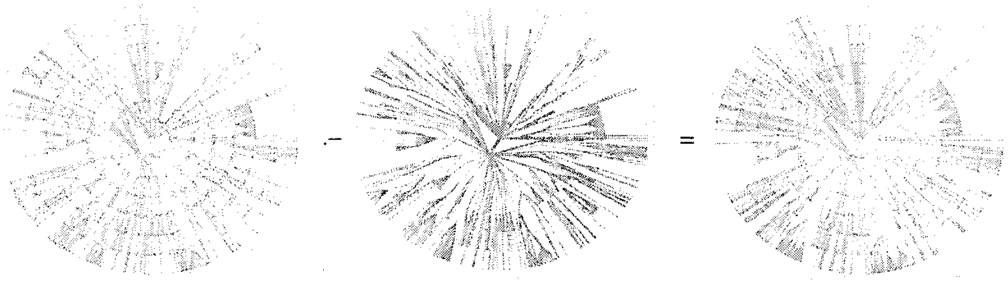


Figure 5.4: Exit Page: $G. - G = G'$

Previous investigation *Entry Page* and *Exit Page* make it possible to generate another advanced user behaviour investigation, *Port Page*. As its name may imply, *Port Page* contains pages that act as ports of the website, i.e, pages that users not only enter but also leave from. For example, the home page of an online book store is usually a port page. After come through it to find some listed books and complete the purchase transaction, customers usually go back, check whether there are some other interesting books to buy and leave if they cannot find any. Therefore, a port page is a great place to put commercial advertisements together with some other information, which a typical entry and exit page should have.

Moreover, the port page investigation can be used to increase the accuracy of the entry page and exit page set. It is quite obvious that a port page is always an entry, therefore, a page that is not a port but an entry can be treated as a "real" entry page, where people visit first not because of the prior structural location, but its attractive information content or excellent link list service. Similarly, "real" exit pages can also be found. Administrators would like to have a look on these pages and find out why users only exit there. This refinement of exit and entry pages can be easily achieved by applying the *EXCEPT* operator.

We generate the web graph for port page investigation by selecting pages that are contained in both entry pages and exit pages using the *COMMON* operator, as Figure 5.5 shows.

Information Page: Which pages are interesting?

Generally speaking, people are surfing all over the WWW looking for information. If web administrators have a clear idea of pages that contain this needed information,

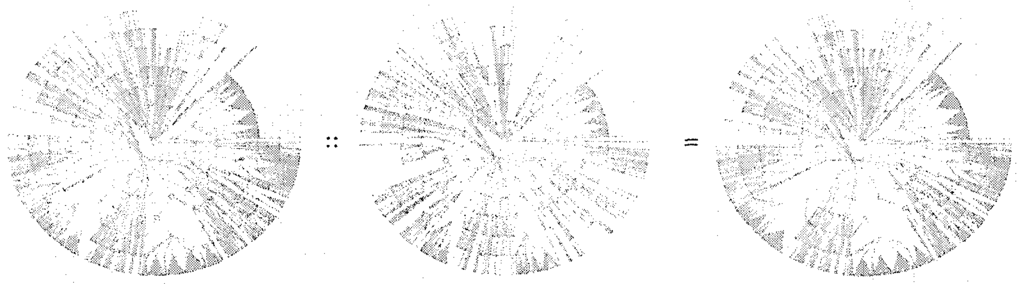


Figure 5.5: Port Page: $G_{Entry} :: G_{Exit} = G_{Port}$

they can facilitate the user searching process by putting links to these pages in entry pages or providing a recommendation list page. Additionally, the content of these information pages indicates the recent popular interests, which can definitely help analyzers understand what users want and which service should be improved. For example, most of the information pages of our department website are assignment or project description pages in March 2003, but in April, they turn to be grade database inquiry and final mark distribution policy pages. Since the majority of the users of our department website are students, we can conclude that students are busy doing their assignments and projects in March, and usually check their marks in April, which is the end month of the winter term.

Web mining researchers have already proposed many methods to identify information pages. However, we use a simple and reasonable way to recognize such pages, based on visualized information layers of usage data and proposed algebra operators. Define that *an information page is a page that has been viewed by lots of users for a long time*, it can be generated by *FILTER* the original graph on *NumofVisit* layer and *ViewTime* layer separately and then *COMMON* the two graphs (Figure 5.6).

