Improving an Evolutionary Multi-objective Algorithm for the Biclustering of Gene Expression Data

Carlos A. Brizuela[1], Jorge E. Luna-Taylor[2], Israel Martinez-Perez[1], Hugo A. Guillen[1], David O. Rodriguez[1], Armando Beltran-Verdugo[1]

1: Computer Sciences Department - CICESE
Ensenada, B.C., Mexico
e-mail:cbrizuel@cicese.mx

2: Department of Systems and Computation - ITLP
La Paz, B.C.S., Mexico
e-mail: eluna@itlp.edu.mx

***Abstract*—*The development of new technologies for the design of DNA microarrays has boosted the generation of large volumes of biological data, which requires the development of efficient computational methods for their analysis and annotation. On these sets of data, the bicluster construction algorithms attempt to identify coherent associations of genes and experimental conditions. In this paper, we introduce an improved version of a multi-objective genetic algorithm to find large biclusters that are, at the same time, highly homogeneous. The proposed improvement uses a group based representation for the genes-conditions associations rather than long binary strings. To assess the proposal performance the algorithm is applied to generate biclusters for two real gene expression data: Saccharomyces Cerevisiae with 2884 genes and 17 conditions, and the human B cells Lymphoma with 4096 genes and 96 conditions. The results of computational experiments show that the proposed approach outperforms current state-of-the-art algorithms on these data sets.*

Keywords- biclustering; gene expression; multi-objective genetic algorithm; group based representation; microarray DNA.

I. INTRODUCTION

The increased use of microarray technology has generated a large volume of biological data, which necessitates the development of efficient computational methods for their functional interpretation. To address this challenge many techniques have been proposed. Among them, clustering has become one of the most used approaches as a first step in the work of discovering new knowledge. However, the results of clustering methods applied to genes have been limited. This limitation makes it difficult to analyze the expression of genes for a given set of experimental conditions, mainly because the expression patterns do not associate the genes over all conditions, rather on a subset of them. To overcome this situation various algorithms have been proposed to cluster genes and conditions simultaneously. These algorithms are called bicluster algorithms and have the aim to identify groups of genes that exhibit a high correlation across a set of given conditions.

The search for biclusters in gene expression data is a very attractive computational challenge. There are a vast amount of methods proposed to deal with this problem. The work of Cheng and Church [1] is of much relevance since it introduces the concept of bicluster applied to the analysis of gene expression for the first time, and proposed an original algorithm for its construction. Despite some limitations, as discussed by Rodriguez et al. [2] and by Aguilar [3], this algorithm been used as a benchmark for evaluating and comparing the performance of a wide variety of more recent and elaborated algorithms.

Madeira and Oliveira [4] presented a classification of biclustering methods mainly based on two aspects: *i*) the type of biclusters that the algorithms are able to find, and *ii*) the computational technique used. There are algorithms that seek biclusters with constant values, e.g. the mClustering [5], based on a divide and conquer approach, and the DCC [6] that uses a combination of clustering of rows and columns. Other methods identify biclusters with columns or rows with constant values, such as the CTWC [7], the δ-Patterns [8], which is a greedy approach, and Gibbs [9]. Some methods such as δ-biclusters [1] and FLOC [10, 11], use greedy approaches, the pClusters [12] uses exhaustive search, Plaid Models [13] and PRM [14, 15] are based on the identification of probability distribution parameters. There are also methods that seek biclusters with patterns of coherent evolution such as OPSMs [16] and xMotifs [17], both using a greedy search, and SAMBA [18] and OP-Clusters [19], which perform exhaustive search.

Rodriguez et al. [2] add to this classification methods that use stochastic search. In this branch, algorithms such as the SEBI [20] and Simulated Annealing [21] are included.

Despite the existence of a large number of biclustering algorithms, there are still many significant challenges to overcome [2]:

- The scarce information available to define the type of specific biclusters to search.
- The amount of noise in the data matrices.
- The large computation time due to the complex calculations often required.
- Missing data in the input matrices.
- The existence of user parameters that strongly influence the final results.
- The scarce number of assessment methods for the generated results.

- The multi-objective nature of the problem, since the *MSR* and the bicluster size, must be optimized at the same time.

In this paper, we introduce an improved version of a recently proposed evolutionary algorithm for the biclustering problem. The improvement uses a more appropriate representation and its corresponding genetic operators. The biclusters are represented by two sequences, one represents the genes and the other the conditions which are present in the bicluster. The algorithm seeks to simultaneously minimize a homogeneity measure of the bicluster known as *MSR*, and to maximize the bicluster size. To show the effectiveness of the proposed approach a set of experiments are performed on two reference sets data (*Yeast Saccharomyces cerevisiae* and *Human Lymphoma B-cells*). The next section formally defines the problem to solve.

II. BICLUSTERING ANALYSIS OF GENE EXPRESSION

Cheng and Church [1] introduced the concept of bicluster within the context of gene expression data analysis. A bicluster is a subset of genes along with a subset of conditions with a high level of similarity. The similarity is considered as a consistency measure between genes and conditions in the bicluster.

Within this context, we can define biclustering as the process of grouping genes and conditions simultaneously, searching for biclusters of maximum size and maximum similarity within a data matrix of gene expression.

