Visualizing and Discovering Web Navigational Patterns

Jiyang Chen, Lisheng Sun, Osmar R. Zaïane, Randy Goebel
Department of Computing Science, University of Alberta
Edmonton, Alberta, Canada

{jiyang, lisheng, zaiane, goebel}@cs.ualberta.ca

ABSTRACT

Web site structures are complex to analyze. Cross-referencing the web structure with navigational behaviour adds to the complexity of the analysis. However, this convoluted analysis is necessary to discover useful patterns and understand the navigational behaviour of web site visitors, whether to improve web site structures, provide intelligent on-line tools or offer support to human decision makers. Moreover, interactive investigation of web access logs is often desired since it allows ad hoc discovery and examination of patterns not a priori known. Various visualization tools have been provided for this task but they often lack the functionality to conveniently generate new patterns. In this paper we propose a visualization tool to visualize web graphs, representations of web structure overlaid with information and pattern tiers. We also propose a web graph algebra to manipulate and combine web graphs and their layers in order to discover new patterns in an ad hoc manner.

Keywords

Web Visualization, Web Graph Algebra, Interactive Web Mining, Visual Data Mining

1. INTRODUCTION

Web information visualization is of interest to two major groups of people[8]. On one hand, web users and information seekers need information visualization tools to facilitate their browsing experience. On the other hand, web site administrators and information publishers need visualization tools to assist them in their maintenance, publishing or analysis work. While web users are assisted by many navigation recommendation systems and other tools for information retrieval, web site administrators and decision makers are still lacking the needed effective visualization tools to help them discover and explain the interesting navigational patterns within their web site usage data. Website administrators are concerned with a site's effectiveness, asking questions such as "Are people entering and visiting 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.

Visualization tools for web site structure, web site usage data and navigational patterns are paramount because to better understand events on a site or consequences of changes and design on a web site, it is important to have a big picture of the state of the site and the activities on the web site. Moreover, while there are myriad algorithms and web mining tools to discover useful patterns from within web usage logs, the discovered patterns are so numerous and complex that it is difficult to analyze them and understand them without their web site context. A visualization tool to represent the discovered patterns and overlap them over a comprehensive web site structure depiction is capital for the evaluation and interpretation of web mining results. A depiction of the facts with cues such as colour, thickness and shape of vertexes and edges is easier to understand and use to spot anomalies or interesting phenomenon than a long list of patterns, rules and numbers provided by data mining algorithms. Furthermore, visualization in its own is not enough. Interactive visualization can help evaluate data mining results more effectively. We believe that visualization can and should be even a means for data mining. Visual Data mining has already been used in particular for classification [1]. We consider here the visualization of cues representing web information, and the means to manipulate and operate on these visual cues to discover useful knowledge in an ad hoc manner. We call this process *visual web mining*.

In this paper, we describe a visual data mining system which uses a new algebra to operate on web graph objects to highlight interesting characteristics of web data, and consequently help discover new patterns and interesting hidden facts in web navigational data. Our system, *Web Knowledge Visualization and Discovery System*(WEBKVDS), is composed of two main parts: *FootPath* and *Web Graph Algebra*. FootPath renders a 2D representation of a web structure overlaid with tiers of various data, called a web Graph. The algebra operates on graphs and their layers to generate other graphs highlighting relevant information.

The contributions of the paper are as follows. First, we propose the concept of Web Graph, a multi-tier object comprised of layers of data or patterns overlaid over a web structure representation. We adopt the idea of a disk-tree for the web structure representation [4]. However, we propose

Seventh International Workshop on the Web and Databases (WebDB 2004), June 17-18, 2004, Paris, France.

a different approach to transform the web topology into a 2D tree. Moreover, the depiction is not static but rather dynamically adapts to the representation of the layers of data and parameter settings (i.e. thickness and number of links). Finally, we propose an algebra to operate on a series of web graphs for the purpose of ad hoc visual web mining. Contrary to most of the previous visualization systems that usually only visualize the web data, the system we present is unique in that several operations are designed to mine the rendered visualizations to achieve more valuable patterns.

The remainder of the paper is organized as follows. We review related web visualization work in Section 2. Section 3 introduces our WEBKVDS system and identifies the different objects we intend to manipulate and operate on. FootPath, which renders web graphs is detailed in Section 4. The Web Graph Algebra is introduced in Section 5. Finally we summarize our contributions in Section 6.

