Visualizing Multi-Attribute Web Transactions Using A Freeze Technique

Ming C. Hao, Daniel Cotting, Umeshwar Dayal,
Vijay Machiraju, Pankaj Garg
Intelligent Enterprise Technologies Laboratory
HP Laboratories Palo Alto
HPL-2003-35
February 20th , 2003*

E-mail: (ming_hao, umeshwar_dayal, vijay_machiraju)@hp.com, dcotting@student.ethz.ch, garg@zeesource.net

visualization, multi-attribute, web transactions, physics-based, classification

Web transactions are multidimensional and have a number of attributes: client, URL, response times, and numbers of messages. One of the key questions is how to simultaneously lay out in a graph the multiple relationships, such as the relationships between the web client response times and URLs in a web access application. In this paper, we describe a *freeze* technique to enhance a physics-based visualization system for web transactions. The idea is to freeze one set of objects before laying out the next set of objects during the construction of the graph. As a result, we substantially reduce the force computation time. This technique consists of three steps: automated *classification*, a *freeze operation*, and *a graph layout*. These three steps are iterated until the final graph is generated. This iterated-freeze technique has been prototyped in several e-service applications at Hewlett Packard Laboratories. It has been used to visually analyze large volumes of service and sales transactions at online web sites.

* Internal Accession Date Only Approved for External Publication
Published in and presented at Visualization and Data Analysis 2003, 20-24 January 2003, Santa Clara, California

Visualizing Multi-Attribute Web Transactions Using A Freeze Technique

Ming C. Hao, Daniel Cotting*, Umeshwar Dayal, Vijay Machiraju, Pankaj Garg*
Hewlett Packard Research Laboratories, Palo Alto, CA
(ming_hao, umeshwar_dayal, vijay_machiraju)@hp.com

ABSTRACT

Web transactions are multidimensional and have a number of attributes: client, URL, response times, and numbers of messages. One of the key questions is how to simultaneously lay out in a graph the multiple relationships, such as the relationships between the web client response times and URLs in a web access application. In this paper, we describe a *freeze* technique to enhance a physics-based visualization system for web transactions. The idea is to freeze one set of objects before laying out the next set of objects during the construction of the graph. As a result, we substantially reduce the force computation time. This technique consists of three steps: automated *classification*, a *freeze operation*, and *a graph layout*. These three steps are iterated until the final graph is generated. This iterated-freeze technique has been prototyped in several e-service applications at Hewlett Packard Laboratories. It has been used to visually analyze large volumes of service and sales transactions at online web sites.

Keywords: Visualization, multi-attribute, web transactions, physics-based, classification, freeze, graph layout, force computation.

1. INTRODUCTION

A transaction starts with a user (client) clicking on a web page (URL). The web browser sends a request (transaction) through applications servers to perform some service, for example, a user clicks on a web page to purchase a book. Recently, the rapid increase of transactions on the Internet has led to the availability of large volumes of transaction data. Business research efforts have focused on how to turn this raw data into valuable information that can be visually analyzed.

Web transactions are multidimensional. For example, clients, URLs, response times, and numbers of messages are the attributes for a web transaction. Product type, country name, price, and quantity are the attributes for a finance revenue transaction. Based on our practical experience, we have discovered that using one single type of relationship for all data items is too general. The relationships can vary significantly depending on the selected attributes. One of the key questions is how to simultaneously lay out in a graph the multiple relationships, such as the relationships between the web client response times and URLs in a web access application, and the relationships between product types and countries in a finance revenue analysis (Market Basket Analysis). Each pair of relationships plays an important role for the overall visual analysis.

To date, many practical applications have shown that the physics-based visualization technique is one of the best methods for web browsing and information retrieval systems [1, 3, 7, 8]. To place every data object in a graph, usually all object pairs must be traversed. As a result, the usefulness of this technique is limited if the volume of data objects grows large. The computation time seriously impacts the performance of a visualization system.

* Correspondence: Email: dcotting@student.ethz.ch; garg@zeesource.net

2. OUR APPROACH

When using the physics-based techniques, we encounter the following problems:

1. The types of relationships among data objects can vary significantly depending on different data attributes.
2. The current physics-based technique is limited to a single relationship and cannot support data with multiple relationships.
3. The data objects placement requires the calculation of the velocity and positions for each data item and cannot scale up with large volumes of data objects.

