

Towards Interactive Exploration of Gene Expression Patterns*

Daxin Jiang

Jian Pei

Aidong Zhang

State University of New York at Buffalo, Email: {djiang3, jianpei, azhang}@cse.buffalo.edu

ABSTRACT

Analyzing coherent gene expression patterns is an important task in bioinformatics research and biomedical applications. Recently, various clustering methods have been adapted or proposed to identify clusters of co-expressed genes and recognize coherent expression patterns as the centroids of the clusters. However, the interpretation of co-expressed genes and coherent patterns mainly depends on the domain knowledge, which presents several challenges for coherent pattern mining and cannot be solved by most existing clustering approaches.

In this paper, we introduce an *interactive exploration* system *GeneX* (Gene eXplorer) for mining coherent expression patterns. We develop a novel *coherent pattern index graph* to provide highly confident indications of the existence of coherent patterns. Typical exploration operations are supported based on the index graph. We also provide a bunch of graphical views as the user interface to visualize the data set and facilitate the interactive operations. To help users to interpret and validate the mining results, we design the *gene annotation panel* that connects the genes with some public annotation databases. The experimental results show that our approach is more effective than the state-of-the-art methods in mining real gene expression data sets.

1. INTRODUCTION

DNA microarray technology has made it now possible to monitor simultaneously the expression levels of thousands of genes during important biological processes (e.g., [21]) and across collections of related samples (e.g., [16]). It is often an important task to identify the co-expressed genes and the coherent gene expression patterns from the gene expression data. A group of *co-expressed genes* present similar expression patterns, while a *coherent gene expression pattern* (or coherent pattern in short) characterizes the common trend of expression levels for a group of co-expressed genes. In other words, a coherent gene expression pattern is a “template”, while the expression profiles of the corresponding co-expressed genes yield to the pattern with small

*This research is partly supported by NSF grants DBI-0234895, IIS-0308001 and NIH grant 1 P20 GM067650-01A1. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation or the National Institutes of Health.

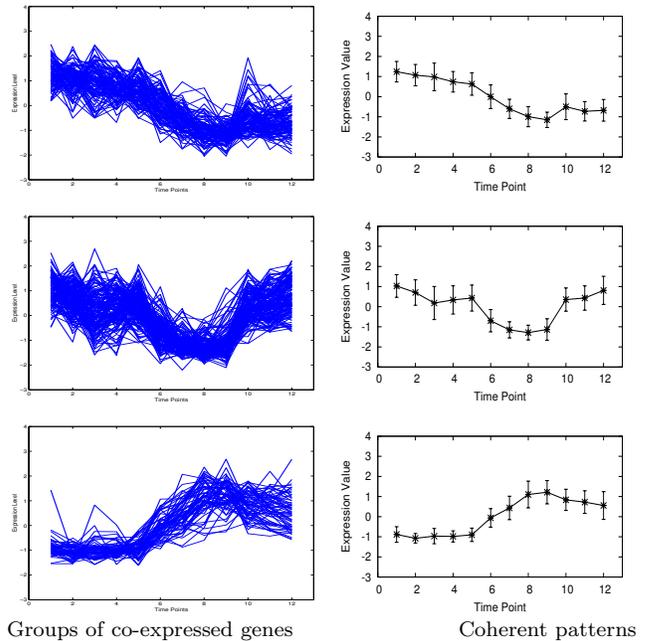


Figure 1: Examples of groups of co-expressed genes and corresponding coherent expression patterns

divergence.

For example, the well known Iyer’s data set [21] records the expression profiles of 517 human genes with respect to a 12-point time-series. A list of 10 groups of co-expressed genes and the corresponding coherent gene expression patterns are also given and well accepted as the *ground truth*. Figure 1 demonstrates three groups of the co-expressed genes in the ground truth (the left column) and the corresponding coherent patterns (the right column). Clearly, the co-expressed genes share the common trends in their expression profiles. The error bar at each time point of the coherent patterns delineates the standard deviation of the expression levels within the corresponding group of co-expressed genes.

Why do we want to find co-expressed genes and coherent gene expression patterns? In practice, co-expressed genes may belong to the same or similar functional categories and indicate co-regulated families [42]. Coherent gene expression patterns may characterize important cellular processes and suggest the regulating mechanism in the cells [33].

In a DNA microarray gene expression data set, there are

usually multiple groups of co-expressed genes and the corresponding coherent patterns. Previous studies (e.g., [2; 11; 21]) suggest that *there is usually a hierarchy of co-expressed genes and coherent patterns in a typical gene expression data set*.

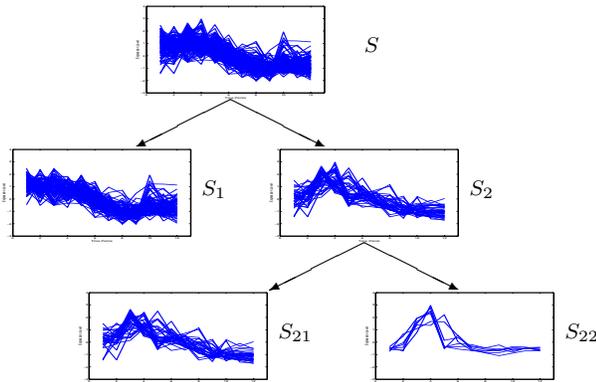


Figure 2: The hierarchy of a co-expressed gene group

For example, Figure 2 shows a group S of co-expressed genes in the Iyer's data set, which can be split into two subgroups, S_1 and S_2 , and the subgroup S_2 can be further split into two sub-subgroups. Genes in each subgroup have more uniform expression profiles, i.e., the pattern is more coherent than the one at the higher level. In other words, at the high levels of the hierarchy, large groups of genes approximately follow some “rough” coherent expression patterns. At the low levels of the hierarchy, the large groups of genes are divided into small subgroups that follow some “fine” coherent expression patterns. The “fine” coherent expression patterns inherit some characteristics from the high level “rough” patterns, and add some specific characteristics.

One subtlety here is that *there is no general and objective standard to identify co-expressed gene groups*. The interpretation of co-expressed genes and coherent patterns mainly depends on the domain knowledge. Typically, two situations may happen in the analysis of gene expression data.

- A microarray experiment often involves thousands of genes. However, only a small subset (e.g., several hundred) of those genes may play important roles in the coherent patterns. As an initial examination, biologists may prefer browsing the “rough” patterns in the data set. They may then choose several patterns of particular interest, and want to decompose them into “finer” patterns in the later analysis. Biologists may have different requirements of “coherence” for different parts of the data set.
- In some cases, whether a group of genes should be further divided depends on the biological hypotheses or domain knowledge. For example, in Figure 2, the subset of genes S_2 can be split into two sub-subsets S_{21} and S_{22} . However, the expression patterns of genes in S_{21} and S_{22} are similar. The critical difference is that the genes in S_{21} are up-regulated at the 3rd time point, while the expression levels of genes in S_{22} are peaking at the 4th time point. Biologically, there could be two hypotheses explaining this phenomenon. On the one hand, the genes in S_{22} are likely up-regulated by the

genes in S_{21} . If so, it is meaningful to split S_2 into S_{21} and S_{22} . On the other hand, the genes in S_{21} and S_{22} may both be up-regulated by some other factors, but the genes in S_{21} respond faster than the the genes in S_{22} . In this case, the genes in S_{21} and S_{22} may have similar functions, and it would be appropriate not to split S_2 . To make this decision, the domain knowledge from biologists is essential.

To find co-expressed genes and discover coherent expression patterns, many clustering algorithms have been developed, including the conventional methods, such as K-means [42], SOM (Self Organizing Map) [41], and the traditional hierarchical approaches (e.g., [11; 2]), as well as some newly proposed methods dedicated to gene expression data, such as CAST [6], CLICK [40] and DHC [22]. Please see Section 2 for a brief survey. Generally, those clustering algorithms partition the set of genes into clusters. Each cluster is considered as a group of co-expressed genes and the coherent expression pattern can be simply the mean (the centroid) of the expression profiles of the genes in that cluster. While previous studies confirm that the clustering algorithms are useful to identify co-expressed gene groups and discover coherent expression patterns, due to the specific characteristics of gene expression data and the special requirements from the biology domain, clustering gene expression data is still facing the following challenges.

