Obtaining Biclusters in Microarrays with Population-Based Heuristics

Pablo Palacios[1], David Pelta[2], and Armando Blanco[2]

[1] Dept. de Ingeniería Electrónica, Sistemas Informáticos y Automática, Universidad de Huelva, Campus UIB - Huelva
pablo.palacios@diesia.uhu.es

[2] Depto. de Ciencias de la Computación e I.A., Calle Periodista Daniel Saucedo Aranda s/n, Universidad de Granada, 18071 Granada, Spain
{dpelta, armando}@decsai.ugr.es

Abstract. In this article, we shall analyze the behavior of population-based heuristics for obtaining biclusters from DNA microarray data. More specifically, we shall propose an evolutionary algorithm, an estimation of distribution algorithm, and several memetic algorithms that differ in the local search used.

In order to analyze the effectiveness of the proposed algorithms, the freely available yeast microarray dataset has been used. The results obtained have been compared with the algorithm proposed by Cheng and Church.

Both in terms of the computation time and the quality of the solutions, the comparison reveals that a standard evolutionary algorithm and the estimation of distribution algorithm offer an efficient alternative for obtaining biclusters.

1 Introduction

One of the research fields which has aroused the greatest interest towards the end of the 20th century and whose future is expected to be as equally promising in the 21st century is the study of an organism's genome or genomics.

By way of a brief history, it was Gregor Mendel who defined the gene concept in his research as the element where information about hereditary characteristics is to be found. At a later stage, Avery, McCleod and McCarty demonstrated that an organism's genetic information stems from a macromolecule called deoxyribonucleic acid (DNA); it was later discovered that genetic information located in specific areas of the DNA (the genes) enabled protein synthesis; this was followed by the sequencing of the genome of certain organisms (including humans). This and future consequences awakened a great deal of interest among scientists.

Since proteins are responsible for carrying out cellular functions, cellular functioning therefore depends on the proteins synthesized by the genes, and is determined by regulation of protein synthesis (gene expression) and control of its activity.

The process whereby the approximately 30,000 genes in the human genome are expressed as proteins involves two steps: 1) the DNA sequence is transcribed in messenger RNA sequences (mRNA); and 2) the mRNA sequences are in turn translated into amino acid sequences which comprise the proteins.

Measuring the mRNA levels provides a detailed vision of the subset of genes which are expressed in different types of cells under different conditions. Measuring these levels of gene expression under different conditions helps explore the following aspects (among others) in greater depth: a) The function of the genes, b) How several genes interact and c) How different experimental treatments affect cell function.

Recent advances in array-based methods enable expression levels of thousands of genes to be measured simultaneously. These measurements are obtained by quantizing the mRNA hybridization with a cDNA array, or oligonucleotide probes fixed in a solid substance.

Technological advances in the development of cDNA arrays simultaneously produce an amazingly large quantity of data relating to the transcription levels of thousands of genes and in specific conditions. For knowledge extraction (function of the genes, implication of certain genes in specific illnesses, etc.), researchers use consolidated methodologies and specific ones are being developed. However, although the results obtained so far are getting better, there is still room for improvement.

2 Gene Expression Matrices

In a gene expression matrix, the rows represent genes and the columns represent samples, and each cell contains a number which characterizes the expression level of a particular gene in a particular sample.

Like most experimental techniques, microarrays measure the final objective indirectly through another physical quantity, for example the relative abundance of mRNA through the fluorescence intensity of the *spots* in an array.

Microarray-based techniques are still a long way from providing the exact quantity of mRNA in a cell. The measurements are naturally relative: essentially we can compare the expression levels of one gene in different samples or different genes in one sample, so that it is necessary to apply a suitable normalization to enable comparisons between data. Moreover, as the value of the microarray-based gene expression can be considerably greater according to the reliability and limitations of a particular microarray technique for certain types of measurements, data normalization is a key issue to consider.

Once we have constructed the gene expression matrix, the second step is to analyze it and attempt to obtain information from it.

In this work we shall use the *biclustering* concept introduced by Hartigan [6] to capture the degree of similarity between a *subset* of elements within a *subset* of attributes. Church applied this technique on DNA *microarrays* [3].

The advantage of biclustering as opposed to traditional clustering when applied to the field of microarrays lies in its ability to identify *groups* of genes that show similar activity patterns under a *specific subset* of the experimental conditions. Therefore, biclustering approaches are the key technique to use when one or more of the following situations applies [7]:

1. Only a small set of the genes participates in a cellular process of interest.
2. An interesting cellular process is active only in a subset of the conditions.
3. A single gene may participate in multiple pathways that may or not be coactive under all conditions.

