Accelerated Force Computation for Physics-Based Information Visualization

Ming C. Hao, Umeshwar Dayal, Daniel Cotting*, Thomas Holenstein*, Markus Gross*

Hewlett Packard Research Laboratories, Palo Alto, CA
(ming_hao, usmeshwar_dayal)@hp.com

Abstract

Visualization of similarity is an emerging technique for analyzing relation-based data sets. A common way of computing the respective layouts in an information space is to employ a physics-based mass-spring system. Force computation, however, is costly and of order N^2. In this paper, we propose a new acceleration method to adopt a well-known optimized force-computation algorithm which drastically reduces the computation time to the order of N log N. The basic idea is to derive a two-pass, "prediction and correction" procedure including a customized potential function. We have applied this method to two different applications: web access and sales analysis. Both demonstrate the efficiency and versatility of the presented method.

1. Introduction

Many techniques try to visualize similarities between related data objects by arranging them in a multidimensional space [1]. By displaying data objects in this manner, it is possible to visualize the relationships between the objects while preserving the essential information in the data. A common way of computing the respective layouts is to employ physics-based mass-spring systems. The goal of these methods is to map similar objects to spatial positions close to each other, while unrelated objects are placed far away from each other. The data placement requires the computation of the forces applied to every object.

Physics-based techniques have been widely used in many different applications. They include visualization [3] for web browsing, visual clustering methods [4,5], and web-based information retrieval systems [6]. Besides the mass-spring paradigm there is also a plenitude of other layout methods. Eick [7], for instance, uses algorithms that position nodes based on weighted links and employ hierarchy and filtering to unclutter the display. Kreuseler [14] describes an enhanced spring model by assigning a data object not only to points but also to small, general shapes to prevent objects from collapsing into each other.

The usefulness of physics-based systems is limited if the volume of the number of objects grows large, as physics-based visualization techniques require computing the forces for every possible pairing of data objects. It is well known that this results in a computational complexity of O (N^2).

As a core limitation, such physics-based methods scale poorly with the number of data objects. As a further limitation, penetration, collision and cluttering of individual data objects may occur during visualization.

Straightforward applications of acceleration methods used in particle physics, such as Barnes-Hut [2] and fast multipole expansions [10] (FME), are not possible, since these methods impose restrictions onto the employed potentials. Specifically the Barnes-Hut method demands particle forces which decrease monotonically as a function of the spatial distance.

In this paper, we overcome the aforementioned limitations of physics-based layout techniques by deriving a novel method for accelerated force computation based on energy minimization. Our contributions are twofold. First, by refining the forces involved into the simulation and by minimizing potentials, we gain much better control over penetration and collision. Second, by introducing a prediction-correction scheme, we extend the conventional Barnes-Hut algorithm to make it work for non-monotonic potentials, such as the ones in information visualization.

Our paper is organized as follows. First, we derive a potential function to provide an alternative to direct force computation. Next, using an error-correction method, we extend the Barnes-Hut force-computation technique. As a result, the computation can drastically be reduced from $O(N^2)$ to O(N log N + # corrections). After briefly explaining the Visual Similarity System (VISIM) [9], which implements

*Presently with Department of Computer Science, Swiss Federal Institute of Technology, Zurich, Switzerland
dcotting@student.ethz.ch; holenst@inf.ethz.ch; grossm@inf.ethz.ch;

this method, we describe some applications that demonstrate the performance and versatility of our approach.

2. Energy Minimization and Customized Potential Functions

Physics-based computations can either be carried out directly on the particle forces (strong form) or minimize the system's energy (weak form) making use of potentials.

In our setting, the potential eventually defines how the data objects are placed at the end of the minimization process. The user can change the current potential during runtime as needed. Thus, we can customize a potential function that avoids collisions and overlap of data objects. We can also customize a potential function that improves the layout of unrelated objects.

2.1 Force And Potential

The potential function is generally a function both of the similarity and of the distance of two objects:

$$e = e(r, s)$$

First, we assume a configuration in 3-dimensional space by placing the objects at different locations. Then, we add up the total energy/potential of all objects:

$$E = \sum e(r_{ij}, s_{ij})$$

Where $i \in \{1,\ldots, n\}$ are items in a three-dimensional space, E is the total potential, r_{ij} is the distance between two items from item i to item j, and s_{ij} is their similarity. This means that the total potential E is the sum of the individual energies associated with a pair of items. An optimal configuration is the one in which the total energy is minimal. Force and energy are closely related to each other when the force is the negative gradient of the energy.