- Most clustering algorithms generate clusters at a single level. It is hard to see the inherent hierarchical relationship among the groups of co-expressed genes as well as the coherent patterns.
- Users may have substantially different requirements on the size and the granularity of clusters for different parts/subsets of the data set. However, most clustering algorithms cannot adapt to local structures according to various clustering demands.
- The structure of gene expression data is often complicated. Different clustering algorithms, or even the same clustering algorithm with different parameter values, can generate very different clustering results. Often, only the domain knowledge can tell the quality of the results. In other words, there is no general guideline to choose appropriate parameter values. For each part of the data set, users may want to compare multiple approaches and/or different parameter settings, and choose the one fitting the domain knowledge/hypotheses the best.
- It is typical that users have some domain knowledge about the data set. For example, some genes are already known to have similar functions. Effective integration of domain knowledge may improve the mining results substantially. However, most clustering algorithms are “pure” unsupervised approaches, i.e., they can hardly integrate the domain knowledge.

Can we provide a flexible tool for biologists so that they can interactively unfold the hierarchy of groups of co-expressed genes and derive the corresponding coherent patterns with their domain knowledge? In this paper, we propose a framework *GeneX* for interactive exploration of coherent patterns for gene expression data, and make the following contributions.

- One important observation about the existing clustering methods is that almost all of them try to find the clusters (i.e., groups of co-expressed genes) first according to some *global criteria*, and then derive the coherent patterns based on the clusters. In this study, we propose a novel strategy of *interactive exploration of gene coherent patterns*: it enables the users to interactively explore the hierarchy of coherent expression patterns and find the groups of co-expressed genes according to the coherent patterns. The users' background knowledge can be integrated by the interaction between the users and the system .
- To implement our new strategy, we develop a novel interactive exploration tool, *coherent pattern index graph*, to give users highly confident indications of the existence of coherent patterns.
- To derive a coherent pattern index graph, we need to extract the information about the relations between genes and their groups. We devise an *attraction tree* structure to record the genes in the data set that summarizes the information needed for the interactive exploration.
- We present fast and scalable algorithms to construct the attraction tree and the coherent pattern index graph from the gene expression data set.

The remainder of the paper is organized as follows. In Section 2, we give a short survey on related work. The attraction tree structure and the interactive exploration of coherent patterns using coherent pattern index graph are presented in Section 3. In Section 4, we introduce a system under construction that sets up a general framework to support the interactive exploration. The paper is concluded in Section 5.

2. RELATED WORK: A BRIEF SURVEY

Clustering is the process of grouping data objects into a set of disjoint classes, i.e., *clusters*, so that the objects within a class are similar to each other, while the objects in different classes are dissimilar. To find co-expressed genes and discover coherent expression patterns, various conventional clustering algorithms are adapted, and some new methods dedicated to gene expression data are proposed.

In this section, we give a brief survey of some representative clustering methods applied to gene expression data. Basically, the methods can be classified into three categories, namely, *partition-based approaches*, *hierarchical approaches* and *pattern-based approaches*. Moreover, based on different clustering assumptions and optimization models, we can further divide the partition-based methods into four subcategories: *the K-means algorithm and its derivatives*, *the Self Organizing Map (SOM) algorithm and its extensions*, *the graph-based algorithms* and *the model-based algorithms*. On the other hand, the hierarchical methods can be further divided into *the agglomerative algorithms* and *the divisive algorithms* based on how the hierarchical structure is formed.

2.1 K-means and Its Derivatives

The K-means algorithm [26] is a typical partition-based clustering method. Given a pre-specified number K , the algorithm partitions the data set into K disjoint clusters such

that the sum of the squared distances of objects from their cluster centers is minimized.

In [42], Tavazoie et al. apply the K-means algorithm on the gene expression data collected by Cho et al. [9]. They find that each cluster contains a significant portion of genes with similar functions. Furthermore, by searching the upstream DNA sequences of genes within the same cluster, they extract 18 motifs, which are promising candidates for novel *cis*-regulatory elements.

The K-means algorithm has some drawbacks. The K-means algorithm requires the number of clusters as the parameter. However, it is often hard for a user to pre-determine the number groups of co-expressed genes in the data set. Moreover, gene expression data is typically noisy and contains a large number of outliers. However, the K-means algorithm forces each gene into a cluster, which may cause the centroids of clusters, i.e., the coherent patterns to be reported, dragged astray by the outliers.

To overcome the above two drawbacks, several new algorithms [32; 19; 13] have been proposed. We call them *derivatives of the K-means algorithm*, since essentially they also minimize the overall divergence of objects from their cluster centers. For example, the k-medoids algorithms use an object closest to the center of a cluster as the representative (medoid) such that the total distance between the k selected medoids and the other objects is minimized. The k-medoids algorithms are more robust to the outliers than the k-means algorithm. Another group of algorithms use some thresholds to control the coherence of clusters. For example, Ralf-Herwig et al. [32] introduce two parameters γ and ρ , where γ is the maximal similarity between two separate cluster centroids, and ρ corresponds to the minimal similarity between a data point and its cluster centroid. In [19], the clusters are constrained to have a diameter no larger than d . Motivated by [19], Smet et al. [13] propose a more efficient algorithm *Adapt_Cluster*. Data object x will be assigned to cluster c if the assignment has a higher probability than threshold S .

The clustering process of the above algorithms turns out to be extracting all the clusters with qualified coherence from the data set. Therefore, users do not need to input the number of clusters. On the other hand, with the coherence control, outliers may only end up with trivial clusters, i.e., clusters with very few members. Thus, they will not compromise the significant groups of co-expressed genes and corresponding coherent patterns.

Basically, K-means algorithm and its derivatives require some global parameters, i.e., either the number of clusters or some coherence threshold. The clustering process is like a "black box". There is no intensive interaction between the user and the mining procedures. Therefore, they are not flexible to the local structures of the data set, and can hardly support interactive exploration for coherent expression patterns.

2.2 SOM and Its Extensions

The Self-Organizing Map (SOM) was developed by Kohonen [24], on the basis of a single layered neural network. The data objects, usually of high dimensionality, are mapped onto a set of neurons organized with low dimensional structures, e.g., a two dimensional $p \times q$ grid. Each neuron is associated with a reference vector, and each data point is mapped to the neuron with the "closest" reference vector. During the clustering process, each data object acts as a

training sample that directs the movement of the reference vectors towards the denser areas of the input vector space, so that those reference vectors are trained to fit the distributions of the input data set. When the training is complete, clusters are identified by mapping all data points to the output neurons.

Tamayo et al. [41] apply the SOM algorithm in a study of hematopoietic differentiation. The expression patterns of 1,036 human genes are mapped to a 6×4 SOM. After the clustering process, the genes are organized into biologically relevant clusters that suggest novel hypotheses about hematopoietic differentiation. For example, they report that Cluster 15 captures 154 genes involved in the “differentiation therapy”, which is part of the standard treatment for patients with acute promyelocytic leukemia. Among the 154 genes, some show unexpected regulation patterns. This provides interesting insights into the mechanism of differentiation, which is uncertain yet.

One of the remarkable features of SOM is that it allows users to impose partial structure on the clusters, and arranges similar patterns as neighbors in the output neuron map. This feature facilitates easy visualization and interpretation of the clusters, and thus partly supports the explorative analysis of gene expression patterns.

However, similar to the K-means algorithm, SOM also requires a user to specify the number of clusters, which is typically unknown in advance for gene expression data. Moreover, as pointed out in [18], if the data set is abundant with irrelevant data points, such as genes with invariant patterns, SOM will produce an output where irrelevant data points will populate the vast majority of clusters, while most interesting patterns will be missed since they are collapsed into only a few clusters.