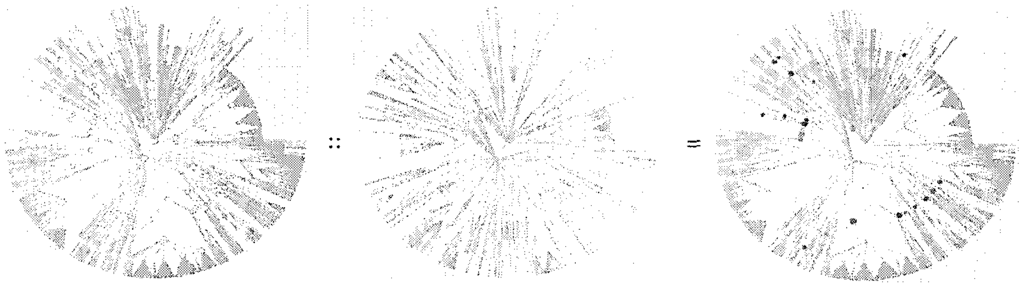


Figure 5.6: Information Page: $(FLT_{Visit,(500,\infty)}(G)) :: (FLT_{Time,(100,\infty)}(G)) = G_{Info}$

Hub Page: Which pages lead to information?

According to the definition in [31], a web page is an *authority* on a topic if it provides good information and is a *hub* if it provides links to good authorities. As shown in the above, the web graph algebra is able to visualize the authority pages in the Information Page Investigation. Then we can generate the hub page visualization based on that using the *CONNECT-TO* operator. Realizing the pages that lead to information would greatly help administrators reorganize and improve the site's informative structure.

Since a hub page is now defined to be *a page that provides links to information pages*, without considering the exact number of providing links, pages that only provide one link (Figure 5.7 (A)) are treated as important as pages that provide many. To make hub pages that provide many links more distinct, we can add an optional parameter to *CONNECT-TO* and *CONNECT-FROM* operators, the *connection number.* If this parameter is given, only pages, whose linking number to visualized pages of the current graph is bigger or equal to the connection number, are retained and visualized (Figure 5.7 (B), *parameter* = 3).

Another possible extension based on investigations of the hub page and information page is the informative topology visualization. We may use the node type (circular or rectangular for example) to indicate whether it is a hub page or an information page and still keep the colour and size for the original layers. The links between the information pages and hub pages are also shown.

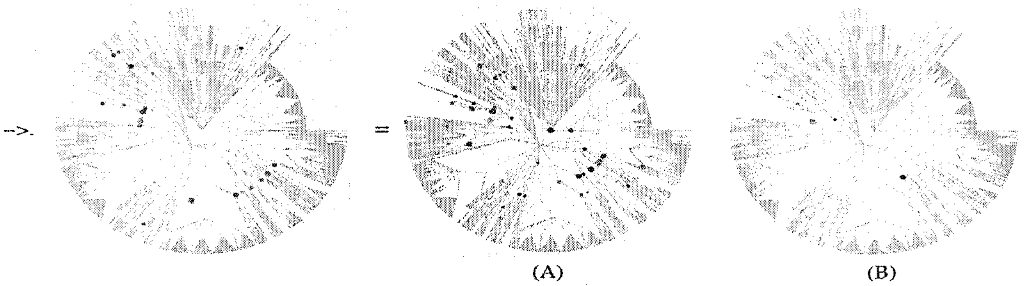


Figure 5.7: Hub Page: $- > .(G_{Info}) = G_{Hub}$

Bridge Page: Which pages are usually clicked through?

Web users may all have the experience of looking for specific information in a website, in which people usually just click through pages until they find the wanted one. Also, in some cases, there is a dominating interesting link in a page so that people who visit that page are all quickly attracted and follow the link. We call these click-through pages, which are highly accessed but hardly get viewed by users, *bridge pages.*

Although researchers usually concentrate on pages where users spend a lot of time viewing, bridge page investigation is interesting and valuable because the visualization not only points out the spots where improvement can be made (or not if the page is supposed to be a link list), but also reveals the navigational behaviour from another aspect. An example of bridge pages is shown in Figure 5.8.