To solve these above problems, we derive an *iterated-freeze* technique to layout a graph with types of relationships. This technique consists of three steps: *automated classification*, a *freeze operation*, and *a graph layout*. These three components (steps) are iterated until the final graph is generated.

The idea is to freeze a set of objects during the construction of the graph. The relationships among the data objects are classified into primary and secondary classes based on (1) data attributes, (2) relationship types, and (3) the user's selection. For different domains, various criteria can be employed to select which class of data objects to freeze and which class of data objects should participate during the process of the graph layout.

The order of the iteration is as follows:

1. Freeze the primary data objects.
2. Optimize and layout the primary relationships using unfrozen data objects.
3. Freeze the primary data objects.
4. Optimize and layout data objects from the secondary relationships with respect to the primary relationship using data objects from the secondary class.

This approach excludes the frozen data objects from participating in the force velocity computation and position movements. As a result, this elimination shortens the graph layout time by at least two orders of magnitude, i.e. graph layout time drops from 80 sec to 2 sec.

Figure 1 illustrates the multi-attribute freeze overall process flow running on a visual data mining platform-VisMine [6]. Each of the above components is described further in the following sections.

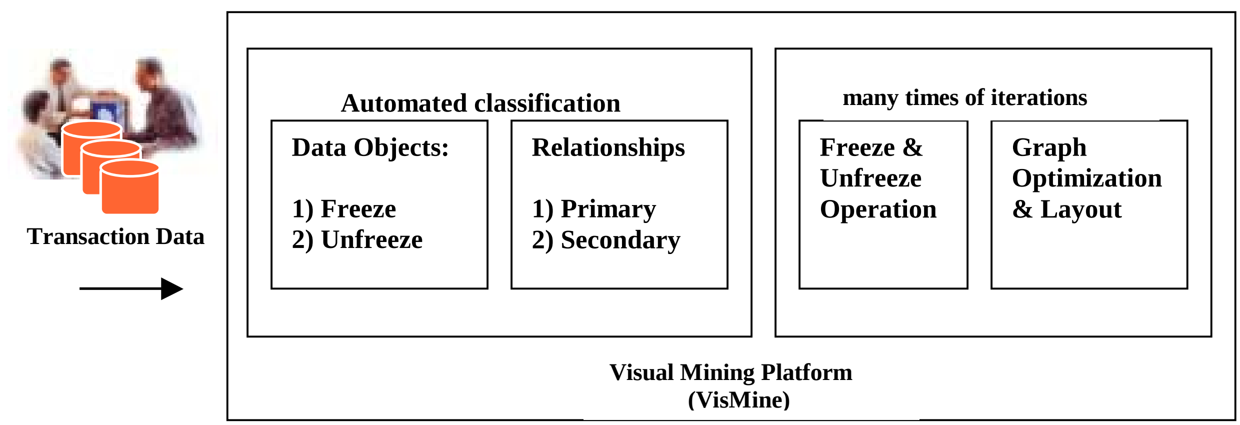

Figure 1: Our Approach

2.1 Automated Classification

A classifier automatically separates relationships among data objects into primary and secondary classes based on data attributes. It determines which class of data objects to freeze and which class of data objects will participate in

the current graph layout. There is only one primary relationship in a graph and many secondary relationships with respect to the primary relationship. For example, in a web access application, the relationship between a pair of client response times is the primary class relationship; the relationship between the client and an accessed URL is the secondary class relationship.
The automated classification processes are as follows:

a) Classifies data objects into classes based on their attributes, relationships, and user-defined features.
   For example, as illustrated in Figure 3 in a web access application, the web transaction data objects are classified into two classes of clients and URLs.
b) Orders the sequence of the relationships as the primary and the secondary relationships.
   There is only one primary relationship class in a graph, but there are many secondary relationship classes with respect to the primary relationship. From the example above, the relationships between web clients are in the primary class. The relationships between clients and the URLs are in the secondary class.
c) Transforms and maps each class's data objects to a different appearance, such as spheres representing clients and cubes representing URLs.
d) Assigns primary class relationship.
   In the web access application, clients can be laid out according to their similarities of median response times.
e) Assign the secondary class data objects and their relationships.
   In the web access application, the relationship is defined as flows:
   Relationship = 1 if the client accesses the URL
   Relationship = 0 if the client does not access the URL
   The URLs are arranged close to the clients that access the corresponding URLs.