Recently, several new algorithms [18; 43; 27] have been proposed based on the SOM algorithm. Those algorithms can automatically determine the number of clusters and dynamically adapt the map structure to the data distribution. For example, Herrero et al. [18] extend the SOM by a binary tree structure. At first, the tree only contains a root node connecting two neurons. After a training process similar to that of the SOM algorithm, the data set is segregated into two subsets. Then the neuron with less coherence is split in two new neurons. This process is repeated level by level, until all the neurons in the tree satisfy some coherence threshold. Other examples of SOM extensions are Fuzzy Adaptive Resonance Theory (Fuzzy ART) [43] and supervised Network Self-Organized Map (sNet SOM) [27]. In general, they provide some approaches to measure the coherence of a neuron (e.g., *vigilance criterion* in [43] and *grow parameter* in [27]). The output map is adjusted by splitting the existing neurons or adding new neurons into the map, until the coherence of each neuron in the map satisfies a user-specified threshold. SOM is an efficient and robust clustering technique. Hierarchical structure can also be built based on SOM (e.g., SOTA). Moreover, by systematically controlling the split of neurons, SOM can easily adapt to the local structures of the data set. However, the current approaches control the splitting process by some coherence threshold, which is hard for users to specify.

2.3 Graph-based Algorithms

The graph-based algorithms model a gene expression data set as a weighted graph $G(V, E)$, where each gene is repre-

sented by a vertex $v \in V$. For example, in Click [40], a pair of genes $x, y \in V$ are connected by an edge $e(x, y) \in E$ with a weight based on the similarity between the expression patterns of x and y . In CAST [6], the similarity of x and y is mapped to $[0, 1]$, and the edge $e(x, y)$ is created in the graph if the mapped value is 1. The problem of clustering a set of genes is then converted to some classical graph-theoretical problems, such as searching for the *minimum cut* [17; 40], the *minimum spanning tree* [46], or the *maximum cliques* [6] in graph G .

Hartuv et al. [17] propose an algorithm *HCS* (for Highly Connected Subgraph), which recursively splits the weighted graph G into a set of highly connected components along the minimum cut. Each highly connected component is considered as a cluster. Motivated by *HCS*, Shamir et al. present the algorithm *CLICK* (for CLuster Identification via Connectivity Kernels) in [40]. *CLICK* sets up a statistic framework to measure the coherence within a subset of genes and determine the criterion to stop the recursive splitting process.

Ben-Dor et al. [6] introduce the idea of a *corrupted clique graph* data model. The input data set is assumed to come from the underlying cluster structure by “contamination” with random errors. Clustering a dataset is equivalent to identifying the original clique graph from the corrupted version with as few errors as possible. A heuristic algorithm *CAST* (for Cluster Affinity Search Techniques) is developed to iteratively identify the “corrupted cliques” (clusters) once at a time. The coherence of the clusters is controlled by a user-specified parameter, called *affinity threshold*.

In [46], Xu et al. first generate a *Minimum Spanning Tree* (MST) from the weighted graph G of data set X . By removing $(K - 1)$ edges from the generated MST, the data set is partitioned into K clusters. Three alternate algorithms are presented to determine the edges to be removed.

The graph-based algorithms stem from some classical graph-theoretical problems. Although with solid mathematical ground, they may not be suitable for gene expression data without adaptation. For example, in gene expression data, groups of co-expressed genes may be highly connected by a large amount of “intermediate” genes [23]. In this case, the approaches based on minimum spanning tree and minimum cut may lead to clusters including genes with incoherent profiles but highly connected by a series of “intermediate” genes.

2.4 Model-based Algorithms

The model-based clustering approaches (e.g., [12; 48; 15; 28]) provide a statistical framework to model the cluster structure of gene expression data. The data set is assumed to come from a mixture of underlying probability distributions, with each component corresponding to a different cluster. The goal is to estimate the parameters $\Theta = \{\theta_i \mid 1 \leq i \leq k\}$ and $\Gamma = \{\gamma_r^i \mid 1 \leq i \leq k, 1 \leq r \leq n\}$ that maximize the likelihood $L_{mix}(\Theta, \Gamma) = \sum_{i=1}^k \gamma_r^i f_i(x_r | \theta_i)$, where n is the number of data objects, k is the number of components, x_r is a data object (i.e., a gene expression profile), $f_i(x_r | \theta_i)$ is the density function of x_r in component C_i with some unknown set of parameters θ_i , and γ_r^i represents the probability that x_r belongs to C_i . Usually, the parameters Θ and Γ are estimated by the *EM (Expectation-Maximization) algorithm* [10].

Several early studies, including [12; 48; 15], impose a model

of multivariate Gaussian distributions on gene expression data. Although the Gaussian model works well for gene-sample data where the expression levels of genes are measured under a collection of samples, it may not be effective for time-series data (the expression levels of genes are monitored during a continuous series of time points). The reason is that the Gaussian model treats the time points as unordered, static attributes, and ignores the inherent dependency of the gene expression levels over time.

To better describe the gene expression dynamics in time-series data, several new models have been introduced, such as [5; 25; 33; 36]. In [5], each gene expression profile is modelled as a *cubic spline* and each time point influences the overall smooth expression curve. Luan et al. [25] independently develop a similar model of *B-splines*. In [33], Ramoni et al. assume that the time-series follow an *autoregressive model*, where the value of the series at time t is a linear function of the values at several previous time points. Schliep et al. [36] propose a restricted *hidden Markov model* to account for the dependencies in time-series data.

An important advantage of the model-based approaches is that they provide an estimated probability γ_r^i that data object x_r belongs to cluster i . Since it is typical for a gene to participate multiple cellular processes, there may be instances that a single gene has a high correlation with two different clusters. Therefore, the probabilistic feature of model-based clustering is particularly suitable for gene expression data. Moreover, model-based clustering does not need to define a distance (or similarity) between two gene profiles. Instead, the measure of coherence is inherently embedded in the statistical framework. This feature makes the model-based clustering more robust to the dimensionality of attributes (samples or time points).

However, model-based clustering relies on the assumption that the data set fits a specific distribution. This may not be true in many cases [48]. The modelling of gene expression data sets, in particular, is an ongoing effort by many researchers, and, to the best of our knowledge, there is currently no well-established general model for gene expression data. Although some models are reasonably robust when the distribution of real expression data is deviated from the assumption, the best results can only be expected when the real data fits the model well.

2.5 Agglomerative Hierarchical Algorithms

Agglomerative algorithms (i.e., bottom-up approaches) initially regard each data object as an individual cluster, and at each step, merge the closest pair of clusters until all the groups are merged into one cluster.

Eisen et al. [11] apply an agglomerative algorithm called UPGMA (for Unweighed Pair Group Method with Arithmetic Mean) and adopt a method to graphically represent the clustered data set. In this method, each cell of the gene expression matrix is colored according to the measured fluorescence ratio, and the rows of the matrix are re-ordered based on the hierarchical dendrogram structure and a consistent node-ordering rule. After clustering, the original gene expression matrix is represented by a colored table (a *cluster image*) where large contiguous patches of color represent groups of genes that share similar expression patterns over multiple conditions.

Hierarchical clustering not only groups together genes with similar expression pattern but also provides a natural way to

graphically represent the data set. The graphic representation gives users a thorough inspection of the whole data set so that the users can obtain an initial impression of the distribution of data. Eisen's method is much favored by many biologists and has become one of the most widely-used tools in gene expression data analysis [11; 2; 1; 21; 31].

However, as pointed out in previous studies [41; 4], traditional agglomerative clustering algorithms may not be robust to noise. They often make the decisions of merging based on local information and never trace back, i.e., any "bad" decisions made in the initial steps may never get corrected later. In addition, hierarchical clustering only provides a tree structure (called *dendrogram*). There is no standard to decide where to cut the dendrogram to derive clusters. Given a typical gene expression data with thousands of genes, it is hard for users to manually inspect the whole tree.

