Visual Analytics Techniques for Trend Detection in Correlation Data

Joshua New*
University of Tennessee, Knoxville

Abstract

Domain scientists hope to address grand scientific challenges by exploring the abundance of data generated and made available through modern high-throughput techniques. However, the impact of this large volume of data is limited unless researchers can effectively assimilate the entirety of this complex information and integrate it into their daily research; interactive visualization tools are called for to support the effort. Specifically, typical scientific investigations can make use of novel visualization tools that enable the dynamic formulation and fine-tuning of hypotheses to aid the process of evaluating sensitivity of key parameters and achieving data reduction. These general tools should be applicable to many disciplines: allowing biologists to develop an intuitive understanding of the structure of coexpression networks and discover genes that reside in critical positions of biological pathways, intelligence analysts to decompose social networks, and climate scientists to model and extrapolate future climate conditions. By using a graph as a universal data representation of correlation, our novel visualization tool employs several techniques that when used in an integrated manner provide innovative analytical capabilities. Our tool integrates techniques such as graph layout, qualitative subgraph extraction through a novel 2D user interface, quantitative subgraph extraction using graph-theoretic algorithms or by querying an optimized B-tree, dynamic level-of-detail graph abstraction, and template-based fuzzy classification using neural networks. We demonstrate our system using real-world workflows from a couple large-scale systems.

Keywords: Parallel coordinates, large data visualization, information visualization, focus+context techniques, visualization in physical sciences, life sciences and engineering, graph and network visualization, bioinformatics visualization, focus+context techniques

Index Terms: I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques.

1 Introduction

Today's scientific research frequently generates large amounts of quantitative data that is specific to the domain being studied. At the bleeding edge of science, specialists often seek out new ways to torture their data until it confesses to some new property, gives rise to process insight, or points toward a paradigm shift. While domain-specific analytic techniques certainly have their use, there exist mature bodies of general mathematical analysis in statistics, graph algorithms, and artificial intelligence that continue to invade a broad range of scientific disciplines.

Perhaps the most common data framework for scientific data is the idea of entities with properties, often referred to as multivariate data, which obviates the rise of the spreadsheet and database storage schemas with distinct entities in rows and a list of properties (aka. features or attributes) in columns. To list only a few examples:

1) Climate data - rows of latitude/longitude grid locations and columns of hundreds of environmental variables over centuries

2) Systems genetics data - microarray probe sets with corresponding genome position information for a given strain, RNA transcript abundance for different tissues, genotype from the genetic progenitor, quantitative trait locus mapping information, correlation cluster to co-operative probe sets (gene clusters), genetic pathway information, and relation to some observable characteristic or behavior of the individual contributing the sample

3) Physical systems - individual grid points or (sub)particles with corresponding position, velocity, acceleration, jerk, or other (possibly simulated) measurements

4) Social networks - individuals with specific relation types and strengths with other individuals, organizations, locations, or materials

Multivariate data can straightforwardly be analyzed using statistics, graph algorithms, and artificial intelligence. In fact, a particular analytic process can be wholly or partially automated by chaining together several techniques in each of these domains. However, the quality and accuracy of such a process is a complex tradeoff due to uncertainty both innate to the data as well as inherent in its method of representation. For example, which method of normalization is appropriate for a specific type of statistical analysis, which threshold is appropriate for a certain complexity of graph analysis, and how do you map text and missing features to representations amenable for artificial intelligence methods? The expert user's domain knowledge plays a vital role in balancing tradeoffs adequately for the scientific question at hand. It is common for such investigations to leverage human-in-the-loop control over an iterative process of data visualization, user input, and computer operations on the data. The unique capabilities of real-time interactivity using modern, hardware-accelerated graphics to take advantage of the human eyes broadband pathway to the brain and artificial intelligence to quickly process human-intuitive, semantically rich inputs are called for to facilitate the process. To facilitate this process, I propose an interactive graph visualization system which uses correlation calculated from a database to generate a weighted, undirected graph. This graph can be displayed in several ways, undergo processing from graph-theoretic algorithms, and has a myriad of data associated with each vertex which can subsequently undergo complex analysis. I propose to integrate several well-known and novel methods for decomposing the correlation matrix in order to discover structure and relationships within the data. In the remainder of this proposal, I describe relevant domain-specific background work in section 2, relevant technical background in section 3, and propose several new techniques and extensions to existing systems that assist interactive exploration in section 4, example results from a couple areas in section 5, and outline an approximate schedule of milestones I aim to achieve in my dissertation work in section 6.

2 Application Domains Background

We will apply the graph-based visual analytics framework to several application-specific domains. While the proposed framework should be general enough for application to nearly any domain, I provide background information on several that are relevant to ongoing collaborative research between The University of Tennessee and Oak Ridge National Laboratory.

*email: new@cs.utk.edu

2.1 Systems Genetics Data

"Making sense of genomics is risky,
But with database builders so frisky
Gene expression in brains
May one day explain
A mouse's obsession with whiskey."
-Poet Laureate of the Neuroscience Program, University of Illinois at Urbana-Champaign, November 27, 2006

Let us consider an analogy familiar to the field of computer science: a variable stored at a location in the main memory of a computer. In genomics, one can consider the entire memory space roughly corresponding to the genome, a location-specific variable as a gene, and the value stored in each variable as the genotype at that location. The value of a genotype is transmitted by each parent. The fact that each location can take on different genotypes is termed polymorphism, since the same genome location for different individuals may hold (parts of) different genes or non-gene DNA sequences.

The entire set of genotypes across the genome defines the genetic makeup of an organism, while a phenotype defines the actual physical properties, or traits, of the organism. Although genetic makeup is not the sole factor influencing an organism's phenotype, it is often a strong causative predictor of the trait. Consider common traits relating to physical appearances as an example. Having exactly the same genotypes, identical twins have strikingly similar appearances (phenotypes), yet due to environmental influences they may not look exactly the same.

It is of great interest to unravel the inner workings of how genotypes influence molecular networks to affect a phenotype such as agility, seizures, and even drug addiction, to name a few. Geneticists have already achieved great success in associating a genotype and phenotype for a trait determined by one gene (i.e. monogenic traits), but much present attention is now focused on traits that are determined by many genes (i.e. complex traits). These traits are continuously distributed random variables and thus referred to as quantitative traits. Linear modeling is used to identify genotypes that predict phenotype values. The location of these genotypes are quantitative trait loci (QTLs) [3]. Detected via statistical methods [9], QTLs are stretches of DNA highly associated with a specific phenotype, analogous to genetic landmarks which roughly indicate the position of the active gene. QTLs are not defined at very fine granularity; they usually correspond to areas large enough to hold several genes. The genetic polymorphism (genotypes) in neighboring areas of a set of loci, as a group, influence structure and function on both molecular and organismic scales.

For decades, scientists have systematically randomized and then stabilized genetic variation in groups of mice to effectively create a population of clones. These mice, called "recombinant inbred" (RI) strains, function as a reference population which is used by groups worldwide in order to allow a basis of comparison and integration across different experiments [8]. This is very important from a statistical standpoint as it implies that the potential size of the combined datasets is theoretically unbounded, resulting in extremely high dimensional data. Sufficient confidence is currently allowing integration of diverse biological data across levels of scale in an approach related to systems biology, "systems genetics." This integrative approach for multiscale and multiorgan phenotypic datasets has only become feasible in recent years and relies heavily on statistical techniques, complex algorithms, high-performance computing and visualization.

Let us consider an analogy familiar to the field of computer science: a variable stored at a location in the main memory of a computer. In genomics, one can consider the entire memory space roughly corresponding to the genome, a location-specific variable as a gene, and the value stored in each variable as the genotype at that location. The value of a genotype is transmitted by each parent. The fact that each location can take on different genotypes is termed polymorphism, since the same genome location for different individuals may hold (parts of) different genes or non-gene DNA sequences.

The entire set of genotypes across the genome defines the genetic makeup of an organism, while a phenotype defines the actual physical properties, or traits, of the organism. Although genetic makeup is not the sole factor influencing an organism's phenotype, it is often a strong causative predictor of the trait. Consider common traits relating to physical appearances as an example. Having exactly the same genotypes, identical twins have strikingly similar appearances (phenotypes), yet due to environmental influences they may not look exactly the same.

It is of great interest to unravel the inner workings of how genotypes influence molecular networks to affect a phenotype such as agility, seizures, and even drug addiction, to name a few. Geneticists have already achieved great success in associating a genotype and phenotype for a trait determined by one gene (i.e. monogenic traits), but much present attention is now focused on traits that are determined by many genes (i.e. complex traits). These traits are continuously distributed random variables and thus referred to as quantitative traits. Linear modeling is used to identify genotypes that predict phenotype values. The location of these genotypes are quantitative trait loci (QTLs) [3]. Detected via statistical methods [9], QTLs are stretches of DNA highly associated with a specific phenotype, analogous to genetic landmarks which roughly indicate the position of the active gene. QTLs are not defined at very fine granularity; they usually correspond to areas large enough to hold several genes. The genetic polymorphism (genotypes) in neighboring areas of a set of loci, as a group, influence structure and function on both molecular and organismic scales.

For decades, scientists have systematically randomized and then stabilized genetic variation in groups of mice to effectively create a population of clones. These mice, called "recombinant inbred" (RI) strains, function as a reference population which is used by groups worldwide in order to allow a basis of comparison and integration across different experiments [8]. This is very important from a statistical standpoint as it implies that the potential size of the combined datasets is theoretically unbounded, resulting in extremely high dimensional data. Sufficient confidence is currently allowing integration of diverse biological data across levels of scale in an approach related to systems biology, "systems genetics." This integrative approach for multiscale and multiorgan phenotypic datasets has only become feasible in recent years and relies heavily on statistical techniques, complex algorithms, high-performance computing and visualization.

2.2 Climate Data

"Prediction is very difficult, especially if it's about the future."
-Niels Bohr, Nobel laureate in Physics

Climate scientists have very large and ever-growing datasets using ground-truth data from centuries of measurements. However, these datasets have many problems that hinder their effective analysis: varying measurement times, irregular grid locations, quality of measurements (occasionally dependent upon untrained individuals or faulty equipment), inaccurate bookkeeping, differing number of climatological

variables over time, a plethora of missing values, and changing standards of measurement over time. These are but a few of the sources of uncertainty inherent in climate data.