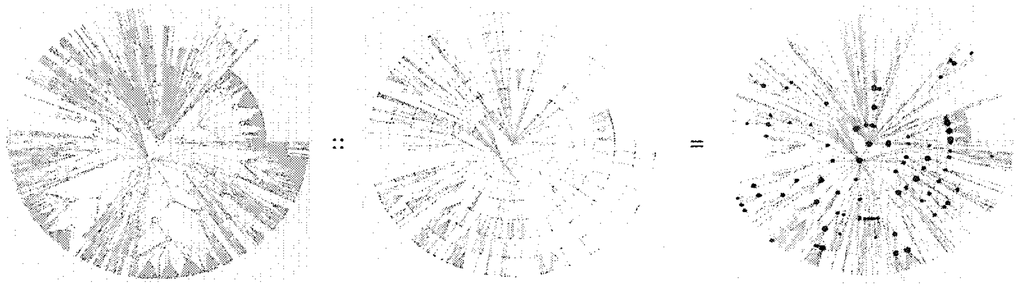


Figure 5.8: Bridge Page: $(FLT_{Visit,(500,\infty)}(G)) :: (FLT_{Time,(1,60)}(G)) = G_{Bridge}$

Other Investigations

We have listed some investigation examples above, showing the navigational behaviour only regarding pages. However, there are still many other possible basic investigations such as:

No usage Pages: Where do users rarely go? The investigation visualizes pages that are not popular in both *NumofVisit* and *ViewTime* layers, using *COMMON* and *FILTER* operators.

Highway Links: Where are links that are highly used? The investigation visualizes links that both have a big clicking number and a high usage probability, using *COMMON* and *FILTER* operators.

No usage Links: Where do users rarely click? The investigation visualizes pages that are not popular in both *LinkUsage* and *ProbUsage* layers, using *COMMON* and *FILTER* operators.

Link Usage: Are the links used as expected? The investigation visualizes the usage of all links in a particular page, using *FOCUS* operator (Figure 4.7).

and so on.

We can generate extended investigations using specific operators based on what we have above. Some of them, the information structure and the "real" entry and exit pages, are already mentioned. They are also classified in the basic investigation category since the operation is still manipulating only one original graph with several layers. We lists some other possibilities in the following:

Select N: What are the most important N objects in one analysis? The investigation selects and visualizes objects that meet the given parameter, using the *SELECT* operator.

Last and Next: Where are the visualized pages from and leading us to? The investigation visualizes pages that connect to or from the visualized objects of the current graph, using the *CONNECT-TO* or *CONNECT-FROM* operators.

Comparison: Comparison can be made between any two investigations which contain at least one same layer, using the *COMMON* or *EXCEPT* operators.

and so on.

Actually, it is impossible to list all the possibility of useful expressions for investigations. The user of the algebra is encouraged to apply operators to any web graphs as long as he finds the result graph is reasonable and meaningful and he should be responsible to explain what he get and why.

5.3.2 Time Tube Investigation

While operating on graphs based on one original usage data set to explain the user behaviour in that particular time slot, we can manipulate graphs from different data sets, separated by time periods, to visualize the usage difference between comparable time slices. These investigations are called *Time Tube[18] Investigation.* We present its four types in the following.

Similarity

The first possibility of the *Time Tube Investigations* is to find the similarity between two graphs, using the *COMMON* operator. For example, it would be interesting to check the pages that are always popular through different times by *COMMON* two information page investigations of two time slices. We can also show pages that were popular but lost their attraction recently by *COMMON* the former information page investigation with the recent no usage page investigation.