Besides, the biclusters should not be exclusive and/or exhaustive: A gene / condition should be able to belong to more than one cluster or to no cluster at all and be grouped using a subset of conditions/genes.

2.1 Biclustering Techniques

In this section, we briefly present some representative strategies in literature for obtaining biclusters.

Cheng and Church in [3], proposed a set of heuristic algorithms whose function, beginning with the complete matrix, is based on the execution of iterative stages: deletion and addition of rows in order to obtain biclusters.

The *FLOC* algorithm (Flexible Overlapped Clustering) [16], is a variant of previous work, which performs an iterative process from an initial set of biclusters in an attempt to improve their overall quality. In each iteration, a row or column is added or deleted from each bicluster in order to produce the ideal set of biclusters in terms of having the greatest similarity. The algorithm finishes when there is no improvement in the overall quality of the previous set of biclusters.

The *CLICK* algorithm is based on the construction of bipartite graphs [13]. This algorithm uses the following three stages to obtain biclusters: a) Identify regions which may contain biclusters; b) Identify the biclusters and c) Refine biclusters to their minimum size.

The *double conjugated clustering* algorithm [2], where the search for biclusters is performed on two different search spaces: one comprising the genes, and the other the experiments. In this way, a clustering algorithm is applied to each space independently. In order to join the clusters induced in both processes, two functions are defined which enable one node from one space to be converted into the conjugated node of the other space, and vice versa. The final adjustment between both search spaces is obtained by means of a new clustering process to correct the clusters conjugated in the other space.

The *pCluster* algorithm [14] uses a more general similarity type. Two objects therefore belong to the same cluster if they display a similar pattern for a subset of dimensions. This enables biclusters to be discovered with elements which comply with the same pattern although they are not close to each other. Discovering these biclusters is essential when revealing gene behavior.

Finally, the *pMafia* algorithm [10] consists of a parallel implementation of a grid algorithm [12, 15] adapted for biclusters. Each dimension is divided into small intervals of a fixed size called "windows". During the clustering process, when two adjacent windows are similar, they merge and a new one is created. The algorithm uses the parallelism to reduce the computation time.

The last years have shown an increasing interest in this field. We suggest the interested reader to check out the excellent survey by Madeira and Oliveira [7].

2.2 Measuring the Quality of a Bicluster

Below, we shall define the main concepts which form the basis of the residue-based bicluster induction methods towards which we have directed our research.

Definition: Let D be a gene expression matrix, of the size $n \times m$ ($D_{n \times m}$), where the set of rows $F = \{G_1, G_2, ..., G_n\}$ represents the genes and the set of columns $R = \{E_1, E_2, ..., E_m\}$ represents the conditions or experiments. Each element d_{ij} in the matrix matches the expression level (absolute or relative) of gene G_i in experiment E_j.

Definition: Given a gene expression matrix $D_{n \times m}$, a bicluster is a pair (I, J), where $I \subseteq \{1, ..., n\}$ is a subset of rows of F and $J \subseteq \{1, ..., m\}$ is a subset of columns of R, in which the genes G_i with $i \in I$ behave in a similar way.

Definition: Given a bicluster (I, J), the residue (r_{ij}) of an element d_{ij} of the bicluster is calculated according to Equation 1.

$$r_{ij} = d_{ij} - d_{iJ} - d_{Ij} + d_{IJ} \tag{1}$$

where

$$d_{iJ} = \frac{\sum_{j \in J_i} d_{ij}}{|J_i|} \tag{2}$$

$$d_{Ij} = \frac{\sum_{i \in I_j} d_{ij}}{|I_j|} \tag{3}$$

$$d_{IJ} = \frac{\sum_{i \in I, j \in J} d_{ij}}{|I| \cdot |J|} \tag{4}$$

The residue is an indicator of the degree of coherence of an element in relation to the remaining elements in the bicluster, additionally providing the bias of the objects and the relevant attributes. Therefore, the smaller the value of the residue, the greater the coherence.

Definition: Given a bicluster (I, J), the residue (r_{IJ}) of the bicluster can be obtained from Equation 5, where r_{ij} is the residue of the element d_{ij} and v_{IJ} is the volume of the bicluster.

$$r_{IJ} = \frac{\sum_{i \in I, j \in J} |r_{ij}|}{v_{IJ}} \tag{5}$$

In order to determine the overall quality of a bicluster, its residue is defined as the mean of the residues of all its elements. This mean could be arithmetic, geometric, etc. Here, we applied the arithmetic mean.