2. RELATED WORK

Most of the existing tools for web visualizations concentrate on representing the relationship among the web usage data, users' behaviour and the web structure. Among them, DiskTree [3, 4], designed by Ed Chi et al., was the first to display the web usage and structure information together by mapping the usage data on the structural objects. The graph representing the web site structure is collapsed into a "disc" using a breadth-first search algorithm. If a page is linked from many other pages, only the first link found with the breadth-first search is kept and represented. This technique has been used to visualize web site evolution, web usage trends over time, and evaluation of information foraging. The concept of Time Tube Visualization, which is a series of visualizations constructed at different time periods but aligned together for comparison, was proposed in [4].

Minar's Crowds Dynamics [5] is an animation visualization system, which treats a web site as a social, active space and illustrate how the web users navigate the web. The dynamic animation is simple but information-rich since it acts like a documentary movie for real-time user behaviour. However, the system doesn't scale well for large web sites. First, the web site structure is hand made, the system does include the function to generate the web site map automatically. More importantly, rather than showing what web users have done, the system visualizes what web users are doing, making any analysis less revealing.

Although web visualization is a complex task, researchers can begin to understand its characteristics by decomposing the task into the three dimension of scale, first proposed in [7], of web structure visualization, representation of aggregate and individual navigation usage behaviour, and the comparative display of navigation improvement methods. The authors also present *Web Knowledge and Information Visualization* (WEBKIV), a system that combines the strategies from several other web visualization tools, as part of the WebFrame project [11], which presents a initial development of a framework for gathering, analyzing, and re-deploying web data. However, WEBKIV lacks the functionality to operate on the graphs, presents a static graph, and uses the same web topology rendering algorithm as DiskTree. Our system is different on these three accounts.

A recent preliminary proposal uses 3D representations rather than a 2D layout techniques to visualize the web structure and map it with web usage data [10]. 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.

Although none of the existing visualization tools presented above include the possibility of data manipulation and operation based on visualization, they do provide a universal background of where the research sits, and motivates our design of visual mining in web domains. To the best of our knowledge, there is no published work describing any kind of operators to manipulate graphs representing web structure or web usage data. The only relevant work worth mentioning is map-algebra [9]. It pertains to geographical information systems and provides operators to combine information layered on geographical maps. Our web algebra and the details of the layered web graphs we manipulate are described in the following sections.

3. VISUALIZATION & DISCOVERY

The Web Knowledge Visualization and Discovery System (WEBKVDS) is mainly composed of two parts:
1- FootPath: for visualizing the web structure with the different data and pattern layers.
2- Web Graph Algebra: for manipulating and operating on the web graph objects for visual data mining.

In this section, we first present our framework with the details of the web graph objects and terminology we use, then discuss the architecture of our visualization and discovery system. The rendering engine, FootPath, and the algebra are presented in subsequent sections.

In addition to visualizing data from web access logs and visualizing patterns derived from web mining processes, the main goal steering the design of our system is to provide means to interactively play with visualization objects in order to perform ad hoc visual data mining and interpretation of the discoveries. To achieve this, our framework encompasses the definition of: 1. objects that include navigational data and patterns with their context, 2. tools to render these objects, and 3. ways to interact with these objects and their visualization. The navigational data are statistics extracted from pre-processed web logs, navigational patterns are patterns discovered with web mining techniques while the context of these is simply the structure of the web site.

There are tools to visualize discovered web patterns such as visitation click-stream sequences [2] or other regularities. However, patterns are visualized outside their context - i.e, the web site where they occur. Our goal is to render discovered patterns overlaid on top of the web structure visualization (i.e. in context).

The idea of visualizing web usage data on relevant web struc-

ture was proposed before to help people better understand the statistics of web accesses [3, 4, 7, 10]. However, the visualization is static and the proposed tools do not provide the visual interaction that leads to new discoveries, which is known as visual data mining.

To overcome the above challenges, we introduce a visualization object that can both be rendered, and manipulated for interpretation and application. This object, which we call a Web Graph, assembles together data and knowledge about navigational behaviour in their web structure context.