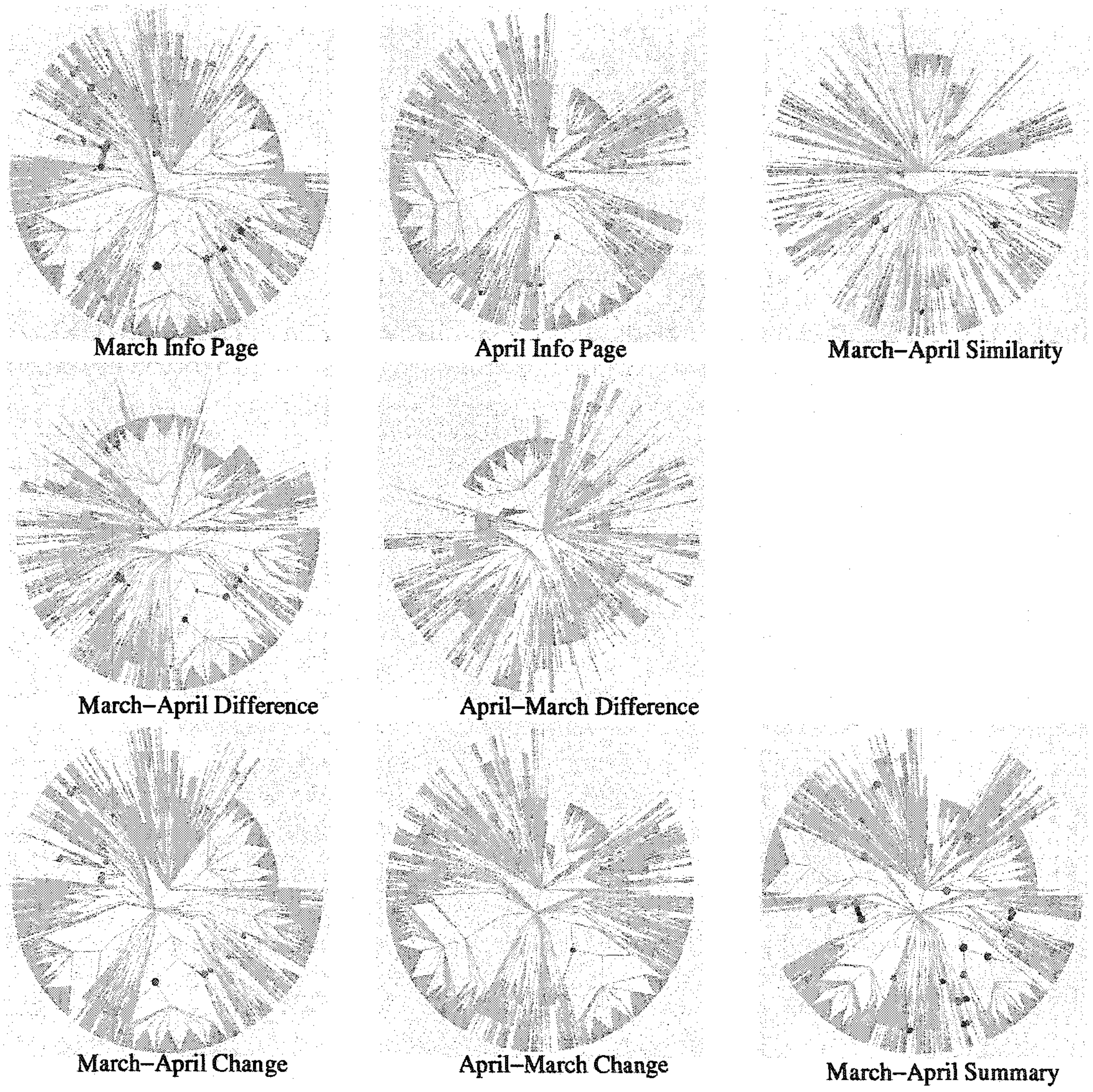
The first row of Figure 5.9 shows the similarity investigation on monthly information pages. The first two graphs are visualizations of information pages of March and April in year 2003. The third graph represents their similarity. Other types of time tube investigation are also shown in the figure.

Difference

It is also possible to show the difference between graphs using the *MINUS* operator. The values of the layer held by visualized objects are subtracted so that the left is the difference. Note that negative layer value usually appears after we do the *MINUS* operation. Although they are kept as ordinary layer value, they are treated as zero in the visualization representation. To show these information, we can do the MINUS operation in the opposite way so that these negative values would become positive.

Change

Thirdly, we can visualize the changing of the usage by the *EXCEPT* operator. Pages, which are not popular previously but really catch people's attentions in the following time slice, represent the new public concerns and focus. Moreover, using the *EXCEPT* operator in the opposite way can find pages that are popular before but fade out people's interests recently.



March Info Page

April Info Page

March–April Similarity

March–April Difference

April–March Difference

March–April Change

April–March Change

March–April Summary

Figure 5.9: Time Tube Investigation

Summary

The final type of *Time Tube Investigation* answers the question about the total usage in one time period, using the *ADD* operator. We add all the values held by visualizing objects, according to existing layers respectively, to make a summary of two operating web graphs and their layers. More importantly, by doing this, we can generate graphs representing bigger time slice based on the default unit. For example, our default time slice is month, but we can also create and manipulate graphs in year after the summary operation.

In this chapter, we first introduce the data we used for the research, then present the implemented system functionalities. Finally, we give several visual web mining examples to show how to use the algebra to discover usage patterns. With no doubt, the algebra is not limited to expressions and investigations that are shown above, but have further possibilities for extensions. In the next chapter, we discuss possible future work of the algebra and conclude this dissertation.

Chapter 6

Summary and Future Work

The previous chapters discussed the web visualization framework and the mining algebra. This chapter summarizes the work that has been done, discusses some possible technical extensions that can be made to improve the performance and broaden the potential usage domain of the visualization algebra concept and ends the dissertation with conclusions.

