CLUSTERING GENE EXPRESSION DATA USING AN EFFECTIVE DISSIMILARITY MEASURE[1]

R. Das,* D.K. Bhattacharyya,* and J.K. Kalita**

Abstract

This paper presents two clustering methods: the first one uses a density-based approach (DGC) and the second one uses a frequent itemset mining approach (FINN). DGC uses regulation information as well as order preserving ranking for identifying relevant clusters in gene expression data. FINN exploits the frequent itemsets and uses a nearest neighbour approach for clustering gene sets. Both the methods use a novel dissimilarity measure discussed in the paper. The clustering methods were experimented in light of real-life datasets and the methods have been established to perform satisfactorily. The methods were also compared with some well-known clustering algorithms and found to perform well in terms of homogeneity, silhouette and the z-score cluster validity measure.

Key Words

Gene expression, dissimilarity measure, clustering, density based, frequent itemset mining, nearest neighbour

1. Introduction

A microarray experiment compares genes from an organism under different developmental time points, conditions or treatments. For an n condition experiment, a single gene has an n-dimensional observation vector known as its gene expression profile. Clustering genes having similar expression profiles is an important research field [1]. Two genes having similar expression profiles have similar functions and/or are co-regulated. To identify genes or samples that have similar expression profiles, appropriate similarity (or dissimilarity) measures are required. Some of the commonly used distance metrics are: Euclidean distance, Pearson's correlation coefficient and Spearman's rank-order correlation coefficient [1]. Euclidean distance imposes a fixed geometrical structure and finds clusters of that shape even if they are not present. It is scale variant and cannot detect negative correlation. Euclidean distance gives the distance between two genes but does not focus on the correlation between them. Pearson's correlation, on the other hand, retains the correlation information between two genes as well as the regulation information. However, as it uses the mean values while computing the correlation between genes, a single outlier can aberrantly affect the result. Spearman's rank correlation is not affected by outliers, however there is information loss w.r.t. regulation because it works on ranked data. Thus, it can also be observed that choosing an appropriate distance measure for gene expression data is a difficult task. In this work, we use our dissimilarity measure which handles the above-mentioned problems and is reported in [2].

1.1 Gene Expression Data Clustering Approaches

Data mining techniques have been widely used in the analysis of gene expression data. According to [1], most data mining algorithms developed for gene expression data deal with the problem of clustering. Clustering identifies subsets of genes that behave similarly along a course of time (conditions, samples, etc.). Genes in the same cluster have similar expression patterns. A large number of clustering techniques have been been reported for analyzing gene expression data, such as partitional clustering such as K-means [3], Fuzzy c-means [4] and self organizing maps (SOMs) [5], hierarchical clustering (unweighted pair group method with arithmetic mean (UPGMA) [6], self-organizing tree algorithm [7]), divisive correlation clustering algorithm (DCCA) [8], density-based clustering [9], [10], shared nearest neighbour-based clustering [11], Model-based methods such as SOMs [5], neural networks [12], graph-theoretic clustering (cluster affinity search techniques (CAST) [13], cluster identification via connectivity kernels (CLICK) [14], E-CAST [15]) and quality threshold clustering [16], genetic algorithms (GAs)-based clustering techniques such as [17], [18]. In [19], a two-stage clustering algorithm for gene expression data (SiMM-TS) is presented. A novel multi-objective genetic

* Department of Computer Science and Engineering, Tezpur University, Tezpur, India; e-mail: {rosy8, dkb}@tezu.ernet.in
** Department of Computer Science, University of Colorado at Colorado Springs, Colorado, USA; e-mail: kalita@eas.uccs.edu
(paper no. 210-1014)
[1]The department is funded by UGC's DRS- Phase I under the SAP.

fuzzy clustering followed by support vector machine classification is presented in [20]. The technique has been found to detect biologically relevant clusters and is dependent on proper tuning of the input parameters.

1.2 Discussion

In this section, we have reviewed a series of approaches to gene clustering. Different clustering algorithms are based on different clustering criteria and the performance of each clustering algorithm may vary greatly with different data sets. For example, K-means or SOM may outperform other approaches if the target data set contains few outliers and the number of clusters in the data set is known, while for a very noisy gene expression data set in which the number of clusters is unknown, CAST, QTC or CLICK may be a better choice. Also, the result of clustering is highly dependent on the choice of an appropriate similarity measure.

From the above discussion, it can be concluded that choosing an appropriate clustering algorithm together with a good proximity measure is of utmost importance. In this paper, we introduce two methods for clustering gene expression data. The first method (DenGeneClus, DGC) clusters the genes from microarray data with high accuracy by exploiting our dissimilarity measure (DBK) [2] and it can also be found to be robust to outliers. The second method (frequent itemset mining approach FINN) attempts to find finer clusters over the gene expression data by integrating nearest neighbour clustering technique with frequent itemset discovery. The advantage of FINN is that it produces finer clustering of the dataset. The advantage of using frequent itemset discovery is that it can capture relations among more than two genes while normal similarity measures can calculate the proximity between only two genes at a time. We have tested both DGC and FINN on several real-life datasets and the results have been found satisfactory. The z-score measure for cluster validity was used to compare our methods with well-known algorithms such as k-means, UPGMA, CLICK, SOM and DCCA and the score obtained by our methods were much higher. Next, we introduce DGC algorithm which is developed based on density-based clustering.

2. DGC

DGC works in two phases which are discussed next.

2.1 Phase I: Normalization and Discretization

The gene expression data is normalized to mean 0 and standard deviation 1. Expression data having low variance across conditions as well as data having more than three fold variation are filtered. The discretization process takes into account the regulation pattern, i.e., up- or down-regulation in each of the conditions for every gene. Let G be the set of all genes and T be the set of all conditions. Let $g_i \in G$ be the ith gene, $t_j \in T$ be the jth condition and $\hbar_{i,j}$ be the expression value of gene g_i at condition t_j. An example of a discretized matrix obtained from Fig. 1 is shown in Fig. 2. The regulation pattern is computed across conditions based on the previous condition value other than the first condition. For the first condition, t_1, its discretized value is directly based on $\hbar_{i,1}$. Discretizing is done using the following two cases:

Case 1: For Condition t_1 (i.e., the First Condition)
The discretized value of gene g_i at condition, t_1.

$$\xi_{i,1} = \begin{cases} 0 \text{ if } \hbar_{i,1} = 0 \\ 1 \text{ if } \hbar_{i,1} > 0 \\ 2 \text{ if } \hbar_{i,1} < 0 \end{cases}$$

Case 2: For the Conditions ($T - t_1$)
The discretized value of gene g_i at t_j:

$$\xi_{i,j+1} = \begin{cases} 0 \text{ if } \hbar_{i,j} = \hbar_{i,j+1} \\ 1 \text{ if } \hbar_{i,j} < \hbar_{i,j+1} \\ 2 \text{ if } \hbar_{i,j} > \hbar_{i,j+1} \end{cases}$$

where $\xi_{i,j}$ is the discretized value of gene g_i at condition t_j $(j = 1, ..(T-1))$. Each gene will now have a regulation pattern ($\wp$) of 0, 1 and 2 across the conditions or time points. Once $\wp$ of each gene is obtained, Phase II, i.e., the clustering process is initiated.

-0.26188	-0.26188	0.662408	1.097367	0.553668
-1.34928	-1.34928	-0.26188	0.009968	-0.2347
0.281818	0.009968	-0.80558	-0.26188	-0.26188
0.825517	0.825517	0.009968	0.281818	0.281818
-1.51239	-1.51239	-1.07743	0.281818	-1.45802
1.097367	1.097367	1.641067	2.184767	1.097367

Figure 1. Example dataset.

2	0	1	1	2
2	0	1	1	2
1	2	2	1	0
1	0	2	1	0
2	0	1	1	2
1	0	1	1	2

Figure 2. Discretized matrix.

2.2 Phase II: Clustering of Genes

The clustering of genes is initiated with the finding of the maximal matching genes with respect to regulation pattern.

2.2.1 A Density-Based Notion of Clusters

Clusters consist of genes having similar expression patterns across conditions, while noise genes are those that do not belong to any of the clusters. The basic idea behind recognizing a cluster is that within each cluster we have a typical density of genes having similar expression patterns which is considerably higher than that outside the cluster. Furthermore, the density within the areas of noise is lower than the density in any of the clusters. In the following, we try to formalize this intuitive notion of clusters and noise in a database G of genes. The key idea is that for each gene of a cluster, the neighbourhood has to contain at least δ number of genes which has similar expression pattern (regPattern). The shape of a neighbourhood is determined by the choice of a distance function for two genes g_i and g_j, denoted by $D(g_i, g_j)$. Note that our approach works with any distance measure and hence there is provision for selecting the appropriate similarity function for some given application. In this paper, we give results for our own dissimilarity measure [2] which has been discussed in detail in the previous section.