To be sure, scientists are always working on ways to remove this uncertainty, collect more accurate measurements with greater certainty, and continue to improve as more advanced technology becomes deployable. Nevertheless, climatologists have been tasked with extrapolating to predict the future of our planet's climate. This is often approached through creating statistical models from the collected data for individual variables to determine the relationships among the variables and incorporating these into complex, number-crunching simulations which can carry the current state of the climate into the future.

Recent work at Oak Ridge National Lab has focused on the quantitative segmentation of the global climate into ecoregions, also known as climate regimes [19]. These areas which have been spatio-temporally clustered based on similar environmental factors can have many uses in management, legislation, ecological triage, and comparison of standard simulation models [22].

Oak Ridge National Lab has several high performance computing resources that are leveraged to keep them on the cutting edge of research and innovation. In the relevant climate work, supercomputers are used to run parallel k-means clustering for ecoregion identification as well as for model-fitting routines to predict one variable in terms of others. This process creates a multitude of data which can be difficult for even an expert in the field to look through efficiently or effectively during the process of knowledge discovery. Visual analytics tools are called for to aid this process.

Building on much related work from statistical plots, we propose the use of parallel coordinates to allow intuitive visualization and interaction with data across any number of dimensions. We also plan to provide automated trend detection algorithms to highlight potential inter- and intra-variable relationships of interest. These trends and data selections are then projected onto a geo- registered map which is the typical method of presentation for climate scientists.

3 Technical Background

A graph is a universal concept used to represent many different problems. While more restrictive layouts, such as trees, should be used when possible, this work will address the general case of graph interaction. In relation to this work, we categorize methods to comprehend graph properties as: (i) those solely depending on algorithms, i.e. the algorithmic approach, and (ii) those incorporating human input as an integral component, i.e. the interactive approach. Let us review both approaches in turn.

3.1 The Algorithmic Approach

Algorithmic research to automatically compute graph properties of various kinds has been extensively studied. Well known examples include clique, strongly connected components, induced subgraph, shortest paths, and k-connected subgraph. Let us use clique analysis as a representative example. By filtering out edges with weights below a certain threshold, a gene network with high co-regulation should appear as a complete subgraph, or a clique. Hence, it is natural to consider clique analysis in gene expression data analysis.

However, clique analysis is an NP-complete problem. Even though more efficient fixed-parameter methods [35] are currently being used, it is still a very time consuming procedure to compute. It is also hard to treat edges with negative weights in the context of clique analysis, so common approaches typically preprocess the graph to convert all edge weights to absolute values. The impact of information loss due to thresholding is hard to evaluate and is further complicated by the presence of noise. While partially resolved by paraclique [35] methods in which a few missing edges are acceptable, additional problems are introduced such as the meaning of paraclique overlap which may be handled differently depending on the working hypothesis.

Such shortcomings apply to different graph algorithms in varying degrees, but are generally inherent with graph theoretic analysis. However, this should in no way prevent graph algorithms from being used for suitable problems. From this perspective, it would be greatly advantageous to develop a visual, effective and efficient feedback framework. In this framework, a human expert is enabled to quickly identify imperfect portions and details of the data, and not only remove irregularities but also to significantly reduce the dataset's complexity by interactively constructing various levels of abstraction. The resulting problem space would be more appropriate for graph theoretic analysis to be applied. In fact, some undertakings in visualization research have already adopted similar approaches [44].

Here we note that our goal is neither to accelerate all computation in a scientist's workflow nor replace computation solely with visualization. We hope to develop a visualization framework which allows navigation through gene expression data and segmentation of the appropriate data for further study. In this way, s/he can flexibly choose and apply the right computational tool on the right kind of problem.

3.2 The Interactive Approach

Much related work in visualization follows the Information Seeking Mantra proposed by Shneiderman [51]. That is: overview first, zoom and filter, and then details on demand. At each of the three stages, there are a number of alternative approaches, many of which are highly optimized for a specific application. A key driving application in this area has been visualization of social networks [43].

To provide an overview, the graph can be rendered in the traditional node-link setting or adjacency matrix [1], and more recently as a self-organizing map [33]. When using the common node-link model, it is pivotal to develop a sufficient hierarchy of abstraction to deal with even moderately sized graphs. Solely relying on force directed methods (i.e. spring embedding [39]) for graph layout cannot resolve visual clutter and may still significantly hamper visual comprehension.

Structural abstraction can be computed either bottom-up or top-down. In bottom-up approaches, one can cluster strongly connected components [34], or by distance among nodes in the layout produced by a spring embedder [56]. Top-down approaches are often used for small scale or largely sparse graphs in which hierarchical clusters are created by recursively dropping the weakest link [4]. More comprehensive systems employ clustering algorithms that consider a number of different node-edge properties [2].

Semantic-based abstraction is a more powerful mechanism for providing an overview, zooming, or giving details. This approach is tied to its intended application since it requires specific domain knowledge of the semantic information [52]. When combined, structural and semantic abstraction can prove to be very effective [49]. Also in [49], it is shown that overview and level-of-detail (LoD) enabled browsing can be based on induced subgraphs of different sizes.

There are many well-known packages that have evolved over time to specifically address visualization of gene correlation data using node-link diagrams such as Cytoscape [47] and VisANT [23]. These tools are built to be web accessible and thus render node-link diagrams using 2D layouts. While 2D layouts are accepted by the community, such packages neglect modern 3D acceleration hardware, rarely scale well

beyond hundreds of nodes, and do not leverage 3D operations that have proven to be the preferred representation and navigation techniques for our users. Due to the common 2d framework, and in contrast to Shneiderman's principle, biologists are typically forced into a workflow in which filtering must be first applied and a global overview of the entire dataset simply isn't possible. Our software leverages both OpenGL and efficient C compilation to facilitate interaction with tens of thousands of nodes while maintaining interactive performance with complex visual analytics tools not currently available in these packages. Current work involves integration with a lightweight API [48] to allow web-based interaction and data-sharing so our software may be used synergistically with such well-developed packages.

In contrast to the node-link model, an adjacency matrix is a clutter free interface. While an adjacency matrix interface for large data is limited by the resolution of the display, it is still ideal for a bird's eye view [1]. Some patterns such as clique and bipartite subgraphs could be very distinctive when examined in an adjacency matrix. However, a proper order of vertices is critical. The community has studied this problem at length. In [21], a comprehensive survey on automatic vertex order is included. In general, binary, undirected graphs are the most straightforward. While weighted graphs needed more complicated algorithms, graphs with negative weights are less studied. Based on adjacency matrices, LoD type of browsing is often supported as well [1].

Due to the complexity involved in computing a high quality overview of a graph, researchers have also attempted to use self-organizing maps [33]. Self-organizing maps are a dimension-reduction technique which adjusts weights in a manner similar to neural networks to discretize the input space in a way that preserves its topology. The end result is (usually) a 2D field that can be conveniently rendered as a terrain.

By creating a spatial layout for a graph, it can be interactively visualized while preserving the data's underlying topological relationships. Typical interaction methods include focus+context methods (i.e. zoom and filter), graph queries using language-based methods [50], and filtering databases of graphs using graph similarity metrics, typically based on non-trivial graph theoretic algorithms [44].

Social networks are currently a primary driving application of interactive methods for graph visualization. This has resulted in non-binary, non-positive definite weights not being as thoroughly studied. Also, tools for extracting highly connected subgraphs from this data in a way that addresses the inherent uncertainty appear to be lacking. Whereas neural networks have already been used for volume segmentation [55], similar approaches have rarely been attempted in graph visualization. In this work, we propose several tools that allow traditional quantitative drill-down as well as qualitative selection and filtering techniques to aid domain experts with their analysis.

3.3 Parallel Coordinate Plots

Parallel coordinates, originally introduced by [26], have become increasingly popular as a scalable technique for visualization and interaction with large multivariate data. A parallel coordinate plot (PCP) is a generalization of a Cartesian scatterplot in which axes are drawn parallel to one another. This type of diagram highlights the more common case of parallelism, rather than orthogonality, present in higher-dimensional geometry. PCPs also allow an arbitrarily large number of dimensions to scale intuitively within the plane, whereas human perception degrades quickly as dimensions higher than three are projected to a 2D display.

PCPs developed as a way to accurately visualize and thereby gain insights from multidimensional geometry. From their onset, several mathematical properties were proven which enhanced their interpretation by defining analogues between parallel coordinate space and two-dimensional Cartesian scatterplots [37]. These included the point-line, translation-rotation, and cusp-inflection point dualities [25, 26]. This technique quickly found its way into Vis [24] and perceptual properties such as the importance of axis orderings were considered as early as [60]. While [60] gives equations for selecting an order from a set of axes, it in no way addresses optimality criteria such as those being proposed in this paper.

There has been much research to alleviate some of the inherent weaknesses of PCPs such as visual clutter when dealing with large data. Techniques for clutter reduction include clustering, subsampling, and axis redirection. In [13], the authors use a clustering algorithm to create a hierarchical representation for PCPs and render a graduated band to visually encode variance within a cluster. In [30], the authors use K-means clustering, a high precision texture to reveal specific types of clusters, multiple transfer functions, and an animation of cluster variance to accurately convey the clustered data while minimizing clutter. In [10], the authors provide a sampling lens and demonstrate that random sampling of lines within the lens is the optimum choice in the tradeoff between their accuracy metric and performance. In [59], the authors use the grand tour algorithm to generate a d-space general rigid rotation of coordinate axes which can be used to confirm separability of clusters.

PCP implementations often operate on binned data and even uncluttered PCPs often demonstrate data ambiguities. The traditional straight-line rendering can be augmented by using energy-minimization to render curved lines as in [62]. Likewise, the authors in [16] used Gestalt principles of good continuation in order to address ambiguitives from line crossovers and shared values. Binning can cause outliers to dominate data expression or be filtered out altogether which has lead [42] to more thoroughly analyze the preservation of outliers while capturing the major trends in PCPs.