To make the traditional agglomerative method more robust to noise, Šášík et al. [35] propose a novel approach called *percolation clustering*. In essence, percolation clustering adopts a statistical bootstrap method to merge two data objects (or two subsets of data objects) when they are significantly coherent with each other. In [4], Bar-Joseph et al. replace the traditional binary hierarchical tree with a k -ary tree, where each non-leaf node is allowed to have at most k children. A heuristic algorithm is also presented to construct the k -ary tree, reduce the susceptibility to noise and generate an optimal order for the leaf nodes. The two approaches above make the derived hierarchical tree more robust. However, neither of them indicates how to cut the dendrogram to obtain meaningful clusters.

Seo et al. [39] develop an interactive tool, *Hierarchical Clustering Explorer (HCE)*, to help users derive clusters from the dendrogram. To be specific, HCE visualizes the dendrogram by setting the distance from the root to an internal node N according to the coherence between the two children N_1 and N_2 of N . That is, the more coherent are N_1 and N_2 , the more distant is N from the root. A user can select how to cut the dendrogram in horizontal by dragging the "minimum similarity bar". However, the system of HCE is only a visualization tool that facilitates the inspection of the dendrogram. In other words, it does not provide any hint to where the dendrogram should be cut.

2.6 Divisive Hierarchical Algorithms

The divisive algorithms (i.e., top-down approaches) start with one cluster containing all the data objects. They iteratively split clusters until each cluster contains only one data object or certain stop criterion is met. For divisive approaches, the essential problem is to decide how to split clusters at each step.

Alon et al. [2] apply an algorithm called the *deterministic-annealing algorithm (DAA)* [34] to split the clusters of genes. First, two initial cluster centroids C_j ($j = 1, 2$) are randomly defined. The expression pattern of gene k is represented by a vector \vec{g}_k , and the probability of gene k belonging to cluster j is assigned according to a two-component Gaussian model:

$$P_j(\vec{g}_k) = \frac{e^{(-\beta|\vec{g}_k - C_j|^2)}}{\sum_j e^{(-\beta|\vec{g}_k - C_j|^2)}}$$

The cluster centroids are recalculated as

$$C_j = \frac{\sum_k \vec{g}_k P_j(\vec{g}_k)}{\sum_k P_j(\vec{g}_k)}$$

The *EM algorithm* is then applied to solve P_j and C_j . For $\beta = 0$, there is only one cluster, $C_1 = C_2$. When β is increased in small steps until a threshold is reached, two distinct, converged centroids emerge. The whole data set is recursively split until each cluster contains only one gene.

In [22], Jiang et al. propose a density-based divisive approach, *DHC*, to clustering genes. The basic idea is that a group of co-expressed genes, i.e., a cluster, forms a dense area in the data object space. Data objects (genes) at the “center” of the dense area have high density and carry the coherent pattern shared by other genes within the whole dense area. On the other hand, genes at the peripheral or boundary area of the cluster have low density and will be “attracted” toward the “core” area level by level.

To measure the density of a data object O , the neighborhood of O is firstly discretized into a series of hyper-shells $\{Shell_k^O | 1 \leq k \leq K\}$, such that each hyper-shell occupies exactly a *unit* volume. Then, a weight w_k is assigned to each hyper-shell $Shell_k^O$ according to the distance between $Shell_k^O$ and object O . Last, the number of data objects falling in each hyper-shell (denoted as n_k) is collected and the density is derived by $density(O) = \sum_{k=1}^K w_k \cdot n_k$ (Figure 3).

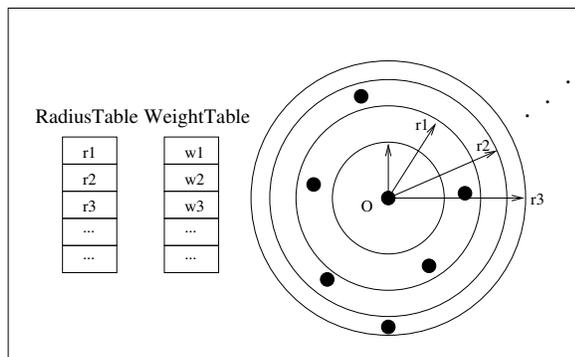


Figure 3: Hypersphere and hyper-shells w.r.t. data object O_i

Once the density of data objects is defined, an *attraction tree* structure is constructed to record the coherence information among genes (See Section 3.1 for details). Algorithm *DHC* splits the data set recursively. At the root, *DHC* regards the whole data set D as one dense area. It searches the attraction tree and identifies the subarea D_1 with the highest density. It then generates two children, D_1 and $(D - D_1)$. *DHC* relies on two parameters to identify the dense subareas: the *similarity threshold* τ and the *minimum number of object threshold* $MinPts$. A group of data objects O form a dense subarea if the data object of the highest density among O has at least $MinPts$ neighbors within a distance of τ . The divisive process is repeated until no leaf node can be further split into two dense subareas.

DHC shares the similar clustering theme with *DAA* [2], *SOTA* [18] and the graph-based approaches *HCS* [17] and *CLICK* [40], but is based on different assumptions of data distribution and uses different models of clustering. The

experimental study shows that the density-based model is particularly robust to noise and well suited to analyze complex cluster structure, e.g., embedded clusters and highly intersected clusters [22]. However, similar to some other approaches, *DHC* splits the data set according to some global thresholds that are hard to set.

2.7 Pattern-based Clustering Algorithms

The clustering algorithms discussed so far are examples of “*global clustering*”. That is, given a data set to be clustered, the attribute space is globally determined and shared by all resulting clusters. However, it is well known in molecular biology that any cellular process may take place only in a subset of the attributes (samples or time points). Recently a series of *pattern-based clustering algorithms* have been proposed to capture coherence exhibited by a subset of genes on a subset of attributes.

In [8], Cheng and Church introduce the concept of *bicluster* to measure the coherence between genes and attributes (either time series or samples). Given a set of genes and a set of conditions, a bicluster is a subset of genes coherent with a subset of attributes. Yang et al. [47] propose a move-based algorithm to find biclusters more efficiently. Both algorithms in [8] and [47] adopt heuristic search approaches, and thus cannot guarantee to find the complete set of biclusters in the data set.

In [45], Wang et al. propose a new model of pattern-based cluster. Given a subset of objects O and a subset of attributes A , pair (O, A) forms a pattern-based cluster if for any pair of objects $x, y \in O$, and any pair of attributes $a, b \in A$, the difference of change of values on attributes a and b between objects x and y is smaller than a threshold δ . In a recent study [30], Pei et al. propose an efficient algorithm, *MaPle*, to mine the complete set of maximal pattern-based clusters.

3. AN INTERACTIVE EXPLORATION APPROACH

In this section, we present an interactive approach to explore gene expression data sets. Our approach is in the following three steps.

Step 1: Extracting distance information. We extract and organize the information about the relationship among genes and their groups into an *attraction tree* data structure. The attraction tree captures the information about the clustering in a data set. Once the attraction tree is built, we do not need to visit the original data set anymore.

Step 2: Indexing coherent patterns and genes. The genes are ordered into an *index list*, such that the genes sharing a coherent pattern stay close to each other in the list. A 2-dimensional *coherent pattern index graph* can be generated. The x-axis is the genes in the order of index list, and the y-axis is the *coherent pattern index value*. If there is a consecutive sublist of genes sharing a coherent pattern, the first gene in the sublist has a significantly high index value (i.e., a “pulse”) and the following genes has a low index value. The coherent pattern index graph provides users an intuitive and informative tool to understand the clustering structure.

Step 3: User interaction. A user may choose the pulses in the coherent pattern index graph, and then the coherent pattern as well as the corresponding co-expressed genes can be derived. The user can recursively examine the selected groups of co-expressed genes as well as its sub-patterns in depth.

We illustrate the technical details of the above steps in the following subsections and elaborate the method using one real case – the Iyer’s data set [21].

3.1 The Attraction Tree

An *attraction tree* records the related information for mining coherent patterns from a gene expression data set. Intuitively, to form clusters, a gene with a high density in a data set can be regarded as “*attracting*” other genes with low density.

