Neighborhood-Based Clustering of Gene-Gene Interactions

Norberto Díaz–Díaz, Domingo S. Rodríguez–Baena,
Isabel Nepomuceno, and Jesús S. Aguilar–Ruiz

BioInformatics Group Seville; Seville and Pablo de Olavide University. Spain
{ndiaz, dsavio, isabel}@lsi.us.es, jsagurui@upo.es

Abstract. In this work, we propose a new greedy clustering algorithm to identify groups of related genes. Clustering algorithms analyze genes in order to group those with similar behavior. Instead, our approach groups pairs of genes that present similar positive and/or negative interactions. Our approach presents some interesting properties. For instance, the user can specify how the range of each gene is going to be segmented (labels). Some of these will mean expressed or inhibited (depending on the gradation). From all the label combinations a function transforms each pair of labels into another one, that identifies the type of interaction. From these pairs of genes and their interactions we build clusters in a greedy, iterative fashion, as two pairs of genes will be similar if they have the same amount of relevant interactions. Initial two–genes clusters grow iteratively based on their neighborhood until the set of clusters does not change. The algorithm allows the researcher to modify all the criteria: discretization mapping function, gene–gene mapping function and filtering function, and provides much flexibility to obtain clusters based on the level of precision needed.

The performance of our approach is experimentally tested on the yeast dataset. The final number of clusters is low and genes within show a significant level of cohesion, as it is shown graphically in the experiments.

1 Introduction

In any biologic process, cells and genes in particular play an important role which can be measured by their different levels of expression. These levels depend on the type of process, on the stage, and on the experimental condition that is analyzed. The knowledge about these, under a specific situation, helps to understand the function that genes play in a particular biological process.

Current works accomplished by researchers in the Bioinformatic field, like SAGE [1] for measuring gene expression, or like [2, 3] to store this gene expression in structure denominated microarray, make possible the simultaneous study of numerous genes under different conditions. Many different approaches have been applied to analyze this structure, including principal component analysis [4] as well as supervised [5] and unsupervised [6–10] learning. In unsupervised learning, clustering techniques are used to identify groups of genes that show the same expression pattern under different conditions.

[6] applied the k–means algorithm to find clusters in yeast data. In [7] graph–theoretic and statistical techniques were used to identify tight groups of highly similar elements. In [8] a memetic algorithm is presented, i.e., a genetic algorithm combined with local search -based on a tree representation of the data - for clustering gene expression data. With this aim, in [9] is explored a novel type of gene–sample–time microarray data sets, which records the expression levels of various genes under a set of samples during a series of time points. Even evolutionary algorithm [10] have been used to discover clusters in gene expression data.

All of these methods are based on the idea of grouping those genes that show the same behavior. In this work, we propose a novel clustering algorithm to identify groups of related genes based on the idea of clustering pair of genes which present the same type of interaction.

In broad outlines, the remainder of the paper is organized as follows. In section 2, the characteristics of our approach are detailed. Later in Section 3, we describe the results of our experiments. Finally, the most interesting conclusions are summarized in Section 4.

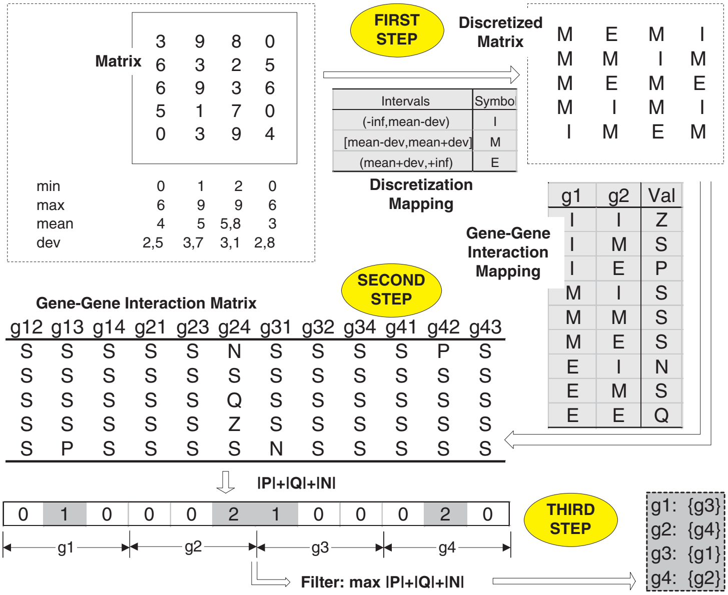
2 Description

The algorithm presented in this paper can be divided into four steps: encoding of each gene expression (*segmentation*), representation of the interaction of every two genes (*gene–gene interaction*), filtering of most representative interactions (*filtering*), and clustering interactions (*neighborhood–based clustering*). The overall approach, named InterClus, is illustrated in Algorithm 1. Each step represents a line of code in the algorithm.