In additional to use solely as a visualization technique, PCPs can be used as an intuivite navigation and query methodology. Recent research has demonstrated the ability to interactively alter axis ordering, interactively brush range queries, and use axis scaling to refine those queries [53]. The introduction of a navigation element to the visualization leads naturally to more complex data mining techniques. In [11], the authors use parallel coordinates, and its circular variant, as multidimensional visualization elements in a taxonomy of other techniques. They also highlight user interface and analysis concerns, highlighting the strengths and weaknesses of PCPs in the context of other visualization methods.

There are also several variants of the traditional parallel coordinates rendering. These include the circular variant in which axes are drawn as spokes of a wheel [11]. This approach was extended by [29] who created three-dimensional parallel coordinate renderings by placing axes like tent pegs in a circle around a central axis. PCPs were also extruded into three-dimensional rendererings by treating each line as a plane that could arbitrarily be transformed [58].

4 Proposed Capabilities

The information age has resulted in an explosion in the quantity of data, both social and scientific. Such voluminous data easily overcomes the limits of human cognition and perception. The proposed graph analytics package has a wealth of algorithmic capabilities which help in the process of drilling down through large, complex data and aid (or automate) the process of knowledge discovery. This necessitates new data management, filtering mechanisms, and visualization tools that leverage efficient data storage, retrieval, and analysis to facilitate

actionable knowledge discovery. My proposed work provides several computational tools which is applicable to nearly any domain and will showcase results in a few of the hottest. Some of my work has been showcased at several events (ACM, KBRIN, SCIDAC, Gaggle), resulted in a publication [40], and more are currently in progress.

4.1 Proposed Data Management System

The only required data is a matrix containing gene-gene correlation values. While all of our testing data use Pearson's correlation, different metrics of correlation are treated no differently in our system. In addition, we handle a database of information corresponding to each gene as can be seen using three relational tables. Specific information about the object of interest is stored in the Gene table while information relating to computed gene networks is stored in the Paraclique table. Since our driving application is to identify the genes that cause variation in complex traits, it is necessary to show the relationship or distance between genes and QTLs. For that, we need an additional relational table describing the exact location of QTLs in the unit of megabases.

Graph theoretic algorithms provide valuable information that is otherwise hard to discern about the data. However, many such algorithms incur long compute times and are far from being interactive. For those algorithms, it is then necessary to pre-compute and store their results for visualization at run-time. In this work, for example, we pre-compute and store each gene's membership in any of the paracliques. The resulting data can easily be stored in a relational table.

We treat all data in the relational tables as attributes of individual vertices, and the correlation values as an attribute specific to each edge. This is a very generic model that is applicable to a variety of application domains and is a boon to scientists typically involved in spreadsheet science. Based on these data, it is then the job of the visualization system to facilitate interactive, hypothesis-driven study by the user.

4.2 Graph Representation

A graph is defined as an ordered pair G=(V,E) where V is a set of elements called vertices (or nodes) and E is a multiset of unordered pairs of vertices called edges. Graphs are a very general dataset that are good at representing many kinds of entities and relationships. Moreover, there is a wealth of information on algorithms for analyzing and determining potentially useful properties about a graph. Due to the constraints of our targeted application domains, we were able to speed up many of the algorithms by supporting only weighted, undirected graphs. There are many ways to represent graphs, each with their own advantages and disadvantages. Two of the most common graph representations are as an adjacency matrix (better for dense graphs) or as an adjacency list (better for sparse graphs). We support either format as file input, but store the graph internally as an adjacency matrix for better caching performance and hashing construction. We have found that storing a weighted, undirected graph with 1e-4 precision in a lower triangular matrix of short integers gives storage overhead low enough to store on one computer while also providing a boost to algorithm interactivity. This also makes it amenable to adaptation and display on current PDAs which only support fixed-precision arithmetic and OpenGL ES. This weighted-edge adjacency matrix of short integers is stored as mat2d inside a graph struct which contains all other data loaded, computed, and ready for display. We maintain data-handling efficiency and expandability by only passing pointers of a graph struct between various algorithms.

4.3 Data Support

We support several data types and formats for many kinds of data. The graph interaction package is written such that, given the filename of one of the files, it will check and load the other data files. If a data file is not there and must exist for the program to run, it will generate the required file which will be loaded on the next run. Below is a list of information that can be used by the application and the file format expected:

Weighted Edge List `[*.wel]` - currently the one and only file that's required (all others can be computed from this file) containing edge weight information.
NumVerts NumEdges
VertexIName VertexJName WeightVitoVj...

Lower Triangular Matrix `[*.ltm]` - can be used instead of a .wel file.
NumVerts NumEdges
WeightV_2toV_1
WeightV_3toV_1 WeightV_3toV_2...

Graph Vertex Layout `[*.gvl]` - file which contains a 3D position (as a short int `[-32k,32k]`), will be computed from input file if it doesn't already exist.
Comment line (parameters used to compute the layout)
NumVerts NumEdges
Vertex1 $V1_x$ $V1_y$ $V1_z$ ...

Database Features `[*.txt]` - floating point data associated with each of the vertices. If not available, query will be for vertex degree or ID if fully connected.
NumRows NumCols
Row_1Col_1 Row_1Col_2 ... Row_1Col_N...
Row_2Col_1 Row_2Col_2 ... Row_2Col_N...

Feature Map `[*.map]` - database index file which provides a mapping from vertex names in the .wel file to the data order in a .txt file. If not available, query will be for either vertex degree or ID if fully connected.
$Vertex_I$Name
$Vertex_J$Name...

Vertex Labels `[*_labels.map]` - vertex names displayed on the graph when desired in same order as the .map file. If not available, vertices will be named according to the .wel file.
$Vertex_I$_DisplayName
$Vertex_J$_DisplayName...

Block TriDiagonalization (BTD) Vector `[*.btv]` - used to compress the adjacency matrix along the diagonal for display and interaction. Entire array is processed in which all rows are permuted and then all columns. If not available, no BTD belt will be shown along the bottom.
NumVerts
$RowCol_I$ $RowCol_J$ $RowCol_K$

4.4 Force-directed Layout

The software has been designed to easily incorporate other graph layout algorithms, such as the fast multipole multilevel method (FM^3) [18] or LinLog [41]. Such layout algorithms have different strengths and weaknesses in conveying specific properties in the resulting topology. Custom algorithms can be easily incorporated in our application either procedurally or by simply loading a graph layout file. By default, a single layout is computed for a graph and its appearance is modified at run-time but the user may also switch interactively between multiple layouts. Additionally, the user may dynamically swap between the customary 2D view and the 3D view which tends to convey more relationship information.

It is also noteworthy that clustering is another term often used by biologists in their research. Although in our work "cluster" and "dense subgraph" have similar meanings, the goal of our approach is quite different from the basic goal of popular clustering algorithms. Our goal is to adequately handle uncertainty in the pursuit of co-regulated (putatively cooperating) genes forming a network, and the nature of the interconnections among those networks. We use BTD belt, queries and neural network to computationally assist the discovery and realtime fine-tuning of those dense subgraphs of interest. We do not solely rely on graph layout algorithms to reveal dense clusters.

4.5 Rendering

Vertices are rendered after the edges without the depth buffer to prevent edges from occluding data points. We support the option of rendering vertices as splats (also known as billboards or impostors) or quadrics which can take arbitrary shape (usually spherical). The splatting implementation is the typical geometry-based primitive scheme using pre-classification and thus results in slightly blurry vertices but has the advantage that it is typically fast. While both options render in realtime on most single computers, framerate tests were conducted for the 7,443-node graph at several resolutions on a powerwall using Chromium. This resulted in 16 fps quads vs. 246 fps splats on an 800x600 viewport, and 5 fps quads vs 17 fps splats at 3840x3072 (9 monitors).

There are many interactive rendering options such as color table generation and weight-mapping mechanisms for coloring edges. A default set of these has been provided to express differentiation between solely positive and negative edges (up and down regulation) or to enhance contrast between edges with similar weights. The rendering mechanism is adaptive and can optionally adjust properties such as the number of subdivisions in the quadrics based upon the current frame rate. Semi-transparent vertex halos [54] are rendered using splatting for enhanced depth perception. Intuitive click-and-drag interaction and continuous rotation during manipulation circumvents problems with 3D occlusion and aids perceptual reconstruction of the topology through motion parallax. The system also has dozens of minor utilities including the ability to change the color table used by all elements of the program, take a screenshot, create video, print statistical information for the current graph, and output gene lists.

4.6 Compound Range Queries

Information contained in relational tables needs to be studied in an integral fashion with gene networks. A common comprehensive tool for accessing information in those formats is compound boolean range queries. Unfortunately, it is a difficult process to efficiently integrate a commercial database management system with realtime visualization systems. For this application, we have extended the functionality of a recent visualization data server to provide essential data management functionalities. The core of that data server is a simplified B-tree that is optimized assuming no run-time insertion or deletion in the B-tree [15]. Using this B-tree, the database in our system, with compound boolean range queries over 8 features, can be queried in $O(log\ n)$ time at a rate of 10 million vertices per second on a 2.2Ghz Athlon64. This results in interactive, sub-millisecond response to sets of dynamic queries even for large graphs.

The effect of querying is data filtering and thereby complexity reduction. One of the most common data filtering operations for biologists is thresholding. With visualization, this threshold choice can be applied to a graph interactively and result in both visual and statistical testing for proper threshold selection. Besides thresholding, we also provide more power by also allowing multiple, dynamic, quantitative queries over any computable attributes of a vertex or edge. Sample queries include all genes within a 100-megabases distance to a target QTL, or all genes in the current subgraph that are significantly related to a paraclique of interest.

The queried genes also form a subgraph, no different from those qualitatively selected via the BTD belt interface. Our querying system attains realtime performance, making it feasible to visually evaluate the effects or sensitivity of key parameters, such as what threshold value to use.