2.2.2 Basis of the Clustering Approach

The three fundamental bases on which the clustering technique (DGC) is designed are:

(i) Regulation Matching: For a particular gene g_i, the maximal matching regulation pattern (MMRP) is found. All those genes having the same MMRP w.r.t. g_i are grouped into the same cluster.

(ii) Order Preserving: We follow order preservation based on [21] in the following way. For a condition set $t \subset T$ and a gene $g_i \in G$, t can be ordered in a way so that the expression values are ordered in ascending order. By order ranking, we search for the expression levels of genes within a cluster which induce ordering of the experiments (conditions). Such a pattern might arise, for example, if the experiments in t represent distinct stages in the progress of a disease or in a cellular process and the expression levels of all genes in a cluster vary across the stages in the same way [21].

(iii) Proximity: The proximity between any two genes g_i and g_j is given by $D(g_i, g_j)$ where D is any proximity measure like Euclidean distance, Pearson's correlation, *etc.*

The identification of clusters is based on the following definitions. The definitions are given based on the density notion available in [22].

Definition 1. *Matching: Let $\wp_{g_i}$ and $\wp_{g_j}$ be the regulation patterns of two genes g_i and g_j. Then, the matching (M) between g_i and g_j will be given by the number of agreements (No_Agreements)* (i.e., *the number of condition-wise common regulation values excluding condition 1) between the two regulation patterns, i.e.,*

$$M(g_i, g_j) = No\_Agreements(\wp_{g_i}, \wp_{g_j}).$$

Definition 2. *Maximal Matching: Gene g_i is referred to as maximally matched (MM) with gene g_j if the number of agreements between ($\wp_{g_i}$, $\wp_{g_j}$) is $\geq \delta$ where $g_j \in G - \{g_i\}$ and G are sets of genes.*

Definition 3. *MMRP: If a gene g_i maximally matches with say, gene g_j, then the regulation pattern $\wp'_{g_i}$ and $\wp'_{g_j}$ formed by taking the subset of conditions where both $\wp_{g_i}$ and $\wp_{g_j}$ match is referred to as the MMRP for g_i and g_j.*

MMRP of genes g_i and g_j is computed as follows:

$$\wp'_{g_i} = \wp'_{g_j} = \begin{cases} 1 \text{ if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 1 \\ 0 \text{ if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 0 \\ 2 \text{ if } \wp_{g_{i,t}} = \wp_{g_{j,t}} = 2 \\ x \text{ otherwise.} \end{cases}$$

Here $(t = 2, 3, \cdots, T-1)$ refers to the conditions.

Each gene will have a rank which will give the permutation order of that gene across conditions $t \subset T$. The rank is calculated according to the expression values of a gene across conditions, i.e., the elements of the rank pattern are given by their ranking in ascending order of their expression values. The rank of a gene is calculated as follows: (i) For a gene g_i, find $\wp'_{g_i}$ and (ii) Rank g_i in ascending order according to the expression values where $\wp'_{g_i,t} \neq x$. For ease of understanding of the rank computation, the example given in Fig. 1 is referred. Here, the rows represent the genes $g_1, g_2, \cdots, g_6$ and the columns represent the corresponding conditions (excluding condition 1 as stated before),

$$\begin{array}{lllll} \wp_{g_1} = 2 & 0 & 1 & 1 & 2 \\ \wp_{g_3} = 1 & 2 & 2 & 1 & 0 \\ \wp_{g_5} = 2 & 0 & 1 & 1 & 2 \end{array} \qquad \begin{array}{lllll} \wp_{g_2} = 2 & 0 & 1 & 1 & 2 \\ \wp_{g_4} = 1 & 0 & 2 & 1 & 0 \\ \wp_{g_6} = 1 & 0 & 1 & 1 & 2. \end{array}$$

Matching among pairs of genes are:

$$\begin{array}{lll} M(g_1, g_2) = 4 & M(g_1, g_3) = 1 & M(g_1, g_4) = 2 \\ M(g_1, g_5) = 4 & M(g_1, g_6) = 4 & M(g_2, g_3) = 1 \\ M(g_2, g_4) = 2 & M(g_2, g_5) = 4 & M(g_2, g_6) = 4 \\ M(g_3, g_4) = 3 & M(g_3, g_5) = 1 & M(g_3, g_6) = 1 \\ M(g_4, g_5) = 2 & M(g_4, g_6) = 2 & M(g_5, g_6) = 4. \end{array}$$

Suppose $\delta = 3$, then Maximal Matching of pairs of genes are:

$$\begin{array}{lll} MM(g_1, g_2) = 4 & MM(g_1, g_5) = 4 & MM(g_1, g_6) = 4 \\ & MM(g_2, g_5) = 4 & MM(g_2, g_6) = 4 \\ & MM(g_3, g_4) = 3 & MM(g_5, g_6) = 4 \end{array}$$

Thus, MMRP is:

$$\wp'_{g_1} = 0 \quad 1 \quad 1 \quad 2 \qquad \wp'_{g_2} = 0 \quad 1 \quad 1 \quad 2$$
$$\wp'_{g_5} = 0 \quad 1 \quad 1 \quad 2 \qquad \wp'_{g_6} = 0 \quad 1 \quad 1 \quad 2$$
$$\wp'_{g_3} = x \quad 2 \quad 1 \quad 0 \qquad \wp'_{g_4} = x \quad 2 \quad 1 \quad 0.$$

From the above example, it is clear that the MMRP of g_1, g_2, g_5, and g_6 are same, as well as the MMRP of g_3 and g_4 are same.

Genes 1, 2, 5, and 6 have the MMRP over conditions 2, 3, 4, 5. Rank order over these four conditions are computed w.r.t. their expression values ($\hbar_{i,j}$, $i = 1, 2, 5, 6$ and $j = 2, 3, 4, 5$, where i refers to gene i and j refers to condition j) and ranks as follows:

$$Rank(g_1) = 1 \quad 3 \quad 4 \quad 2 \qquad Rank(g_2) = 1 \quad 2 \quad 3 \quad 4$$
$$Rank(g_5) = 1 \quad 3 \quad 4 \quad 2 \qquad Rank(g_6) = 1 \quad 2 \quad 3 \quad 1.$$

Similarly, genes 3 and 4 can be found to have the MMRP over Conditions 3, 4, 5 and ranks obtained are as follows:

$$Rank(g_3) = 1 \quad 2 \quad 2 \qquad Rank(g_4) = 1 \quad 2 \quad 2.$$

Definition 4. *θ-neighbourhood: The θ-neighbourhood of a gene g_i, denoted by $N_\theta(g_i)$ is defined by, $N_\theta(g_i) = g_i \in G$, such that $D(g_i, g_j) \leq \theta$, where, D may be any distance measure such as Euclidean, Pearson's correlation, our dissimilarity measure, etc.*

Definition 5. *Core Gene: A gene g_i is said to be a core gene w.r.t. θ if there is at least one gene g_j such that: (i) $g_j \in N_\theta(g_i)$, (ii) $|N_\theta(g_i)| \geq \sigma$, (iii) $Rank(g_i) \approx Rank(g_j)$ and (iv) $\wp'_{g_i} \approx \wp'_{g_j}$.*

where σ is a user-defined threshold for the minimum number of genes in the θ-neighbourhood of g_i.

Definition 6. *Directly Reachable Gene: A gene g_i is directly reachable from gene g_j w.r.t. θ if (i) g_j is a core gene, (ii) $g_i \in N_\theta(g_j)$ and (iii) $\wp'_{g_i} \approx \wp'_{g_j}$.*

Directly reachable relation of a gene is symmetric for pairs of core genes. However, in case of a pair of core and non-core genes, it may not be valid.

Definition 7. *Reachable Gene: A gene p is said to be reachable from gene q w.r.t. θ if there is a chain of genes $P_1, P_2, \cdots, P_n$, where $P_1 = q$, $P_n = p$ such that P_{i+1} is directly reachable from P_i.*

Thus, reachability relation is a canonical extension of direct reachability [22]. This relation is transitive, but is not symmetric. However, over this gene expression domain reachability is symmetric for core genes.