Algorithm 1. Interclus

```
INPUT M: microarray ($Conditions$,$Genes$)
   $\Omega$: alphabet of discretization
   $\alpha$: discretization mapping
   $\Pi$: alphabet of interactions
   $\beta$: interactions mapping
   $F$: Filter
OUTPUT $S$: Set of Clusters
begin
   $M'$=Segmentation(M,$\Omega$,$\alpha$)
   $M''$=Encoding_Gene–Gene_Interactions($M'$,$\Pi$,$\beta$)
   $L$=Filtering($M''$,$F$,$\Pi$)
   $S$=Build_Set_of_Clusters($L$)
end
```

The first three steps of the process are depicted in Figure 1. Each of these steps is described in detail in the next subsections. In addition, the last step, neighborhood–based clustering is also explained.



Matrix				
	3	9	8	0
	6	3	2	5
	6	9	3	6
	5	1	7	0
	0	3	9	4
min	0	1	2	0
max	6	9	9	6
mean	4	5	5,8	3
dev	2,5	3,7	3,1	2,8

Intervals	Symbol
(-inf,mean-dev)	I
[mean-dev,mean+dev]	M
(mean+dev,+inf)	E

M	E	M	I
M	M	I	M
M	E	M	E
M	I	M	I
I	M	E	M

g1	g2	Val
I	I	Z
I	M	S
I	E	P
M	I	S
M	M	S
M	E	S
E	I	N
E	M	S
E	E	Q

g12	g13	g14	g21	g23	g24	g31	g32	g34	g41	g42	g43
S	S	S	S	S	N	S	S	S	S	P	S
S	S	S	S	S	S	S	S	S	S	S	S
S	S	S	S	S	Q	S	S	S	S	S	S
S	S	S	S	S	Z	S	S	S	S	S	S
S	P	S	S	S	S	N	S	S	S	S	S

g1			g2			g3			g4		
0	1	0	0	0	2	1	0	0	0	2	0

g1: {g3}
g2: {g4}
g3: {g1}
g4: {g2}

Fig. 1. First three steps of Algorithm 1. First step: definition of the discretization mapping function to obtain a discretized matrix. Second step: definition of the gene–gene interaction mapping function to obtain the gene–gene interaction matrix. Third step: selection of gene–gene interactions that satisfy the filtering criterion.

2.1 Segmentation

The first step addresses the segmentation of each gene expression level. Due to the fact these levels are represented by numerical values, the segmentation is done by discretizing the range of values. In this way, different labels are obtained according to the gene expression level under particular stimulus (experimental condition). However, the discretization is local, i.e., the same expression level for two different genes might transform into different labels.

To carry out the discretization, we need to define an alphabet Ω, which is used to provide labels for the mapping, and a mapping function α, which is used to assign labels from Ω to the numerical values. The definition of Ω and α is provided by the user: characters for Ω and a discretization mapping table for α, in which the user can also make use of symbols ∞, μ and σ, standing for *infinite*, *mean* and *standard deviation*. Any expression that uses these special symbols is valid, together with arithmetical operators and numbers. For instance, in Figure 1, the first step transforms the gene expression level matrix into a discretized matrix by using the discretization mapping α, defined over a three–symbol alphabet $\Omega = \{I, M, E\}$. If the gene expression level is in $(-\infty, \mu - \sigma)$ then the label "I" is assigned (inhibited); if it is in $[\mu - \sigma, \mu + \sigma]$, then the label is "M" (medium); and finally, if it is in $(\mu + \sigma, +\infty)$, then "E" (expressed). An expression like $\mu + 2\sigma$ is also feasible, and any number of labels as well.