The main advantage of using a potential function is that it gives us a much finer control over the particle behavior and thus allows us to overcome the limitations described above. As explained in the next section, customized potential functions can reverse force directions when objects get too close to each other and thus prevent collisions and penetration.

2.2 Customized Potential Functions for Information Visualization

From many experiments in Hewlett Packard Laboratories, we have derived a customized potential function to avoid collisions of very similar objects. We have applied this function to many different physics-based applications, including IT user search behavior analysis, web access response time analysis, and sales revenue analysis. We found that the following potential function [8] achieves the best results:

$$e(r, s) = \frac{a}{r} + bsr^2 + cr$$

Here, a, b, and c are fixed numbers which can be used to calibrate the potential. The first term a/r is used to prevent two objects from penetrating each other, even in cases of large similarity values. The second term bsr^2 controls the attraction of objects as a function of the similarity. This ensures that two objects, which are more similar, end up at a smaller distance than unrelated objects. The third term cr, usually with a very small c, makes sure that objects do not drift to infinity, as this would lead to a poor visual layout.

A picture of the potential for different similarities can be seen in Figure 1.

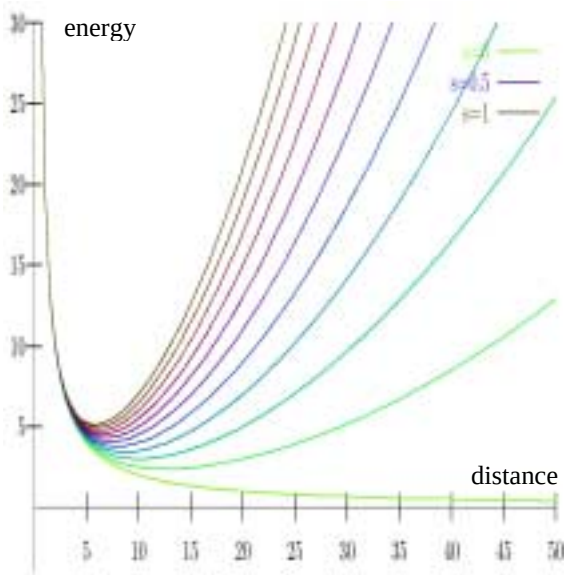


Figure 1. Customized potentials for different similarities. On the horizontal axis, the distance between two objects is depicted. The different functions describe the energy for different similarities.

3. Accelerated Force Computation

To calculate the force for every data object, usually all object pairs must be traversed. This is computationally very expensive resulting in a slow

runtime performance of the visualization system. In astrophysics, various object methods exist to accelerate the computation including Barnes-Hut [2] and fast multipole expansions (FME) [10]. The complexity of the forces as a function of the data relations, however, prohibits a straightforward application of these methods in applications using similarity.

3.1 Concept

Specifically, the aforementioned algorithms approximate the force of a group of objects by replacing the group by some heavier "super object". However, the forces in the visualization system do not depend only on the object distance r, but also heavily on the data similarity s. Hence, it seems to be impossible to build a single data structure which can be used to calculate forces, as the methods from astrophysics usually do. The problem is illustrated in Figure 2. In this example, most object pairs have similarity 0 and only a few pairs have similarity 1. These are the pairs (A, B), (C, D) and (D, E). When calculating the force for object B, the conventional Barnes-Hut algorithm assembles objects C and D to a single "superobject", as they are far away from B and close to each other. When computing the force for object E one would want to apply the same optimization. Unfortunately, forces strongly depend on object similarities. That is, the force from pair (C, D) onto B can be completely different from the force from (C, D) onto E. Therefore, straightforward application of algorithms like Barnes-Hut or FME is not possible as these algorithms critically depend on forces being only a function of the object distance. A detailed discussion of conventional particle simulation algorithms is beyond the scope of this article and we refer to textbook literature [15,16] for further information.