4.7 Graph Operations

Many graph algorithms are simple and can shed much light on a graph with very little computational overhead. We allow several to be invoked at any point. Others are NP-complete and must be pre-computed and stored in the database information (for later query) in order to maintain interactive performance. Several integrated graph properties for graph similarity classification we plan to include are degree of vertex, transitive closure, connected components, edge expansion, and shortest path. Each of these properties has a unique biological interpretation. Degree is one of the simplest useful metrics that can be calculated and allows biologists to determine statistical correlation between genes and gene networks as there are often many loosely connected genes and few highly related genes. Transitive closure minimizes the longest distance to all other nodes and can be used to find the core of a genetic structure. Connected components allow biologists to visualize only related data within a given subgraph. Edge expansion shows relationships within a given subgraph without regard to threshold. Shortest path

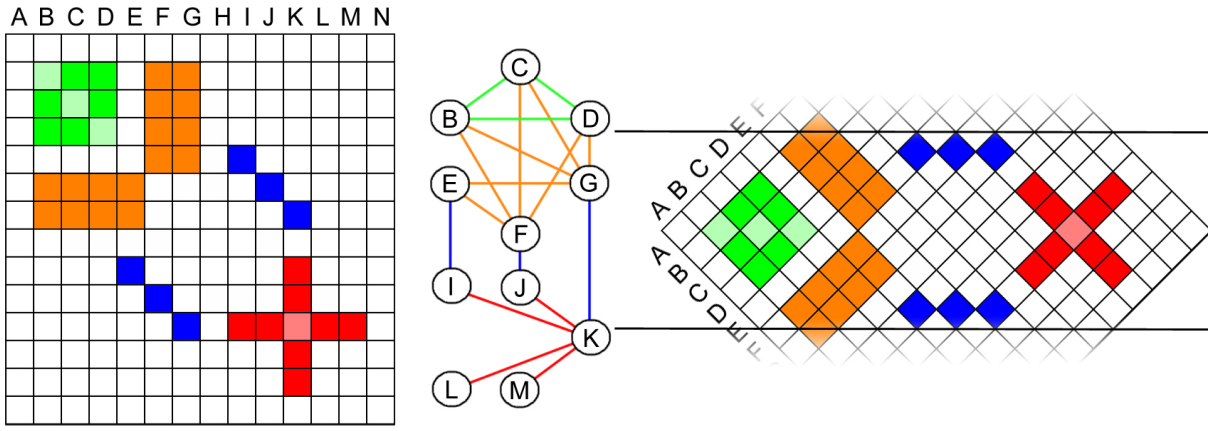

Figure 1: Illustration of a permuted adjacency matrix with common graph patterns (left), and extraction of the BTD belt for qualitative selection (right).

allows biologists to see how specific genes most directly regulate one another. It could also be used to further investigate a favorite location in the graph and gather only the local correlates view of the specific gene(s) under study. These properties may be used as extra variables in the feature vector used for training a neural network. By adding these to the relational data that are queried, we provide more information to be leveraged for fuzzy classification.

4.8 Block Tridiagonalization

A major difficulty with graph visualization is the visual clutter caused by the sheer complexity of the data. An adjacency matrix provides a concise interface for overviewing the data in a way that is free from visual clutter. However, the 3D space is still a natural domain for user cognition. The added dimension can be used to convey additional data, allowing navigation through node-link rendering and full appreciation of structural cues in the data. For our application, we have designed a framework where an overview is provided through a 2D adjacency matrix. Users can arbitrarily select subsets of interesting vertices and create abstractions for further interactions in 3D.

Unlike popular datasets studied in most previous works, the range of edge weights in our data is `[-1.0,1.0]`. In addition, to let users decide about uncertainty issues, we would like to avoid taking a threshold at this stage of processing due to possible information loss. This makes it hard to directly leverage existing vertex reordering algorithms like those surveyed by Mueller [38] or Henry et al. [21].

Block tridiagonalization (BTD) is a mature numerical algorithm that permutes row and column elements of a matrix in such a way as to cluster nonzero elements in blocks along the diagonal [5]. This algorithm always preserves the eigenvalues of the original matrix within a specified error tolerance. It iterates until the following criteria are met: (1) the final matrix has small bandwidth relative to the size of the matrix, and (2) any off-diagonal blocks in the final matrix have either low dimension or are close to a low-rank matrix.

The BTD algorithm was developed to improve both performance and storage efficiency for solving eigen problems. The smaller a block is in a matrix, the lower the corresponding rank in most cases. Thus, the optimization goal of BTD is to minimize block sizes on the diagonal, and correspondingly reduce block sizes off-diagonal as well.

The result of BTD is often characterized as minimization of bandwidth, because non-zero entries are clustered around the diagonal. It is very significant to our research. In our application, the minimization of diagonal block sizes through global optimization provides a reliable means to abstract a large graph into a set of minimal basic “building blocks,” each of which represents a densely correlated subgraph. The vertices in these subgraphs appear in contiguous segments along the diagonal. The off-diagonal blocks determine how these “building block” subgraphs are interconnected. In this way, we can conveniently reassemble the original graph using the minimized diagonal blocks, and show more appreciable structures with significantly less clutter.

Let us consider the illustration in Figure 1 from a biological perspective. In this example, we show four graph patterns that are often of interest to geneticists. Since every data entry in an adjacency matrix represents an edge, selections made in adjacency matrices are on edges and only indirectly on vertices. The green subgraph is a clique of highly correlated genes that potentially operate as a unit. The orange subgraph is a bipartite graph, used in gene-phenotype mapping, in which the trait is correlated with a number of related genes which would be of experimental interest. The blue subgraph is a perfect matching graph which functions as bridges between subgraphs. The red subgraph is a star containing a “hub” gene which could be targeted for knock-out and affect expression in many structures.

For our real world datasets, BTD has been able to consistently generate permutations that compress the majority of non-zero to the diagonal. This enabled us to crop a stripe along the diagonal and rotate that stripe to a horizontal position, as shown in Figure 1, bottom. We refer to the horizontal stripe as the BTD belt.

BTD belt is a more efficient use of precious screen resources. Our datasets typically contain several thousand genes so the adjacency matrix is typically very large. Although it is possible to downsample the matrix for on-screen viewing, the essential high frequency details in the matrix could be hard to distinguish.

From the BTD belt interface, a user can select diagonal blocks that are perceived to be “highly correlated”. Letting a human expert decide what can be considered as “highly correlated” is our way of handling data uncertainty. We note that the functionality of detecting high correlation in a general setting is a hard problem, particularly when the acceptable tolerances of error can only be qualitatively determined in a subjective manner. In this regard, BTD can be considered as a computational tool for creating data abstraction.

Since the BTD-belt is a permuted and rotated adjacency matrix, much information is visible along the diagonals which correspond to an individual vertex. Therefore, the BTD belt immediately shows the major graph structures in which each vertex participates. These properties

can be used to quickly determine the role of specific genes in varying numbers and types of networks.

To allow further data abstraction, users are able to dynamically generate level-of-detail (LoD) representations that facilitate simultaneous operation across multiple levels of scale. At any point in our system, the current working graph can be saved and is rendered, along with all other saved subgraphs, in the background of the current working graph. The LoD graph can be generated by taking all user-defined subgraphs and treating them as supernodes. By default, the edge connecting two supernodes has a weight defined as the average of all edge weights between vertices within the two corresponding subgraphs. Optionally, feature vectors of topological metrics can be calculated on each subgraph for querying and graph pattern matching to find similar graph structures rather than similar data items. This LoD graph can subsequently be treated as a normal graph and all provided analytics tools in our system can be used to perform increasingly complex analysis at higher levels of abstraction.

4.9 Neural Networks

When interacting with complex data, it is often useful to provide an automated arbitrator between the user and the data so that repetitive tasks are off-loaded from the human user. One such important task is to exhaustively search for genes that match a certain pattern and classify the data accordingly while handling the innate uncertainty. A key motivation is to alleviate users from the need to manually browse through the entire dataset and thus specifying the feature they think they are seeing in an overly rigorous manner. It is desirable to have a proper level of fuzziness into this iterative feedback loop. While elaborate agents can take a long time to develop and are too complex to train in realtime, we have implemented a simpler AI system which users can train at run-time.

In our system, we use a feedforward, multilayer, back-propagation neural network capable of quickly learning items of interest and displaying similar items to the user. From this perspective, qualitative selection allows users to visually perceive uncertainties and decide how to best guide the computational process, while quantitative queries provide an exact means to request a subset of data. With all datasets, the neural network classifier can be trained with subsecond efficiency.

We use a multilayer, feedforward, back-propagation, online neural network for realtime classification which is able to learn while in use by employing stochastic gradient descent. Our implementation closely follows the description in [45]. Results are given for a neural network with the number of input nodes corresponding to the number of provided attributes, 30 hidden nodes, 2 output nodes, a learning rate of 0.4, a sigmoid threshold function, and a hard max used to simply select the most likely output. The unusually large number of hidden nodes provides sufficient degrees of freedom for any problem domain and could be reduced if training speed or overfitting become issues. Each of the two output nodes corresponds to the likelihood that the user does or does not want to see the object.

Neural network interaction involves only a few easy steps. The user left-clicks on an arbitrary number of vertices to select them as examples. Similarly, right clicking on a vertex adds the vertex as a counter-example. The user may use any filtering or processing techniques previously mentioned to aid the process of defining the training set. Once all examples and counter-examples have been selected, the entire dataset is processed to only show items like the examples or to segment the data using color. Training the neural network and classifying all other data items is in the order of dozens of milliseconds and is transparent to the user.

Deciding the proper training set is critical when attempting to achieve accurate classification for multivariate data due to the high-dimensional decision space. In our system, we provide two alternative approaches for the user definition of large training sets. First, we allow selection of the training set using supernodes in the LoD graph. Each supernode represents a network of genes, typically segmented through BTD selections or database queries, such that a few example supernodes can correspond to hundreds of individual data points. In this way, users can visually interact with the data while quickly selecting entire groups of vertices as examples or counterexamples. Second, the application allows the import/export of vertex data for the current working graph using ASCII files. This can be used to analyze the corresponding data with much more sophisticated statistical packages such as Statistical Analysis Software (SAS) or statistics programming languages such as R. These analytics tools or their batch programs can be called during run-time to either fully segment or partially classify the data. The result can be stored in a vertex list file and then utilized as a training set.