The density of a data object reflects the distribution of the other objects in its neighborhood. We adopt the the density definition from a recently proposed method Denclue [20], since it is particularly suitable for drilling down to the subsets of genes. Denclue uses an *influence function* to describe the influence between two objects. For example, the *Gaussian influence function* is defined as follows.

$$f(O_i, O_j) = e^{-\frac{d(O_i, O_j)^2}{2\sigma^2}} \quad (1)$$

where $d(O_i, O_j)$ is the distance between objects O_i and O_j , and σ is a parameter. We will address how to determine an appropriate value for parameter σ in Section 3.3.

We normalize the attribute values of the data objects as follows. Given an object O , for each attribute d , let $O'_d = \frac{O_d - \eta_O}{\sigma_O}$, where η_O and σ_O are the mean and the standard deviation of all the attributes of O , respectively. O' is the normalized object of O .

The similarity and distance between data objects O_i and O_j are defined as the Pearson’s correlation coefficient and Euclidean distance between the transformed objects O'_i and O'_j , respectively. That is,

$$\text{similarity}(O_i, O_j) = \text{pearson}(O'_i, O'_j), \quad (2)$$

and

$$\text{distance}(O_i, O_j) = \text{euclidean}(O'_i, O'_j). \quad (3)$$

Given a data set \mathcal{D} , the density of an object O is the sum of the influences from all the objects in the data set except for itself. That is,

$$\text{density}(O) = \sum_{O_j \in \mathcal{D}, O_j \neq O} f(O, O_j). \quad (4)$$

The attraction between two data objects O_i and O_j ($O_i \neq O_j$) is defined by the influence function (Equation 1). The attraction is said *from* O_i *to* O_j if $\text{density}(O_i) < \text{density}(O_j)$, denoted as $O_i \rightarrow O_j$. In the case that two objects are tie in density, we can artificially assign $O_i \rightarrow O_j$ for ($i < j$). Thus, an object O is attracted by a set of objects $A(O)$ whose density are larger than that of O , i.e., $A(O) = \{O_j | \text{density}(O_j) > \text{density}(O)\}$. The *attractor* of O is the object $O_j \in A(O)$ with the largest attraction to O , i.e.,

$$\text{Attractor}(O) = \arg \max_{O_j \in A(O)} f(O_j, O)$$

According to the influence function, the attractor of an object O is its closest neighbor with a higher density. The only

exception is object O_{hd} whose density is the highest in the data set. We define the attractor of O_{hd} is O_{hd} itself.

The attraction from object O_i to object O_j (i.e., $O_i \rightarrow O_j$) forms a partial order. Based on this partial order, we can derive an *attraction tree* T . The root of the tree is O_{hd} , the object that is the attractor of itself. Each non-root node in the tree corresponds to an object O , whose parent node is its attractor. We define the *weight* of each edge $e(O_i, O_j)$ in the attraction tree T as the similarity between O_i and O_j . The attraction tree has two nice properties. On the one hand, *an attraction tree is self-closed*. That is, a group of objects following the same coherent pattern forms an attraction subtree. Objects following different coherent patterns are not mixed in the same attraction subtree. On the other hand, *the attraction tree is robust to noises*. The root of each attraction subtree has the locally maximal density and represents the coherent pattern in this attraction subtree. Objects closely matching the coherent pattern stay at the high levels of the tree, while noises (or intermediate objects) stay at the low levels of the tree. Even in a data set having a large amount of noises or intermediate objects, since the density of noises or intermediate objects are relatively lower than that of the co-expressed objects, the structure of the high levels of the attraction tree will not be affected and the representatives of coherent patterns will not be deviated by the noises or the intermediate objects.

3.2 Coherent Pattern Index

To plot the genes as well as its probability to be a “leader” in a group of co-expressed genes in a 2-dimensional space, we need to order the genes into a list. An ordering, *index list*, can be devised based on the following three observations.

1. In the attraction tree, an edge connecting a pair of objects O_1 and O_2 has a heavy weight if the two objects follow the same coherent pattern P . Genes connected by those edges should stay close to each other in the list.
2. An edge has a moderate weight if it connects a pair of intermediate objects O_1 and O_2 or a pattern correlated object and an intermediate object. Genes connected by those edges should stay close to each other in the list, too, but not as close as the ones in case 1.
3. The edges connecting a pair of objects O_1 and O_2 following different coherent patterns P_1 and P_2 have light weights. Genes connected by those edges should stay far away in the list.

Based on the above idea, we develop an algorithm to order the genes, as shown in Figure 4. In the algorithm, we maintain an FIFO list, called *processedVertex*, to record the visiting order of the nodes in the attraction tree T . We start from the root of T . All the edges connecting the root with its children are put into a heap, where the edges are sorted in the weight descending order. Then, we iteratively extract the edge with the highest weight from the heap. At this point, the start vertex of the edge must have been processed (Otherwise, the edge could not be put into the heap.) We put the end vertex of the edge *currentVertex* into the list *processedVertex* and put all the edges connecting *currentVertex* and its children into the *edgeHeap*. The loop continues until all of the edges in the tree have been visited. The *processedVertex* is the *index list* of the data genes.

```

Procedure ordering(AttractionTree root){
  // Initialize the processedVertex and the edgeHeap
  processedVertex.add(root)
  for each child ch of root do edgeHeap.insert(edge(root, ch))
  // Iteration
  while ( !edgeHeap.isEmpty() ) do {
    currentEdge = edgeHeap.extract()
    currentVertex = currentEdge.endVertex
    processedVertex.add(currentVertex)
    for each child ch of currentVertex
      edgeheap.insert(edge(currentVertex, ch)) } }

```

Figure 4: The algorithm ordering the genes.

Suppose that we check the genes one by one in the order of index list, if we find a consecutive subsequence S of genes such that the genes in S are much more coherent to their parents in the attraction tree than the genes in the precedent subsequence of S do, then it may strongly suggest that S is the starting segment of a group of co-expressed genes. Remember that in the constructions of attraction tree and index list, co-expressed genes are located in subtrees and thus are arranged as neighbors in the index list. This is the intuition of the design of coherent pattern index. The philosophy here is similar to that in [3].

Then, the problem becomes *how to find those probes – the short subsequences of genes at the beginning of the groups of co-expressed genes.*

For a gene g_i in an index list $g_1 \dots g_n$, let $Sim(g_i)$ be the similarity between g_i and its parent in the attraction tree. $Sim(g_i) = 0$ if $(i < 1)$ or $(i > n)$. Let p be the minimum size of probe as a parameter. We define the coherent pattern index $CPI(g_i)$ as follows.

$$CPI(g_i) = \sum_{j=1}^p Sim(g_{i+j}) - \sum_{j=0}^{p-1} Sim(g_{i-j}) \quad (5)$$

Intuitively, a high coherent pattern index value indicates a strong potential that the gene is the starting one of a group of co-expressed genes. The graph plotting the coherent pattern index values with respect to the index list is called the *coherent pattern index graph*. In particular, from the above definition, the first $(p - 1)$ genes in the index list always bring the first sharp pulse.

Figure 5 is the coherent pattern index graph for Iyer’s data set with $p = 5$. The coherent pattern index graph indicates the existence of coherent patterns clearly.

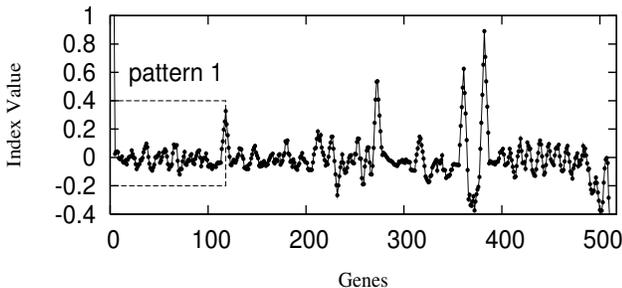


Figure 5: The coherent pattern index graph for the Iyer’s data

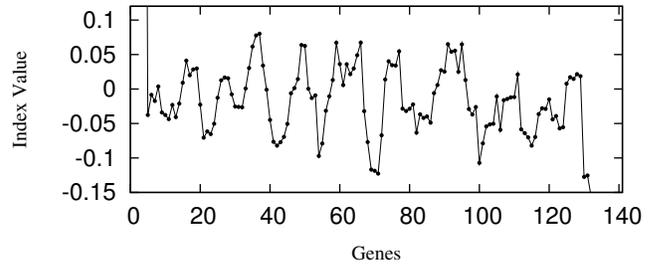


Figure 6: The coherent pattern index graph for a subset of genes in the Iyer’s data set

3.3 Drilling down to Clusters

From Figure 5, we can immediately tell that there are 5 major coherent patterns in the data set. However, *can we further investigate the groups of co-expressed genes following the coherent patterns and identify subgroups of co-expressed genes that follow any derivation patterns?*

Suppose that a user accepts the 5 major coherent patterns reported by the system and clicks on the corresponding peaks in the coherent pattern index graph. The system will split the attraction tree T for the whole data set into 5 exclusive attraction subtrees. Each subtree corresponds to one coherent pattern and the genes following that coherent pattern are gathered in the subtree.