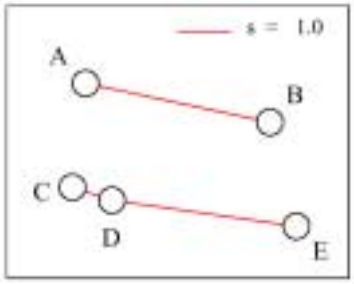


Figure 2: The particle force is calculated. Here, the force on the two objects depends on the similarity between the objects. The similarity of most pairs is 0 (which implies the unified force), but some pairs have similarity 1 (pairs (A, B), (C, D), and (D, E)).

We propose to solve this problem using a two-pass method consisting of a *prediction step* and a *correction step* as described in the following sections.

3.2 A Prediction Step

In the prediction step, all potential functions are replaced by functions that depend purely on the distance. This enables us to use any of the conventionally optimized, hierarchical force-computation algorithms, including Barnes-Hut or FME.

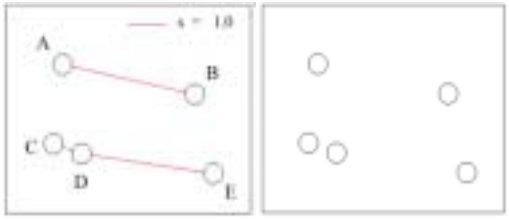
To speed up the computation, we start by setting the similarity of all the object pairs to 0 in the prediction step. Hence, we only need to process the links with non-zero similarities in the correction step.

Our observations have shown that most of the forces are based on distance. We can calculate these forces on each object by assuming that all the forces follow the same direction. Doing so, the force can be calculated using the Barnes-Hut or any other well-known algorithms.

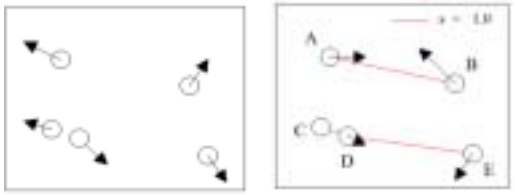
3.3 A Correction Step

In the correction step, we adjust for the simplifying assumption above and update the forces for all links having substantially different behaviors. In the prediction step, for each of those links the previously computed force of both objects is subtracted. In a correction step we add the real force by explicit force computation. Because this step involves only $K << \frac{1}{2}N(N-1)$ edges (links) between pairs of objects with non-zero similarity, we only add a complexity of O(K) to the overall computational effort. Thus, the total complexity of the algorithm is $O(N \log N + K)$. In our application, the user can control the quality of the approximation by changing the edge parameter K, such as setting edges with low similarity values to similarity 0. Here K determines the amount by which forces need to be adjusted in the correction set. That is, by changing K we set links with similar forces to a lump force. Furthermore, we can adjust the fixed accuracy parameter ϑ of the Barnes-Hut algorithm [2] to give us total control over the algorithm and to adjust the tradeoff between accuracy and speed.

The accelerate control algorithm is illustrated in Figure 3. Figure 3(a) shows a set of objects for which we want to calculate the forces. The lines symbolize links with similarity 1.0, but the aforementioned algorithm could handle any similarity other than 0. In Figure 3(b) we illustrate the prediction step. By setting the similarities of all links to 0 we make the situation amenable to the astrophysical object process. After applying the object algorithm for the force computation as in Figure 3(c), we correct the errors by reintroducing the initial similarities and eventually obtain the final force field as depicted in Figure 3(d).



(a) Compute the forces (b) Set similarities to 0



(c) Compute the force prediction using conventional Barnes-Hut

(d) Correct the force prediction with similarities

Figure 3: A new accelerated potential function with prediction and correction.

3.4 Minimization

In order to compute the final layout for visualization, the total energy needs to be minimized. Our goal is to find a configuration of particle positions such that

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

Where items $i \in \{1,\ldots, n\}$, and the position corresponding to item i in 3-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 $|X_i - X_j|$ denoted by r_{ij}.

The minimization step can be achieved by using either any numerical minimization method, such as steepest descent [11], or by simulating the scene, using leapfrog [12,13], Conjugate Gradients, and Runge Kutta.

4. The Visual Similarity System (VISIM)

To analyze large volumes of transaction data with multiple attributes, the methods described in this paper have been integrated into a data analysis visualization system, VISIM. The system uses a web browser with a Java activator to allow real-time interactive visual data mining on the web.