Definition 8. *Density Connected Genes: A gene g_i is said to be connected to another gene g_j if both g_i and g_j are reachable from another gene g_k w.r.t. θ.*

Connectivity is a symmetric relation. For reachable genes, the relation of connectivity is also reflexive.

Definition 9. *Cluster: A cluster C w.r.t. θ is a non-empty subset of G and $|C| \geq \sigma$ satisfying the following conditions: (i) $\forall g_i, g_j$ if $g_i \in C$ and g_j is reachable from g_i w.r.t. θ then, $g_j \in C$ (reachability) and (ii) $\forall g_i, g_j \in C : g_i$ is density connected to g_i w.r.t. θ (connectivity).*

Therefore, a cluster can be defined as a set of reachable and/or connected genes.

Definition 10. *Noise: Let C be the set of clusters of the dataset G w.r.t. parameter θ. Noise is defined as the set of genes not belonging to any cluster $C_i \in C$. In other words, noise $= \{g_i \in G \mid \forall i : g_i \notin C_i\}$. Also, a gene g_i is said to be a noise gene if it does not satisfy the θ-neighbourhood condition, i.e., $|N_\theta(g_i)| < \sigma$.*

Note that in this paper, any cluster C_i w.r.t. θ contains at least two genes (i.e., $\sigma = 2$) to satisfy the core gene condition.

cluster_creation()
Precondition: All genes in D_G are unclassified

```
FOR all g_i ∈ G do
  Compute ℘(g_i);
END FOR
FOR i = 0 to G do
  IF g_i.classified ≠ CLASSIFIED then
    Compute ℘'(g_i) & Rank(g_i);
    IF get_core(g_i) == TRUE then
      expand_cluster(g_i, cluster_id);
      cluster_id = cluster_id + 1;
    END IF
  END IF
END FOR
```

Figure 3. Algorithm for Cluster Formation.

2.2.3 Finding the Maximal Coherent Clusters

Cluster identification starts with an arbitrary gene and finds the MMRP ($\wp'$) with the other unclassified genes (Fig. 3). For regulation pattern matching, two genes are matched w.r.t. the regulation across the conditions starting from Condition 2. Condition 1 is not considered because it has no previous condition. If the arbitrary gene is a core gene then cluster expansion proceeds with this core gene and finding reachable and connected genes from this core gene. All reachable and connected genes in a particular iteration of the clustering process are grouped into the same cluster. The process then recursively continues until all genes are classified. This expansion process is given in Fig. 4. Here, *get_core*(g_i) is a function which checks the core condition as stated in Definition 5. Assuming G is a set of genes and C is a set of clusters, following lemmas are trivial to DGC. Intuitively they state, given the parameter θ we can discover a cluster in a two-step approach. First, an arbitrary gene is chosen as the seed which satisfies the core gene condition. Second, all genes reachable from the seed are retrieved. These two steps result in a cluster containing the seed.

Lemma 1. *Let g_i be a core gene in G in C_i (where $C_i \in C$) and let g_j be any gene $\in C_i$. Then g_j is reachable from g_i w.r.t. θ.*

expand_cluster(g_i, cluster_id)

```
IF g_i.classified == CLASSIFIED then
  RETURN;
END IF
g_i.classified = CLASSIFIED;
g_i.cluster_id = cluster_id;
FOR j = 0 to G do
  IF g_i ≠ g_j
    IF ℘'_{g_i} ≈ ℘'_{g_j} && g_j ∈ N_θ(g_i) then
      IF get_core(g_j) == TRUE then
        expand_cluster(g_j, cluster_id);
      END IF
      g_j.classified = CLASSIFIED;
      g_j.cluster_id = cluster_id;
    END IF
  END IF
END FOR
```

Figure 4. Algorithm cluster expansion.

Lemma 2. *Assume genes g_i, $g_j \in G$ and let C_1, C_2 be two clusters, where $g_i \in C_1$ and $g_j \in C_2$, then g_i and g_j are not connected.*

Lemma 3. *Assume gene $g_i \in G$ and C be the set of all clusters. If g_i is a noise gene, then $g_i \notin C$.*

The following observations have been made in DGC:

Observation 1. Any core gene $g_i \in C_k$ (where $i = 1, 2, \cdots, m$ and C_k is a cluster) w.r.t. θ have the same MMRP and Rank with the other core genes in C_k.

Observation 2. All genes in a cluster C_k have same MMRP with the core gene(s) $\in C_k$.

The clustering result of DGC using our dissimilarity measure is reported in Section A.

3. Frequent Itemset Mining and Nearest Neighbour Clustering (FINN)

FINN works in three phases. In the first phase, the gene expression data G_D is transformed into a 0–1 transaction matrix. The second phase finds the maximal frequent itemset using a frequent itemset mining algorithm such as Apriori or FP-tree. The third phase is dedicated to the task of clustering using a shared nearest neighbour-based approach. Below, we discuss these phases in detail.

3.1 Phase I: Transformation From Gene Expression Matrix to Transaction Matrix

The gene expression dataset is a $G \times T$ matrix of expression values where G is the number of rows (genes) and T is the number of columns (time points) as shown in (1). Using DBK between the genes across time series is used to build a $G \times G$ dissimilarity matrix for the whole dataset. We introduce some definitions as we proceed with the description of our method.

Definition 11. *Nearest Neighbour of a gene: A gene g_i is the nearest neighbour of a gene g_j if $D(g_i, g_j) \leq \theta_1$, where θ_1 is a dissimilarity threshold and D is our dissimilarity measure (DBK) discussed before.*

From the nearest neighbour lists, we build the $G \times G$ gene-gene transaction matrix, T_G, of zeroes and ones (2). For each gene g_i, a 0−1-pattern of size G is obtained, where "1" is set if a gene g_j is neighbour of g_i and 0 otherwise, as given in 3:

$$G_D = \begin{bmatrix} a_{11}\ a_{12} \cdots\cdots a_{1T} \\ a_{21}\ a_{22} \cdots\cdots a_{2T} \\ \vdots \\ a_{G1}\ a_{G2} \cdots\cdots a_{GT} \end{bmatrix} \tag{1}$$

$$T_G = \begin{bmatrix} t_{11}\ t_{12} \cdots\cdots t_{1G} \\ t_{21}\ t_{22} \cdots\cdots t_{2G} \\ \vdots \\ t_{G1}\ t_{G2} \cdots\cdots t_{GG} \end{bmatrix} \tag{2}$$

$$T_G = t_{ij} = \begin{cases} 1 \text{ if } D(g_i, g_j) \leq \theta_1, \text{ where } i = 1, 2, \cdots, p; \\ \qquad\qquad\qquad\qquad\qquad j = 1, 2, \cdots, p \\ 0 \text{ otherwise.} \end{cases} \tag{3}$$

Pruning

Those transactions are pruned which satisfy the following conditions:

i. In the transaction matrix, the value of t_{ij}, where $i = j$ is set to zero because the same gene does not contribute to frequent itemset generation.
ii. In the transaction matrix if for a particular row i the value of t_{ij} across all j conditions are zero and the same applies for column j and all i rows, then that ith row and jth column both are discarded.

These two steps reduce the size of the transaction matrix considerably.

Phase II now uses this matrix, T_G, to calculate the frequent itemset using FP-tree.

3.2 Phase II: Maximal Frequent Itemset Generation

In this phase, we use the FP-tree to generate the maximal frequent itemset(s) (MFIS) at support threshold s%. The gene-gene $G \times G$ transaction matrix, T_G is fed as input along with the user-defined support threshold to get the frequent itemsets. The MFIS obtained from this phase gives us the set of core genes. The identification of core genes is done as follows:

- If only one MFIS is obtained at s% support, the genes within that set become the set of core genes for a particular cluster.
- If more than one MFIS is obtained at s% support and there is a chain of genes (items) from one MFIS to the other, the genes are merged together into the set of core genes for a particular cluster.
- If more than one MFIS is obtained at s% support and there is no chain of genes (items) from one MFIS to

the other, each MFIS will give the set of core genes for different clusters.

This set of core genes contain the seeds for cluster expansion which gives the core clustering of the dataset. Different clustering approaches such as hierarchical or density-based clustering can be applied on these core genes to get the final cluster. The next phase gives a detailed overview of the clustering process.

The following definitions provide the foundation for the clustering process.