6.1 Summary

Although web site topology is only an abstract data structure with neither usage records nor any real spatial attributes, it would be smart for web visualization systems to take it into consideration while visualizing the corresponding web usage data. We follow the steps of a few former systems to take the topology as the visualizing background to display the implicit relationship between web usage and structure. In comparison with other existing web data visualization tools and systems, our contributions are:

- Manipulate data based on Visualization.

To the best of our knowledge, none of the existing visualization tools include the possibility of data manipulation and operation based on visualization. However, we believe that data visualization manipulation, i.e, data visual mining, can be and should be a means of data mining. Therefore, in our visualization system, several operators of an algebra are designed for this purpose. One user can select several of them and generate a new visualization based on what he or she has in hand. By providing these operators, the system becomes an ad-hoc and WYSIWYG (What you see is what you get) mining tool for web administrators to produce hundreds of meaningful visual analysis possibilities.

- Propose a visual mining framework

We propose a complete framework for interactive visual web mining in this dissertation. Important concepts such as web image, web graph and information layers are well defined. By these terms, we can easily and accurately describe and visualize the characteristics and status of the usage data and user behaviour within their structure context. The framework is then implemented using the web visualization engine [53] and acts as the theoretical basis of foundation for the WEBKVDS system.

- Information is well categorized.

Although all the user activities are recorded in the web log files, it is usually an information mess. For example, session information, user ID, accessing time, accessing page, etc, all these kinds of information are usually mixed together in one line of ordinary log files or navigational knowledge patterns. This information representing style surely is not the best for visualization and would make it uncertain to answer questions such as: Which kinds of data should be emphasized in visualization for particular intention? (We can not show everything, which would result in showing nothing.) Which kind of data should be kept or thrown away in further data manipulation? The situation would be better for navigation pattern data, but the problem still exists. Fortunately, we are able to generate one layer for each kind of data, based on the proposed information layer conceptualization. Putting these layers on the web topology image much more helps the user to explain the visualizations and understand the information it contains.

- Information layer acts as a middleware between web mining and visualization.

The information layer concept would complete the web information visualization scheme and improve its ability to share knowledge in the process from navigational data to visualizations. Although web mining researchers had already achieved great success in mining technique and pattern collection, the visualization issues are usually ignored. Because it is not practical to show all the information together in one graph and the visualization emphases are distinct for different parts of the discovered knowledge, the performance of the web mining visualization is still unsatisfactory: the graphs are usually hard to understand and the used visualization principles are always the same for diverse patterns or web log files, which inevitably leads to confusion and misinterpretation. As we can see in Figure 6.1, there are two ways to visualize the knowledge. The classical method is a straight-forward process, while the other is separated into two steps: first transform all the knowledge into information layer format and then visualize these information layers. The advantages of the second way are quite obvious. By introducing the information layer concept as interface between the knowledge and its visualization, the formal direct procedure splits into two parts. On one hand, web mining researchers need no more consideration regarding the possible visual effect when they discover and save new patterns. On the other hand, visualization system designers do not have to take various knowledge types and their content into account, but only concentrate on how to better display the information layers, whose format is fixed and easy to understand. In other words, the information layer concept is actually a data and pattern representing standard, whose purpose is to facilitate the knowledge visualization procedure.

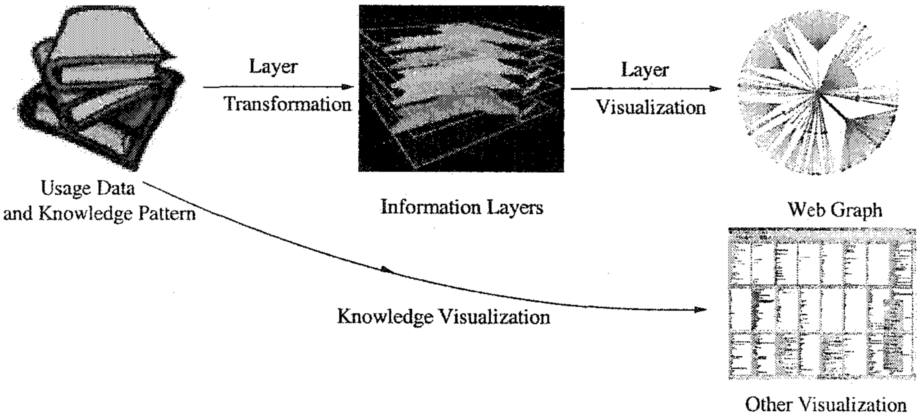


Figure 6.1: Interface between Knowledge and Visualization

6.2 Future Work

Although future work of the web algebra and the related visualization system is not limited in particular areas, some extensions may have high possibility to improve the visual mining framework and the algebra and thus have priority over the others. They are listed in the following.