3.1 Visualization Objects

The object that we use both for visualizing the web information and for expressing web mining operations is a web graph. It combines all the necessary data regarding the web structure, the usage data and whatever patterns available. A web graph is also a multi-tier object. The first tier, called a web image, is a tree representing the structure of a given web site (or subset of it). Each other tier, called an information layer, represents some statistics about web pages or the links connecting them. Layers can also contain patterns about the pages and links. Combining the tiers allows putting the navigational data in its web structural context. Each layer is identified separately and when the object is displayed, layers can be inhibited or rendered. The tree representing the web structure is always displayed as a background of the information layers, allowing the localization of any information vis-à-vis the web site. Here we enumerate and summarize the terms and concepts we use:

Web Image: Also referred to as *bare graph*, is a tree representing the structure of a web site or a subset of it. It has a root, which is a starting page, and a certain given depth. Each node in the tree represents a web page and an edge represents a hyperlink between pages. A web site is in reality a graph, but it is collapsed into a tree as explained in Section 4. When visualized, the tree is displayed as a disc with the centre being the root, and the nodes of each level displayed on a circular perimeter, each level successively away from the centre. Figure 1 illustrates such discs. Note that all graphs illustrated in this paper are only 2 levels deep, but could be arbitrarily deeper.

Information Layers: Similar to the map layers geographers manipulate in GIS, we represent various web usage information as layers. An information layer is a logical collection of web data abstractions that can be laid over the web image. We identify here four different layers. These basic layers are prescribed by the identified useful expressions of the Web Graph algebra we will present later. However, other information layers are also possible. The four current basic layers are: (1) *NumofVisit* layer giving the visit statistics per page; (2) *LinkUsage* layer showing the usage statistics of links; (3) *ViewTime* layer representing the average access time per page; and (4) *ProbUsage* layer giving the access probability of the links. Visualizing these information layers is achieved using visual cues such as colour, size and shape of nodes, as well as colour and thickness of links. Figure 1 illustrates the visualization of these layers overlaid on the web image. Note that combining different visual cues, different information layers can be shown together.

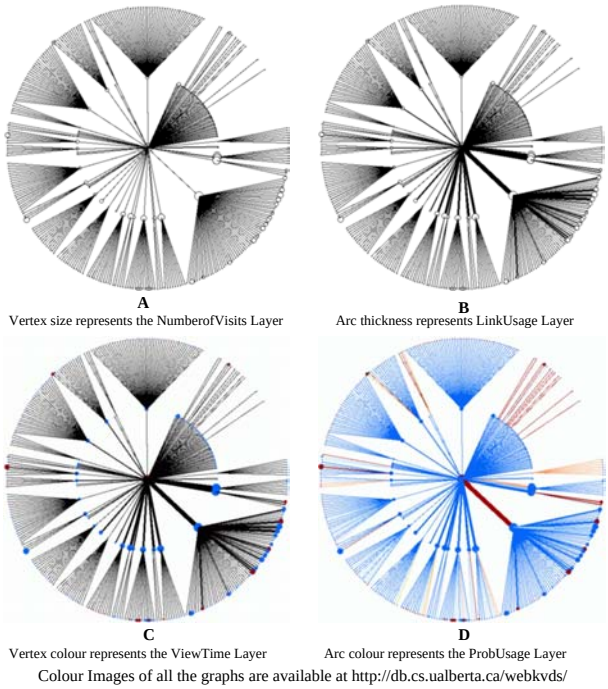

Figure 1: Visualization of Multiple Layers

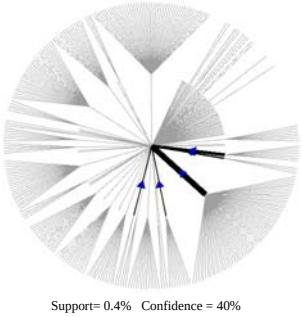

Figure 2: Association Rule Layer

Pattern layers: While information layers compile statistical data collected after cleaning and pre-processing the web access log, pattern layers hold patterns discovered using data mining algorithms such as sequence analysis, clustering, and association mining, etc. When displayed, they play the role of visualization for evaluation of patterns in the knowledge discovery process. Currently we have identified three such layers: *Association Rules*, *Page Clustering*, and *Page Classification*. Figure 2 shows a visualization of the Association Rule layer. The thickness represents the support of a rule, the arrow head size represents the confidence of the rule, while the colour can represent any defined interestingness measure. For clustering and classification, the class labels are identified by the colour of nodes in the graph.

Web Graph: A web image with one or more information or pattern layers constitutes a web graph.

3.2 System Architecture

Figure 3 illustrates the architecture of our system WEBKVDS. Given a web site and its web access log, a web image is generated using the structure of the site and the pre-processed web log. The data obtained after pre-processing the web access log is also used both to generate information layers and as input to some data mining modules. The patterns discovered by these data mining modules are used to generate pattern layers. The different layers and the web images are used to create web graphs, which are displayed using a rendering engine: FootPath. The web graph algebra allows the creation of new graphs, again displayed with FootPath. The combination of the algebra expressions and the visualization with FootPath constitutes the basis of our *Visual Web Mining* system.