Definition 12. *Density of a gene: The density of a gene g_i is the number of nearest neighbours of that gene in the gene–gene transaction matrix, T_G.*

$$Density(g_i) = \sum_{j=1}^{G} t_{ij}, \quad \text{where } t_{ij} = 1 \tag{4}$$

Definition 13. *Core genes: The set of core genes Cr can be defined as the set of MFIS, i.e., the maximal frequent itemset(s) generated by the FP-tree algorithm. For a set of MFIS of cardinality k it is formalized into three cases as given below. Cr is formed by either one, two or three cases or a combination of them:*

1. *if $k=1$, $Cr=\{MFIS\}$,*
2. *if $k>1$ and $MFIS_i \bigcap MFIS_j \neq \phi$, $Cr=\{\bigcup_{i=1}^{k} MFIS\}$, where $j \neq i$, and $j=k-i$,*
3. *if $k>1$ and $MFIS_i \bigcap MFIS_j = \phi$,, $Cr=\{MFIS_i, MFIS_j, \cdots, MFIS_k\}$, where $j \neq i$, and $j=k-i$.*

Each MFIS will give the core genes of a particular cluster.

Definition 14. *Shared Neighbours: Assume $Cr=\{MFIS_1, \cdots, MFIS_k\}$ is the set of core genes. A gene g_k is said to be the shared neighbour of each of the genes $\{g_a, \cdots, g_m\}$ in $MFIS_i$, i.e., $sn(MFIS_i, g_k)$, if it satisfies the following:*

$$sn(MFIS_i, g_k) = \begin{array}{l} D(g_a, g_k) \leq \beta \ \wedge D(g_b, g_k) \\ \leq \beta \wedge \cdots \wedge D(g_m, g_k) \leq \beta \end{array} \tag{5}$$

where β is the shared neighbour threshold.

Definition 15. *Cluster: A cluster C_i can be defined as the set of all shared neighbours of $MFIS_i$, i.e.,*

$C_i = \bigcup_{j=1}^{p} sn(MFIS_i, g_j)$, where, $sn(MFIS_i, g_j)$ is the set of p shared neighbors of $\{g_a, \cdots, g_m\} \in MFIS_i$.

Definition 16. *Noise genes: A gene g_k is said to be a noise gene, if it has no nearest neighbour gene g_m, where $g_m \in G$.*

The following lemmas provide the foundation of FINN.

Lemma 4. *A gene belonging to an MFIS will have nearest neighbors to it.*

Proof: A gene g_j can be a member of $MFIS_i$ iff g_j is frequent over T_G at $s\%$ support. Therefore, g_j has nearest neighbours to it and hence the proof. ■

Lemma 5. Seeds selected for cluster expansion cannot be noise.

Proof: Assume g_{ij} be a seed and be the jth gene in the ith MFIS, i.e., $g_{ij} \in MFIS_i \in Cr$. Then g_{ij} will have nearest neighbours to it according to Lemma 4. Again, according to Definition 16, a gene with nearest neighbour cannot be a noise gene and hence the proof. ■

3.3 Phase III: Clustering

We have used a shared neighbour approach to expand the cluster from the core clusters to obtain the final clusters. The clustering procedure is initiated from the core genes identified in Phase II. First, these genes are classified. The set of core genes are classified using either of the following cases:

1. If $Cr=\{MFIS\}$ and $MFIS=\{g_1, g_2, \cdots, g_x\}$ then Classify $\{g_1, g_2, \cdots, g_x\}$ with the same cluster_id.
2. If $Cr=\{MFIS_1, MFIS_2, \cdots, MFIS_k\}$ and $MFIS_1=\{g_{11}, g_{12}, \cdots, g_{1x}\}$, $MFIS_2=\{g_{21}, g_{22}, \cdots, g_{2y}\}, \cdots, MFIS_k=\{g_{k1}, g_{k2}, \cdots, g_{kz}\}$ then Classify the genes corresponding to each MFIS (i.e., say ${g_{il}}^s$ corresponding to $MFIS_i$ with same cluster_id.

For a classified MFIS of cardinality k, an arbitrary unclassified gene g will be a shared neighbour, if g is a nearest neighbour of each of the genes of that MFIS. A major advantage of FINN is that it eliminates the exhaustive neighbour search over T_G. If g has dissimilarities lesser than a given *shared neighbour threshold* (β) with each of the core genes of $MFIS$ then g is classified with the same cluster_id as that of the core genes of that $MFIS$ and grouped into the same cluster. This process of cluster expansion is iterated until there are no more genes that can be merged into this cluster. The cluster thus obtained gives a final cluster.

Once cluster expansion terminates, the row and column of the classified genes in the transaction matrix T_G are discarded from further consideration. This step reduces the number of items (genes) which have to be checked for itemset generation. The process then restarts Phase II with the new compact transaction matrix T_G.

The steps of FINN are given below:

i. Calculate the $G \times G$ dissimilarity matrix using DBK and generate the $G \times G$ gene–gene transaction matrix.
ii. Generate the MFIS using FP-tree algorithm on T_G.
iii. Classify the genes of $MFIS_i$ as core genes and give cluster_id to them.
iv. Select a gene from the nearest neighbours of the core genes of $MFIS_i$ which is a shared neighbour of each of the core genes and classify this gene with the same cluster_id as $MFIS_i$.
v. Repeat step iv till no more genes satisfy the shared neighbour condition.
vi. Discard the rows and columns of the classified genes from the gene–gene transaction matrix.
vii. Increment i and go to step iv.
viii. Repeat steps ii. through vii. till all genes in T_G are classified.

Table 1
Datasets Used in This Paper

Serial No.	Dataset	No. of Genes	No. of Conditions	Source
1	Yeast diauxic shift [23]	6,089	7	http://www.ncbi.nlm.nih.gov/geo/query
2	Subset of yeast cell cycle [24]	384	17	http://faculty.washington.edu/kayee/cluster
3	Rat CNS [25]	112	9	http://faculty.washington.edu/kayee/cluster
4	*Arabidopsis thaliana [26]*	138	8	http://homes.esat.kuleuven.be/thijs/Work/Clustering.html
5	Subset of human fibroblasts serum [27]	517	13	http://www.sciencemag.org/feature/data/984559.hsl
6	Yeast cell cycle [28]	698	72	Sample input files in Expander [29]

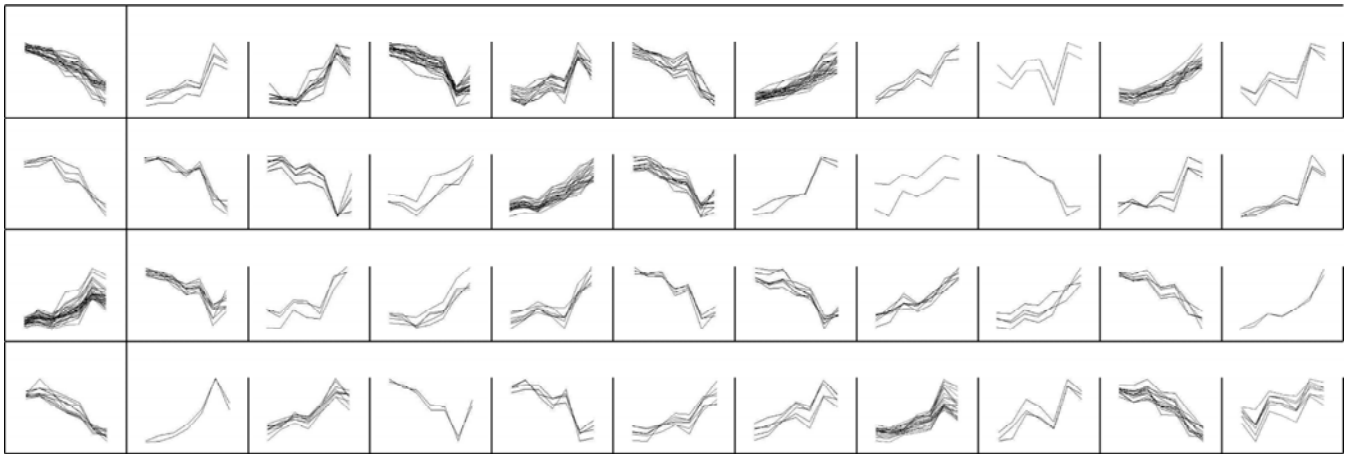

Figure 5. Result of DGC on the reduced form of Dataset 1 using our dissimilarity measure.

The clustering result of FINN using DBK is reported in Section 4.