4.10 Hierarchical Representation

The true power of the package is in the way all these tools can be arbitrarily combined by an expert user to investigate or discover specific patterns in the data. Similar to the power-of-ten paradigm, we allow the user to operate on multiple levels of abstraction simultaneously. This is done by allowing the representation of an entire graph as a single vertex (aka supernode), possibly within the context of another graph. By computing the weight between supernodes and other super/nodes as the average normalized weight or a measure of connectivity, users can see relationships among entire clusters of objects. Each of the tools listed in this section operate on supernodes as simply as they do on normal vertices. For most tools this is achieved by keeping a list of the vertex IDs that each supernode encodes. For example, selected supernode(s) for neural network training behave as a template-based search for similar data items in the original database. Possible uses of the level-of-detail capabilities in our application domains include seeing the connectivity between paracliques of genes in systems genetics data, relationship of variables across multiple disparate ecoregions in climate data, or relationships between organizations of individuals and materials in social networking data.

4.11 Parallel Coordinates

Visualization and computational tools are necessary for the analysis of large multivariate data. Parallel coordinates rendering has proven very useful in this area as it allows intuitive visualization of a multidimensional attribute space. It is very natural to visually "chain together" a sequence of variables in a linear layout. Ideally, a parallel coordinate rendering should provide a preview containing sufficient information to guide users to the most interesting parts of the underlying data.

This goal is quite achievable when the data at hand is manageably small, both in terms of total size as well as total number of variables that must be considered at any particular time. However, without sufficient computational tools at our assistance, this task is quite daunting, particularly when we increasingly need to handle datasets with hundreds to thousands of variables.

From a practical point of view, often a parallel coordinates rendering can only effectively a few axes in one setting. Among the $O(N^2)$ bi-variate relationships in a N-variable dataset, to choose the right handful of those relationships in an optimal fashion is still an open problem, as is the question of how to best order the axes in a computational way.

The novelty of our work stems from our taking an optimization driven perspective to explore the full potential of using parallel coordinates to visualize datasets with a large number of concurrent variables. Our work is also novel due to our way of characterizing trends in the context of parallel coordinates, and the resulting new possibilities of clutter reduction. Finally, we also devise a novel rendering method to better reveal trenads visually in parallel coordinates.

Our goal is to develop a general system which generates a near-optimal axis ordering to obviate key trends for a given dataset. The only required data is a matrix containing axis-axis correlation values which roughly denote the importance or strength of a trend between two attributes based upon a user-selected or computationally defined metric.

Once given an *NxN* matrix corresponding to all pairwise metric computations, the problem can be solved in the domain of graph algorithms by treating it as an adjacency matrix. In this framework, there is a graph of N vertices and N^2 edges from which we want to extract what we shall refer to as an "optimized k-walk" where k is the number of axes desired in the parallel coordinate plot. This can be seen as a generalization of the TSP problem in which the salesperson must visit $k \leq N$ cities and reduces to the classical problem for $k = N$. Similarly, the problem can be rephrased in terms of the 0/1 knapsack problem as well as a system of linear equations for solution by methods such as the simplex method.

The brute force method for solving an "optimized k-walk" is to simply take every possible *N choose k* subsets and permute every subset's k variables to find the maximum sum of edge weights between consecutive pairs. This method is $O((N \text{ choose } k) * k!)$ or $O(\frac{n!}{(n-k)!})$ and can be used to find a subset of k values which maximizes:

$$\sum_{i=1,k-1} Weight(i, i-1) \tag{1}$$

While this method is guaranteed to find a globally optimum axis ordering, it is NP-complete and therefore computationally intractable for all but the smallest datasets. Even for a dataset with only 63 variables and 7 axes, if calculating the optimum layout from 7! took only 1 millisecond, the entire calculation would still take approximately 6.5 days. For this reason, we developed some simple alternatives to this brute force approach that typically calculate a layout so near to optimal that the difference is neglible.

The greedy algorithm simply finds the largest weight in the graph, represents the two corresponding variables as axes, and then greedily adds a vertex to one of the end vertices until the requested number of vertices is reached. This algorithm is $O(|V|^2 + 2k|V|)$, incrediby simple to code, and obtains surprisingly high performance.

When we calculate an axis layout, it would make sense to keep the pairs with the highest value next to one another. In this pairwise greedy-based algorithm, we begin by finding the k largest weights in the graph. Since each weight has two associated axes, each pair is permuted to find the pairwise ordering which maximizes the sum of weights from the first k consecutive axes. This algorithm is $O(k|V|^2 + k!)$ and typically outperforms the pure greedy approach slightly.

It is useful to constrain the axis ordering to maintain an intuitive interpretation of the final rendering. In the results presented, we do require that axes are non-repeating. This was necessary since certain variables, such as latitude and longitude which exist at every point on the axis, often have high image-space metric evaluations which would lead to interleaving of these axes throughout the PCP. While latitude and longitude could be removed as potential variables, we found they were key axes at identifying equatorial trends resulting from the sun's direct rays.

In the context of time-dependent data, a constraint on the temporal spaces of variable axes may be appropriate. In order to demonstrate our technique in the context of thousands of variables, we take 61 variables per monthly timestep of climate simulation data and treat each subsequent timestep as another set of variables. Multiple variables from other timesteps may be treated independently. However, climate scientists typically think of trends across time rather than across variables within the same timestep. Furthermore, most people are accustomed to visual representations temporal trends unfolding from left to right. In this context, we can limit the axis algorithm to only show axes whose temporal distance between each axis is the same. This creates the ability to see seasonal trends throughout the year as well as how hot the summers get throughout a decade.

While we present only one example for PCP renderings of time-dependent data, there are many contexts in which algorithmic constraints should be imposed. It should be noted that such restrictions can significantly prune the search space for the presented algorithms and can lead to tractability for larger problems.

The system allows multiple ways to determine a set of axes to be used in a layout, by: number of dimensions to visualize, threshold for the most important trends, or graph-based methods. There are also additional parameters which may be modulated to allow for repeats of the same axis or trend.

4.12 Procedural Karyotyping

Throughout this work, we attempt to implement only techniques, algorithms, and visualizations that are general enough to apply across practically any domain while also making the API open enough to allow integration with domain-specific codes. While much if this is discussed in the section below, we felt it appropriate to showcase the one-day integration of a domain-specific visualization. Context-specific requirements of visualization types are sometimes needed to convey information to users in a format that are native to them. For systems genetics data, a karyotype is a common way to display information which simply colors individual genes of interest based on some property within the view of the chromosomes in which those genes reside. In this way, biologists can quickly determine the precise locations that may require further study. In one day we were able to integrate procedural karyotyping based on database data of chromosomal position for any working graph. In this way, one can easily determine how clusters of genes are correlated with one another across genes.

4.13 Synergy

There are many mature graph interaction packages, standards, bioinformatics tools, statistical programming languages and other such workbench platforms that one might consider superior to the developed graph interaction package. While many of the functions listed in this section are exclusive to or can be carried out faster than any other program, my dissertation is in no way competing with them but makes substantial effort to work synergistically with such packages. My current application has several built-in handles and exposes API to several other tools including Cytoscape, R, and Gaggle bioinformatics tools. The package can call scripts for the statistical programming language R to operate or perform statistical analysis on any working graph dynamically as requested by the user. Other graph visualization packages

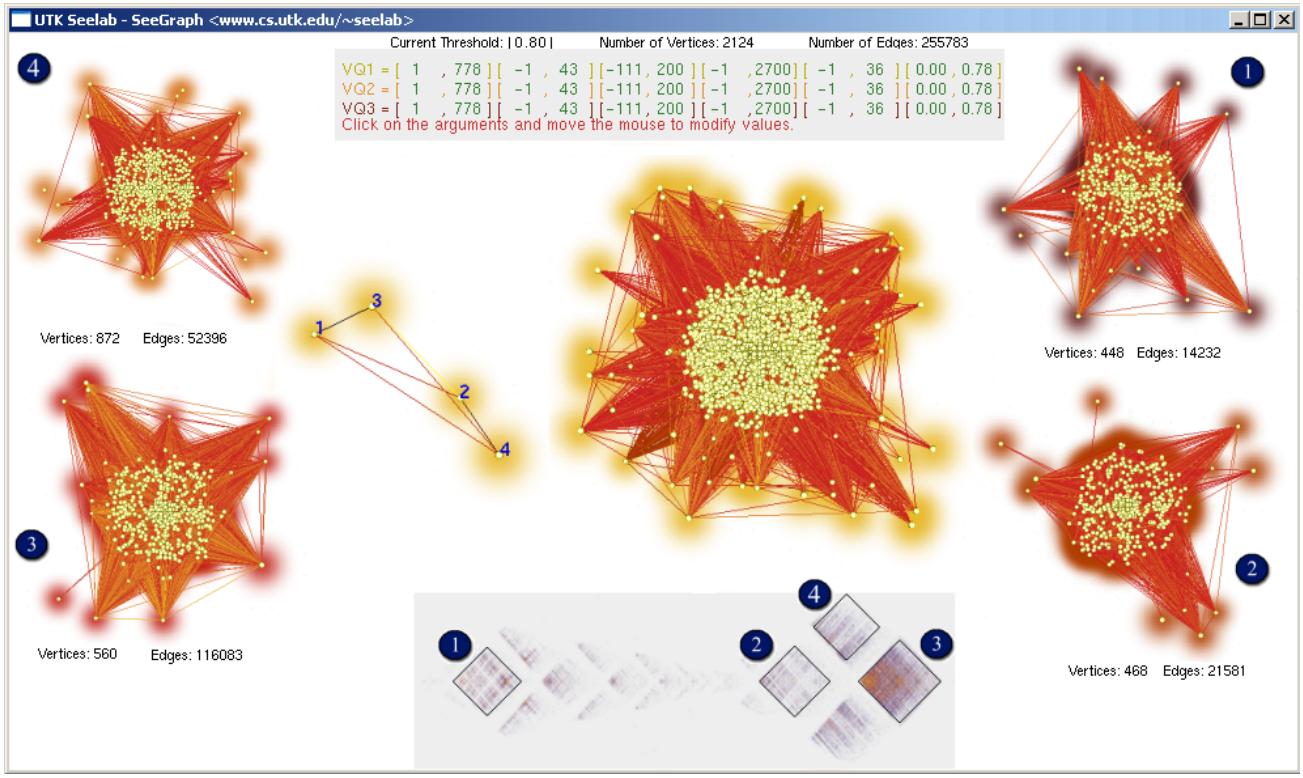

Figure 2: BTD selections (bottom) qualitatively extract gene networks (sides), are rendered using dynamic level-of-detail (center left), and used for template-based classification of entire subgraphs in the original data (center right) for other regulatory mechanisms.

such as Cytoscape can communicate with my application (and vice versa) so that each application can be used for the tasks at which it is most adequate. Minor revisions would be necessary for converting my graph representation to a format recognizable by the Boost Graph Library (BGL) in order to leverage its mature base of efficient graph- processing algorithms. I have created a Java-based "interpreter" goose which acts as a translator between other Gaggle geese and my application. The Gaggle, developed by the Institute for Systems Biology, is essentially a Java interface which, when implemented by a given application, allows data to be broadcast/receive to/from any other such applications (known as geese). In this way, mature software packages can be easily integrated into the "Gaggle" of geese with relatively few lines of code. My interpreter goose is able to accept the data types required by the Gaggle API and send those to my graph interaction package. In this way, my application can easily and interactively share data with other (mostly bioinformatics) applications.