If a user now selects the first subset of genes \mathcal{D}_1 (as shown in Figure 5) and wants to zoom in \mathcal{D}_1 . Figure 6 shows the local coherent pattern index graph for the selected subset of genes. Please note that Figure 6 is not simply extracted from Figure 5 with a higher resolution. Instead, we collect \mathcal{D}_1 from the attraction tree such that only the genes following the coherent pattern are selected. Then, the attraction tree, the index list and the coherent pattern index graph are generated, respectively. Only the genes in the selected subset are considered. The user can specify local parameters (e.g., σ) for computing the influence and density in the subset of genes.

Now, let us discuss how to set the value of σ . According to the influence function (Equation 1), a smaller σ will boost the relative influence of a gene to its neighborhood. A detailed discussion on the effect of σ on the influence calculation can be found in [20]. We use the standard deviation of the pairwise distance between genes as σ . When the data set is split into smaller subsets, the standard deviation will decrease. To lower the computational cost, we use a small sample of the data set to approximate the standard deviation.

3.4 A Case Study on the Iyer’s Data Set

Figure 7 illustrates the exploration process. At the beginning, the coherent pattern index graph for the whole data set indicates five “major” coherent expression patterns. Suppose that the user accepts the indication and asks the system to split the data set accordingly. Some subsets show clear coherent patterns, such as the 2nd and the 4th subsets in the second row of the figure. The others need to be further investigated.

The system generates the coherent pattern index graphs for the remaining subsets, respectively (i.e., the 1st, the 3rd and

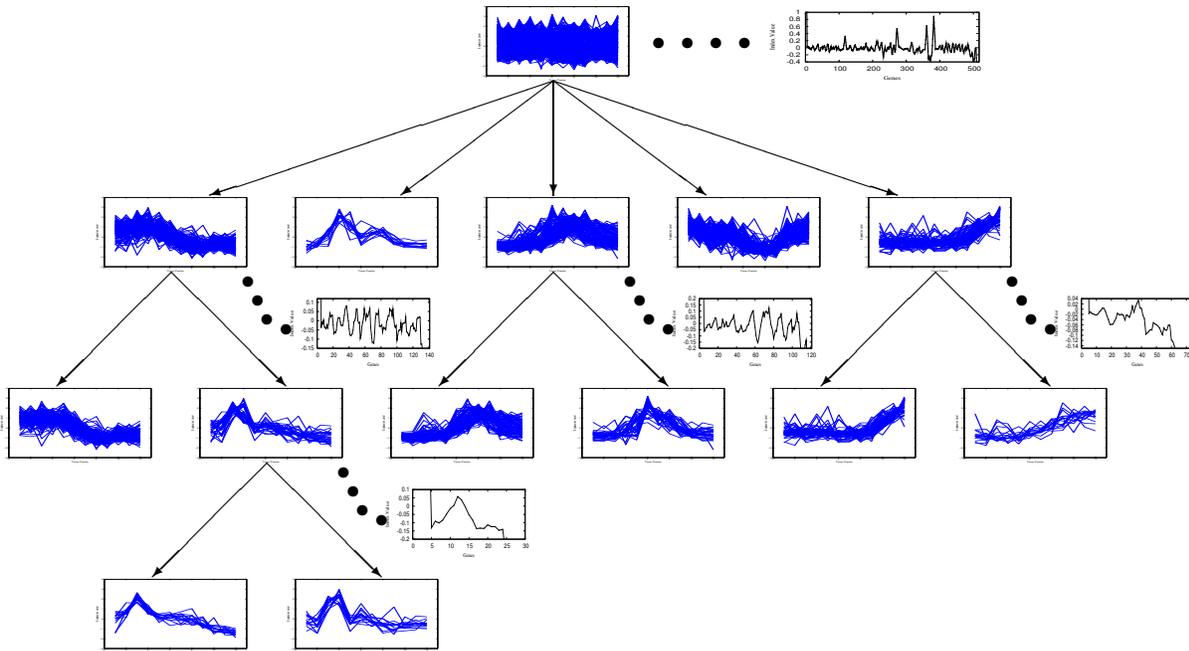


Figure 7: The hierarchy of co-expressed gene groups in the Iyer’s data set

the 5th subsets in the second row of the figure). In each of the coherent pattern index graphs for the subsets, there are multiple significant pulses.

How should we further split the data set and explore the finer patterns? Generally, the highest peak acts as a good signal. The user can ask the system to split the subset according to the highest peak in the graph. If the splitting result is not satisfying, the user can easily “roll back” to the previous level and choose the second highest peak to split the data set. (In our experiment, we just assume the user chooses the highest peak in each subset and split the data set accordingly.) The hierarchy extends to the third level. Such an interactive exploration can be conducted recursively, until the user is satisfied with the patterns and the groups of co-expressed genes.

Comparison with Ground Truth and Other Methods

We also compare the coherent expression patterns discovered by GeneX with the ground truth and the results from some other methods. In our experiment, at each step, GeneX takes the strong pulses to split.

Let $\{P_1, \dots, P_n\}$ be the set of coherent expression patterns in the ground truth and $\{\tilde{P}_1, \dots, \tilde{P}_m\}$ be the set of coherent expression patterns discovered by one mining method. For each pattern P_i in the ground truth, we find the most similar pattern \tilde{P}_j from the mining results, and call \tilde{P}_j match the ground truth pattern P_i . In Figure 8, we list the similarity between the patterns in the ground truth and the matching patterns from various methods. The ground truth coherent expression patterns that are matched with a similarity value larger than 0.9 are highlighted in bold. The numbers in the parenthesis in the first row are the numbers of coherent patterns returned by the methods. Please note that *not every pattern returned by a method is necessarily a pattern in the ground truth*.

The results show that GeneX discovers most of the patterns

Pattern	GeneX (9)	Adapt (11)	CLICK (7)	CAST (9)
1	0.993	0.956	0.884	0.955
2	0.957	0.911	0.991	0.887
3	0.984	0.993	0.994	0.997
4	0.980	0.984	0.883	0.968
5	0.958	0.855	0.868	0.855
6	0.952	0.989	0.970	0.984
7	0.967	0.976	0.990	0.719
8	0.991	0.997	0.914	0.999
9	0.702	0.824	0.844	0.800
10	0.974	0.981	0.976	0.996

Figure 8: Coherent patterns discovered in Iyer’s data set by different approaches.

in the ground truth in the Iyer’s data set, and outperforms all other methods. In particular, we can observe the following.

- GeneX is more accurate than the other methods. On the one hand, GeneX identifies 9 out of the 10 patterns in the ground truth, which is the best record among all the methods. On the other hand, every pattern identified by GeneX turns out to be a pattern in the ground truth. Each of the other methods returns some false patterns. For example, all the other methods further split the co-expressed gene group following pattern 2 in the ground truth into several subsets. Some subsets are merged into other groups of co-expressed genes, while each of the remaining subsets becomes a separate group of co-expressed genes.
- Pattern 5 in the ground truth is *only* identified by GeneX.
- The only pattern in the ground truth that GeneX misses, pattern 9, cannot be identified by any other method, either. The reason is that it is too similar to pattern 6 and thus hard to be separated.