2.2 A Freeze Operation

This component constructs a 3-dimensional array (x-axis, y-axis, z-axis) to record the position of the frozen data objects. As illustrated in Figure 2, the frozen data objects will not be moved during the layout of non-frozen data objects. The order of freezing operation depends on the sequence of the relationships. To layout the primary class relationships, the freeze operation needs to freeze the data objects in the secondary class before placing the primary class relationships. To layout the secondary class relationship, the data objects in the primary class need to be frozen so that they do not participate in the graph generation.

A) Before Freeze

The three objects pointed to by arrows are fixed before layout

B) After Freeze

The three data objects pointed to by arrows are not moved after running the layout

Figure 2: The Freeze Operation

2.3 Graph Optimization and Layout

This component applies the "physics-based mass-spring" algorithms [2] through relaxation (many iterations). In order to compute the final layout for visualization, the total force (E) needs to be minimized. Our goal is to find a configuration of particle positions such that a local minimal of energy level is reached.

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{\substack{j=1 \\ j \neq i}}^{n} \epsilon\left(\left|X_i - X_j\right|, s_{ij}\right)$$

Where data objects $i \in \{1, \ldots, n\}$, and the position corresponding to item i in three-dimensional space is denoted as X_i. X represents all the points in a matrix. The distance between two points X_i and X_j is $|X_i - X_j|$. The relationship between two data objects is denoted as $S_{ij.}$

The minimization step can be achieved by any numerical minimization method, such as simulating the scene, using leapfrog [4] or Conjugate Gradients [5].

The above computation applies only on the non-frozen objects under certain boundary conditions. There are no force computations and position movements over time on frozen objects. As a result, this elimination shortens the graph layout time by at least two orders of magnitude.

3. APPLICATIONS

This iterated-freeze technique has been prototyped in several e-service applications at Hewlett Packard Laboratories. It has been used to visually analyze large volumes of service and sales transactions at online web sites. Various interaction capabilities, such as layered drill-down, fade in/out, linked views, and dynamic search are provided to make it easy for analysts to navigate through their data.

3.1 Web Access Analysis

The freeze technique has been applied to web access patterns and behaviors. Figure 3 contains 400,000 transaction records with 986 clients and over 2,000 URLs. The cubes represent clients that make transactions on the web. The spheres represent URLs. Clients are arranged in such a way that clients with similar behavior with respect to average medium response times are arranged close together, dividing clients into those with low-, medium-, and high-response times. The URLs are then arranged close to the clients that access them.

.

There are two types of similarities described as follows.

1. Similarity Between Clients

   The client data objects contain a client name, a median response time and the accessed URLs. The similarity value is based on the difference in median response times between a pair of clients. The reciprocal value of the difference is scaled by half the average difference value between zero and one on a logarithmic scale. After the first minimization, the clients will be positioned in an almost linear layout from low response times (colored green) to high response times (colored burgundy).

2. Similarity Between URLs and Clients

   For the client-URL pairs the similarity is set to 0 if the client does not access the URL; else the similarity is set to 1. (The similarity between URLs is set to 0 to indicate there are no relationships defined between them). After running the second minimization, The URLs will be positioned around the clients that access them.

Figure 3A, 3B, and 3C illustrate the construction of a web access graph. Figure 3D illustrates a user searching for a specific client “15.4.90.244” (red cube) and its linked URLs (spheres) from the graph.

3A: Initialize graph:

Classify data objects into two classes; all the clients (cubes, primary class) and URLs (spheres, secondary class) are placed on a spherical surface.

3B: Freeze URLs--layout clients and client relationships:

Employ force-based algorithms to layout clients in such a way that clients with similar response times are arranged close together (automatically classify groups as low, medium, high).

3C: Freeze clients - layout URLs and client relationships:

URLs are placed close to the clients to show the client access patterns.

3D: Navigation:

Interactivity is an important aspect of a web-based visualization system. To make large volumes of web transactions easy to explore, users are allowed to navigate through the graph. A dynamic search capability is provided for discovering relationships among certain clients and URLs.

3.2 Sales Transaction Analysis

The second example (Figure 4) shown is the Hewlett Packard sales transaction analysis application. It shows a sales history of 40,752 transactions. Sales managers want to use this sales history to discover the relationships between products and the relationships between the products and the countries.