Madeira and Oliveira [4] present a formal definition of the bicluster problem. The input data is defined by a matrix A of n by m, where each element a_{ij} is a real value. In the case of gene expression arrays, a_{ij} represents the level of expression of gene i under condition j.

The matrix A with n rows and m columns is defined by its set of rows, $X = \{x_1,\ldots,x_n\}$ and its set of columns, $Y = \{y_1,\ldots,y_m\}$. (X, Y) is used to denote the matrix A. If $I \subseteq X$ and $J \subseteq Y$ are subsets of rows and columns of A, respectively, then $A_{IJ} = (I, J)$, which denotes the submatrix A_{IJ} of A containing only the elements a_{ij} that belong to the submatrix with the set of rows I and the column set J.

Given the matrix A, a cluster of rows is a subset of rows that have a similar behavior through the set of all columns. This means that a cluster of rows $A_{IY} = (I, Y)$ is a subset of rows defined by the set of all columns Y, where $I = \{i_1,\ldots,i_k\}$ is a subset of rows $I \subseteq X$ and $k \leq n$. A cluster of rows (I, Y), can thus be defined as a submatrix k by m of the data matrix A. Similarly, a cluster of columns is a subset of columns which have a similar behavior across the set of all rows. A cluster $A_{XJ} = (X, J)$ is a subset of columns defined on the set of all rows of X, where $J = \{j_1,\ldots,j_s\}$ is a subset of columns ($J \subseteq Y$ and $s \leq m$). A cluster of columns $A_{XJ} = (X, J)$ can be defined as a submatrix of n by s of the data matrix A.

A bicluster is a subset of rows that have a similar behavior through a subset of columns, and vice versa. The bicluster $A_{IJ} = (I, J)$ is a subset of rows of X and a subset of columns of Y, where $I = \{i_1,\ldots,i_k\}$ is a subset of rows ($I \subseteq X$ and $k \leq n$), and $J = \{j_1,\ldots,j_s\}$ is a subset of columns ($J \subseteq Y$ and $s \leq m$). A bicluster (I, J) can be defined as a submatrix of k by s of the data matrix A.

The specific problem addressed by the biclustering algorithms is defined as: given a data matrix A it is required to identify a set of biclusters $B_k = (I_k, J_k)$ such that each bicluster B_k satisfies some property of homogeneity. The exact features of homogeneity of biclusters vary according to the statement of the problem.

In this work we concentrate on optimizing two properties of the bicluster:

i) The homogeneity $G(I, J)$ of bicluster (I, J) is expressed as a mean squared residue (MSR) score defined as:

$$G(I,J) = \frac{1}{|I| \times |J|} \sum_{i \in I, j \in J} \left(e_{ij} - e_{iJ} - e_{Ij} + e_{IJ}\right)^2$$

where,

$e_{iJ} = \frac{1}{|J|}\sum_{j \in J} e_{ij}$, $e_{Ij} = \frac{1}{|I|}\sum_{i \in I} e_{ij}$, and

$e_{IJ} = \frac{1}{|I| \times |J|}\sum_{i \in I, j \in J} e_{ij}$

ii) The bicluster size $|B| = |I| \times |J|$

The MSR has to be minimized while the bicluster size maximized.

Although the complexity of the biclustering problem depends on the exact formulation of the problem, and specifically the function used to evaluate the quality of a bicluster, the variant analyzed here is NP-hard.

III. RELATED WORK

Recently there have been several algorithms based on a variety of techniques to find biclusters, for example, BBC [22], Reactive GRASP [23], RAP [24], GS Binary PSO [25] and TreeBic [26], among others.

In general, it is difficult to evaluate and compare biclustering methods, since the obtained results strongly depend on the scenario under consideration. Prelic et al. [27] present an evaluation and comparison of five outstanding methods. The evaluated methods are: CC [1], Samba [18], OPSM [16], ISA [28, 29] and xMotif [17]. To evaluate the methods both artificial and real data sets are used. The artificial data include biclusters with constant and additive values. Also, a systematic increase in noise with an increasing overlap between the created biclusters is considered. As for the real data, biological information takes into account GO annotations [30, 31], maps of metabolic pathways [31], and information on protein-protein interaction [32, 31]. In general, the methods ISA, Samba and OPSM perform well. While some methods perform better under certain scenarios, they show lower performance in others.

Mitra and Banka [33] introduce a multiobjective evolutionary algorithm (MOEA) with the addition of local search. The objective is to find large size biclusters, with *MSR* values below a predefined threshold. Their method was evaluated using two sets of gene expression data referenced in the literature: *Saccharomyces Cerevisiae* and *Human B*

Cell Lymphoma. The yeast data they use is a collection of 2884 genes under 17 conditions, with 34 null entries identified with value -1, indicating a missing value. The expression data of Human B cells [34] contains 4026 genes under 96 conditions, with 5.08% of missing values. The results of this method are compared with FLOC [11], DBF [35] and CC [1], using as comparison criteria the *MSR*, and the size of the biclusters obtained by each method. In addition, they determined the biological significance of the biclusters in connection with information on the yeast cell cycle. The biological relevance is evaluated based on the statistical significance determined by the GO annotation database [36]. As for the comparison based on the *MSR* and the size of the biclusters obtained, the MOEA results outperform the ones generated by other methods.