4.1 System Architecture and Components

The VISIM system connects to a data warehouse server and uses the database to query for detailed data as needed. The data structure is kept in memory to support real-time manipulation and correlation. As illustrated in Figure 4, the visual clustering system architecture contains three basic components.

4.1.1 Similarity

The relationship between data objects is characterized according to their "similarities". Similarity varies according to the characteristics of the applications. For example, in a product sales application, the similarity is defined as how often these products are bought together.

4.1.2 Force Computation

First, force computation uses potentials to find a placement such that similar objects are close and dissimilar objects are far. Second, VISIM applies the Barnes-Hut algorithm to calculate the force. Third, VISIM corrects the forces for item pairs with non-zero similarity values. Last, VISIM performs relaxation to minimize the total energy.

4.1.3 Interactive Navigation

This system provides for navigation of relationships and patterns.

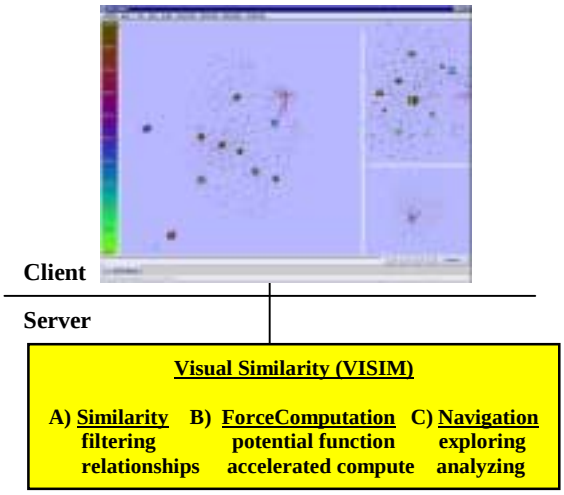


Figure 4: System architecture and components.

4.2 User Interactions

Interactivity is an important aspect of a web-based visualization system. To make large volumes of web

transactions easy to explore and to allow users to navigate the graph and to discover patterns and trends, VISIM provides some of the following interaction capabilities, such as clustering, searching and freezing, as illustrated in Figure 5A, 5B, and 5C.

(1) Clustering Capability
The clustering capability enables users to simplify complex layouts by grouping spatially adjacent objects. Users are allowed to click on a cluster and drill down for detail information.

A scene clustered with BLOBS is shown as follows:

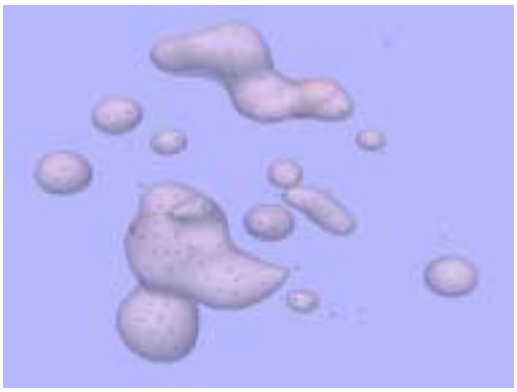


Figure 5A: User Interaction – Clustering

(2) Freezing Capability
The freezing sets the objects to stay at the same position during the minimization (object movement) step. This capability is required to construct a special user-controlled layout.

Figure 5B: User Interactions–Freezing

(3) Searching Capability
Search capability allows the user to locate an object in the scene and to draw the paths to the related objects.

A search dialog is used to find an object in the scene.

The found object – printer will be highlighted. All connected lines will be drawn as follows.