4. Performance Evaluation

The methods were implemented in Java in Windows environment and to evaluate the methods the six real-life datasets were used as given in Table 1. All the datasets are normalized to have mean 0 and standard deviation 1.

4.1 Results: DGC

We exhaustively tested DGC on the above datasets with $\sigma = 2$. The value of σ was taken to be 2 as we went for an exhaustive search for the different patterns. We have used our dissimilarity measure [2] for D and the value of $\theta = 2$. We compared our algorithm with k-means, hierarchical clustering (UPGMA), CLICK, SOM, DCCA and GA. The k-means and UPGMA algorithms were evaluated using the built-in MATLAB implementation. CLICK and SOM algorithms were executed using the implementation provided by the Expander tool [29]. CLICK was run with the default parameter provided by Expander. Expander was also used for finding the homogeneity of the k-means clustering. For k-means, k varied from 2 to 30 by increments of two. The results obtained by our method over a reduced form of Dataset 1 are shown in Fig. 5. The dataset was reduced by filtering out the low variance and low entropy genes from the data. We note here that the clusters obtained by our algorithm are detected automatically and unlike k-means no input parameter for number of clusters is needed. We have tested k-means with $k = 16$, 20, 30, 40, 48. As our method gave a total of 47 clusters (when Euclidean distance was used) and 44 clusters (when DBK was used) for the reduced form of Dataset 1, we also tested k-means algorithm for $k = 44$ and 47, respectively. Similarly, UPGMA algorithm was tested for cutoff = 43, 44, 47 and also for various other values. Some of the clusters obtained by our method over full Dataset 1 are shown in Fig. 6. A total of 118 clusters were generated from the full Dataset 1. In Fig. 7 the clusters generated by k-means on the reduced form of Dataset 1 is given. In Figs. 8 and 9, clusters generated from the reduced form and full form of Dataset 1 using UPGMA at cutoff = 46 and 176 are shown, respectively. In Fig. 10, some of the clusters generated from the full Dataset 2 using our method are

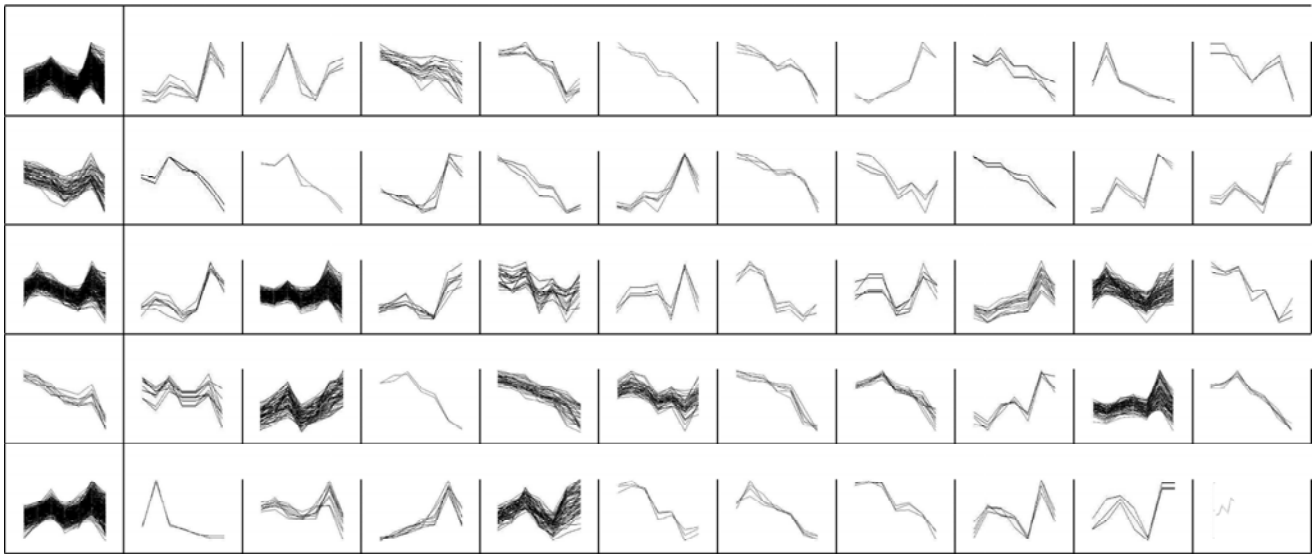

Figure 6. Result of DGC on the full Dataset 1 using our dissimilarity measure.

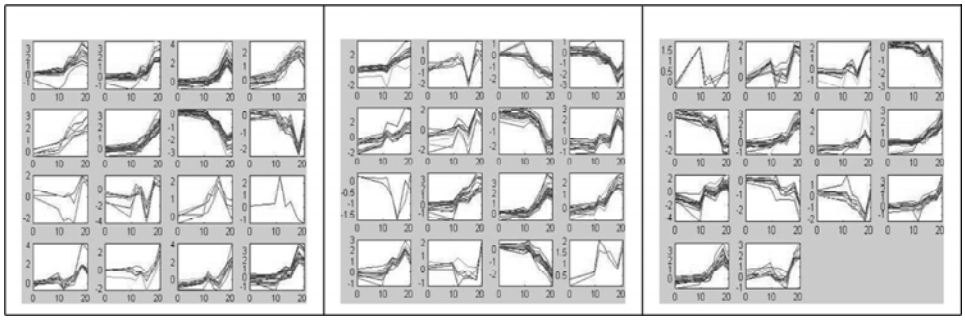

Figure 7. Result of k-means on the reduced form Dataset 1 at cutoff = 46.

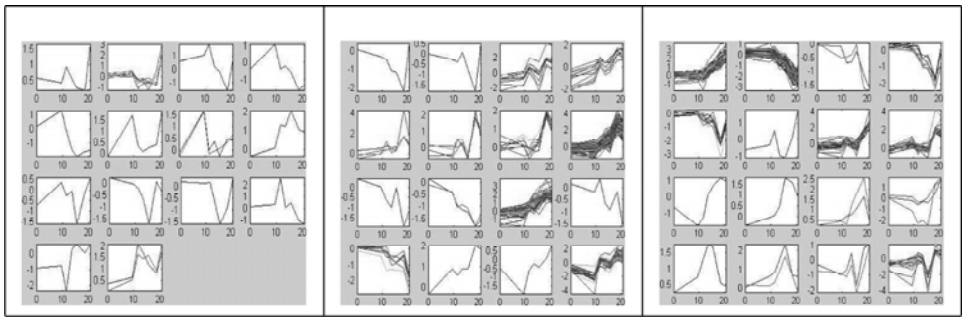

Figure 8. Result of UPGMA on the reduced form Dataset 1 at cutoff = 46.

shown and in Fig. 11 the clusters identified for Dataset 5 using DBK is depicted. Finally, to validate the cluster results, cluster validity measures like z-score, homogeneity and silhouette index were used and the results were compared with the different clustering algorithms (Tables 4–7).

4.2 Results: FINN

We exhaustively tested FINN on all the datasets. Using FINN, eight clusters were obtained from the Dataset 3. When the method was executed on Dataset 1, the clusters obtained agreed well with the functional classification