We also test the performance of GeneX on other related aspects, including the effect of the probe size, the scalability and the coherent pattern index graph versus the reachability-plots of Optics [3] on both other real world gene expression data sets and synthetic data sets. The results reported in [22; 23] show that our method is insensible to the setting of probe size, and is scalable w.r.t. the number of genes. The index list is also more effective than the reachability-plot by Optics. Limited by space, we omit the details here.

4. GENEX: THE INTERACTIVE EXPLORATION SYSTEM

Gene expression data sets are often complex in clustering structure. Moreover, biology domain knowledge should be integrated into the mining process. Interactive exploration of gene expression data may lead to more biologically meaningful mining results and substantially improve the interpretability of the results. Therefore, we are building *GeneX*, a gene expression data mining system supporting interactive exploration of coherent gene expression patterns. *GeneX* has the following distinct features:

- It presents a *hierarchy* of the coherent patterns and the corresponding co-expressed genes.
- It uses the *coherent pattern index graph* as an exploration “compass” to illustrate the local clustering structure and guide the users in the exploration.
- It provides the flexibility to *plug-in various components* such as various distance measures and different density definitions. By using various plug-in components, users can compare various approaches and select the one fitting the data set the best.
- It contains a *gene annotation panel*. Using the panel, users can connect the derived groups of co-expressed genes with some public annotation databases, such as the *GO* [7]¹.
- It supports an integrated environment of multiple *graphical views*, such as parallel coordinates, to visualize the data set from different aspects and in different resolutions.

As shown in Section 3.4, our system starts with the root node corresponding to the whole data set. With the exploration and visualization tools provided by the system, the users can unfold the hierarchical structure of coherent patterns inherent in the gene expression data. Each node on the hierarchical tree represents a coherent pattern and the corresponding group of co-expressed genes. To better appreciate the local structure of a specific node, users can choose alternate plug-ins and adjust parameters repeatedly until a satisfying result is achieved.

4.1 The Exploration Operations

Given a specific subset of genes, i.e. a node on the hierarchical tree, our system will generate the coherent pattern index graph as described in Section 3.2. Each pulse in the graph

¹Please also see Gene Ontology, <http://www.geneontology.org>

indicates a potential coherent pattern existing in the subset. Based on the index graph that demonstrates the clustering structure of the gene subset and users’ domain knowledge, the online analytic processing (OLAP) operations can be defined (the theoretical foundation is justified in [29]). *GeneX* supports the exploration operations as follows.

- **Drill-down.** A user can selectively clicks on the pulse(s) in the index graph, and the system will split the subset of genes accordingly. Or, if the user asks the system to split the subset, but does not select any pulses, the system will automatically identify all the significant pulses and split the subset accordingly.
- **Roll-up.** A user can revoke any drill-down operation. The user can either select a node and undo the drill down operation from this node. In this situation, all the children of this node will be deleted. Or, the user can roll up one node *A* to its parent *P*. That is, skip the selection of the pulse in *P*’s index graph that corresponds to *A*, and undo the drill down operation for *P*.
- **Slice.** A user can choose alternate plug-ins, adjust parameters, and compare the mining results. For example, there are various coherence measures [33], e.g., Pearson’s correlation coefficient, delayed correlation, Euclidean distance, Kullback-Leiber distance, and difference density definition [22], e.g., radius-based density, *k*-nearest neighbor density and the density definitions in Denclue and DHC. It is hard to tell which coherence measure or which density definition best fits the underlying local clustering structure. The system allows users to try different approaches and compare the results.

The clustering structure of gene expression data is usually complicated. There may be several different ways to split a specific subset of genes. Different splitting results may correspond to different hypothesis about the gene function and gene regulation. To avoid missing any valuable hypothesis, users may want to try each meaningful split and use their domain knowledge to interpret the splitting results.

4.2 The Gene Annotation Panel

There exists very rich literature about the functions and regulation mechanisms of genes collected during the past studies. Some genes have been well annotated about the molecular function they perform, the biological process they participate in and the cellular component where they locate. It would be very helpful to integrate such domain knowledge into the system.

For example, given a group of co-expressed genes, biologists may postulate the functions of the novel genes in the group based on those well annotated ones in the same group. Moreover, the gene panel can also help validate the mining results. If a group of co-expressed genes scatter into diverse functional categories, biologists may further split this group or try other split path.

To meet this need, we design a *gene annotation panel*. Given a specific node on the hierarchical tree, the panel will display the name and the annotation (if any) for each gene belonging to the node. The gene annotations can be downloaded from some public databases, such as the Gene Ontol-

ogy Consortium (<http://www.geneontology.org>) and MIPS (<http://mips.gsf.de>).

The genes in the panel follow the same order as in the *index list* that we described in Section 3.2. Therefore, the expression profile of the first gene in the panel represents the coherent pattern shared by all the genes in the panel. The similarity measures between the coherent pattern and the expression profile of the individual genes decrease gradually along the list. The genes in the front part of the panel have high coherence with the coherent pattern. Those genes are considered as the co-expressed genes. The genes in the rear part of the panel may have low coherence with the coherent pattern. As explained in Section 3.2, those genes are mostly “intermediate genes”, which do not belong to any co-expressed gene groups. Users can manually select and remove those genes from the panel.

4.3 Graphical Views

In *GeneX*, we provide the following graphical views to visualize the data set from various aspects and in different resolutions.

- A tree structure (e.g., Figure 7) is an overview of the hierarchical clustering structure of the gene expression data set.
- For a given node in the tree structure, the expression profiles of the genes within the node as well as the corresponding coherent pattern can be shown. The coherent pattern is accompanied by an error bars at each experiment condition, which delineates the standard deviation of the expression levels among genes within the node (Figure 1).
- Given a tree structure as the current exploration result, all the leaf nodes in the tree are put into a graph where each vertex in the graph corresponds to a leaf node. Nodes with similar coherent patterns are presented in the graph as neighbors close to each other, while nodes with different coherent patterns are allocated remotely. Such a graph may give good indications for underlying genetic network. For example, a set of nodes that are allocated close to each other may correspond to a pathway.
- In the gene panel, we borrow the visualization method from [11] and represent the expression profile for each gene g_i by a series of colored tile, where each tile c_j corresponds to an experiment condition, and the color suggests the expression level of g_i at c_j .

The four views above serve as the graphical user interface for the interactive exploration. The users not only can have a thorough overview of the clustering structure in the data set, but also can zoom in a view to a specific coherent pattern of particular interest, or even focus on individual genes.

5. DISCUSSION AND CONCLUSIONS

The development of microarray technology provides a great opportunity for functional genomics. Identifying co-expressed genes and coherent expression patterns in gene expression data can help biologists understand the molecular functions of the genes and the regulatory network between the genes.

However, due to the distinct characteristics of gene expression data and the special requirements from the biology domain, mining coherent patterns from gene expression data presents several challenges, which cannot be solved by traditional clustering algorithms.

In this paper, we introduce an *interactive exploration* system *GeneX* for mining coherent expression patterns from gene expression data. The system consists of three major components, the coherent pattern index graph, the gene annotation panel and a bunch of graphical views. Instead of a direct partition of the data set, our system provides users a graphical representation of the clustering structure. Users can try splitting the data set in different ways based on the clustering structure as well as their domain knowledge.

In the future, we plan to extend our system in the following two aspects. On the one hand, besides the gene expression data, recent technological advances have enabled collecting many different types of data at a genome-wide scale, e.g., protein-protein interactions from the *yeast two-hybrid assay* [44] and *mass spectrometry* [14]. A joint mining from combined data with more than one type of the above data may improve the mining results [37; 38]. For example, the complex process of microarray experiment typically brings in a large amount of errors in the resulted gene expression data. Combining gene expression data with protein interaction data may increase the signal-to-noise ration and achieve more robust results. In our interactive system, we provide a coherent pattern index graph to demonstrate the (local) data structure purely based on the expression profiles of genes. In the future, we will study whether we can integrate other types of data and make the index graph more robust and biologically meaningful.