4.14 Scripting

In using the graph visualization package to develop scientifically meaningful results, it was often the case that it was difficult to remember, after several complex steps of iterative refinement, how one arrived at the current meaningful result. Also, entire corpuses of previous analysis or experiments had to be painstakingly recreated as new data became available. In order to alleviate the burden on the user, we introduced a scripting capability which can automate experiments or portions of a pipeline. This script is capable of recreating users mouse movements and key commands to emulate any setting or procedure invoked by the user. The individual operations to be carried out can be easily inspected from the ASCII file which the application will then automatically or interactively recreate on screen with a customizable time delay for quickly generating the requested results or actually seeing results of each step on-screen for demo purposes.

5 Results

5.1 Exemplary Application Areas and Preliminary Results

My initial results have already been published in the IEEE Transactions on Visualization and Computer Graphics, 2008 [40] and several more are currently being written. In the published paper, I have several examples using systems genetics information from gene expression data from mouse cerebellum. In the other papers I am using various data sets from different application areas, including climate data and other bioinformatics data, to demonstrate the capabilities of my system. Below are a few examples of the plethora of results already generated.

5.2 Bioinformatics Data Sets

5.2.1 Overview: Data and Workflow

While early microarray studies emphasized differential expression and comparative analyses, modern applications [27] emphasize correlation of gene expression across large sets of conditions, including environments, time points and tissues. Increasingly, this data is being collected in a context of natural genetic variation [7], where it can be integrated with multiple data sources such as genome sequencing, QTL analysis, and disease relevant phenotypic data. For this application we focus on gene expression analysis conducted with a particular emphasis on those traits related to brain and behavior in the laboratory mouse.

A primary source of covariation in gene expression are single nucleotide polymorphisms (SNPs). Studies in genetical genomics [27] attribute variation and covariation in gene expression to the influence of these differences in DNA sequence. The use of recombinant inbred strains allows biologists to study replicate populations of mice with identical genomes. These populations allow indefinite aggregation of data across studies as new technologies for characterization of mice become available. When traits are assessed across genetically identical individuals, the correlations among traits are assumed to be due to common genetic regulation. By finding and analyzing statistical correlations between genotypes and phenotypes, geneticists hope to discover and interpret the network of causal genotype-phenotype relationships that determine a trait of interest.

Systems genetics research often follows a workflow of finding a gene network, finding regulators of that network, and then performing a focused gene perturbation experiment to determine the role of the associated network on gene expression or function. To begin, a "large" gene correlation graph must be sifted through, to find a highly connected subgraph which corresponds biologically to a gene network in which genes are expressed together, presumably to regulate or subserve a common function. They must then find a small set of causative genes, highly correlated with the subgraph and likely to regulate co-expression, to be used as targets of focused investigation. By manipulating the expression of these genes, the function of the gene network can be determined through observation of expressed phenotypes. Proof of causality occurs when the gene manipulations recapitulate network relations. It should be noted that while standards of "large" are highly application dependent, even graphs with less than 10k vertices exhibit a combinatorial space that is overwhelming and, indeed, presents a rather large and unique problem unlike dealing with volume datasets.

In this section, we showcase results for publicly available biological data which has been the subject of several previous studies. Whole brain mRNA gene expression data was obtained using the Affymetrix U74Av2 microarray for each of the strains in the BXD mouse population and subsequently processed using Robust Multi-Array (RMA) normalization [7]. Throughout the paper, we use Pearson's correlation over 7,443 genes of this dataset as our driving application. The associated database in our system is used for querying, interactive neural network training, and constructing dynamic level-of-detail (LoD) graph features; it contains information relating to typical systems genetic analyses for each gene such as: the chromosome, position (in megabases), paraclique membership and connectivity, broad-sense heritability indices, and QTL mapping [57] locations with p-values from QTL Reaper (sourceforge.net/projects/qtlreaper).

5.2.2 Discovery of Novel Networks

Typical biologists bring a large amount of domain-specific knowledge to their investigative process, for which many tools exist but are usually challenged by purely data driven investigation of networks. One approach to discovery of novel systems genetics networks is the use of computational tools which allow extraction of highly connected subgraphs in a qualitative fashion. By providing block tridiagonalization in which clusters around the diagonal constitute highly related genes, biologists can easily select potentially novel gene networks. Indeed, this $O(|V|^2)$ algorithm quickly extracts dense subgraphs and can be treated as a rough approximation to the NP-complete problem of paraclique enumeration in this context.

In Figure 2, the user has selected four BTD regions and dynamically generated a level-of-detail graph. As is expected, the selection 1 is most unrelated to the rightmost selections and therefore placed far away from the other selections with a negative correlation to selection 3. By selecting LoD vertices 2-4 as examples and 1 as a counterexample, neural network training on entire subgraphs is used to perform template-based search for similar genes in the original data. The resulting classification from the database information is the graph in the right center which has been extracted through the application of domain-specific knowledge in combination with several computational tools (BTD selection, LoD graphs, and NN template matching). This highly-connected subgraph contains genes which are similar to cliques 2 and 3 and bipartite structure 4 selected from the BTD and gives biologists a potentially novel network of two highly-related dense subgraphs to inspect for related function(s). This is currently being applied to create a comprehensive bipartite graph of gene networks (represented by LoD vertices) on one side and all network regulators (network interface genes) on the other. The ability to interactively and qualitatively search across multiple levels of detail has given biologists several tools for which they can not only solve current problems but also find new ways to address more difficult problems.

The central motivation of our system is to enable more in-depth and flexible expert-driven analysis by providing a diverse set of computational tools. However, there are other more established algorithmic tools, such as graph analysis, that are of value to scientific research. Those tools can be leveraged from within our system through our internal B-tree based data structure which allows queries from algorithmic solutions at a rate that facilitates realtime rendering and interaction. In the section below, we present a significant use case that demonstrates parts of our system to discover network interface genes.

5.2.3 Use Case: Discovery of Network Interface Genes

We now demonstrate our application with a biologically significant use case. Once gene networks have been extracted, it is of primary interest to determine the identity of the gene products that regulate these networks. Using either qualitative BTD selection or algorithmic network extraction, the total decomposition of a genetic correlation matrix into disjoint subgraphs can be achieved. With each disjoint subgraph treated as a structure, finding mRNA transcripts with strong correlations to multiple structures would lead to the discovery of "interface genes". These mRNA transcripts regulate expression of genes in those structures, and thereby couple multiple networks and biological processes. The detection of these transcripts and the analysis of their gene's regulatory polymorphisms could lead to the discovery of major genetic modifiers of large biological networks.

Our domain experts have found paraclique extraction to be the most useful and general algorithmic technique. Although choosing the proper threshold is a hard problem in general, by way of repetitive experimentation, statistical and combinatorial analysis, 0.85 is a preferred threshold for extracting paraclique in the dataset at hand [6]. Paracliques with more than ten vertices were extracted, resulting in thirty-seven dense subgraphs, and stored in the systems database. We show the largest and third largest paracliques corresponding to gene networks ready for further study in Figure 3.

In this use case, the challenge is to identify candidate genes that may be the common regulators of expression for a large number of other genes, and to determine which functional biological characteristics or disease related traits the network may be involved with. Some facts are known about this situation: 1) each gene's physical location within the genome is near the location of one of the genotype markers associated with the gene expression level; 2) the interface gene may be a member of one of the dense subgraphs, but it must be highly connected to members of both dense subgraphs; and 3) the biological regulator is likely to be in the same pathway as the genes it regulates.

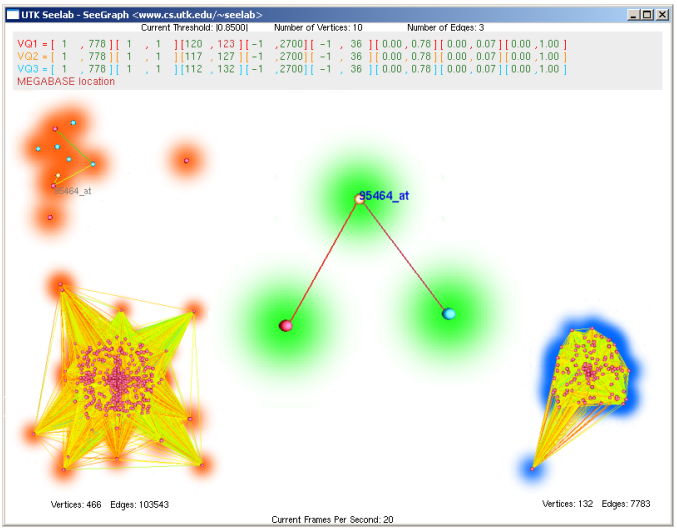

Figure 3: In this screenshot, two gene networks (bottom left and right) have been discovered with a single putatively co-regulating gene as a potential target of knock-out study (center) with proximity information for other potential regulatory genes (top left) undergoing further study. This illustrates the discovery of candidate genes which can affect expression of several genes throughout the genome that play a role in the locomotor response of mice exposed to methamphetamine and cocaine.