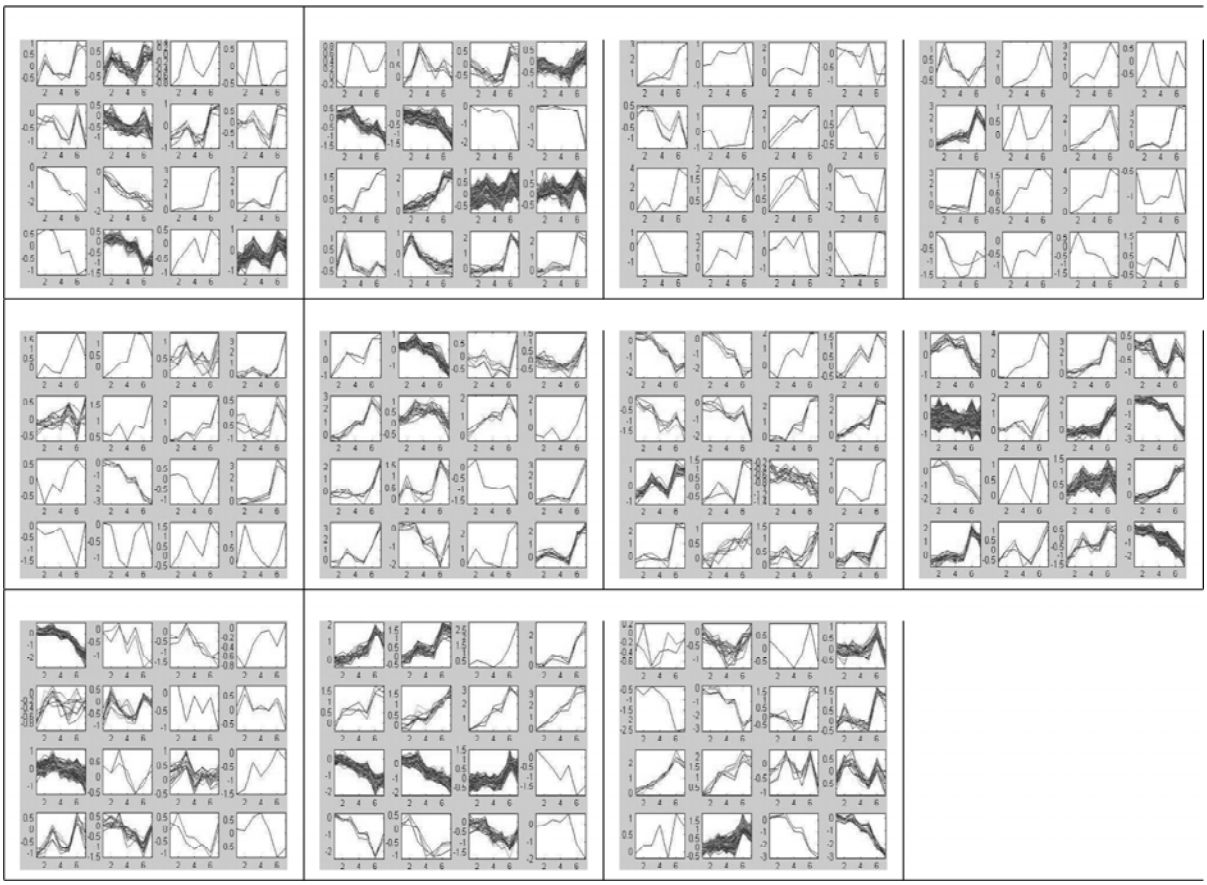

Figure 9. Result of UPGMA on the full Dataset 1 at cutoff = 176.

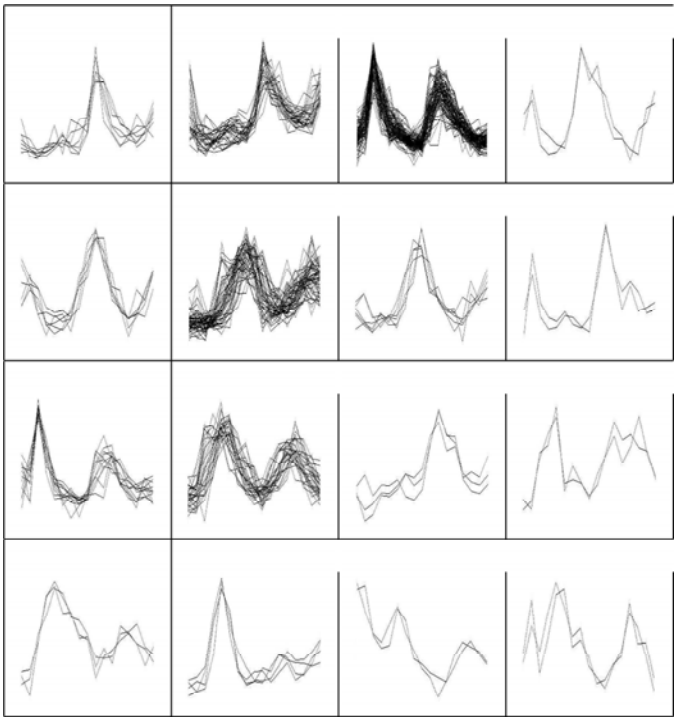

Figure 10. Some clusters generated using DGC on Dataset 2. A total of 17 clusters were detected.

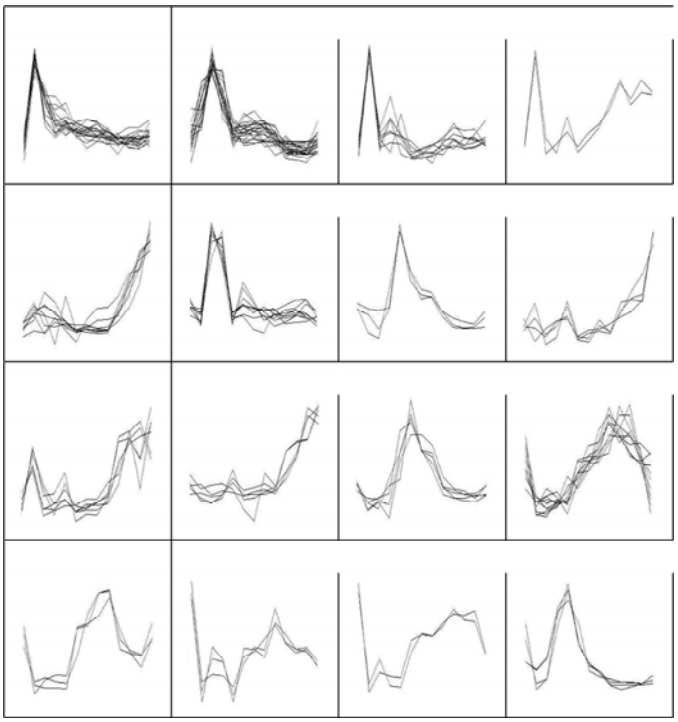

Figure 11. The clusters obtained by DGC on Dataset 5.

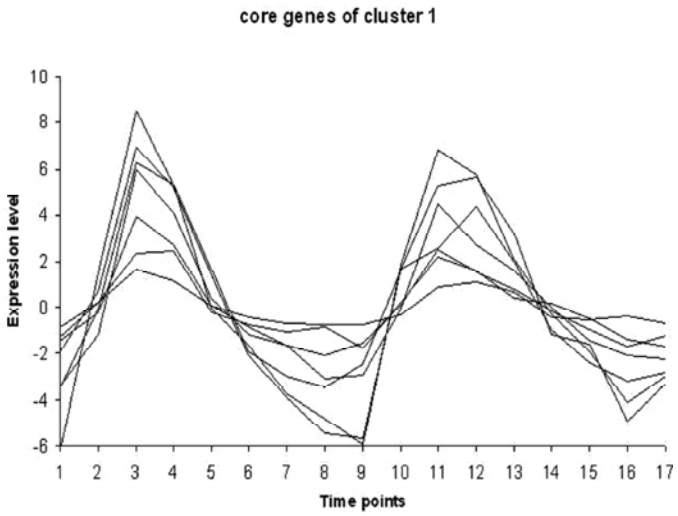

Figure 12. The core genes of Cluster 1.

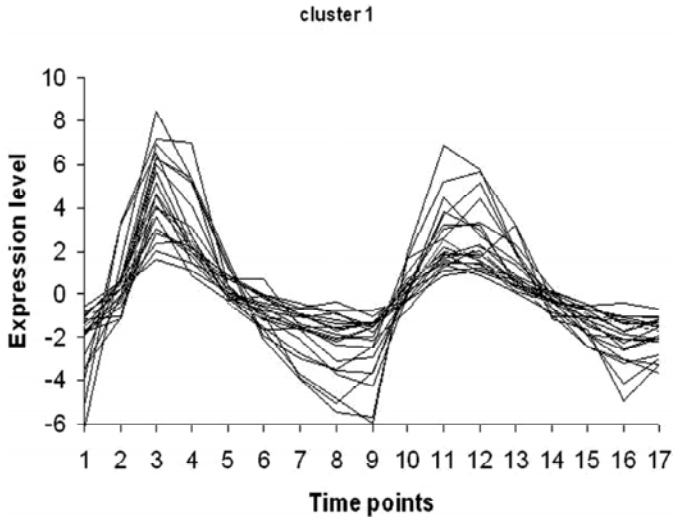

Figure 13. Final Cluster 1 based on the core genes of Fig. 12.