4. FOOTPATH

FootPath is the rendering engine of our visualization and discovery system. A web graph is displayed by first rendering the web image and then attributing visual characteristics to nodes and edges such as colour, thickness etc., to represent data from information layers. For efficiently visualizing the web topology in a web image, we adopted the Disktree representation proposed in [4], in which a node indicates a page and an edge represents the hyperlink that connects two pages; the root, a node located in the center of the graph, is usually the home page of a site or a given starting page for the analysis at hand. Nodes on the same perimeter are from the same level with regard to the root. The consecutive levels form a disc on top of which statistics and patterns are displayed. However, the novelty of FootPath is two folds, first we adopt a new strategy for collapsing the web topology into a tree structure exploiting usage data in addition to the structure of the site; second, we deal with the problem of occlusion, due to the additional visual characteristics representing information layers, by using a dynamic layout that redistributes the nodes of each level along the 360 degree perimeter.

4.1 Web Image rendering

The Disktree algorithm, and many other other algorithms representing web topology in a tree structure, use a simple Breadth First Search (BFS) method to convert a connected graph into a tree. The idea is that for each node, only one link coming to it is represented. The BFS strategy is simply to represent the link that first leads to the node following the scan of the web topology. The popularity of this approach is due to its simplicity. Depth First Search (DFS) is another alternative. However, a study comparing both the BFS and the DFS strategies for a tree representation of web topology indicates BFS as a better choice in terms of ballance of the tree[6]. One disadvantage of BFS is the sheer number of nodes in the first levels of the disc making it difficult to overlay additional information without occluding or compressing the edges and nodes in the graph.

To better visualize the website's structure, we introduce a new method, a *Usage-Based method*. The idea is simple and is integrated into the BFS. Since, in order to collapse a connected graph into a tree only one incoming link is kept per page, we chose to keep the link with the highest usage count in the web access log while avoiding cycles in the graph, and avoiding disconnecting the tree. This strategy leads to deeper trees but less nodes per level. This in turn gives a more "aerated" web image (i.e. sparse) without missing any page. Notice that no page is lost from the web graph, but like with any other strategy, some links in the site topology are not represented.

4.2 Dynamic Layout

The more links are used or pages are visited, the thicker or bigger they appear in our web graph visualization, hence the name *FootPath*. In addition to the size of nodes and the thickness of edges, Footpath also uses colours and shapes of nodes as well as colours of edges to represent other information. It is the user that associates these visual cues to the different available information layers. The user, at display time, also chooses some parameters to indicate ranges inside which nodes or links are interesting. Other nodes and edges are eliminated from the visualization. To avoid occlusion because of various thicknesses and sizes, and to take advantage of space made available by dropping irrelevant edges and nodes identified by the algebra operators, we provide a dynamic layout. This method uses more efficiently the disc area by dividing the 360 degrees of the out most perimeter by the remaining number of nodes on the last level weighted by their respective edge thickness or node diameter. Parents, on the next level, are positioned in the middle of their children, and the same is done for each upper level up to the root. This process for distributing the nodes and edges on the disc is redone whenever necessary during the interactive visualization.

5. WEB GRAPH ALGEBRA

We propose an algebra, Web Graph Algebra, to manipulate and produce web graphs. A combination of unary and binary operators can form expressions and equations. Variables in our algebra are web graphs. Unary operators extract sub-parts of the underlying web structure or given information layers. Binary operators combine two web graphs by evaluating arithmetic expressions on the common information layers as well as the nodes and edges constituting the web images. 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 tool.

The idea of the algebra is similar to the *Map Algebra*[9], a general set of conventions, capabilities, and techniques that have been widely adopted for use with geographic information systems (GIS). Analogically, while the GIS researchers operate the geographic layers on a map, we intend to do web usage mining and data operations on a web graph. In the following, because of lack of space, we enumerate only one unary operator as example, and some of the proposed binary operators of the algebra for web graphs.

Operator FILTER: $\theta = FLT_{Layer,threshold}(\alpha)$
The unary operator FILTER selects from α the objects (i.e. nodes and edges), whose layer contents fits within the given threshold parameters. The FILTER operator not only is used to refine the graph, but also acts as a basic step for many other operations and further analysis manipulations.