Dharan and Nair [23] proposed the Reactive GRASP method. Statistical significance of the generated biclusters is assessed to see how well they correspond with the known gene annotation [33]. For this purpose the package SGD GO gene ontology term finder [36] is used. The performed tests show that the Reactive GRASP is able to find biclusters with higher statistical significance than the basic GRASP [23] and the CC [1] methods.

Das and Idicula [25] propose a greedy search algorithm combined with PSO (GSPSO). The tests are conducted on expression data of the cell cycle of the *Yeast Saccharomyces Cerevisiae*. The data used is based on [34], and consists of 2884 genes under 17 conditions. The results are compared with those of SEBI [20], CC [1], FLOC [11], DBF [35], and Modified Greedy [25]. The comparison criteria are the *MSR* (named as MSE) presented by [1], and the bicluster size. The GS Binary PSO outperforms the other methods, except the DBF, on the *MSR*, and shows competitive results in the size of the biclusters found.

Caldas and Kaski [26] propose TreeBic, a hierarchical model. The method assumes that the samples or conditions in a microarray are grouped in a tree structure, where nodes correspond to subsets in the hierarchy. Each node is associated with a subset of genes, for which, samples are highly homogeneous. The tests were conducted on a collection of 199 miRNAs profiled from 218 human tissues from healthy and tumor cell lines. The results are compared with those obtained by Samba [18], Plaid [13], DC [1], and OPSM [16] methods. TreeBic performs better both, in terms of the proportion of biclusters enriched to at least one tissue or GO category, and in terms of the total number of tissues and GO categories enriched. Despite these results, the TreeBic method ranks second regarding the number of generated biclusters.

In a recent work [37] a multi-objective genetic algorithm (MOGB), based on the well known NSGA-II [38], has been proposed. The algorithm, that outperforms some state-of-the-art approaches, uses a standard binary encoding scheme. This encoding scheme generates long binary strings mainly composed of zeros. By observing this fact we propose a different encoding scheme whose length depends only on the number of genes and conditions actually considered in the bicluster, consequently producing shorter chromosomes.

IV. PROPOSED ALGORITHM

We propose a multi-objective genetic algorithm, where each individual in the population encodes a bicluster. The goal is to minimize the *MSR* and to maximize the bicluster size, both at the same time. Unlike the MOEA proposed in [33], the proposed algorithm does not require a local search to keep biclusters under the *MSR* threshold δ. Instead, the selection process prefers individual with their MSR under the threshold over individuals violating the threshold. This represents two important advantages, first it avoids the use of the parameter α required in local search, which influences the results. Second, it reduces the computation time, allowing the use of a larger number of individuals and generations. The algorithm details are presented in the following subsections.

A. Representation of biclusters

A bicluster is represented by two sequences of integers, one for the genes (G) and the other for the conditions (C). If the gene sequence has a value *j* it indicates that gene *j* is part of the bicluster, the same applies for the condition sequence. Note that under this representation the individuals are of variable size. Fig. 1A shows an example of the sequences representation. The bicluster corresponding to the sequences of Fig. 1A is shown in Fig. 1C, it was extracted from the expression matrix presented in Fig. 1B.

B. The main steps

Algorithm 1 starts by creating a population of *n* biclusters. Each bicluster is created by selecting at random two genes and two conditions of the matrix expression, so that the *MSR* do not exceed the threshold δ. If the threshold is exceeded the selected pair are discarded, and the process repeated until a bicluster with an *MSR* value below the threshold is obtained. Since the initial size of biclusters is small the chances to get one under the threshold is high.

Algorithm 1: Enhanced MO Genetic Biclustering (eMOGB)

Input: A gene expression matrix, *MSR* threshold δ, *n, pc, pm, ng*
Output: A set of optimized biclusters

1. generate a random initial population of *n* individuals with *MSR* below δ
2. compute the nondominated fronts
3. compute the crowding distance for each individual
4. **repeat**
5. select the best biclusters
6. apply crossover with probability *pc*
7. apply mutation with probability *pm*
8. combine parent and children populations
9. compute the nondominated fronts of the combined populations
10. compute the crowding distance for the individuals in the combined population
11. sort the biclusters of the combined population
12. define the new population of *n* biclusters
13. **until** the number of generations without improvement is *ng*
14. **return** the biclusters corresponding to the nondominated individuals of the last generation

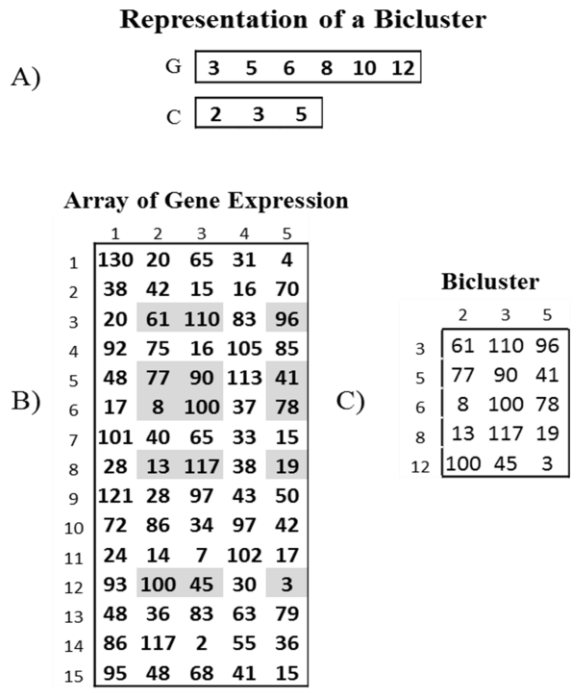

Figure 1. Representation of a bicluster, G = genes, C = conditions. A) The integer array representing the bicluster. B) An array of gene expression data. C) Bicluster values comprising selected expression (shaded) values of the matrix in B).