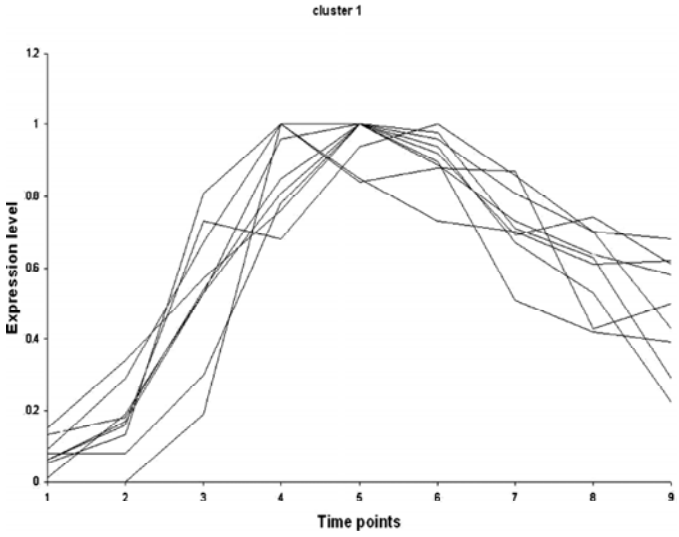

Figure 14. The final Cluster 1 obtained from the core genes.

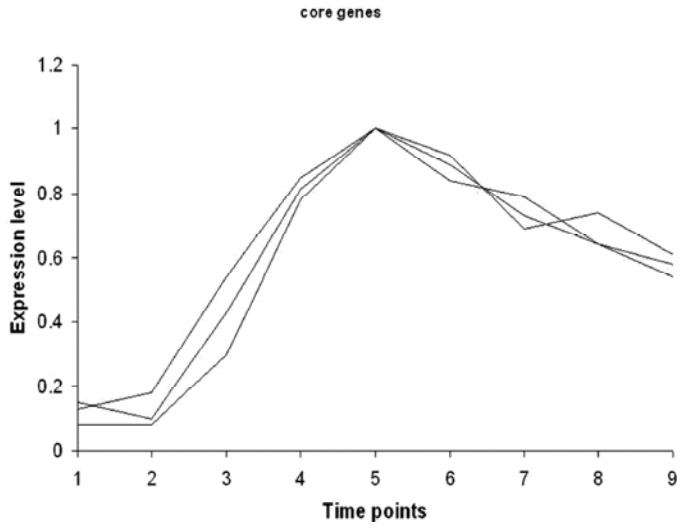

Figure 15. The core genes at $s = 40\%$.

of [24]. Because of space constraint, only one cluster from each of the datasets 2 and 3 are presented here. Of the different clusters obtained from Dataset 2, one is shown in this paper. The cluster along with its core genes is shown in Figs. 12 and 13. One of the clusters obtained from the Dataset 3 is shown in Fig. 14 and its respective core genes is shown in Fig. 15. From the results of FINN, it can be concluded that the core genes give the overall trend of the cluster. Therefore, this approach can also be used to detect the embedded clusters in the dataset. From our exhaustive experiments on FINN, it is seen that by varying the value of β, the quality of the clusters can be increased further. The support count in the frequent itemset generation has a pivotal role in the detection of the core genes. With the increase in the support count, a more compact set of core genes can be obtained. Moreover, for higher values of support count, frequent itemset generation also becomes faster. Taking these factors into count, more compact clusters may be obtained.

4.3 Cluster Quality

In this section, the performance of DGC is demonstrated on the six publicly available benchmark microarray data sets. Comparative studies of several widely used microarray clustering algorithms are reported. To judge the performance of DGC, silhouette index [30], average homogeneity score [14] and z-score [31] were used. Tables 2 and 3 show the homogeneity and silhouette values for the different cluster algorithms on the real-life datasets mentioned before.

To validate our clustering result, we used z-score [31] as the measure of agreement. Higher value of z indicates that genes would be better clustered by function, indicating

Table 2
Homogeneity Values for DGC and Its Counterparts

Datasets	Method Applied	No. of Clusters	Threshold Value	Homogeneity
Dataset 2	k-means	4	NA	0.553
	k-means	5	NA	0.591
	k-means	6	NA	0.601
	k-means	16	NA	0.771
	k-means	29	NA	0.787
	k-means	30	NA	0.8
	SOM	4	2×2 grid	0.624
	SOM	9	3×3 grid	0.723
	SOM	25	7×7 grid	0.792
	SOM	41	8×8 grid	0.840
	SOM	33	10×10 grid	0.823
	CLICK	3	Default value	0.549
	DGC	17	2	0.877
Dataset 4	k-means	4	NA	0.603
	k-means	5	NA	0.635
	SOM	4	2×2 grid	0.555
	CLICK	4	Default value	0.754
	DGC	4	4	0.741
Dataset 5	k-means	6	NA	0.475
	k-means	10	NA	0.531
	k-means	25	NA	0.604
	SOM	16	4×4 grid	0.571
	SOM	32	10×10 grid	0.616
	CLICK	5	Default value	0.483
	DGC	14	1.3	0.969
	DGC	17	1.5	0.959
	DGC	25	2	0.923
Dataset 5	k-means	5	NA	0.452
	k-means	11	NA	0.528
	k-means	30	NA	0.602
	SOM	16	4×4 grid	0.571
	SOM	26	6×6 grid	0.612
	SOM	28	7×7 grid	0.599
	CLICK	5	Default value	0.483
	DGC	11	6	0.833

a more biologically relevant clustering result. The result of applying the z-score on the reduced form of Dataset 1 is shown in Table 4. Table 4 clearly shows that our method outperforms k-means, DCCA and SOM w.r.t. the cluster quality. Table 5 shows the z-score values when the proposed method is executed at different values of θ. It can be seen that the cluster result gives better clustering at $\theta = 2$ for the full Dataset 1. The z-score values obtained from clustering the full Dataset 1 is given in Table 6. As can be seen in the table, our method performs better than K-means and hierarchical clustering. We note here that unlike k-means our method does not require the number of clusters as an input parameter. It detects the clusters present in the dataset automatically and gives the rest as noise. Also, UPGMA requires the parameter cutoff as input to the algorithm.

The z-score value of DGC compared with DCCA is given in Table 7. It can be observed that unlike the other datasets, DCCA performs better for Dataset 2. However, for most of the datasets DGC performs better than its counterparts other than Dataset 4 where CLICK performs better in terms of average homogeneity.

Table 3
Silhouette Index for DGC and Its Counterparts

Datasets	Method Applied	No. of Clusters	Silhouette Index
Dataset 2	MOGA-SVM (RBF)	5	0.4426
	MOGA (without SVM)	5	0.4392
	FCM	6	0.3872
	Average linkage	4	0.4388
	SOM	6	0.3682
	DGC at $\theta = 2$	17	0.7307
Dataset 3	MOGA-SVM (RBF)	6	0.45127
	MOGA (without SVM)	6	0.4872
	FCM	5	0.4050
	Average linkage	6	0.4122
	SOM	5	0.4430
	DGC at $\theta = 4$	8	0.489
Dataset 4	MOGA-SVM (RBF)	4	0.4312
	MOGA (without SVM)	4	0.4011
	FCM	4	0.3642
	Average linkage	5	0.3151
	SOM	5	0.2133
	DGC at $\theta = 0.3$	5	0
	DGC at $\theta = 0.4$	10	0.8
Dataset 5	MOGA-SVM (RBF)	6	0.4154
	MOGA (without SVM)	6	0.3947
	FCM	8	0.2995
	Average linkage	4	0.3562
	SOM	6	0.3235
	k-means	6	0.509
	DGC at $\theta = 2$	26	0.4077
	DGC at $\theta = 1.5$	16	0.688
	DGC at $\theta = 1.3$	14	0.738

Table 4
z-Scores for DGC and Its Counterparts for the Reduced Form of Dataset 1

Method Applied	No. of Clusters	z-score
k-means	19	10.6
DCCA	2	-0.995
SOM	35	4.46
DGC at $\theta = 0.7$	7	12.6

5. Conclusion

This paper presents two methods for clustering gene expression data, DGC and FINN. The clusters obtained by DGC have been validated using several cluster validity measures over six microarray data sets. The regulation-based cluster expansion also overcomes the problem of maintaining the pattern information usually linked with the different clustering approaches due to traditional similarity measures. In FINN, the frequent itemset generation step gives the innermost or the fine clusters from the gene

Table 5
z-Scores for DGC at Different Values of θ for the Full Dataset

DGC at	No. of Clusters	z-Score
$\theta = 0.7$	176	8
$\theta = 1$	128	9.6
$\theta = 1.5$	120	10.6
$\theta = 2$	118	13.2
$\theta = 2.7$	120	12.9
$\theta = 3.2$	119	11.3
$\theta = 3.7$	119	12.5
$\theta = 4.7$	119	10.5

Table 6
z-Scores for DGC and Its Counterparts for the Full Dataset 1

Method Applied	No. of Clusters	z-Score	Total no. of Genes
UPGMA	176	9.7	6,089
k-means	176	NA	6,089
DGC at $\theta = 0.7$	176	9.12	6,089
DGC at $\theta = 1$	128	7.02	6,089
DGC at $\theta = 1.5$	120	11.2	6,089
DGC at $\theta = 2$	118	12	6,089
DGC at $\theta = 2.7$	120	11.2	6,089

Table 7
z-Scores for DGC and Its Counterparts of Dataset 2

Method Applied	No. of Clusters	z-Score
DCCA	12	7.19
DGC	17	5.69

expression data and the shared neighbour clustering approach gives the final clusters in the dataset. Compared with other clustering approaches, our method was found better capable of identifying finer clusters of the dataset and may also be used to detect embedded clusters.