3 Proposed Population Based Techniques

In this section, we shall describe the particular characteristics of the genetic algorithm (GA), the memetic algorithms (MA), and the estimation of distribution algorithm (EDA) which have been implemented.

3.1 Genetic Algorithm

As the simplest population based technique, we shall implement a classical genetic algorithm, with elitism. The main characteristics are described next.

Codification: The solution's representation is as follows: given a data matrix D of size $n \times m$, the biclusters are encoded in a vector of size $n + m$, where the first n positions represent the rows and the last m positions represent the columns of the bicluster. Each position of the vector can have one of two values (1 or 0) indicating whether the corresponding row or column is to be found (1) or not (0) in the bicluster.

Selection: Baker's stochastic universal sampling was chosen as the selection method. This is a roulette wheel method with slots which are sized according to the fitness of each chromosome.

Crossover: the uniform operator is used: given two parents, the offspring keep the values common to both of them, while every other value is randomly taken from any of the parents.

Mutation: a *BitFlip* mechanism was chosen: given an individual in the population, one of its bit values is changed for its complementary one.

Fitness: is measured as the residue associated to the bicluster represented by a given individual.

Restart: For the restart strategy, we have chosen to move the best individual to the new population. In addition, 20 % of the new population will be the best individual in the current mutated generation and the rest shall be generated randomly. This restart shall be applied when 10 % of the generations to be made have taken place with no change in the best element in the population.

3.2 Memetic Algorithms

Memetic algorithms have been studied from practical and theoretical points of view since 15 years ago [5] and, while very complex strategies are being developed, in their simplest form they can still be considered as a genetic algorithm hybridized with a local search operator.

Here, and departing from the genetic algorithm described before we implemented several basic memetic algorithms using two different local search techniques, namely:

– **K-opt:** the chromosomes can also be seen as a permutation of the rows and columns of the bicluster so that we could apply *k-opt* movements on

them, which in particular would consist in exchanging with each other k bits of a given solution. If this exchange leads to an improvement, a new k-opt movement would be undertaken and this process would be continued until no improvement is made in certain number of trials.
We constructed 4 different memetic schemes for values of $k \in \{2, 3, 4, 5\}$.

– **Taboo Search (TS):** a very basic TS strategy is used. The neighborhood is sampled using the mutation operator described for the GA.

3.3 Estimation of Distribution Algorithms

Estimation of Distribution Algorithms (EDA) were described for the first time by H. Muehlenbein and G. Paab [8]. In EDAs, the solution space is represented by means of a probability distribution associated with the individuals selected in each generation and not with a population. This probability distribution is calculated from a set of individuals selected from the previous generation. Once obtained, it is sampled in order to generate descendants so that neither the mutation nor the crossover is applied in the EDAs.

The easiest way to calculate the probability distribution consists in considering all the variables of interest to be independent. So, the probability estimation is converted into the product of the marginal probabilities of n variables as:

$$p(x) = \Pi_{i=1}^{n} p(x_i) \tag{6}$$

Different approximations to the methodology can be found, including: the univariate marginal distribution algorithm [9], population-based incremental learning [1] and the compact genetic algorithm [4].

In this work we shall focus on the Univariate Marginal Distribution Algorithm (UMDA, in what follows). UMDA maintains a population of N individuals, to which a selection method is applied in order to create a new population. From this new population, the frequencies of each gene are obtained and used to generate a new population of N individuals. This mechanism for generating the population is a type of crossover operator which replaces the traditional GA crossover operator.

The process in detail is as follows:

1. Choose the M best individuals in the current population which are in the set $D_{l-1}^{S_e}$.
2. Estimate the probability distribution of the current population using Eq. 7.
3. Generate a new population of N individuals from the probability distribution which is stored in the set $D_l^{S_E}$.

The probability distribution is expressed as the product of invariable marginal probabilities, which is estimated from the marginal frequencies:

$$p_l(x_i) = \frac{\sum_{j=1}^{M} \delta_j(X_i = x_i | D_{l-1}^{S_e})}{M} \tag{7}$$

where $\delta_j(X_i = x_i|D^{S_e}_{l-1})$ is 1 for the j^{th} individual in M if the value of gene X_i is equal to x_i, in any other case it will be 0.

The outline of our UMDA-based algorithm can be seen in Algorithm 1.