The nondominated front is calculated based on the concept of dominance. An individual *i* dominates individual *j*, if either of the following conditions hold:

1. The *MSR* of *i* (MSR_i) is less than or equal to MSR_j, and the size of *i* ($size_i$) is larger than $size_j$.
2. $size_i$ is greater than or equal to $size_j$, and MSR_i is less than MSR_j.

An exception to these two conditions is the following rule. If individual *i* has only one gene or one condition and indivual *j* has more than one, then *j* dominates *i*. This especial case of domination helps the algorithm to avoid having a population of individuals mainly with one-gene or one-condition.

For an individual (bicluster) to belong to a nondominated front, it should not be dominated by any other in the population. Once the individuals are identified in the first front, they are discarded to initiate the identification of individuals in the second front. This process is repeated successively until there are no more dominated individuals.

Line 3 computes the crowding distance of each individual as it is done by Mitra and Banka [33]. This distance is a measure of the degree of saturation of the search space (in terms of bicluster size and *MSR*). The closer the *MSR* and size of an individual is to the rest of the population, the lower its crowing distance becomes. This distance is used as a means to maintain diversity in the population.

Once the nondominated fronts and the crowding distance are computed, the selection of the best individuals is performed. The selection is done by applying the binary tournament with crowding distance [38]. First, the population is randomly rearranged, and two adjacent individuals are selected to participate in the tournament. An individual *i* is chosen over an individual *j* if it meets any of the following conditions:

1. MSR_i is below the threshold δ, and the MSR_j is above the threshold.
2. Both *MSRs* are on the same side of the threshold δ, and *i* is in a front with lower index than *j*.
3. Both *MSRs* are on the same side of the threshold of δ, both belong to the same front, and the crowding distance of *i* is greater than the one corresponding to *j*.

Crossover is applied (with probalility *pc*) to the selected individuals in line 6. For this process, individuals are taken in pairs (parents) and two new biclusters (offspring) are generated. Two random crossover points are selected from Parent 1, one from the gene sequence and the other from the condition sequence. The selected crossover points contain the alleles that work as pivots, P.G and P.C, for the genes sequence and for the conditions sequence, respectively. Child 1 takes from Parent 1 alleles that are less than or equal to the pivot while Child 2 receives alleles from Parent 1 greater than the pivot. Child 1 is completed with alleles form Parent 2 greater than the pivot, while Child 2 receives from Parent 2 alleles less than or equal to the pivot. This way, it is guaranteed that no repeated alleles will appear in the offspring. Fig. 2 shows an example of this newly proposed crossover operator.

Mutation is applied (line 7) with probability *pm* to the individuals in the children population. Mutation of a bicluster is done by selecting a random index from the set of genes or conditions (genes are selected 80% and conditions 20%, every time the mutation is applied). If the index is already in the bicluster then it is erased, otherwise it is added to the bicluster. Therefore, this operator adds a new gene or condition, or removes a selected gene or condition.

Fig. 3 shows an example of a mutation in a bicluster. In this example the gene number 10 is randomly selected, and added to the array, since it was not originally present in the bicluster. This introduces in the bicluster the values of expression of gene number 10 in the matrix expression for the selected conditions (shaded values).

After the mutation is performed a process which combines both populations (parent and children) is carried out. This process consists in considering only as a single population all individuals from both populations. For this combined population, of size 2*n*, nondominated fronts and crowding distances are recalculated.

Subsequently, biclusters are ordered for this combined population, according to the following criteria:

1. First filter out the individuals with an MSR value above the threshold δ.
2. Then fit those in the lower fronts.

3. If the population is overflow (population size > *n*) with individuals belonging to a given front then fit first those with a larger crowding distance.

The resulting *n* individuals after these steps will be considered the next generation of biclusters. This process stops after a number of generations *ng* without changes in the size of the largest bicluster with *MSR* below δ is reached (line 13).

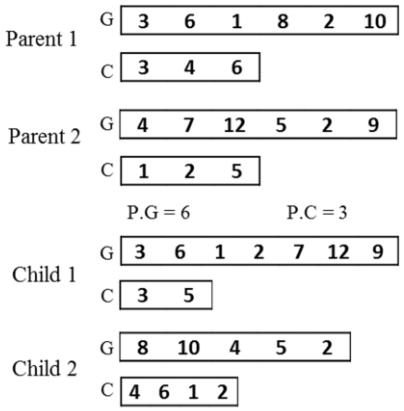

Figure 2. Example of a crossover between two individuals. The pivot for the gene sequence is P.G = 6, while for the condition sequence P.C = 3.