References

[1] D. Stekel, *Microarray bioinformatics* (Cambridge, UK: *Cambridge University Press*, 2005).

[2] R. Das, D.K. Bhattacharyya, & J.K. Kalita, A new approach for clustering gene expression time series data, *International Journal of Bioinformatics Research and Applications*, *5*(3), 2009, 310–328.

[3] J.B. McQueen, Some methods for classification and analysis of multivariate observations, *Proceedings of the Fifth Berkeley Symposium Mathematics Statistics and Probability*, *1*, 1967, 281–297.

[4] J.C. Bezdek, *Pattern recognition with fuzzy objective function algorithms* (New York: Plenum Press, 1981).

[5] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E.S. Lander, & T.R. Golub, Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation, *Proceedings of National Academy of Sciences*, *96*(6), 1999, 2907–2912.

[6] M. Eisen, P. Spellman, P. Brown, & D. Botstein, Cluster analysis and display of genome-wide expression patterns, *Proceedings of National Academy of Sciences*, *95*, 1998, 14863–14868.

[7] J. Dopazo & J.M. Carazo, Phylogenetic reconstruction using an unsupervised neural network that adopts the topology of a phylogenetic tree, *Journal Molecular of Evolution*, *44*, 1997, 226–233.

[8] A. Bhattacharya & R. De, Divisive correlation clustering algorithm (DCCA) for grouping of genes: Detecting varying patterns in expression profiles. *Bioinformatics*, *24*(11), 2008, 1359–1366.

[9] G. Shu, B. Zeng, Y.P. Chen, & O.H. Smith, Performance assessment of kernel density clustering for gene expression profile data, *Comparative and Functional Genomics*, *4*, 2003, 287–299.

[10] D. Jiang, J. Pei, & A. Zhang, DHC: A density-based hierarchical clustering method for time series gene expression data, *Proc. of BIBE2003: 3rd IEEE International Symposium on Bioinformatics and Bioengineering*, Bethesda, Maryland, 2003, 393.

[11] R.A. Jarvis & E.A. Patrick, Clustering using a similarity measure based on shared nearest neighbors, *IEEE Transactions on Computers*, *11*, 1973, 1025–1034.

[12] J. Herrero, A. Valencia, & J. Dopazo, A hierarchical unsupervised growing neural network for clustering gene expression patterns, *Bioinformatics*, *17*, 2001, 126–136.

[13] A. Ben-Dor, R. Shamir, & Z. Yakhini, Clustering gene expression patterns. *Journal of Computational Biology*, *6*(3–4), 1999, 281–297.

[14] R. Sharan & R. Shamir, Click: A clustering algorithm with applications to gene expression analysis, *Proc. of 8th International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, Menlo Park, California, 2000.

[15] A. Bellaachia, D. Portnoy, Y. Chen, & A.G. Elkahloun, E-cast: A data mining algorithm for gene expression data, *Proc. of the BIOKDD02: Workshop on Data Mining in Bioinformatics (with SIGKDD02 Conference)*, Edmonton, Alberta, 2002, 49.

[16] L.J. Heyer, S. Kruglyak, & S. Yooseph, Exploring expression data: Identification and analysis of co-expressed genes, *Genome Research*, *9*(11), 1999, 1106–1115.

[17] U. Maulik & S. Bandyopadhyay, Fuzzy partitioning using a real-coded variable-length genetic algorithm for pixel classification, *IEEE Transactions on Geoscience and Remote Sensing*, *41*(5), 2003, 1075–1081.

[18] S. Bandyopadhyay, U. Maulik, & A. Mukhopadhyay, Multiobjective genetic clustering for pixel classification in remote sensing imagery, *IEEE transactions on Geoscience and Remote Sensing*, *45*(5), 2007, 1506–1511.

[19] S. Bandyopadhyay, A. Mukhopadhyay, & U. Maulik, An improved algorithm for clustering gene expression data, *Bioinformatics*, *23*(21), 2007, 2859–2865.

[20] U. Maulik, A. Mukhopadhyay, & S. Bandyopadhyay, Combining pareto-optimal clusters using supervised learning for identifying co-expressed genes, *BMC Bioinformatics*, *10*(27), 2009.

[21] A. Ben-Dor, B. Chor, R. Karp, & Z. Yakhini, Discovering local structure in gene expression data: The order-preserving submatrix problem, *Proc. of the 6th Annual International Conf. on Computational Biology*, ACM Press, New york, USA, 2002, 49–57.

[22] M. Ester, H.P. Kriegel, J. Sander, & X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *Proc. of International Conference on Knowledge Discovery in Databases and Data Mining (KDD-96)*, Portland, Oregon, 1996, 226–231.

[23] J.L. DeRisi, V.R. Iyer, & P.O. Brown, Exploring the metabolic and genetic control of gene expression on a genomic scale, *Science*, 278, 1997, 680–686.

[24] R.J. Cho, M. Campbell, E. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T.G. Wolfsberg, A.E. Gabrielian, D. Landsman, D.J. Lockhart, & R.W. Davis, A genome-wide transcriptional analysis of the mitotic cell cycle, *Molecular Cell*, *2*(1), 1998, 65–73.

[25] X. Wen, S. Fuhrman, G.S. Michaels, D.B. Carr, S. Smith, J.L. Barker, & R. Somogyi, Large-scale temporal gene expression mapping of central nervous system development, Proceedings of National Academy of Science, *95*(1), 1998, 334–339.

[26] P. Reymonda, H. Webera, M. Damonda, & E.E. Farmera, Differential gene expression in response to mechanical wounding and insect feeding in arabidopsis, *Plant Cell*, 12, 2000, 707–720.

[27] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J. Lee, J.M. Trent, L.M. Staudt, J.J. Hudson, M.S. Boguski, D. Lashkari, D. Shalon, D. Botstein, & P.O. Brown, The transcriptional program in the response of the human fibroblasts to serum, *Science*, 283, 1999, 83–87.

[28] P.T. Spellman, M.Q. Sherlock, G. Zhang, V.R. Iyer, K. Anders, M.B. Eisen, P.O. Brown, D. Botstein, & B. Futcher, Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization, *Molecular Biology of the Cell*, *9*(12), 1998, 3273-3297.

[29] R. Sharan, A. Maron-Katz, & R. Shamir, Click and expander: A system for clustering and visualizing gene expression data, *Bioinformatics*, *19*(14), 2003, 1787–1799.

[30] P. Rousseeuw, Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, *Journal of Computational Applied and Mathematics*, *20*, 1987, 153–165.

[31] F. Gibbons & F. Roth, Judging the quality of gene expression based clustering methods using gene annotation, *Genome Research*, *12*, 2002, 1574–1581.

Biographies

R. Das is an assistant professor in the Department of Computer Science and Engineering, Tezpur University, Tezpur, India. She is currently pursuing her Ph.D. in Computer Science in the Department of Computer Science and Engineering, Tezpur University. Her research interests include clustering and bioinformatics. She has published several papers in international journals and referred conference proceedings.

D.K. Bhattacharyya is a professor in the Department of Computer Science and Engineering, Tezpur University, Tezpur, India. He received his Ph.D. from Tezpur University in the year 1999. His research interests include data mining, network security and content-based image retrieval. He has published more than 100 papers in international journals and referred conference proceedings and has edited two books.

J. K. Kalita is a professor of Computer Science at the University of Colorado at Colorado Springs. He received his Ph.D. from the University of Pennsylvania. His research interests are in natural language processing, machine learning, artificial intelligence and bioinformatics. He has published more than 70 papers in international journals and referred conference proceedings and has written a book.