On the other hand, as more and more gene expression data are accumulated (e.g., 134 data sets in the *Stanford Microarray Database*), in addition to analyzing genes and conditions within a single microarray experiment, it is important to answer analytical queries about various “summarizations” over gene expression data sets. For example, biologists may ask “*Which genes of yeast are co-expressed both in the Diauxic Shift Sporulation processes and what are the trends in their expression patterns?*” Our hierarchical approach provides a flexible model to organize the expression patterns in individual data sets. In the future, we will focus on how to store, index and retrieve expression patterns across multiple data sets and support complex analytical queries efficiently.

6. REFERENCES

- [1] Alizadeh, A.A., et al. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, Vol.403:503–511, February 2000.
- [2] Alon U., et al. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide array. *Proc. Natl. Acad. Sci. USA*, Vol. 96(12):6745–6750, June 1999.
- [3] Mihael Ankerst, et al. OPTICS: Ordering Points To Identify the Clustering Structure. *Sigmod*, pages 49–60, 1999.
- [4] Bar-Joseph Z., et al. K-ary clustering with optimal leaf ordering for gene expression data. *Bioinformatics*, 19(9):1070–1078, 2003.
- [5] Bar-Joseph Z., et al. A new approach to analyzing gene expression time series data. In *Proc. 6th Annual International Conference on Computational Molecular Biology*, pages 39–48, 2002.

- [6] Ben-Dor A., Shamir R. and Yakhini Z. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.
- [7] Blake J.A. and Harris M. *Current Protocols in Bioinformatics*, chapter The Gene Ontology Project: Structured vocabularies for molecular biology and their application to genome and expression analysis. Wiley and Sons, Inc., 2003.
- [8] Cheng Y. and Church GM. Biclustering of expression data. *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 8:93–103, 2000.
- [9] Cho, R. J., et al. A Genome-Wide Transcriptional Analysis of the Mitotic Cell Cycle. *Molecular Cell*, Vol. 2(1):65–73, July 1998.
- [10] Dempster A.P., Laird N.M. and Rubin D.B. Maximal Likelihood from Incomplete Data Via the EM Algorithm. *Journal of the Royal Statistical Society, Ser B(39)*:1–38, 1977.
- [11] Eisen, et al.. Cluster Analysis and Display of Genome-wide Expression Patterns. *Proc. Natl. Acad. Sci. USA*, 95(25):14863–14868, December 1998.
- [12] Fraley C. and Raftery A.E. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *The Computer Journal*, 41(8):578–588, 1998.
- [13] Frank De Smet, et al.. Adaptive quality-based clustering of gene expression profiles. *Bioinformatics*, 18:735–746, 2002.
- [14] Gavin A.C., et al. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415(6868):123–4, Jan 2002.
- [15] Ghosh, D. and Chinnaiyan, A.M. Mixture modelling of gene expression data from microarray experiments. *Bioinformatics*, 18:275–286, 2002.
- [16] Golub T. R., et al. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science*, Vol. 286(15):531–537, October 1999.
- [17] Hartuv, Erez and Shamir, Ron. A clustering algorithm based on graph connectivity. *Information Processing Letters*, 76(4–6):175–181, 2000.
- [18] Herrero J., Valencia A. and Dopazo J. A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, 17:126–136, 2001.
- [19] Heyer L.J., Kruglyak S., Yooseph S. Exploring expression data: identification and analysis of coexpressed genes. *Genome Res*, 9(11):1106–1115, 1999.
- [20] Hinneburg, A. and Keim, D.A. An efficient approach to clustering in large multimedia database with noise. *Proc. 4th Int. Con. on Knowledge discovery and data mining*, 1998.
- [21] Iyer, V.R., et al. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999.
- [22] Jiang, D., Pei, J. and Zhang, A. DHC: A Density-based Hierarchical Clustering Method for Time Series Gene Expression Data. In *In Proceedings of the 3rd IEEE Symposium on Bio-informatics and Bio-engineering (BIBE'03)*, Washington, DC, USA, March 10-12 2003.
- [23] Jiang, D., Pei, J. and Zhang, A. Interactive Exploration of Coherent Patterns in Time-Series Gene Expression Data. In *In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, Washington, DC, USA, August 24-27 2003.
- [24] Kohonen T. *Self-Organization and Associative Memory*. Springer-Verlag, Berlin, 1984.
- [25] Luan Y. and Li H. Clustering of time-course gene expression data using a mixed-effects model with B-splines. *Bioinformatics*, 19(4):474–482, 2003.
- [26] MacQueen J.B. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Univ. of California, Berkeley, 1967. Univ. of California Press, Berkeley.
- [27] Mavroudi S., Papadimitriou S. and Bezerianos A. Gene expression data analysis with a dynamically extended self-organized map that exploits class information. *Bioinformatics*, 18:1446–1453, 2002.
- [28] McLachlan, G.J., Bean R.W. and Peel D. A mixture model-based approach to the clustering of microarray expression data. *Bioinformatics*, 18:413–422, 2002.
- [29] Pei, J. A general model for online analytical processing of complex data. In *Proceedings of the 22nd International Conference on Conceptual Modeling (ER'03)*, Chicago, IL, October 13-26 2003.
- [30] Pei J., et al. MaPle: A Fast Algorithm for Maximal Pattern-based Clusterin. *Proceedings of the Third IEEE International Conference on Data Mining (ICDM'03)*, November 19-22 2003.
- [31] Perou C.M., et al. Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. *Proc. Natl. Acad. Sci. USA*, Vol. 96(16):9212–9217, August 1999.
- [32] Ralf-Herwig, et al. Large-Scale Clustering of cDNA-Fingerprinting Data. *Genome Research*, 9:1093–1105, 1999.
- [33] Ramoni M.F., Sebastiani P. and Kohane I.S. Cluster analysis of gene expression dynamics. *PNAS*, 99(14):9121–9126, July 2002.
- [34] Rose, K. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc. IEEE*, 96:2210–2239, 1998.
- [35] Šašik R., et al. Percolation Clustering: A Novel Algorithm Applied to the Clustering of Gene Expression Patterns in Dictyostelium Development. In *Pacific Symposium on Bio-computing*, pages 335–347, 2001.
- [36] Schliep A., Schönhuth A. and Steinhoff C. Using hidden Markov models to analyze gene expression time course data. *Bioinformatics*, 19(Suppl. 1):i255–i263, 2003.
- [37] Segal E., Wang H. and Koller D. Discovering molecular pathways from protein interaction and gene expression data. *Bioinformatics*, 19:i264–i272, 2003.
- [38] Segal E., Yelensky R. and Koller D. Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. *Bioinformatics*, 19:i273–i282, 2003.
- [39] Seo, Jinwook and Shneiderman, Ben. Interactively exploring hierarchical clustering results. *IEEE Computer*, 35(7):80–86, July 2002.
- [40] Shamir R. and Sharan R. Click: A clustering algorithm for gene expression analysis. In *In Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB '00)*. AAAI Press., 2000.
- [41] Tamayo P., et al. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA*, Vol. 96(6):2907–2912, March 1999.
- [42] Tavazoie, S., et al. Systematic determination of genetic network architecture. *Nature Genet*, pages 281–285, 1999.
- [43] Tomida S., et al. Analysis of expression profile using fuzzy adaptive resonance theory. *Bioinformatics*, 18:1073–1083, 2002.
- [44] Uetz P., et al. A comprehensive analysis of protein-protein interactions in *Saccharomyces Cerevisiae*. *Nature*, 403(6770):601–3, Feb 2000.
- [45] Wang H., et al. Clustering by Pattern Similarity in Large Data Sets. In *SIGMOD 2002, Proceedings ACM SIGMOD International Conference on Management of Data*, pages 394–405, 2002.
- [46] Xu Y., Olman V. and Xu D. Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees. *Bioinformatics*, 18:536–545, 2002.
- [47] Yang, J., et al. δ -cluster: Capturing Subspace Correlation in a Large Data Set. In *Proceedings of 18th International Conference on Data Engineering (ICDE 2002)*, pages 517–528, 2002.
- [48] Yeung, K.Y., et al. Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17:977–987, 2001.