By creating a level-of-detail graph in which the edge weight for supernodes is defined as the percent of connectivity to the adjacent gene or supernode, we can use the hot-cold coloring scheme to visually elucidate the correlation between multiple structures. This allows for an intuitive representation of network distance which can allow biologists to identify functional modules and their relationship to each other in forming the pathways underlying specific biological processes. The data is then filtered via threshold to retain only the strongest correlations between individual transcripts and entire networks. Once such a network is constructed, additional queries may be posed that relate to additional candidate genes and their association to the genetic polymorphisms that regulate the network.

In this use case, the technique described above was applied to visually identify a specific gene of interest whose transcript is connected to over 99% of both the largest and third largest paracliques as shown in Figure 3. The gene of interest, Pr1 or Tmem37 (Affymetric probe set ID 95464_at), has thus been implicated as part of the regulatory pathway that co-regulates the two large networks. This gene encodes a voltage-dependent calcium channel gamma subunit-like protein which modulates electrical properties of neuronal cells. It can be hypothesized that the paracliques are related to neuronal activity. Further testing was then accomplished through data export and communication capabilities to specialized bioinformatic pathway analysis tools. By further analysis of expression for the gene of interest, using GeneNetwork.org, it was revealed that its transcript abundance is correlated with stereotyped locomotor behavior induced in BXD mice by drugs such as cocaine [31] or methamphetamine [17].

The next stage in the workflow is to enumerate candidate genes which reside at chromosomal locations nearby the gene of interest's regulatory QTLs. To do this, we must first run an analysis of genotype associations to the expression of interface gene Pr1 in order to identify QTLs that regulate its expression. This analysis, which we have performed using GeneNetwork.org and linear modeling in SAS v9.1, reveals putative regulatory loci at specific locations on chromosomes 1, 2, 6 and 14. By using our tool's integrated data query capability, we can then highlight co-expressed genes that reside in regions provided by the other tools. The candidate genes may help regulate the networks since they are located in close physical proximity to one another, their transcripts are highly correlated to many paraclique members, and the QTL that regulates them also regulates many members of both paracliques. This leads to the identification of a limited number of candidate regulators including Pax3, Lama5, Mkrn2, and Dhrs4.

We have now derived testable hypotheses regarding the mechanisms by which the networks are co-regulated and can validate this new knowledge with in vivo experimentation. It follows from the analysis of co-expression that these two paracliques are controlled by a common genetic regulator. A high genetic correlation implies that the biomolecules represented by the connected vertices have values that are determined by the same genotype. Expression of the interface gene Pr1 can also be correlated with biological function and traced back to a regulatory QTL. Pr1 and the two dense subgraphs have been associated with specific traits measured in BXD mice. A small number of candidate regulators from positions near the regulatory QTL has also been identified. The resulting visualization reveals networks which go from locomotor responses to specific drugs down to the connected molecular pathways which underly them.

While Figure 3 represents only a few steps from a typical workflow, the result of this finding captures overwhelming complexity. As early forays into systems genetic analysis have demonstrated, biological processes such as mammalian behavior involve a complex interplay of expression from many hundreds of genes across multiple chromosomes and biological systems. For this reason, we expect similar linked views of multiple tools to be the norm for systems genetic visualization. The tool presented herein allows users to retain networks and their relationships as well as rapidly isolate genes and subgraphs based on their connectivity. The tools demonstrated represent a flexible and dynamic approach which allows users to scale up from single genes or traits of interest found in web based tools to results of global analyses such as clustering and high-performance combinatorial analyses.

5.3 Climate Research Data Set

The climate research data set used to evaluate my system consists of 71 variables output from a global land surface model running within a general circulation model performing a projection of Earths climate from the year 2000 to 2099. The data set contains monthly averaged data

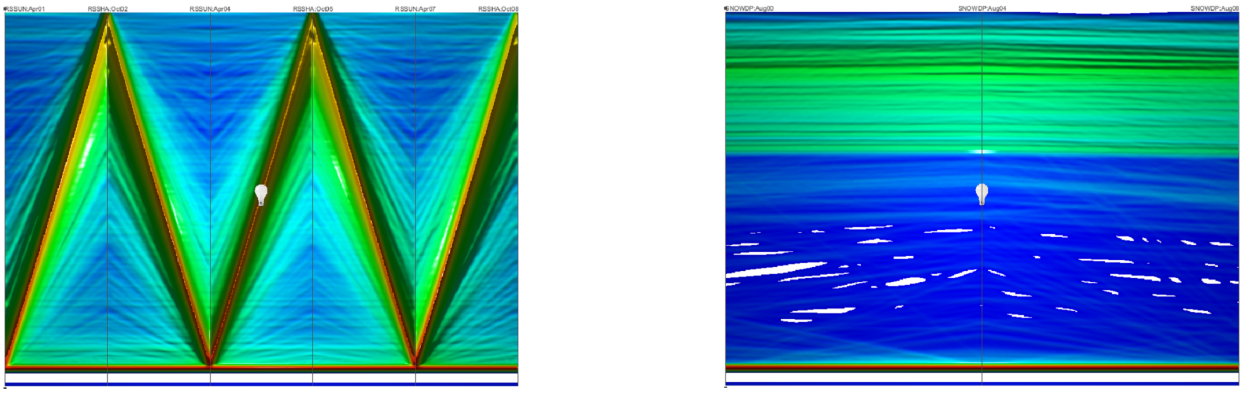

Figure 4: Parallel coordinate plot showcasing intuitive behavior of variables who are most highly inversely correlations (left) and evenly-spaced time-series of variables which exhibit the most highly correlated behavior (right).

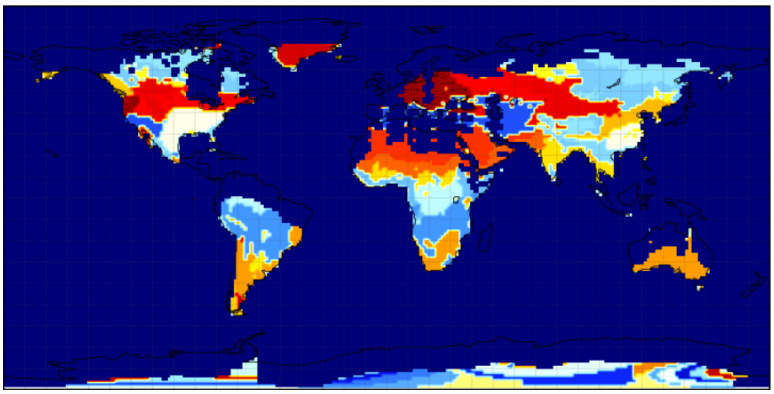

Figure 5: Color-coded areas of the 32 distinct ecoregions extracted using k-means clustering over temperature, precipitation, and soil moisture.

with 12 time steps per year, comprising 1200 time steps in total on a grid of size 256 x 128. For each grid point, we create a polyline in a set of parallel coordinates and use the variable covariance matrix to pick out the most correlated or anti-correlated variables and render them as a parallel coordinate plot using code developed by Chris Johnson.

Figure 4 shows six inversely correlated variables/timepoints detected by our system. This inverse correlation of RSSUN (sunlit leaf stomatal resistance) and RSSHA (shaded leaf stomatal resistance) is intuitive in that it tells the tales of two sides of the same leaf. It should be pointed out that while these relationships are well- known, our system was able to pick these trends out of the data with no prior training or direction. There are several parallel coordinate plots with trends that are currently undergoing investigation in order to determine the likely cause of unusual trends picked out by the system.

Figure 4 shows examples of correlated variables. The system detects the common-sense correlation of a variable with itself for the same month across evenly-spaced consecutive years (exhibits temporal locality). Such detections are common and useful in determining temporal tracking of a variable over time in relations such as global warming.

Based on [22], we use the 3 most common variables of temperature (TBOT), precipitation (RAIN), and soil moisture (SOILLIQ) across all 12 months for 100 years to create a 3600-element feature vector at each grid point. Many of these points have missing values which we estimate based on similar vectors. We then use a parallel k-means clustering algorithm to separate the earth's climate in 32 distinct ecoregions as color-coded in Figure 5.

The meaning of each extracted ecoregion is denoted by the centroid vector for each cluster. Above, we show a graph for each of the 3 variables used to denote the color-coded clusters. In this way, interesting seasonal trends can be visually assessed and further queried. For example, one cluster demonstrates an excessively high soil liquidity which happens to correspond to a cluster present only in Antarctica.

Since the data for all 100 years of a given ecoregion can fit in memory, model-fitting is greatly facilitated. We have used the super-computers at Oak Ridge National Lab to create models for dozens of variables using hundreds of thousands of models for several different model types and is currently undergoing efforts to visualize the accuracy and appropriateness of specific climate models from the space of all possible models.

6 MILESTONES

DATE	TASK
Fall 2008	Conclude work on parallel coordinates axis ordering automation and submit to IEEE Transactions on Visualization and Computer Graphics. Perform literature review in application sciences. Complete a joint paper with climate scientists from Oak Ridge National Laboratory for a journal submission in the field of climate modeling research.
Spring 2009	Finalize research results for dissertation. Prepare final dissertation with intent to defend in April 2008.
April 2009	Defend dissertation. Submit final copy of dissertation to Graduate School.

ACKNOWLEDGMENTS

The author wishes to thank PhD advisor Dr. Jian Huang as well as Dr. Elissa Chesler, Dr. Michael Langston, and Dr. Lynne Parker for their direction on various aspects of this dissertation.

REFERENCES

[1] J. Abello and J. Korn. Mgv: A system for visualizing massive multidigraphs. *IEEE Trans. Visualization and Computer Graphics*, 8(1):21–38, Jan.-Mar. 2002.

[2] J. Abello, F. van Ham, and N. Krishnan. Ask-graphview: A large scale graph visualization system. *IEEE Trans. Visualization and Computer Graphics*, 12(5):669–677, Sep.-Oct. 2006.

[3] O. Abiola, J. M. Angel, P. Avner, A. A. Bachmanov, and J. K. Belknap et al. The nature and identification of quantitative trait loci: a community's view. *Nature Reviews Genetics*, 4(11):911–916, 2003.