The iterated-freeze technique has been used experimentally to visually analyze both relationships as shown in Figure 4A and 4B:

4A: Product and Product Relationships:

Products (spheres) are laid out based on how often they are bought together at the same time (indicated by solid colored lines between pairs of spheres).

4B: Country and Product Relationships:

Countries (cubs) are laid out with respect to products (spheres) sold.

Related products or countries are linked with colored dashed lines (Color represents quantity).

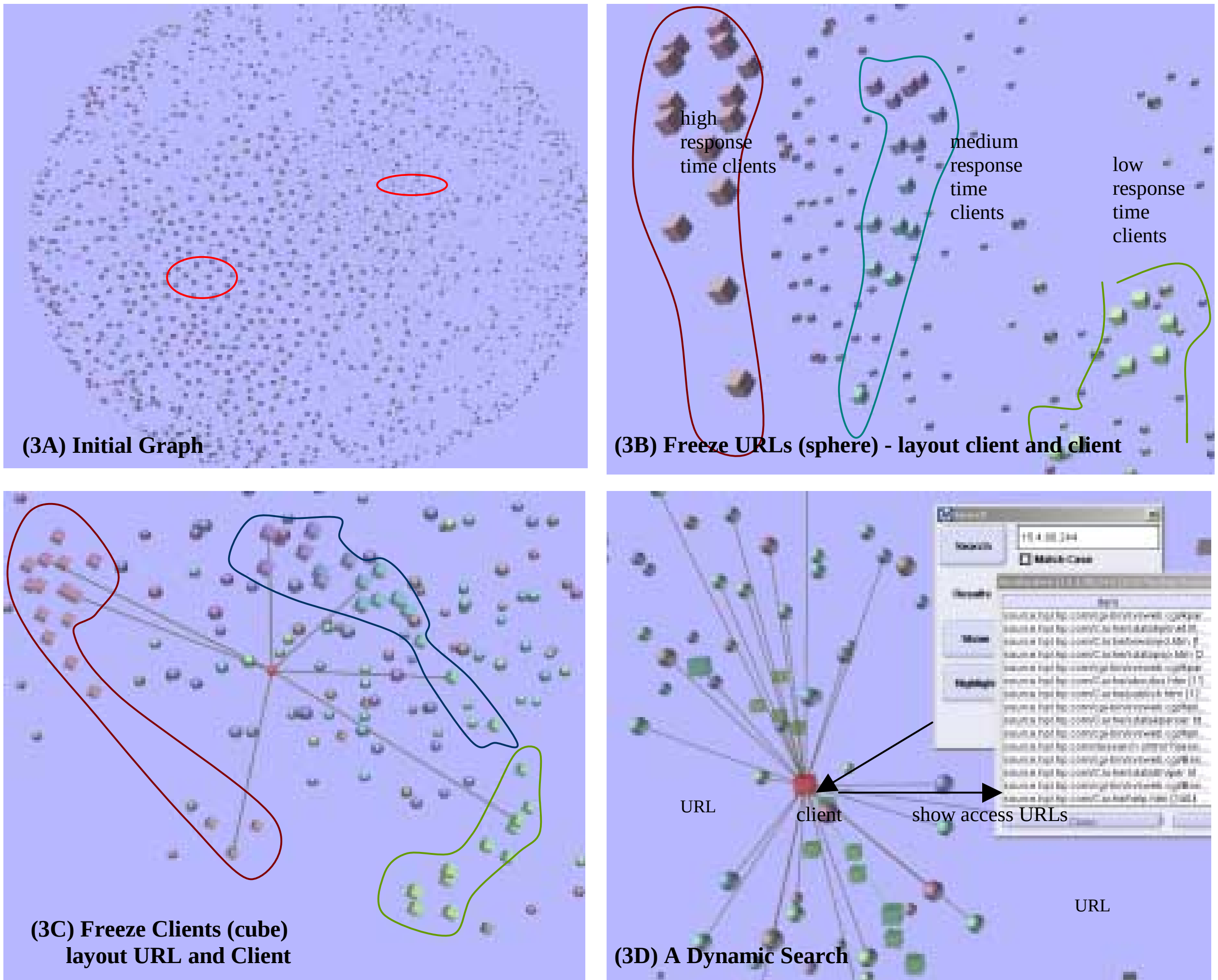

Figure 3: Web Access Analysis

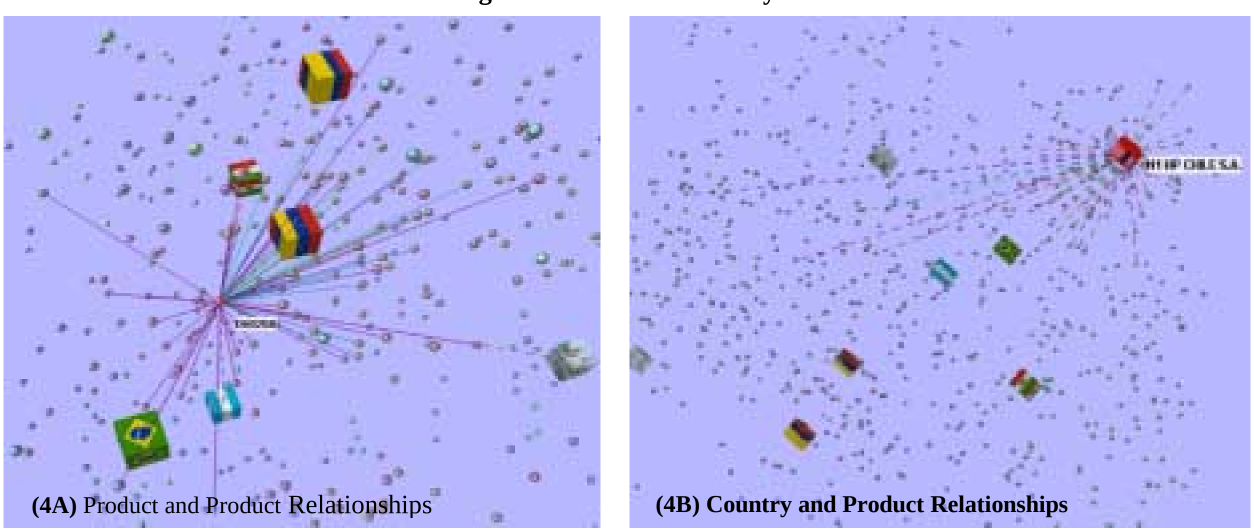

Figure 4: Sales Transaction Analysis

After applying this visualization technique, the analysts are able to understand their product sales. For example, they were able to find out which countries buy which products and which product sales are the largest in which countries. Also, they discovered product combinations such as the fact that printer, paper, ink, black print head, and cartridge tape appear together frequently on the same invoice.

4. CONCLUSIONS

To address the requirements of multiple relationship graph visualization, this paper presents a new iterative freezing technique enabling analysts to quickly visualize large multi-attribute web transactions and to understand service patterns and correlations. Our future work will be extended to animation and to watching the web service or sales changes over time.