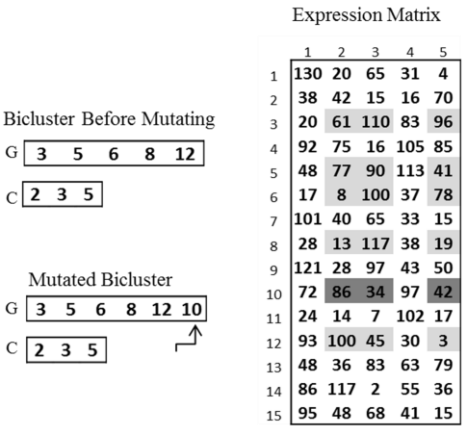

Figure 3. Example of mutation. Gene number 10 is randomly selected from the set of genes and added to the genes sequence.

V. EXPERIMENTAL SETUP AND RESULTS

The enhanced multi-objective genetic algorithm was applied to two real data sets. The first benchmark tested was the expression of 2884 genes under 17 conditions from *Yeast Saccharomyces Cerevisiae*, containing 34 nulls. The second set corresponds to the expression of 4026 genes under 96 conditions of *Human B cells Lymphoma*, with 19,667 null values corresponding to 5.08% of the full set. Both sets of data were taken from the site http://arep.med.harvard.edu/ [34]. The experiments were performed using an *MSR* threshold of δ = 300 for the Yeast, and a threshold δ = 1200 for the Lymphoma. Although there is no a profound justification from the point of view of biology, these values have been extensively used to evaluate and compare a variety of biclustering methods. In the case of the Yeast assembly, null values were replaced by random values (identified by -1) in the range 0 to 800. In the case of Lymphoma null values (identified by 999) were replaced by random values in a range of -800 to 800. Both threshold values selected for the *MSR*, as well as the strategy and range to replace the missing values were established as they were just described. This is done to perform a fair comparison with results reported in other studies.

The algorithm was run 30 times for each data set, using a population size of 50 individuals, and setting a value of *ng* = 400 for the number of generations without improvement in the bicluster size, as a termination criterion. The crossover and mutation probabilities were of *pc* = 1.0 and *pm* = 0.5, respectively. The method was coded and implemented in C#, experiments were performed under Windows XP OS version Service Pack 2 , using Visual Studio Ultimate 2010 in a PC of 2.41 GHz of speed and 2.5 GB of RAM. The algorithm receives as input a text file with the matrix of expression data to be processed. Returns as output another text file with the built biclusters, the values that were used to replace the null values in the array, and a descriptive information on the best biclusters built.

The average *MSR* value, the number of genes, number of conditions, and the average and maximum size of the discovered biclusters were used as assessment criteria. Table I shows a comparison of the results obtained from the Yeast dataset. For this comparison the FLOC algorithms [10], DBF [35], MOEA [33], and the one presented by Cheng and Church [1] are considered. This is a representative group of algorithms for biclustering, which have been analyzed frequently in the literature. The results reported for these algorithms were taken from the work of Mitra and Banka [33]. MOGB [37] is a recently proposed evolutionary approach, the one improved in this work. Average results shown in Table I are taken in MOEA [33], MOGB [37], and eMOGB over all nondominated solutions. The nondominated solutions in MOGB [37] and in eMOGB are the consolidated solutions of 30 runs for each algorithm.

Table I. Comparative results of biclustering methods on data from the *Yeast Saccharomyces Cerevisiae*, using a threshold *MSR* δ = 300.

Method	Average MSR	Average bicluster size	Size of the largest bicluster /Ave. CI
FLOC [10]	187.54	1825.78	2000/0.103
DBF [35]	114.70	1627.20	4000/0.071
Cheng-Church [1]	204.29	1576.98	4485/0.129
MOEA [33]	234.87	10301.71	14828/0.023
MOGB[37]	282.45	14112.60	16488/0.020
eMOGB	290.68	**15189.40**	**16944/0.019**

Table II. Best biclusters found on the data set of the Yeast Saccaromice Cerevisae, using a threshold *MSR* δ = 300.

Method	MSR	Bicluster size	CI
MOEA [33]	286.27	14828	0.019
MOGB[37]	299.95	16728	**0.018**
eMOGB	299.83	**16944**	**0.018**

The proposed algorithm, eMOGB, outperforms the other algorithms in the size of the biclusters discovered under the defined threshold (see third column in Table I). The eMOGB obtains larger biclusters (average and best cases), even larger than those of MOEA and MOGB, which already exceeds the performance of previous algorithms. The *CI* average values is also better for eMOGB (see column 4 in Table I). The *CI* (Consistency Index) introduced by Mitra and Banka, represents the relationship between the *MSR* of a bicluster and its size. This ratio indicates how well the two requirements of biclusters are met: *i*) the expression levels of genes are similar over a range of conditions, i.e., must have a low *MSR*, and *ii*) the size is as large as possible.

Table III. Best biclusters found on the data set of the *Human B-Lymphoma cells*, using a threshold *MSR* δ = 1200.

Method	MSR	Bicluster size	CI
MOEA [33]	1199.98	37560	0.032
MOGB[37]	1199.38	43834	0.027
eMOGB	1199.82	**45708**	**0.026**

A bicluster is considered better as its *CI* value is smaller. A very important advantage of MOGB and eMOGB with respect to MOEA is that they do not require a local search to keep the biclusters below the threshold, which avoids the handling of parameter α (used in various methods [1], [33], [27]), whose proper choice largely influences the results. The yeast data best biclusters, according to the *CI* criterion, generated by each method are shown in Table II. We can see here that both methods MOGB and eMOGB outperform the MOEA [33].