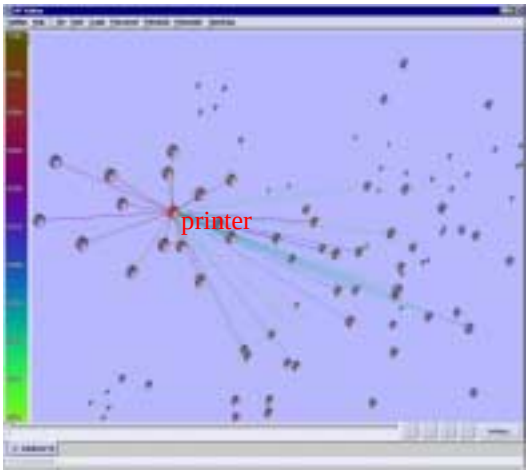


Figure 5C: User Interactions–Searching

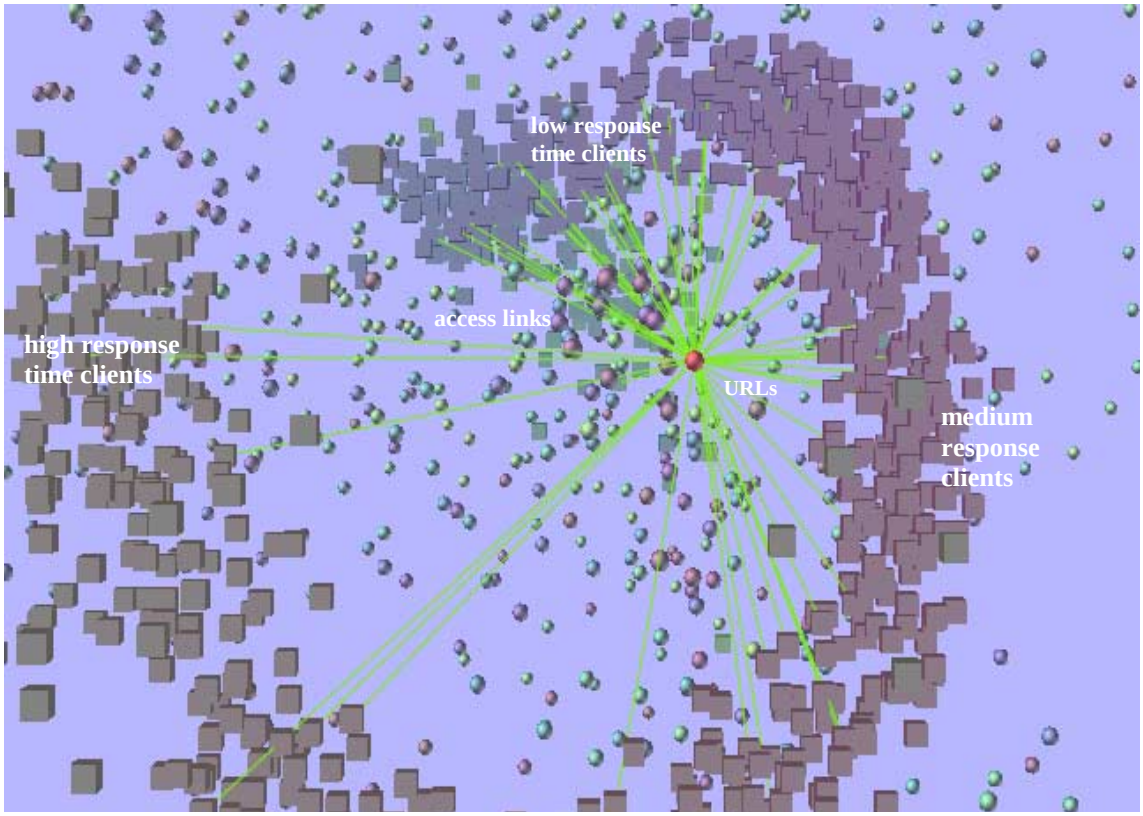


Figure 6: A visual clustering for analyzing 400,000 web transactions with 986 clients and over 2,150 web URLs. (The clients are grouped by the response time and the URLs are placed close to the clients accessing the URLs.)

5. Application and Evaluation

VISIM has been prototyped in several e-service applications at Hewlett Packard Laboratories. It has been used to visually mine large volumes of service transactions and customer shopping activities at online web sites.

5.1 Web Access Response Time Analysis

The visual clustering system has been applied to web access patterns and behaviors. A web transaction starts with a user clicking on a web page. The client (web browser) sends the request through several components, such as applications servers, to perform some service. For example, a user clicks on a web page to purchase an airline ticket. The data access patterns through various components play an important role for the overall transaction. Often, they impact the quality of end-user experience. In order to provide faster service, web analysts need to analyze the data and to balance the workload among their servers.

Figure 6 contains 400,000 transaction records with 986 clients and over two thousand URLs. The rectangles 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 response times are arranged close together, as arranged from low-, to medium-, to high-response time. The URLs are then arranged close to the clients that access the corresponding URL. A colored line, the access link, connects the link between the URL and the client. The color of the line represents the access time. Different from a single similarity, this application requires a multi-step minimization. The first step is used to lay out the clients and the second step is used to lay out the URLs while the clients are frozen in their previously calculated positions. We exploit the freeze capability of our system for that purpose.