At first, the current provided operators may not include all the visualization manipulation possibilities. Also, they may not be able to represent all existing investigations and useful analysis. More operators need to be designed and added to the algebra as long as it can help to present more interesting analysis and investigations in the algebra scheme, e.g, structure operators, which "freeze" certain part of the visualization and re-render the rest for more focused comparison, can be interesting.

Secondly, since the Disktree representation of the web topology is vertex and edge based, rather than user or path based, it would be difficult for our visualization to display any user information and their navigational path. However, these kinds of analysis are also critical for web mining. For instance, is the user fumbling for a particular page and unable to find it? Can the clickstreams be mined to cluster user into several behavioral categories, so that page layout can be optimized for each category? These types of investigations do not seem to be natural within the current framework but are usually requested by the web administrators. Therefore, it would be important for the visualization system and algebra to be able to show these kinds of information. In fact, one operator, *FOCUS*, has already been designed to display the traverse of the links on one selected page, which is a typical user based investigation. More user or path based analysis need to be represented appropriately.

Moreover, the fact that our earlier visualization works[50, 61] have used animation techniques to visualize user traversal also reminds us the possibility to incorporate animation ideas into our algebra operator properties. We believe that animation is important visualization technique. While rapid animations in our earlier works visualize usage change at a fast time frame speed, we can sample the timetube to watch things change at a different and much slower time frame.

We have applied operators to find and visualize popularity of the pages, therefore, we can further give explanation of the reasons: why do the pages become popular or lose its attraction? The system can possibly provide suggestions for improvement, e.g, provide an extra link from the home page or move the resource to a different location.

Finally, it is paramount to build a complete family of possible investigations and prove that the algebra is able to capture them all. This work is correlated with adding operators. If we find some interesting investigations but cannot represent it using any of the provided operators, more operators should be designed and added into the algebra to keep its complete investigation coverage. Unfortunately, we know neither the actual set of useful and feasible expressions nor the complete set of possible interesting analysis in present. More work is needed.

Except for these major possible extensions, there are still other improvement possibilities. For instance, the computation orders of the operators are not defined, as well as their property, such as commutativity, associativity and distributivity. These could be interesting extensions leading to the implementation of web graph algebra expression optimizers.

6.3 Conclusion

In this dissertation, a visual web mining framework has been proposed. The web visual mining terminology have been presented and discussed. Then the algebra for interactive web data manipulation is proposed and explained. The algebra is encapsulated in the WEBKVDS system. Several examples of how to use the algebra and the system to mine useful pattern with the help from the system are then illustrated. The whole framework aims at showing how a web site is organized as well as how it is used, and the correlation between them. The goal is to help web mining researchers and WWW information publishers find implicit user navigational patterns, understand their meaning and even generate more interesting patterns using the algebra operators.

The presented study is the first to design an algebra and use operators to manipulate graphs representing web usage data within their web topology context. Although much work is needed to make the young algebra mature, the idea of data visualization operation, i.e, visual web mining, is promising and interesting. We believe that the visual mining algebra is useful and will be able to receive great success in many areas such as the network performance analysis, machining learning algorithm comparison, and many other possible visualization related fields.

Bibliography

[1] http://www.edmonton.ca.

[2] http://www.inxight.com/map/.

[3] K. Andrews. Visualizing cyberspace: Information visualization in the harmony internet browser. In *Proceedings of First IEEE Symposium on Information Visualization.*

[4] E. Z. Ayers and J. T. Stasko. Using graphic history in browsing the world wide web. In *Proceedings of the Fourth International World Wide Web Conference.*

[5] B. Berendt. Web usage mining, site semantics, and the support of navigation. In *R. Kohavi, B. Masand, M. Spiliopoulou, and J. Srivastava, editors, Working Notes of the Workshop Web Minig for E-Commerce - Challenges and Opportunities. 6th ACM SIGKDD International Conference of Knowledge Discovery and Data Mining*, Boston, MA, August 2000.

[6] B. Berendt. Detail and context in web usage mining: Coarsening and visualizing sequences. In *R. Kohavi, B. Masand, M. Spiliopoulou, and J. Srivastava, editors, WEBKDD 2001- Mining Web Log Data Across All Customer Touch Points*, pages 1–24, Springer Verlag, 2002.