The algorithm results obtained with the Lymphoma data were compared with the results reported by Mitra and Banka [33], which were the best results in the literature (see Table III). This table shows that eMOGB outperforms the best MOEA and MOGB results, both in terms of the largest size of biclusters found, as in *CI* value.

Table IV. BCoverage measures for the Yeast and Lymphoma data sets. A = MOGB, B= MOEA y D = eMOGB

Method	C(A,B)	C(B,A)	C(A,D)	C(D,A)
Yeast	0.889	0	0.118	0.466
Lymphoma	1	0	0.219	0.667

Table V. Binary epsilon indicator for the Yeast and Lymphoma data sets. A = MOGB, B = MOEA y D = eMOGB

Method	I_ε (A,B)	I_ε (B,A)	I_ε (A,D)	I_ε (D,A)
Yeast	1.184	0.844	0.927	1.079
Lymphoma	1	1.094	1.037	0.965

Although these results are informative what each algorithm is actually generating is a set of nondominated solutions, therefore we need some criteria to properly compare two sets of solutions. To this aim the set coverage (C), and the binary epsilon-indicator (I_ε) measures were used. For the coverage indicator [39], a value C(A,B) = 1 means that all decision vectors in B are weakly dominated by A.

On the other hand, C(B,A) = 0 means that none of the solutions of A are dominated by solutions in B. If both conditions hold at the same time then solutions in A dominate all solutions in B. Although the coverage indicator is capable of detecting dominance between approximation sets, it does not provide any information regarding the closeness of fronts generated by the algorithms. To fill this gap the multiplicative binary-epsilon indicator $I_\varepsilon(A,B)$ was also computed. This criterion gives the minimum factor ε by which the objective functions of the approximation set B can be multiplied such that the solutions in A weakly dominate them. Since the epsilon indicator is not symmetric, it is necessary to compute $I_\varepsilon(B,A)$ as well (see [39] for details).

The results for these two measures are shown in tables IV and V. Table IV shows the coverage values, in the case of Yeast we can see that almost all nondominated solutions generated by MOEA are equal to or dominated by nondominated solutions generated by MOGB, C(A,B) = 0.889, while no solution of MOGB is dominated by solutions of MOEA, C(A,B) = 0. This result can also be observed in Figure 4, where we can see that all but a couple of points in MOEA are dominated by solutions in MOGB. The situation is better for MOGB in the case of the Lymphoma data, here all solutions of MOEA are dominated by solutions of MOGB, C(A,B) = 1, C(B,A) = 0. This is supported by what is observed in Figure 5. When comparing MOGB with eMOGB we can see that results favors eMOGB, C(A,D) = 0.118 and C(D,A) = 0.466 for the Yeast, and C(A,D) = 0.219 and C(D,A) = 0.667 for the Lymphoma. Regarding the closeness of fronts, for both data sets, we can see that the largest distance is between the MOEA and both MOGB and eMOGB (see columns 1 and 2 in Table V). However, the fronts generated by MOGB and eMOGB are not far from each other as can be observed in columns 3 and 4 in Table V, and in figures 6 and 7.

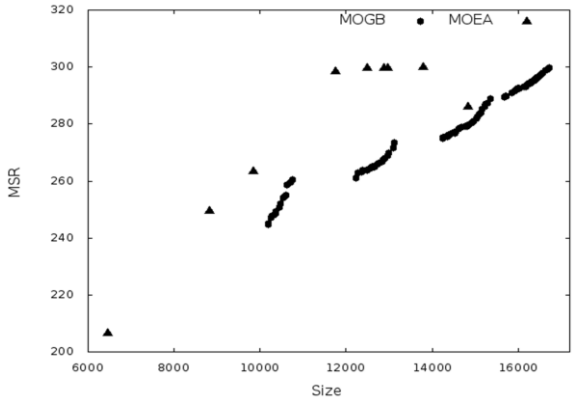

Figure 4. Comparison of nondominated fronts generated by MOGB and MOEA for the Yeast Saccaromice Cerevisae data set. These are the consolidated nondominated fronts after 30 runs of each algorithm.

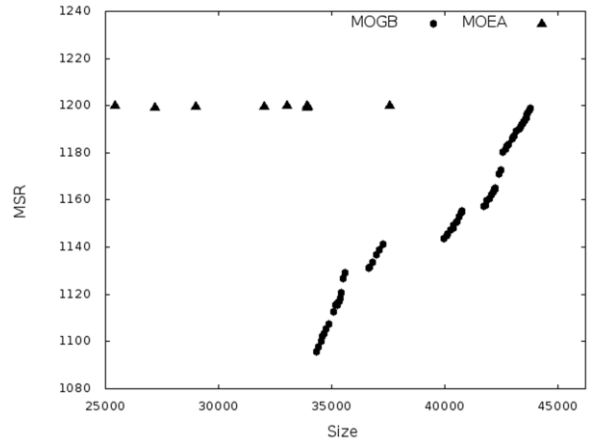

Figure 5. Comparison of nondominated fronts generated by MOGB and MOEA for the Lymphoma data set. These are the consolidated nondominated fronts after 30 runs of each algorithm..