There are two types of similarities described as follows.

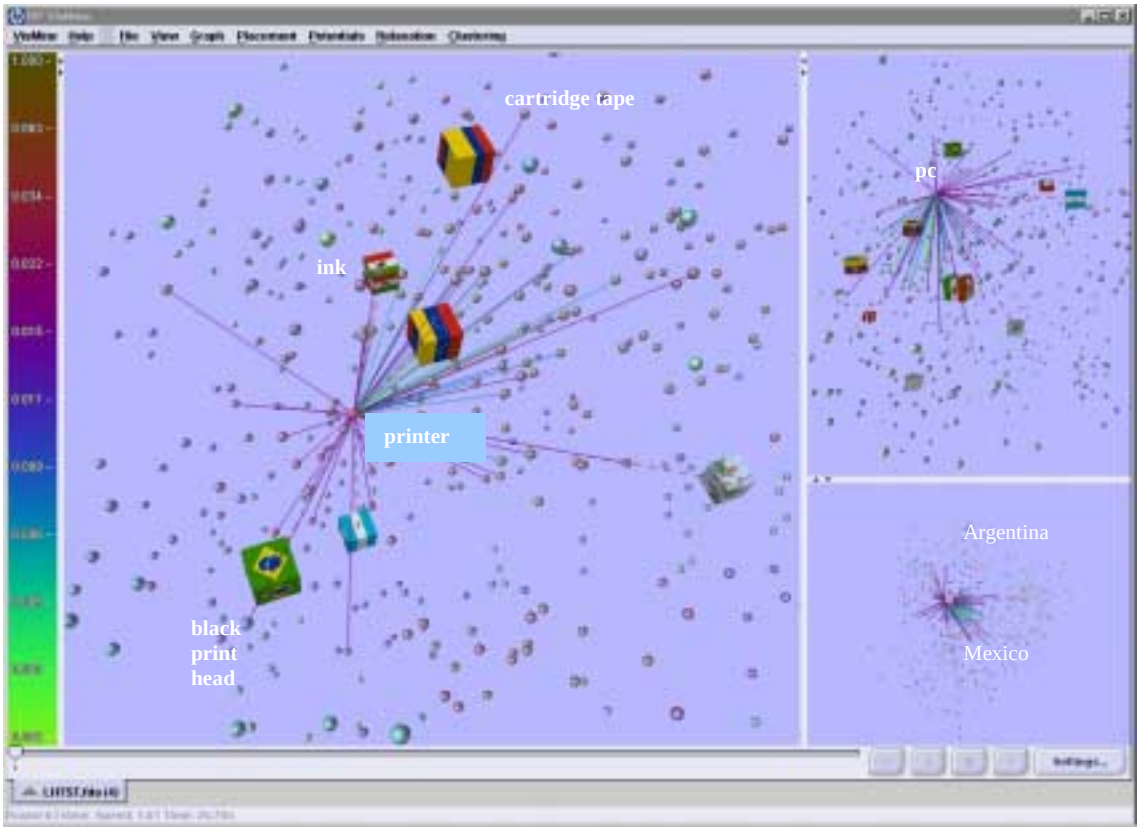


Figure 7: Product-Country Relationship: The selected product is sold only in Mexico and Argentina. There were 47,753 product sales transaction records in the year 2000.

5.1.1 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 time 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 time (colored green) to high response time (colored burgundy) as shown in Figure 6.

5.1.2 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 URLs. After running the second minimization, The URLs will be positioned around the clients which access the URLs as shown in Figure 6.

5.2 Sales Transaction Analysis

One of the common questions electronic store managers ask is how to use the customer purchase history for improving product sales and promotion. Product managers want to understand which products are being bought together. For example, people often buy printer and paper at the same time.

As illustrated in Figure 7, the sales transaction analysis application offers two different operational modes. In the "Products Only Relationship" mode, products are laid out based on how often these products are bought together. In the "Country Relationship" mode, countries are also displayed, and products are laid out with respect to these countries.