5. ACKNOWLEDGEMENTS

Thanks to Stephany Kalil and Peter Wright for providing sales transaction data and for many technical discussions and suggestions.

6. REFERENCES

[1] M. Kreuseler, N. Lopez, and H. Schumann: A Scalable Framework for Information Visualization. Proc. Information Visualization 2000, Salt Lake City, UT.
[2] T. C. Sprenger, R. Brunella, and M. H. Gross. H-BLOB: A Hierarchical Visual Clustering Method Using Implicit Surfaces. IEEE/VIS2000, Salt Lake City, UT, 2000.
[3] R. J. Hendley, N. S. Drew, A. M. Wood, and R. Beale. Narcissus: Visualizing Information. Information Visualization. Using Vision to Think, 1999, pp. 503-509.
[4] I. Bruss and A. Frick: Fast Interactive 3-D Graph Visualization. Proceedings of Graph Drawing 95, Spring Verlag, LNCS 1027, pp. #99-110
[5] J. R. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Carnegie Mellon University, Pittsburgh, PA, 1994.
[6] M. Hao, U. Dayal, M. Hsu, B. D'eletto, and J. Becker. A Java-based Visual Mining Infrastructure and Applications. Proceedings of IEEE InfoVis99, San Francisco, CA., 1999.
[7] A. Morrison, G. Ross and M. Chalmers. A Hybrid Layout Algorithm for Sub-Quadratic Multidimensional Scaling. Proceedings of IEEE Information Visualization 2002, Boston, MA.
[8] M. Chalmers. A Linear Iteration Time Layout Algorithm for Visualising High-Dimensional Data. Proc IEEE Visualization ’96, San Francisco, CA.