Operator ADD: $\theta = \alpha + \beta$
The binary operator ADD selects the objects that exist in both α and β, and transposes them into θ with the sum

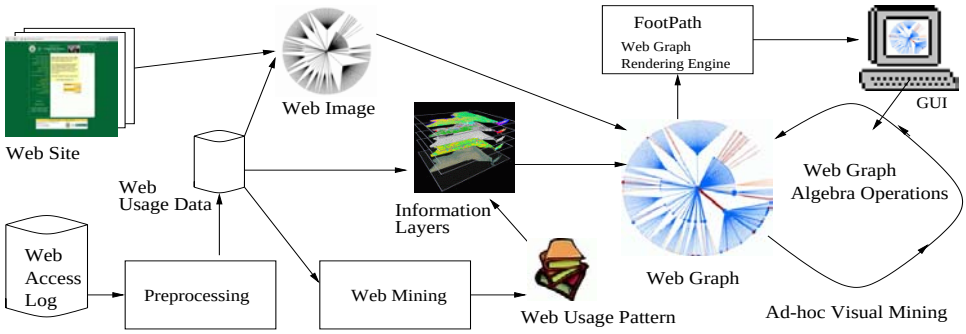

Figure 3: Visualization System Architecture

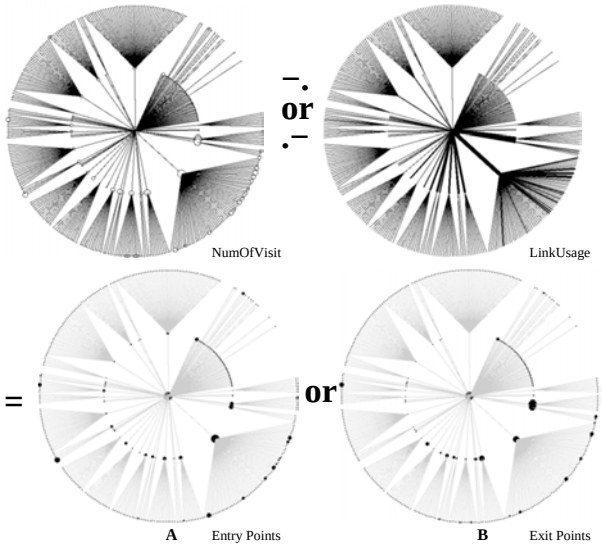

Figure 5: The MINUS_IN & OUT Operators (Finding Entry and Exit points)

of their respective content from the available layers in both graphs. The sums are done at each individual information layer. If an object exists only in one graph, it is kept with no change and transposed into θ. This operator makes the summary analysis possible. For example, to see a visualization of two consecutive months' data together, simply adding the graphs of the two months achieves that. Figure 4 shows the ADD operation for web graphs of two months with 2 layers, NumOfVisit and LinkUsage layers. The result represents an aggregation over a time tube as defined in [4].

Operator MINUS: $\theta = \alpha - \beta$
The binary operator MINUS also acts on individual information layers. The operator MINUS selects the objects that exist in both α and β, and transfers them to θ with the content being the difference of their content in α and β. If an object only exists in one graph, it is kept with positive values if it is from α and negative values if it is from β. The MINUS operator is mostly used in a Time Tube[4] to compare the graphs of different time periods.

Operator COMMON: $\theta = \alpha :: \beta$
Operator COMMON selects the objects that exist in both α and β, and transposes them into θ with the minimum content value for those layers that belong to both graphs. For example, a node with the layers $NumOfVisit$ and $ViewTime$ values respectively 100 and 50 in α, and 80 and 30 in β will be transposed in θ with $NumOfVisit$ layer value 80 and $ViewTime$ layer value of 30.

The COMMON operator is similar to the notion of intersection and can help achieve many interesting analysis. For example, to find in a web site the set of pages considered content pages, we can combine the information about the viewing time and the visit statistics. This is performed by filtering the graph to get only those nodes that are highly visited (based on a threshold) and filtering the same graph to get those nodes that are viewed for a long time, and find the commonality between these two resulting graphs.

$$Graph = FLT_{NumOfVisit,t_n}(G) :: FLT_{ViewTime,t_v}(G)$$

Operator MINUS_IN: $\theta = \alpha - .\beta$ and
Operator MINUS_OUT $\theta = \alpha. - \beta$
The two similar operators subtract values across different information layers. Both variables , α and β, have to have the information layers *NumOfVisit* and *LinkUsage*. These operators are basically similar to the COMMON operator except that the *NumOfVisit* layer of the resulting graph θ is calculated using both layers, *NumOfVisit* and *LinkUsage*, the following way:

For MINUS_IN, the *NumOfVisit* value of a node N in θ is calculated by subtracting the sum of the values of all links pointing **to** N from the layer *LinkUsage* of β from the *NumOfVisit* of N in α.