[7] J. Berry. Fundamental operations in computer-assisted map analysis. *International Journal of Geographic Information Systems*, 2:119–136, 1987.

[8] J. Berry. Cartographic modeling: The analytic capabilities of gis. In *Geographic Information Systems and Environmental Modeling, Oxford, England: Oxford University Press*, 1993.

[9] J. Borges and M. Levene. Data mining of user navigation patterns. In *WEBKDD*, pages 92–111, 1999.

[10] T. Bruns and M. Egenhofer. User interfaces for map algebra. *Journal of the Urban and Regional Information Systems Association*, 9(1):44–54, 1997.

[11] I. V. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Knowledge Discovery and Data Mining*, pages 280–284, 2000.

[12] D. R. Caldwell. Extending map algebra with flag operators. In *Proceedings of the 5th International Conference on GeoComputation, University of Greenwich, United Kingdom*, 23 - 25 August 2000.

[13] S. Card, G. Robertson, and J. Mackinlay. The information visualizer, an information workspace. In *Proceedings of CHI'91*, pages 181–188, 1991.

[14] S. Chakrabarti, B. E. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the Web's link structure. *Computer*, 32(8):60–67, 1999.

[15] J. Chen, L. Sun, O. Zaïane, and R. Goebel. Visualizing and discovering web navigational patterns. In *Proceedings of Seventh ACM SIGMOD International Workshop on the Web and Databases (WebDB 2004), Paris, France*, June 2004.

[16] E. H. Chi. Improving web usability through visualization. *IEEE Internet Computing*, 6(2):64–71, March/April 2002.

[17] E. H. Chi and S. K. Card. Sensemaking of evolving web sites using visualization spreadsheets. In *Proceedings of the Sypmposium on Information Visualization.*

[18] E. H. Chi, J. Pitkow, J. Mackinlay, P. Pirolli, R. Gossweiler, and S. K. Card. Visualizing the evolution of web ecologies. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'98*, 1998.

[19] E. H. Chi and J. T. Riedl. An operator interaction framework for visualization systems. In *Proceedings of the Symposium on Information Visualization '98.*

[20] A. Cockburn, S. Greenberg, B. McKenzie, M. Jasonsmith, and S. Kaasten. Webview: A graphical aid for revisiting web pages. In *Proceedings of the OZCHI'99 Australian Conference on Human Computer Interaction*, November 1999.

[21] R. Cooley, B. Mobasher, and J. Srivastava. Grouping web page references into transactions for mining world wide web browsing patterns, 1997.

[22] R. Cooley, J. Srivastava, and B. Mobasher. Web mining: Information and pattern discovery on the world wide web. In *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97)*, November 1997.

[23] O. Etzioni. The world-wide web: Quagmire or gold mine? *Communications of the ACM*, 39(11):65–68, 1996.

[24] U. Fayyad, G. Grinstein, and A. Wierse. *Information Visualization in Data Mining and Knowledge Discovery.* Morgan Kaufmann, 2001.

[25] N. Gershon. From perception to visualization. In *Scientific visualization - Advances and Challenges, Academic Press, London*, pages 129–139, 1994.

[26] Hendley, N.S.Drew, A.M.Wood, and R.Beale. Narcissus: Visualizing information. In *Proceedings of IEEE Information Visualization Symposium, Los Alamitos*, pages 90–97, Oct 1995.

[27] Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1), January/March 2000.

[28] R. R. Hightower, L. T. Ring, J. I. Helfman, B. B. Bederson, and J. D. Hollan. Graphical multiscale web histories: A study of padprints. In *Proceedings of ACM Hypertext 98 Conference.*

[29] H. Hofmann, A. Siebes, and A. Wilhelm. Visualizing association rules with interactive mosaic plots. In *Proceedings of the sixth ACM SIGKDD International Conferece on Knowledge Discovery and Data Mining.*

[30] J.Larkin and H. Simon. Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science*, 11(1):65–99, 1987.

[31] J.M.Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of ACM-SIAM Symposium. Discrete Algorithms*, 1998.

[32] R. Kohavi and R. Parekh. Ten supplementary analyses to improve e-commerce web sites. In *Fifth International Workshop on Knowledge Discovery in the Web, WEBKDD'2003, Washington, DC, USA*, pages 29–36, August 2003.

[33] Kosala and Blockeel. Web mining research: A survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery and Data Mining, ACM*, 2, 2000.