[4] D. Auber, Y. Chiricota, F. Jourdan, and G. Melancon. Multiscale visualization of small world networks. In *IEEE Syposium on Information Visualization*, pages 75–81, 2003.

[5] Y. Bai, W. N. Gansterer, and R. C. Ward. Block tridiagonalization of effectively sparse symmetric matrices. *ACM Trans. Math. Softw.*, 30(3):326–352, 2004.

[6] E. J. Chesler and M. A. Langston. Combinatorial genetic regulatory network analysis tools for high throughput transcriptomic data. In *RECOMB Satellite Workshop on Systems Biology and Regulatory Genomics*, pages 150–165, 2005.

[7] E. J. Chesler, L. Lu, S. Shou, Y. Qu, J. Gu, J. Wang, H. C. Hsu, J. D. Mountz, N. E. Baldwin, M. A. Langston, J. B. Hogenesch, D. W. Threadgill, K. F. Manly, and R. W. Williams. Complex trait analysis of gene expression uncovers polygenic and pleiotropic networks that modulate nervous system function. *Nature Genetics*, 37(3):233–242, 2005.

[8] E. J. Chesler, J. Wang, L. Lu, Y. Qu, K. F. Manly, and R. W. Williams. Genetic correlates of gene expression in recombinant inbred strains: a relational model system to explore neurobehavioral phenotypes. *Neuroinformatics*, 1(4):343–357, 2003.

[9] R. W. Doerge. Mapping and analysis of quantitative trait loci in experimental populations. *Nature Reviews Genetics*, 3(1):43–52, 2002.

[10] G. Ellis and A. Dix. Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):717–724, 2006.

[11] M. C. Ferreira and H. Levkowitz. From visual data exploration to visual data mining: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 09(3):378–394, 2003.

[12] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software - Practice and Experience*, 21(11):1129–1164, 1991.

[13] Y. H. Fua, M. O. Ward, and E. A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of IEEE Visualization '99*, pages 43–50, 1999.

[14] D. H. Geschwind. Mice, microarrays, and the genetic diversity of the brain. *Proc. National Academy of Sciences*, 97(20):10676–10678, 2000.

[15] M. Glatter, C. Mollenhour, J. Huang, and J. Gao. Scalable data servers for large multivariate volume visualization. *IEEE Trans. on Visualization and Computer Graphics*, 12(5):1291–1298, 2006.

[16] M. Graham and J. Kennedy. Using curves to enhance parallel coordinate visualisations. In *IV '03: Proceedings of the Seventh International Conference on Information Visualization*, pages 10–16, 2003.

[17] J. E. Grisel, J. K. Belknap, L. A. O'Toole, and M. L. Helms et al. Quantitative trait loci affecting methamphetamine responses in bxd recombinant inbred mouse strains. *Journal of Neuroscience*, 17(2):745–754, 1997.

[18] S. Hachul and M. Junger. The fast multipole multilevel method. *GD '04: Proceedings of the Symposium on Graph Drawing*, pages 286–293, 2004.

[19] W. Hargrove and F. Hoffman. Potential of multivariate quantitative methods for delineation and visualization of ecoregions. *Environmental Management*, 34(5):39–60, 2004.

[20] N. Henry, J. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007.

[21] N. Henry and J. D. Fekete. Matrixexplorer: a dual-representation system to explore social networks. *IEEE Trans. Visualization and Computer Graphics*, 12(5):677–685, Sep.-Oct. 2006.

[22] F. Hoffman, W. Hargrove, D. Erickson, and R. Oglesby. Using clustered climate regimes to analyze and compare predictions from fully coupled general circulation models. *Earth Interactions*, 9(10):1–27, 2005.

[23] Z. Hu, J. Mellor, J. Wu, and C. DeLisi. Visant: an online visualization and analysis tool for biological interaction data. *BMC Bioinformatics*, pages 5–17, 2004.

[24] A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Proceedings of IEEE Visualization'90*, pages 361–378, 1990.

[25] A. Inselberg and B. Dimsdale. Multidimensional lines i: Representation. *SIAM J. Appl. Math.*, 54(2):559–577, 1994.

[26] Alfred Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–91, 1985.

[27] R. C. Jansen and J. P. Nap. Genetical genomics: the added value from segregation. *Trends in Genetics*, 17(7):388–391, 2001.

[28] J. Johansson. Perceiving patterns in parallel coordinates: determining thresholds for identification of relationships. 2008.

[29] J. Johansson, M. Cooper, and M. Jern. 3-dimensional display for clustered multi-relational parallel coordinates. In *IV '05: Proceedings of the Ninth International Conference on Information Visualisation*, pages 188–193, 2005.

[30] J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays. In *INFOVIS '05: Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, page 17, 2005.

[31] B. C. Jones, L. M. Tarantino, L. A. Rodriguez, and C. L. Reed et al. Quantitative-trait loci analysis of cocaine-related behaviours and neurochemistry. *Pharmacogenetics*, 9(5):607–617, 1999.

[32] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1):7–15, 1989.

[33] M. Kreuseler and H. Schumann. A flexible approach for visual data mining. *IEEE Trans. Visualization and Computer Graphics*, 8(1):39–51, Jan.-Mar. 2002.

[34] S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling emerging cyber-communities automatically. In *Proc. 8th Intl World Wide Web Conf.*, 1999.

[35] M. A. Langston, A. D. Perkins, A. M. Saxton, J. A. Scharff, and B. H. Voy. Innovative computational methods for transcriptomic data analysis. In *SAC'06: Proceedings of the 2006 ACM Symposium on Applied Computing*, pages 190–194, New York, NY, USA, 2006. ACM Press.

[36] L. Linsen, J. Locherbach, M. Berth, Jorg Bernhardt, and Dorte Becher. Differential protein expression analysis via liquid-chromatography/mass-spectrometry data visualization. In *IEEE Conference Visualization*, 2005.

[37] R. E. A. Moustafa and E. J. Wegman. On some generalizations of parallel coordinate plots. In *Seeing a Million: A Data Visualization Workshop*, 2002.

[38] C. Mueller, B. Martin, and A. Lumsdaine. A comparison of vertex ordering algorithms for large graph visualization. *International Asia-Pacific Symposium on Visualization*, pages 141–148, 2007.

[39] P. Mutton and P. Rodgers. Spring embedder preprocessing for www visualization. *IEEE Syposium on Information Visualization*, 00:744–749, 2002.

[40] J. New, W. Kendall, J. Huang, and E. Chesler. Dynamic visualization of coexpression in systems genetics data. *IEEE Trans. on Visualization and Computer Graphics*, 14(5):1081–1094, 2008.

[41] A. Noack. An energy model for visual graph clustering. *GD '04: Proceedings of the Symposium on Graph Drawing*, pages 425–436, 2004.

[42] M. Novotny. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):893–900, 2006.

[43] A. Perer and B. Shneiderman. Balancing systematic and flexible exploration of social networks. *IEEE Trans. on Visualization and Computer Graphics*, 12(5):693–700, 2006.

[44] J. Raymond, E. Gardiner, and P. Willett. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45(6):631–644, 2002.

[45] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (2nd Ed)*. Prentice Hall, 2002.

[46] P. Saraiya, Chris North, and Karen Duca. An evaluation of microarray visualization tools for biological insight. In *IEEE Symposium on Information Visualization*, pages 1–8, 2004.

[47] P. T. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 11:2498–504, 2003.

[48] P. T. Shannon, D. J. Reiss, R. Bonneau, and N. S. Baliga. The gaggle: an open-source software system for integrating bioinformatics software and data sources. *BMC Bioinformatics*, 7:176, 2006.

[49] Z. Shen, K. L. Ma, and T. Eliassi-Rad. Visual analysis of large heterogeneous social networks by semantic and structure. *IEEE Trans. on Visualization and Computer Graphics*, 12(6):1427–1439, November/December 2006.

[50] L. Sheng, Z. M. Ozsoyoglu, and G. Ozsoyoglu. A graph query language and its query processing. In *Proceedings of the 15th International Conference on Data Engineering, 23-26 March 1999, Sydney, Austrialia*, pages 572–581. IEEE Computer Society, 1999.

[51] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *IEEE Visual Languages*, number UMCP-CSD CS-TR-3665, pages 336–343, College Park, Maryland 20742, U.S.A., 1996.

[52] B. Shneiderman. Network visualization by semantic substrates. *IEEE Trans. Visualization and Computer Graphics*, 12(5):733–741, Sep.-Oct. 2006.

[53] C. A. Steed, P. J. Fitzpatrick, T. J. Jankun-Kelly, and A. N. Yancey. Practical application of parallel coordinates to hurricane trend analysisy. In *Proceedings of IEEE Visualization'07*, 2007.

[54] M. Tarini, P. Cignoni, and C. Montani. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1237–1244, 2006.

[55] F. Y. Tzeng, E. B. Lum, and K. L. Ma. A novel interface for higher-dimensional classification of volume data. In *Proceedings of IEEE Visualization '03*, pages 505–512, 2003.

[56] F. van Ham and J. van Wijk. Interactive visualization of small world graphs. In *IEEE Syposium on Information Visualization*, pages 199–206, 2004.

[57] J. Wang, R. W. Williams, and K. F. Manly. Webqtl: web-based complex trait analysis. *Neuroinformatics*, 1(4):299–308, 2003.

[58] R. Wegenkittl, H. Loffelmann, and E. Groller. Visualizing the behavior of higher dimensional dynamical systems. In *Proceedings of IEEE Visualization '97*, pages 119–126, 1997.

[59] E. J. Wegman and Q. Luo. High dimensional clustering using parallel coordinates and the grand tour. Technical Report 124, 1996.

[60] J. E. Wegman. Hyperdimensional data analysis using parallel coordinates. *J. Am. Stat. Assn*, 85(411):664–675, 1990.

[61] B. Zhang, S. Kirov, J. Snoddy, and S. Ericson. Genetviz: A gene network visualization system. In *UT-ORNL-KBRIN Bioinformatics Summit*, 2005.

[62] H. Zhou, X. Yuan, B. Chen, and H. Qu. Visual clustering in parallel coordinates. In *Proceedings of IEEE Visualization'07*, 2007.