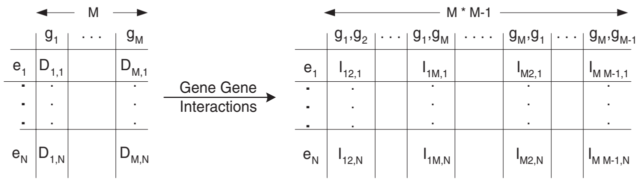


Fig. 2. Gene-Gene Interactions

Note that although we use values like μ or σ, these values are different for each gene, so the discretization is local. A value of 0.6 for a gene can mean "expressed", and perhaps "inhibited" for another one, where both states translate further into labels.

2.2 Gene–Gene Interaction

Once each gene expression level has been labelled, we will focus on the interaction between every pair of genes. Firstly, another alphabet Π is needed to assign a label to any possible combination of gene pairs. For example, we might be interested in differentiating the interaction *inhibited–expressed* from the interaction *expressed–expressed.* In general, the size of the set Π is, at maximum, the square of the size of the set Ω, although usually should be lower. In Figure 1, it is shown in the first step that $|\Omega| = 3$, and in the second step, the gene–gene interaction mapping has exactly 9 combinations, but the size of the alphabet Π is 5, corresponding to {Z,S,P,N,Q}. In this example, Z stands for *null*, S for *similar*, P for *positive*, N for *negative*, and Q for *both expressed.* The interaction mapping function β is also defined by the user, as a mapping table, $\beta : \Omega \times \Omega \rightarrow \Pi$.

As the microarray has M genes and N experiments, for each gene, $M - 1$ interactions with the remaining genes are needed. In short, there will be $M \times (M - 1)$ interactions, as it is illustrated in Figure 2. The left–hand side of Figure 2 represents the discretized matrix obtained after the first step, in which rows mean experiments and columns mean genes. The values D_{ij} of a specific row and column are discrete, belonging to the alphabet Ω. To the right, any possible pair of different genes is enumerated in columns. In general, gene i can interact with other $M - 1$ genes. The value $I_{ij,k}$ of a row k and a column represents the symbol from the alphabet Π obtained after analyzing the two genes i and j involved in the interaction under the experiment k.

The new matrix M'' encodes the information of all possible interactions, although not every one might be interesting. For example, in Figure 1, we see in the table generated by the second step that many columns have only the symbol "S", which means similar, i.e., there is no significant up– or down–regulation in this case. The first column shows that genes 1 and 2 have similar behavior, so its interaction is not relevant. In this way, we might withdraw much irrelevant information if we were able to select the most interesting patterns in columns. That is the aim of the third step, described in the next subsection.

Algorithm 2. Step–3 Filtering

```
INPUT $M''$: Interaction Matrix
  $F$: Filter
  $\Pi$: alphabet of interactions
OUTPUT $L_F$: List of gene subsets
begin
  $L_F := \{\}$
  for all pair of gene $(g_i, g_j)$ with $i \neq j$ do
    $S_e := \{\}$
    for all experiment $e_k$ do
      $S_e := S_e + I_{ij,k}$
    end for
    $S'_e := Filter(S_e, F)$
    $L_F := L_F + S'_e$
  end for
end
```

2.3 Filtering

The fact that two genes are inhibited under most or all of the experimental conditions, has no biological importance. Therefore, this situation can be easily ignored. When two genes are both expressed under most or all of the experimental conditions, that might have biological meaning. In fact, many studies only focus on this aspect: the interaction expressed–expressed. In this work, we are also interested in other cases: for example, when most of the time an inhibited gene is related to an expressed gene, and vice verse. And this situation is especially interesting when the complementary is true as well, i.e., if gene 1 is expressed then gene 2 is inhibited and if gene 1 is inhibited then gene 2 is expressed. The last situation is more difficult to detect and is one of the main goals in this work.

Another interesting issue is that what means "most of the time" for a pair of genes may not have the same meaning for another pair. For example, in Figure 1 the gene 1 is related to genes 2, 3 and 4, in the first three columns. The most significant behavior is shown by the interaction 1–3, because for the last experiment the label is "P". However, if we analyze the gene 2 against genes 1, 3 and 4, the most significant behavior is shown by the interaction 2–4, because for the experiments 1 and 3 the labels are "N" and "Q", respectively. This gives some clues about the strength of interactions, and provides us a specific criterion for each gene regarding the remainder. Therefore, although the filtering function is global, the value provided by the filtering function might be different for each gene. That happens in Figure 1, in the third step, as gene 1 is related to gene 3 (the filter function value is 1), gene 2 is related to gene 4 (the filter function value is 2), etc. Note that if gene 2 were also related to gene 3 with filter function value equal to 1, this interaction will not be chosen as the maximum value for the filter function was 2.