Algorithm 1. Proposed Estimation of Distribution Algorithm

```
1. Generate initial population of size $N$
2. Do $itersNow = 0$
3. While $itersNow < maxIters$
    (a) Choose $\frac{N}{2}$ individuals
    (b) Estimate the probability distribution of the $M$ individuals
    (c) Sample the distribution in order to obtain new individuals
    (d) Replace the old population with the new one
4. End While
5. Return best individual
6. End
```

4 Experiments

In order to evaluate and analyze the implemented algorithms, the yeast expression data set has been used, comprising 17 experiments (columns) on 2900 genes (rows). This gene expression data set was chosen since it is one of the most used in literature by the majority of experts in this field, thereby enabling our results to be compared.

The results obtained with the proposed tools have been compared using the algorithm proposed by Church in [3] as a reference algorithm.

Following an empirical study, the following parameters were fixed: a population of 200 individuals and 200 generations. The crossover and mutation probabilities were fixed at 0.8 and 0.6, respectively.

Each algorithm was executed 30 times, and the seed of the random number generator was changed in each execution. At the end of each execution, the best bicluster found was recorded.

4.1 Results

This section includes the results obtained by the proposed algorithms and the reference algorithm. We also performed a random sampling of biclusters in order to check the expected residue value for a random bicluster.

Table 1 shows the corresponding residues for the best and worst biclusters, the mean and typical deviation on 30 executions, and also an indication of the time taken for each execution. The average size of the resulting biclusters are also displayed. Results for Reference algorithm were taken over 100 bicluster while 10^4 random solutions were generated.

Figure 1 shows the histograms of residue (a), rows (b) and columns (c) of the best biclusters found by every algorithm. Results for k-opt for $k \geq 3$ were omitted for visualization purposes (although they were extremely similar to those of 2-opt). These figures give us a global view of the whole set of best biclusters available, enabling to quickly check out the region of the search space covered by every strategy.

Several aspects can be highlighted from the table and the histograms. First one is that the GA achieves very good residue values despite the simplicity of its components' definitions. On average, the biclusters generated are quite big (825 rows and 8 columns).

The joint use of GA with local search, leading to different memetic schemes, does not seem to be useful. The simpler local search schemes (k-opt) increase the residue while the average number of rows in the bicluster is significantly reduced. In this case, and given the standard deviation values, it is clear that there is a problem of convergence which is independent of the k value. Moreover, no statistical differences were detected for different values of k.

As the complexity of the local search is increased, from 2-opt to TS, the residue values also increase. This becomes clear if we look at the corresponding histogram. In turn, the sizes of the biclusters obtained are slightly higher than those obtained by k-opt.

The EDA strategy achieves the lowest average residue value, while the corresponding bicluster sizes are about 200 rows and 8 columns. The average residue for the reference algorithm is almost three times higher than that of EDA, while the biclusters are smaller on average(although the number of columns is increased from 8 to 12). The reference algorithm presents the highest variability in residue, number of rows and columns (this is clearly seen in the histograms).

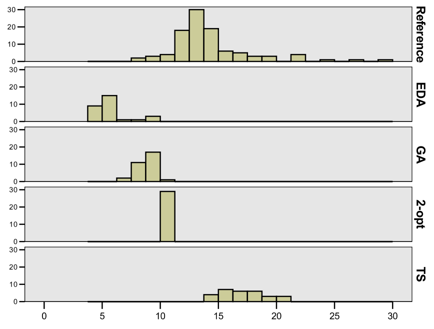
In order to determine what differences in terms of residue are significant, a Kruskal-Wallis test was performed. The test reveals significant differences among the median of the residues of the algorithms. Then, pairwise U Man-Witney non parametrical test were performed and they confirm that the differences among the algorithms were significant.

Another element to analyze is the computational time used. The faster algorithm, and the best one on bicluster quality, is EDA, followed by the GA. The addition of local search to GA increases the computational time considerably while not having the same counterpart in biclusters quality. The Church's algorithm has quite acceptable running times.