For MINUS_OUT, the *NumOfVisit* value of a node N in θ is calculated by subtracting the sum of the values of all links pointing **from** N from the layer *LinkUsage* of β from the *NumOfVisit* of N in α.

These operators can generate some interesting analysis, such as "Entry Points", pages where visitors typically enter the web site represented in the web graph, and "Exit Points", pages where visitors leave the web site. For Web Entry Analysis, we use MINUS_IN operator, subtract the sum of the access to 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. For Web

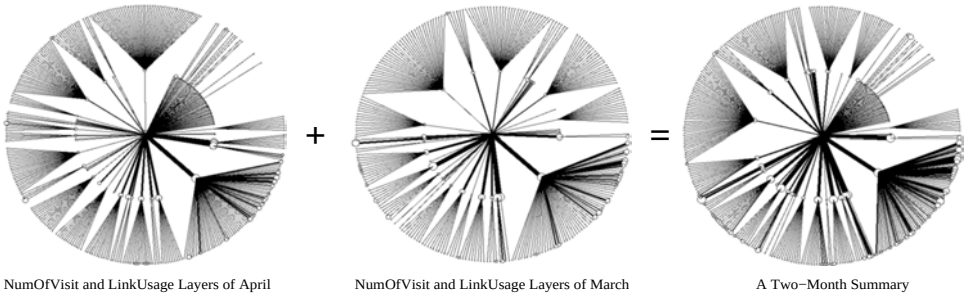

Figure 4: The ADD Operator (Cumulative navigational behaviour)

Exit Analysis, we use MINUS_OUT operator, subtract the sum of the access to any other pages from the specific page from the number of visit of that page. Therefore, the left value is how often a user stops the navigation and leaves the site in that page. Figure 5 shows the described examples for Entry Points and Exit Points.

Operator EXCEPT: $\theta = \alpha \ni \beta$
Operator EXCEPT selects the objects that exist in α but not in β, and transposes them into θ with no changes in the layers. 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 for a given month. 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 have just become "hot" recently.

These operators look simple but are in fact powerful. We have yet to discover all the possible expressions and useful equations in this algebra. Based on web log data visualizations, we can interactively generate many other explainable and information-rich visualizations.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced the concepts of web image and web graph and proposed an algebra for web graphs. We presented the idea of layering data and patterns in distinct layers on top of a disktree representation of the structure of the web, allowing the display of information in context which is more suited for the interpretation of discovered patterns. Our web knowledge visualization and discovery system visualizes multi-tier web graphs, and with the help of the web graph algebra, provides a powerful means for interactive visual web mining

The work we presented in this paper is still preliminary. We have implemented a prototype. However, more theoretical work is needed for the algebra to mature. The few operators we presented are powerful and useful for the web mining and analysis we targeted, but more operators could be defined. Moreover, we have yet to study interesting properties such as commutativity, associativity, or distributivity of operators if coefficients are introduced later in the algebra. Also the order of operations in the algebra is not defined. This could be an interesting investigation leading to the implementation of web graph algebra expression optimizers.

Acknowledgments: This research is supported by the Natural Sciences and Engineering Research Council of Canada, and by the Alberta Ingenuity Centre for Machine Learning. We would like to thank Tong Zheng for providing the data used as layers for the web graphs visualized in this paper.

7. REFERENCES

[1] M. Ankerst, M. Ester, and H.-P. Kriegel. Towards an effective cooperation of the user and the computer for classification. In *ACM SIGKDD Conference*, pages 179–188, 2000.

[2] B. Berendt. Understanding web usage at different levels of abstraction: coarsening and visualising sequences. In *ACM SIGKDD - WEBKDD workshop*, San Francisco, USA, August 2001.

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

[4] 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.

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

[6] T. Munzner and P. Burchard. Visualizing the structure of the world wide web in 3rd hyperbolic space. In *ACM VRML Conf.*, pages 33–38, 1995.

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

[8] R. Spence. *Information Visualization*. Addison-Wesley (ACM Press), 2000.

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

[10] 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.

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