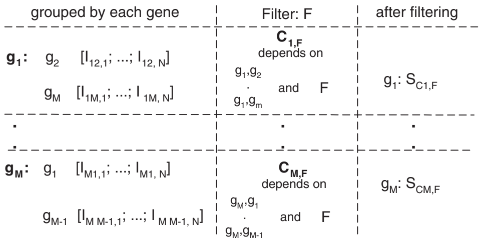

grouped by each gene
Filter: F
after filtering
g_1: g_2 $[I_{12,1}; ...; I_{12,N}]$
g_M $[I_{1M,1}; ...; I_{1M,N}]$
$C_{1,F}$ depends on g_1,g_2 · g_1,g_m and F
g_1: $S_{C1,F}$
g_M: g_1 $[I_{M1,1}; ...; I_{M1,N}]$
g_{M-1} $[I_{M\,M-1,1}; ...; I_{M\,M-1,N}]$
$C_{M,F}$ depends on g_M,g_1 · g_M,g_{M-1} and F
g_M: $S_{CM,F}$

Fig. 3. Filtering process

In Figure 3 is depicted the use of filtering, where $C_{i,F}$ denotes the conditions established for the g_i–interactions using the filter F, and $S_{C_{i,F}}$ represents the subset of genes whose interactions satisfy the condition $C_{i,F}$. As explained earlier, for the example in Figure 1, the condition $C_{1,F}$ would be $max(|P| + |Q| + |N|) = 1$, but $C_{2,F}$ would be $max(|P| + |Q| + |N|) = 2$.

The filtering algorithm is illustrated in Algorithm 2, where L_F denotes the list of all the subsets $S_{C_{i,F}}$. That is, $L_F = \{S_{C_{1,F}}, S_{C_{2,F}}, ..., S_{C_{M,F}}\}$. After this process, the filtering algorithm will generate the list of subsets of genes related to each one, if exists. In Figure 1 is provided, in the third step, the list of four subsets of genes, each of them with only one gene, by using the filter $max(|P|+|Q|+|N|)$.

Also, in this filtering process is possible to establish a minimum threshold. This value will have been satisfied for each $C_{i,F}$, so that if the condition established for g_i-interactions do not satisfy it, $S_{C_{i,F}}$ will be empty and, therefore, it will not be part of L_F. In this way, we manage to give greater power to the filter function, since it is possible to select those gene interactions that fulfil the filtering criterion a minimum number of times.

Note that it does not make sense to establish this threshold in a value greater than the number of experiments of the original dataset, because all of the $S_{C_{i,F}}$ subsets will be empty, and so, the L_F list as well.

2.4 Neighborhood–Based Clustering

Once the relevant interactions between each pair of genes have been obtained, it is time to cluster them. The clustering algorithm, named SNN (Similar Nearest Neighbor), is based on the similarity of groups, instead of analyzing pairs of elements. It builds clusters by grouping genes whose neighbors are similar. SNN stars considering each gene as a separate cluster and at each step merges clusters which have exactly the same neighbors. Thus, the concept of neighborhood is redefined to handle correctly with clusters of neighbors.

Definition 1 (Neighborhood of a gene). *The neighborhood $N_g(i, F)$ of a gene g_i using the Filter F, is the set of genes whose amount of relevant interactions with regards to the gene i fulfils the condition $C_{i,F}$.*

$$N_g(g_i) = S_{C_i} \tag{1}$$

Algorithm 3. STEP–4 *SNN*

```
INPUT $L_F$: List of gene subsets
OUTPUT RSC: Set of Clusters
begin
  $SC := \theta$
  for all gene $g_i$ do
    $RSC[i] := \{g_i\}$
  end for
  repeat
    for all cluster $C_h \in RSC, 1 \leq h \leq |RSC|$ do
      $NSC[h] := N_c(C_h)$
    end for
    $SC := RSC$
    $RSC := Reduction(SC, NSC)$
  until $SC = RSC$
end
```

Algorithm 4. Reduction

```
INPUT C: Set of Cluster
  NSC: Neighbor Set of Cluster
OUTPUT R: Reduced set of clusters
begin
  $R := C$
  for all pair $(i, j)$, with $1 \leq i \leq j \leq |C|$ do
    if $S[i] = S[j]$ then
      $R[i] := R[i] \bigcup C[j]$
      remove $R[j]$
    end if
  end for
end
```