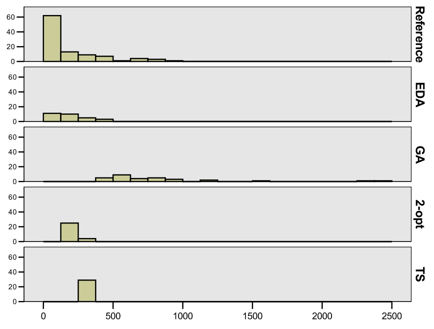
In Fig. 2 we plot the volume of the best solutions (calculated as $rows \times columns$) against residue for algorithms GA, EDA and Reference). This plot reveals several things. First one is the existence of many alternative solutions with similar residue. See for example the vertical range for residue between 5-10. This fact is most notably for GA and EDA. In second place we can see that the Reference algorithm is able to obtain very similar solutions in size while very different in residue. Both facts clearly encourage the use of population based techniques that allow to simply manage a set of solutions of diverse characteristics.

Table 1. Statistical Values of residue and size of the biclusters found by every algorithm. Time is in minutes per run.

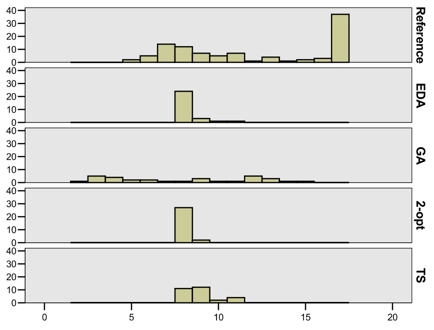
	Residue			Average Size		
Algorithm	Avg (sd)	Best	Worst	Rows (sd)	Cols (sd)	Time
GA	8.83 (0.81)	7.30	11.82	825.19 (488.32)	7.94 (4.11)	5
EDA	5.72 (1.45)	4.81	9.87	213.28 (109.74)	8.28 (0.70)	3
GA+2-opt	10.46 (0.04)	10.35	10.50	235.62 (9.95)	8.07 (0.26)	10
GA+3-opt	10.45 (0.05)	10.36	10.50	241.59 (10.14)	8.07 (0.26)	14
GA+4-opt	10.45 (0.05)	10.37	10.49	243.48 (11.71)	8.14 (0.35)	15
GA+5-opt	10.47 (0.04)	10.33	10.50	240.83 (15.67)	8.04 (0.21)	17
GA+TS	17.07 (1.94)	13.90	20.68	280.20 (10.99)	8.00 (0.12)	14
Church	14.19 (3.59)	7.77	29.78	166.70 (226.37)	12.09 (4.40)	5-10
Random	23.51 (2.60)	5.75	34.26	1407.30 (841.16)	9.48 (4.60)	–



(a)



(b)



(c)

Fig. 1. Histograms of residue (a) , row (b) and column's (c) values of the best biclusters obtained by every algorithm. Results for k-opt for $k \geq 3$ were omitted. TS and 2-opt stands for the memetic algorithm using such local search scheme.

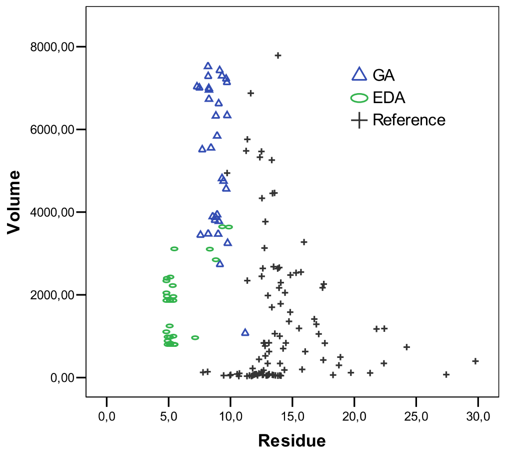


Fig. 2. Volume (*row* × *column*) vs Residue of the best biclusters found by EDA, GA and Reference Algorithm

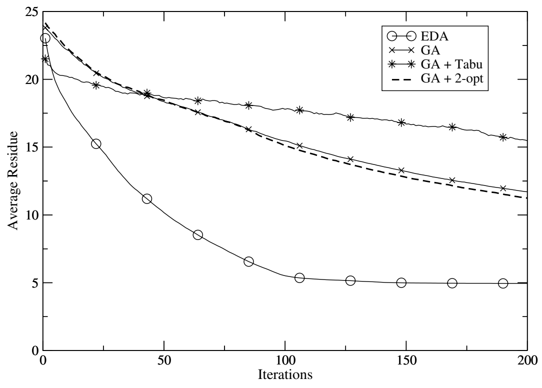


Fig. 3. Time vs Residue EDA, GA, GA+TS and GA + 2-opt Algorithms

Finally, Fig. 3 shows the evolution of the average residue over the time for typical runs of EDA, GA, GA+TS and GA + 2-opt. The faster convergence is achieved by EDA; given that no restart mechanism is included, EDA becomes stagnated during the last half of the time available. The curves for GA and GA+2-opt are pretty similar: both algorithms show a continuous but very slow convergence. Also, GA+TS is the worst method: it seems like the algorithm can not made the population to converge. We have two hypothesis for these behaviors: first one there may be a problem in the local search parameters; second one may be related with the fact that a small change in the genotype can give raise to a big change in the phenotype and, when this fact occurs, the use of local search is not recommendable. Both situations are under study but we suspect the second reason may be more relevant.