[34] J. Lamping, R. Rao, and P. Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 401–408, New York, May 1995.

[35] S. Lawrence and C.Lee.Giles. Accessibility of information on the web. *Nature*, 400:107–109, 1999.

[36] J. Lin, M. W. Newman, J. I. Hong, and J. A. Landay. Denim: Finding a tighter fit between tools and practice for web site design. In *Proceedings of CHI'2000*, April 2000.

[37] B. Liu, W. Hsu, and Y. Ma. Pruning and summarizing the discoverered associations. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining.*

[38] S. K. Madria, S. S. Bhowmick, W. K. Ng, and E.-P. Lim. Research issues in web data mining. In *Data Warehousing and Knowledge Discovery*, pages 303–312, 1999.

[39] H.-P. K. Mihael Ankerst, Martin Ester. Towards an effective cooperation of the user and the computer for classification. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery & Data Mining(KDD)*, pages 179–188, Boston, MA, 2000.

[40] N. Minar and J. Donath. Visualizing the crowds at a web site. In *Proceedings of CHI99.*, 1999.

[41] B. Mobasher, R. Cooley, and J. Srivastava. Creating adaptive web sites through usage-based clustering of ulrs. In *Proceedings of the 1999 IEEE Knowledge and Data Engineering Exchange Workshop KDEX'99*, pages 19–25, 1999.

[42] M.SarKar and M.H.Brown. Graphical fisheye views of graphs. In *Proceeding of ACM SIGCHI.*

[43] S. Mukherjea and J. Foley. Visualizing the world-wide web with the navigational view builder. *Computer Networks and ISDN Systems*, 27(6):1075–1087, 1995.

[44] T. Munzner. H3: Laying out large directed graphs in 3d hyperbolic space. In *Proceedings of the 1997 IEEE Symposium on Information Visualization*, pages 2–10, 1997.

[45] T. Munzner. Drawing large graphs with H3Viewer and Site Manager. In *Graph Drawing*, pages 384–393, 1998.

[46] T. Munzner. *Interactive Visualization of Large Graphs and Networks.* Ph.D Thesis, Stanford University, 2000.

[47] T. Munzner and P. Burchard. Visualizing the structure of the world wide web in 3d hyperbolic space. In *Proceedings of VRML'95, San Diego, California*, pages 33–38, Decemeber 1995.

[48] J. Nielsen. The art of navigating through hypertext. *Communications of the ACM*, 33(3):296–310, 1990.

[49] Y. Niu. *Web Knowledge Information Visualization.* Master Thesis in University of Alberta, 2003.

[50] Y. Niu, T. Zheng, J. Chen, and R. Goebel. Webkiv: Visualizing structure and navigation for web mining applications. In *Proceedings of IEEE WIC, Halifax, Canada*, Oct 13-17 2003.

[51] J. Pitkow and K. Bharat. Webviz: A tool for world wide web access log analysis. In *Proceedings of First International World-Wide Web Conference*, pages 271–277, 1994.

[52] R. Spence. *Information Visualization.* ACM Press, 2001.

[53] L. Sun. *Web Topology and Navigation Visualization.* Master Thesis in University of Alberta, 2004.

[54] C. Tomlin. *Geographic Information Systems and Cartographic Modeling.* Englewood Cliff, NJ: Prentice-Hall, 1990.

[55] C. Tomlin. Map algebra-one perspective. *Landscape and Urban Planning*, 30(1-2):3–12, Oct 1994.

[56] C. Tomlin and J. Berry. A mathematical structure for cartographic modeling in enviromental analysis. In *Proceedings of ACSM*, pages 269–283, 1979.

[57] A. Wexelblat. History-based tools for navigation. In *HICSS*, 1999.

[58] A. Wexelblat and P. Maes. Visualizing histories for web browsing. In *RIAO'97:Computer-Assisted Information Retrieval on the Internet,Montreal*, 1997.

[59] A. Wexelblat and P. Maes. Footprints: History-rich tools for information foraging. In *CHI*, pages 270–277, 1999.

[60] A. Youssefi, D. Duke, M. Zaki, and E. Glinert. Toward visual web mining. In *Proceeding of Visual Data Mining at IEEE Intl Conference on Data Mining (ICDM), Florida*, 2003.

[61] T. Zheng, Y. Niu, and R. Goebel. Webframe: In pursuit of computationally and cognitively efficient web mining. In *PAKDD*, pages 264–275, 2002.