In addition to the similarity between products we define a similarity between products and countries. This similarity is calculated as:

$$S = = \varepsilon^* Nc/Nt$$

Where

S= Similarity between products and countries
Nc= Number of times product was bought in country
Nt = Number of times product was bought in total

The factor value ε can be adjusted as needed. The similarity between countries is set to 0 in the current implementation.

The analysts have used this method to analyze their product sales. For example, they were able to find out which country buys what products and what product sales are the largest in which countries. Also, they discovered product combinations such as printer, paper, ink, black print head, and cartridge tape appear together frequently on the same invoice.

Table 1 shows the sales transaction layout time using the accelerated force computation versus conventional methods.

Force computation	#Trans	#Products and Countries N	#Edges (Links) K	Layout Time
Without accelerated method	47,753	1,436	1,060,660	49min
With accelerated method	47,753	1,436	2,075 (# non-zero similarities)	16sec

Table 1: Sales transaction analysis performance comparison.

6. Conclusion and Future Work

In this paper, we have attacked two main problems while using physics-based layout techniques. First, we use a customized potential function to have control over collisions of very similar objects. Second, we employ a prediction-correction scheme by extending the conventional Barnes Hut algorithm to drastically reduce the force computation time. We have applied both ideas to visualize large-scale services and sales at Hewlett-Packard web sites. As a result, the speed of force computation is orders of magnitude faster than before. In addition, the display has substantially improved the clarity of the layout. Our future work will be focus on animation to observe the changes over time.

Acknowledgements

Thanks to Pankaj Garg for his encouragement, suggestions and data on web access response time analysis, to Stephany Kalil and Peter Wright for providing sales transaction data and for technical discussions and suggestions.

References

[1] J. B. Kruskal, and M. Wish. Multidimensional Scaling. Sage Publications, Beverly Hills, CA, 1978.

[2] J. E. Barnes and P. Hut. A hierarchical O(N log N) force-computation algorithm. Nature, 324(6270): pp. 446–449, 1986.

[3] R. J. Hendley, N. S. Drew, A. M. Wood, and R. Beale: *'Narcissus: Visualising Information*, Information Visualization Using Vision to Think, 1999, pp. 503-509.

[4] T. C. Sprenger, R. Brunella, and M.H. Gross. "H-BLOB: A Hierarchical Visual Clustering Method Using Implicit Surfaces", IEEE/VIS2000.

[5] T. C. Sprenger, and M. H. Gross, Ivory – An Object Oriented Framework for Physics-Based Information Visualization in Java, IEEE InfoVis98, NC.

[6] A. Leuski, and J. Allen: *Lighthouse: Showing the Way to Relevant Information*, Proc. Visualization '2000, Salt Lake City, UT, 2000.

[7] S. G. Eick, J. L. Steffen, and E. E. Sumner, Jr. SeeSoft-A Tool for Visualizing Line Oriented Software Statistics, IEEE Transactions on Software Engineering, 1992.

[8] M. Hao, D. Cotting, and T. Holenstein VISIM Internal Architecture Document, 2002.

[9] M. Hao, U. Dayal, M. Hsu, B. D'eletto, and J. Becker. A Java-based Visual Mining Infrastructure and Applications, Proceeding of IEEE InfoVis99, San Francisco, CA. 1999.

[10] L. Greengard, and V. Rokhlin. A fast algorithm for particle simulations. Journal of Computational Physics, 73: pp. 325–348, August 1987. (This paper is credited as the origin of the fast multipole method, with an O(N) algorithm. It was reprinted in the same journal, vol. 135, pp. 280–292, August 1997).

[11] W. H. Press. Numerical Recipes in C. Cambridge University Press, Cambridge, 1990.

[12] I. Bruss and A. Frick. Fast Interactive 3-D Graph Visualization Proceedings of Graph Drawing 95, Spring Verlag, LNCS 1027, pp. 99-110.

[13] S. K. Card, J. D. Mackinlay, and B. Shneiderman. Information Visualization Using Vision to Think. Morgan Kaufmann, 1999.

[14] M. Kreuseler, N. Lopez, and H. Schumann: *A Scalable Framework for Information Visualization*. Proc. Visualization '2000, Salt Lake City, UT, 2000.

[15] L. Greengard, and V. Rokhlin. A fast algorithm for particle simulations. Journal of Computational Physics, August 1987.

[16] A. Graps. Amara Greps' recap of particle simulation methods, www.amara.com/papers/papers.html.