5 Conclusions and Future Research

The population based techniques tested here showed as robust and efficient strategies for coping with the obtention of biclustering in microarray matrices.

More specifically, the simple EDA implemented was able to obtain better solutions than the other algorithms (including the reference one) while using less computational time.

We are aware that the analysis of biclustering results is somehow controversial because it is not clear what makes a bicluster "good" or not. In principle, it seems desirable for the biclusters to be as large as possible and to maintain low residue levels. This would indicate that a high number of genes have been identified with a similar expression profile in a large number of conditions.

Also, Aguilar [11] proved that the mean squared residue is not precise enough, from the mathematical point of view, to discover shifting and scaling patterns simultaneously and he pointed out that the characterization of an objective function H that allows to detect both patterns simultaneously would be very beneficial. To the best of our knowledge, such function is still not available.

A byproduct of using these population-based techniques is that, at the end of each run, we have available a set of high quality solutions. This is extremely important because the ultimate goal is to obtain a set of genes "biologically" related, so the optimization point is somehow secondary. In this context, the use of more complex strategies like memetic algorithms without having problem specific operators seems not to be recommendable.

Acknowledgments

This work was supported in part by projects TIC2003-09331-C02-01 and TIN2005-08404-C04-01 from the Spanish Ministry of Science and Technology.

References

1. S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithm. In A. Prieditis and S. Russel, editors, *The Int. Conf. on Machine Learning*, pages 38–46. Morgan Kaufmann Publishers, 1995. San Mateo, CA.
2. S. Busygin, G. Jacobsen, and E. Kramer. Double conjugated clustering applied to leukemia microarray data. *SIAM ICDM, Workshop on clustering high dimensional*, 2002.
3. Y. Cheng and G. Church. Biclustering of expression data. *8th International Conference on Intelligent System for Molecular Biology*, pages 93–103, 2001.
4. G. R. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. *IEEE-EC*, 3(4):287, November 1999.
5. W. Hart, N. Krasnogor, and J. Smith, editors. *Recent Advances in Memetic Algorithms*. Studies in Fuzziness and Soft Computing. Physica-Verlag, 2004.
6. J. Hartigan. *Clustering Algorithms*. John Wiley, 1975.
7. S. Madeira and A. Olivera. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on computational biology an bioinformatics.*, 1(1):24–45, 2004.

8. H. Muehlenbein and G. Paab. From recombination of genes to the estimation of distributions. In *Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature*, volume IV, pages 178–187. PPSN, 1996.
9. H. Muhlenbein. Evolutionary computation: The equation for response to selection and its use for prediction. *Evolutionary Computation*, (5):303–346, 1998.
10. H. Nagesh, S. Goil, and A. Choudhary. pmafia: A scalable parallel subspace clustering algorithm for massive data sets. *International Conference on Parallel Processing*, pages 477–, 2000.
11. J. A. Ruiz. Shifting and scaling patterns from gene expression data. *Bioinformatics*, 21(20):3840–3845, 2005.
12. E. Schikuta. Grid-clustering: An efficient hierarchical clustering method for very large data sets. *In Proc.13th Int. Conf. Pattern Recognition, IEEE Computer Society.*, 2:101–105, 1996.
13. R. Sharan and R. Shamir. Click: A clustering algorithm with applications to gene expression analysis. *Proceedings of the Eighth International Conference on Intelligent Systems*, pages 307–316, 2000.
14. H. Wang, W. Wang, J. Yang, and P. Yu. Clustering by pattern similarity in large data sets. *SIGMOD Conference*, 2002.
15. W. Wang, J. Yang, and R. Muntz. Sting: A statistical information grid approach to spatial data mining. *In Proc. 23rd Conf. Very Large Databases*, pages 186–195, 1997.
16. J. Yang, H. Wang, W. Wang, and P. Yu. Improving performance of bicluster discovery in a large data set. *Proceedings of the 6th ACM International Conference on Research in Computational Molecular Biology (RECOMB)*, 2002.