Definition 2 (Neighborhood of a cluster). *The neighborhood $N_c(C, F)$ of a cluster c (*cluster neighborhood*) using the Filter F, is the set formed by all the neighborhoods of each gene belonging to the cluster C.*

$$N_c(C) = \bigcup_{g \in C} N_g(g) \qquad (2)$$

Once every necessary definition to support the algorithm at this step have been presented, we will describe the code depicted in Algorithm 3. The input parameter is L_F, containing in each position i the neighbors of g_i. And the output parameter is RSC, the reduced set of clusters, where each one comprises a group of genes. SC is an auxiliary set of clusters and RSC is initially set with clusters containing only one gene. The process is repeated until RSC has no change at an iteration. The neighborhood of every cluster is calculated in order to analyze the possible reduction of the set of cluster, task done by the Reduction function

(Algorithm 4). The reduction of a set of cluster follows the next criterion: two clusters are joined if both have exactly the same neighborhood. We are aware of the restrictive character of this criterion and a relaxation of it is considered among our future research directions.

3 Experiments

In this section, we address the evaluation of the performance of our approach, which is experimentally tested on the yeast dataset [6]. This dataset has information on 2884 genes under 17 different experimental conditions.

In Table 1 it is shown the discretization mapping. The symbols μ_i and σ_i denote the mean and the standard deviation, respectively, of the expression levels of g_i under the whole set of experiments. Thus, the g_i expression level under e_k will be labelled as **I** (inhibited) if it belongs to $(-\infty, \mu_i + \sigma_i)$, or as **E** (expressed) if it belongs to $[\mu_i + \sigma_i, +\infty)$.

The alphabet Π, used to encode each pair of gene–gene interaction, and the interaction mapping function β are shown in Table 2. Highly relevant interactions are those where genes change their state from inhibited to expressed (P) or from expressed to inhibited (N).

The interaction encoded as Z means that the gene does not take part in the experiment, and the interaction encoded as S means that there is no visible influence on each other. Thus, the used filter aims to select those interactions in which the highest number of P and N is reached. For this dataset we will establish a threshold value equal to 14 (note that 17 is the maximum). In this way, we will manage to select those gene–gene interactions which change their state from inhibited to expressed or from expressed to inhibited in at least 14 of the 17 experiments. With this filter, those genes whose interaction with others are P or N are selected, and those whose interaction is S or N are not. These two last interactions might be chosen as well, although not because of their biological relevance, but to make possible the comparison of the clusters obtained by using the filter *highest(P,N)*. Thus, the InterClus process will be repeated three times with the same configuration but with different filter functions. These filters will be *highest(P,N)*, *highest(Z)* and *highest(S)*, respectively. However, we do not show the cluster obtained with *highest(Z)* because of its lack of biological interest.

The results obtained using our approach over the yeast dataset has been shown in Table 3, in which it is shown the five clusters with the highest size for each filter function. These clusters are ordered decreasingly according to their sizes. The dimension of each cluster will be shown at column "Size". The other column, "Number", represents the number of clusters which have been obtained

Table 1. Disretization mapping α

Intervals	Ω
$(-\infty, \mu_i + \sigma_i)$	I
$[\mu_i + \sigma_i, +\infty)$	E

Table 2. Gene–Gene Interaction Mapping Function β

$\Omega \times \Omega$		Π
I	I	Z
I	E	P
E	I	N
E	E	S

Table 3. Results obtained using the yeast dataset

	F_1 =Highest(Z)		F_2 =Highest(P,N)		F_3 =Highest(S)	
	Number	Size	Number	Size	Number	Cluster
1°	1	164	1	89	1	5
2°	1	116	1	2	-	-
3°	1	84	-	-	-	-
4°	1	54	-	-	-	-
5°	1	45	-	-	-	-