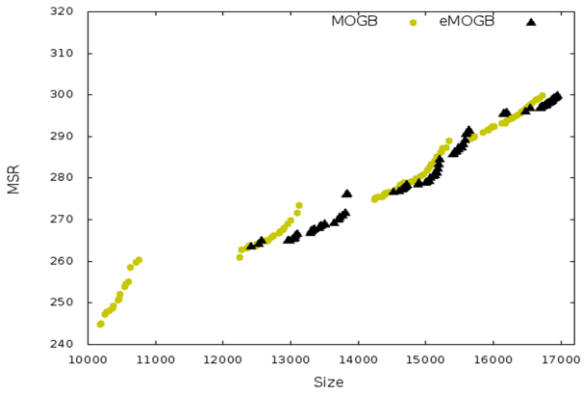

Figure 6. Comparison of nondominated fronts generated by MOGB and eMOGB for the Yeast Saccaromice Cerevisae data set. These are the consolidated nondominated fronts after 30 runs of each algorithm.

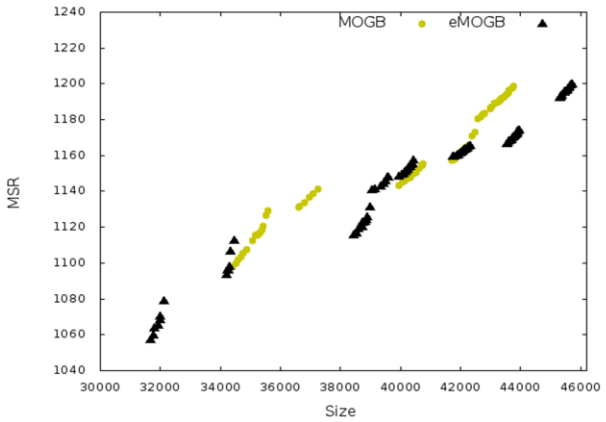

Figure 7. Comparison of nondominated fronts generated by MOGB and eMOGB for the Lymphoma data set. These are the consolidated nondominated fronts after 30 runs of each algorithm.

VI. Conclusions

A new encoding scheme for a multi-objective genetic algorithm applied to the biclustering of gene expression data has been proposed. The encoding follows a group based representation and incorporates appropriate crossover and mutation operators. The performance of the proposed algorithm is tested on two real gene expression data, which have been widely used as benchmarks for this problem. Experiments were focused on the discovery of large biclusters with *MSR* below predefined thresholds for both sets of data. The results have shown that the proposed algorithm performs better than others currently reported in the literature in terms of the average bicluster size and the largest bicluster size when the MSR value is kept under a predefined threshold. When both criteria, bicluster size and MSR are optimized at the same time our algorithm shows also the best performance considering the MOEA and MOGB results.

An important feature of our algorithm is that it does not require a local search, contrary to some current algorithms which require maintaining the *MSR* below the threshold by means of this technique.

Future work will assess the biological significance of the generated biclusters, based on ontological annotations on these and new data sets.

Acknowledgment

This work was partially supported by the National Council of Science and Technology under grant SEP-CONACYT-CB-2010-154737. The authors would like to thanks Najash Marron for his collaboration in the algorithm implementation.

References

[1] Y. Cheng and G. M. Church, "Biclustering of expression data," Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB'00), 2000, pp. 93–103.

[2] D. S. Rodriguez, J. C. Riquelme, and J. S. Aguilar, "Analisis de datos de expresion genética mediante tecnicas de biclustering," tech. rep., Universidad de Sevilla, 2000.

[3] J. Aguilar, "Shifting and scaling patterns from gene expression data," Bioinformatics, vol. 21, 2005, pp. 3840–3845.

[4] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: a survey," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 1, no. 1, 2004, pp. 24–45.

[5] J. A. Hartigan, "Direct clustering of a data matrix," Journal of the American Statistical Association (JASA), vol. 67, no. 337, 1972, pp. 123–129.

[6] S. Busygin, G. Jacobsen, and E. Kramer, "Double conjugated clustering applied o leukemia microarray data," in Proceedings of the 2nd SIAM International Conference on Data Mining, Workshop on Clustering High Dimensional Data, 2002.

[7] G. Getz, E. Levine, and E. Domany, "Coupled two-way clustering analysis of gene microarray data," Proceedings of the Natural Academy of Sciences USA, 2000, pp. 12079–12084.

[8] A. Califano, G. Stolovitzky, and Y. Tu, "Analysis of gene expression microarays for phenotype classification," in Proceedings of the International Conference on Computacional Molecular Biology, 2000, pp. 75–85.

[9] Q. Sheng, Y. Moreau, and B. D. Moor, "Biclustering microarray data by gibbs sampling," Bioinformatics, vol. 19, no. 2, 2003, pp. ii196–ii205.

[10] J. Yang, W. Wang, H. Wang, and P. Yu, "δ-clusters: Capturing subspace correlation in a large data set" Proceedings of the 18th IEEE International Conference on Data Engineering, 2002, pp. 517–528.

[11] J. Yang, W. Wang, H. Wang, and P. Yu, "Enhanced biclustering on expression data," Proceedings of the 3rd IEEE Conference on Bioinformatics and Bioengineering, 2003, pp. 321–327.

[12] H. Wang, W. Wang, J. Yang, and P. S. Yu, "Clustering by pattern similarity in large data sets," Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, 2002, pp. 394–405.