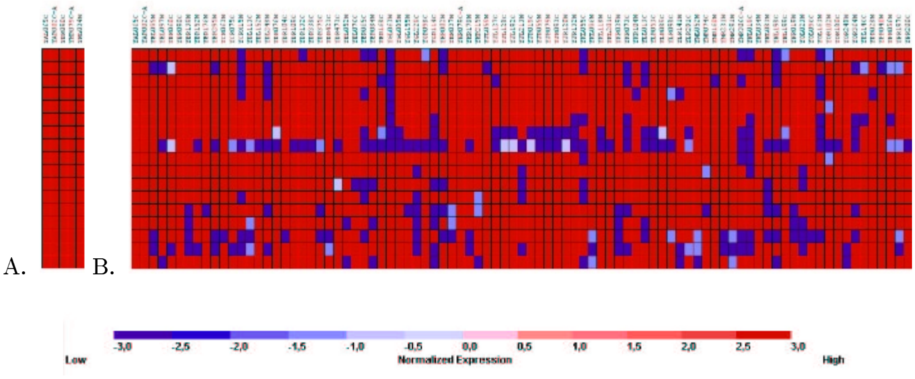


Fig. 4. A. Cluster (5 genes) using F_3 =Highest(S); **B**. Cluster (89 genes) using F_2 =Highest(P,N).

with that size. The symbol "-" means that no cluster has been found with at least two genes. For example, the size of the bigger cluster obtained using F_1 is 164 genes, using F_2 is 89 genes and 5 using the filter F_3. The next clusters found with these filters (second row) have been one with 116 genes, one with 2 genes and none, respectively.

We will show two examples of clusters. Figure 4.A shows the first cluster (5 genes) obtained with F_3 =Highest(S). Obviously all of the genes are highly expressed. Figure 4.B shows the first cluster (89 genes) obtained with F_2 =Highest(P,N). In this case, we are mainly interested in the interactions that lead to changes in the regulation, from inhibited to expressed and vice versa. These expression levels are encoded using the *GenePattern* tools [11]. For each gene under one experimental condition is generated a color which represents the expression level for this pair gene–experiment. The meaning of this colors is depicted at the bottom in Figure 4. A preprocessing of the expression level (standardization and normalization by column) was carried out in order to draw the clusters using using regular levels of blue (inhibited) and red (expressed).

Figure 4 shows that each cluster groups genes with very similar behavior pattern, as the colors are almost alike.

4 Conclusions

In this work, we propose a new greedy clustering algorithm to identify groups of related genes. The approach is based on neighborhood of gene–gene interactions instead of on expression levels. One of the main features is that the algorithm allows the researcher to modify all the criteria: discretization mapping function, gene–gene mapping function and filtering function, and provides much flexibility to obtain clusters based on the level of precision needed. The performance of our approach is experimentally tested on the yeast dataset. The final number of clusters is low and genes within show a significant level of cohesion, as it is shown graphically in the experiments.

References

1. Velculescu, V.E. *Characterization of the yeast transcriptome.* Cell, **88**, 243-251,1997
2. Schena, M. (1996) *Genome analysis with gene expression microarray.* Bioessaya, **18**, 427-431, 1996
3. Lipshutz,R.J *High density synthesis oligonucleotide arrays*, Nature Genetics Supplement, **21**, 20-24, 2000
4. K.Y. Yeung and W.L. Ruzzo *Principal component analysis for clustering gene expression data* Bioinformatics, **17**, 763-774, 2001
5. T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeed, J.P. Mesirov, H. Coller, M.L. Loh, F.R. Downing, M.A. Caliguri, C.d. *Molecular classification of cancer: Class discovery by gene expression monitoring* Science, **286**, 531-537,1999
6. S. Tavazoie, J.D. Hughes, M.J. Campbell, R.J. Cho, and G.M. Church *Systematic determination of genetic network architecture* Nature Genetics, **22**, 281-285, 1999
7. R Sharan and R Shamir *CLICK: A clustering Algorithm for Gene Expression Analysis* Proc Int Conf Intell Syst Mol Biol, **8**, 307-316, 2000
8. N. Speer and P. Merz and C. Spieth and A. Zell *Clustering Gene Expression Data with Memetic Algorithms based on Minimum Spanning Trees* IEEE Press, **3**, 1848-1855, 2003
9. Daxin Jiang, Jian Pei, Murali Ramanathan, Chung Tang, and Aidong Zhang *Mingin Coherent Gene Clusters from Gene-Sample-Time Microarray Data* KDD, 430-439, 2004
10. Patrick C.H. Ma, and Keith C.C. Chan *Discovering Clusters in Gene Expression Data using Evolutionary Approach* ICTAI, 459-466, 2003
11. http://www.broad.mit.edu/cancer/software /genepattern/