[13] L. Lazzeroni and A. Owen, "Plaid models for gene expression data," Statistica Sinica, vol. 12, 2002, pp. 61–86.

[14] E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller, "Rich probabilistic models for gene expression," Bioinformatics, vol. 17, no. Suppl. 1, 2001, pp. S243–S252.

[15] E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller, "Decomposing gene expression into cellular processes," Proceedings of the Pacific Symposium on Biocomputing, vol. 8, 2003, pp. 89–100.

[16] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini, "Discovering local structure in gene expression data: The order-preserving submatrix problem," in Proceedings of the 6th International Conference on Computacional Biology (RECOMB'02), 2002, pp. 49–57.

[17] T. M. Murali and S. Kasif, "Extracting conserved gene expression motifs from gene expression data," Proceedings of the Pacific Symposium on Biocomputing, vol. 8, 2003, pp. 77–88.

[18] A. Tanay, oded Sharan, and R. Shamir, "Discovering statistically significant biclusters in gene expression data," Bioinformatics, vol. 18, no. Suppl. 1, 2002, pp. S136–S144.

[19] J. Liu and W. Wang, "Op-cluster: Clustering by tendency in high dimensional space," Proceedings of the 3rd IEEE International Conference on Data Mining, 2003, pp. 187–194.

[20] F. Divina and J. S. Aguilar, "Biclustering of expression data with evolutionary computation," IEEE Transactions on Knowledge and Data Engineering, vol. 18, 2006, pp. 590–602.

[21] K. Bryan, P. Cunningham, and N. Bolshakova, "Biclustering of expression data using simulated annealing," 18th IEEE Symposium on Computer Based Medical Systems (CBMS'05), 2005, pp. 383-388.

[22] J. Gu and J. S. Liu, "Bayesian biclustering of gene expression data," BMC Genomics, vol. 9, no. Suppl. 1, 2008, p. S4.

[23] S. Dharan and A. S. Nair, "Biclustering of gene expression data using reactive greedy randomized adaptive search procedure," BMC Bioinformatics, vol. 10, no. Suppl. 1, 2009, p. S27.

[24] G. Pandey, G. Atluri, M. Steinbach, C. L. Myers, and V. Kumar, "An association analysis approach to biclustering," ACM SIGKDD, ACM New York, NY, USA, 2009, pp. 677-686

[25] S. Das and S. M. Idicula, "Greedy search-binary pso hybrid for biclustering gene expression data," International Journal of Computer Applications, vol. 2, no. 3, 2010, pp. 0975–8887.

[26] J. Caldas and S. Kaski, "Hierarchical generative biclustering for microrna expression analysis," RECOMB, 2010, pp. 65–79.

[27] A. Prelic, S. Bleuler, P. Zimmermann, A. Wille, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler, "A systematic comparison and evaluation of biclustering methods for gene expression data," Bioinformatics, vol. 22, 2006, pp. 1122–1129, 2006.

[28] J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai, "Revealing modular organization in the yeast transcriptional network," Nature Genetics, vol. 31, 2002, pp. 370–377.

[29] J. Ihmels, S. Bergmann, and N. Barkai, "Defining transcription modules using large-scale gene expression data," Bioinformatics, vol. 20, 2004, pp. 1993–2003.

[30] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock, "Gene ontology: tool for the unification of biology," Nature Genetics, vol. 25, no. 1, 2000, pp. 25–29.

[31] A. P. Gasch, P. T. Spellman, C. M. Kao, O. Carmel-Harel, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown, "Genomic expression programs in the response of yeast cells to environmental changes," Molecular Biology of the Cell, vol. 11, 2000, pp. 4241–4257.

[32] A. Wille, P. Zimmermann, E. Vranova, A. Furholz, O. Laule, S. Bleuler, L. Hennig, A. Prelic, P. von Rohr, L. Thiele, E. Zitzler, W. Gruissem, and P. Buhlmann, "Sparse graphical gaussian modeling of the isoprenoid gene network in Arabidopsis thaliana," Genome Biology, vol. 5, 2004, p. R92.

[33] S. Mitra and H. Banka, "Multi-objective evolutionary biclustering of gene expression data," Journal of the Pattern Recognition Society, vol. 39, 2006, pp. 2464–2477.

[34] Harvard Molecular Technology Group and Lipper Center for Computational Genetics. http://arep.med.harvard.edu

[35] Z. Zhang, A. Teo, B. Ooi, and K. Tan, "Mining deterministic biclusters in gene expression data," Proceedings of the fourth IEEE Symposium on Bioinformatics and Bioengineering (BIBE'04), 2004, pp. 283–292.

[36] SGD GO Termfinder.
http://db.yeastgenome.org/cgi-bin/GO/goTermFinder

[37] J. E. Luna-Taylor and C. A. Brizuela, "A Multiobjective Genetic Algorithm for the Biclustering of Gene Expression Data," In proceedings of the 3rd International Supercomputing Conference in Mexico ISUM 2012.

[38] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation 6(2), 2002, pp. 182–197.

[39] E. Zitzler, L. Thiele, M. Laumanns, C.M. Foneseca, V. Grunert da Fonseca, "Performance Assessment of Multiobjective Optimizers: An Analysis and Review," IEEE Transactions on Evolutionary Computation 7(2), 2003, pp. 117–132.