MOSCFRA: A Multi-objective Genetic Approach for Simultaneous Clustering and Gene Ranking

Kartick Chandra Mondal, Anirban Mukhopadhyay, Ujjwal Maulik, Sanghamitra Bandhyapadhyay, Nicolas Pasquier

MOSCFRA : A Genetic Approach for Simultanious Clustering and Gene Ranking

Kartick Chandra Mondal[1], Anirban Mukhopadhyay[2], Ujjwal Maulik[3], Sanghamitra Bandhyapadhyay[4], and Nicolas Pasquier[1]

[1] I3S Laboratory (CNRS UMR-6070), University of Nice Sophia-Antipolis, Nice-06108, France,
keto004@gmail.com, nicolas.pasquier@unice.fr
[2] Department of Theoretical Bioinformatics, German Cancer Research Center (DKFZ), Heidelberg-69120, Germany,
a.mukhopadhyay@dkfz-heidelberg.de
[3] Department of Computer Science and Engineering, Jadavpur University, Kolkata-700032, India,
umaulik@cse.jdvu.ac.in
[4] Machine Intelligence Unit, Indian Statistical Institute, Kolkata-700108, India,
sanghami@isical.ac.in

Abstract. Microarray experiments generate a large amount of data which is used to discover the genetic background of diseases and to know the gene characteristics. Clustering the tissue samples is an important tool for partitioning the dataset according to co-expression patterns. This clustering task is even more difficult when we try to find the rank of each gene (Gene Ranking) according to their abilities to distinguish different classes of samples. Finding clusters for samples and rank of each gene for a specific gene expression data in a single process is always better. In the literature many algorithms are available for finding the clusters and gene ranking or selection separately. A few algorithms for simultaneous clustering and feature selection are also available. In this article, we propose a new approach to cluster the samples and rank the genes, simultaneously. A novel encoding technique is proposed here for the problem of simultaneous clustering and ranking. Results have been demonstrated for both artificial and real-life gene expression data sets.

Keywords: Multi-objective Evolutionary Algorithm, Gene Ranking, Clustering, Gene Expression Data.

1 Introduction

Now a days, Bio-technology and Molecular Biology [1] is getting much importance in the field of research. A small unit, called gene, is use to repesent the genomic information. The microarray technology generates the global and simultaneous view of expression levels for thousands of genes over different time points of different biological experiments. This is an important tool in the research area of Molecular Biology and Bio-Technology [1]. The biological information of a cell

or a gene, is described by the micro-array expression pattern which is called gene expression data. Analysis of such data finds the relationships among the patterns present in the data. This data analysis has two parts: forming gene expression matrix from raw data generated by microarray technology and analysis of this matrix.

Appropriate mining strategies, e.g. clustering [2] and gene selection [3] are needed for analysis of such information. Clustering of co-expressed genes into biologically meaningful groups, helps in inferring the biological role of an unknown gene that is co-expressed with the known gene(s). Clustering is a process for organizing the objects from an object set into set of subsets of objects where the objects of a subset are similar but objects from different subsets are dissimilar in some ways. The clustering process is sometimes also called the *unsupervised learning process*. Clustering helps to partition the input space into K regions, $C_1, C_2, \cdots, C_K$, on the basis of some similarity/dissimilarity metrics, where the value of K may or may not be known previously. One frequently used such measure is called distance functions (dist(x, y) for x = $(x_1, x_2, \cdots, x_d)$ and y = $(y_1, y_2, \cdots, y_d)$). This distance function mainly depends on the type of applications where it is used, i.e., in numerical data, categorical data or in text document. Examples of such kind of useful distance functions are Eu- clidean distance, Manhattan distance, Mahalanobis distance,Mankowski Distance, Hamming distance and Maximum norm. One important issue in cluster analysis is the evaluation of clustering results to find the partitioning that best fit the underlying data. The process of evaluating cluster is known as *cluster validity* [4]. Several clustering algorithms are proposed in the literature. These algorithms are divided into different types according to their nature of operation (e.g. Hierarchical, Partitional, Density-Based, Grid-Based). A brief discussion on each of them are available in [5]).

Another important subject of matter is the *gene ranking* [6]. Gene Selection is a combinatorial problem. So, instead of selecting a subset of genes, we can give the weight or rank depending on the relevance, which is called *gene weighting* or *gene ranking* [7–9]. Gene ranking is used because of its simplicity, scalability, and good empirical success. Most of the gene ranking methods are based on the wrapper approaches or filter methods. Some heuristic methods for gene weighting are: a) Gradient descent on the input space [10]. b) AdaBoost when each model is trained on one feature only [11].

In this article, we have proposed a multi-objective approach for simultaneous clustering and gene ranking. To the best of our knowledge, the process of simultaneous clustering and gene ranking by using multi-objective optimization is new in this area. The rest of the article is organized as follows: In Section 2, we present an overview on multi-objective Evolutionary paradigm with different concepts of MOO (Multi-objective Optimization). Section 3 presents a detailed discussion of our proposed algorithm with different components used in the algorithm. Section 4 presents the experimental design methods and results obtained during the experiments with a small discussion on them. Section 5 concludes the

article and gives some future direction for further improvement of the proposed method.

2 Multi-Objective Optimization

Genetic Algorithms (GAs) are very popular meta-heuristic optimization method but could not apply directly for multi-objective problems. Traditional GA are modified to reuse for multi-objective problems by using specialized fitness functions and introducing methods to promote solution diversity. Two general approaches are available for optimizing multiple objective. The first method is to combine every objective function into a single composite function (e.g., utility theory, weighted sum method). The second solution is to move all but one by one objective to the constraint set, a constraining value must be established for each of these former objectives. In all cases, the optimization method would return a single solution rather than a set of solutions that can be examined for trade-offs. For this reason, decision-makers often prefer a set of good solutions considering all the multiple objectives.

Most of the real world engineering problems are generally have multiple conflicting objectives, e.g., minimize cost, maximize performance, etc.. So, another solution for solving such multi-objective problem is to determine an entire *Pareto Optimal Solution Set* or a representative subset. In the Pareto optimal solution set, while moving from one solution to another, there is always a certain amount of sacrifice in some objective(s) to achieve a certain amount of gain in the other(s).

Consider that we want to optimize k objectives that are non-commensurable and equally important. Without loss of generality, we consider that all objectives are of the minimization type.

We also assume that the solution of this problem can be expressed by *decision variable vector* $\{x_1, x_2, \cdots, x_n\}$. The solution space X is generally restricted by a series of *constraints*, such as $g_j(x) = b_j$ for j = 1, $\cdots$, m and bounds on the decision variables. A function $f\colon X \to Y$ evaluates the quality of a specific solution by assigning it an *objective vector* $(y_1, y_2, \cdots, y_k)$ in the *objective space* Y. Our aim is to find a vector x^* that minimizes a given set of k objective functions $y(x^*) = y_1(x^*), \cdots, y_k(x^*)$.

A formal definition of Pareto optimality from the viewpoint of the minimization problem may be given as follows: A decision vector $\overline{x}^*$ is called Pareto optimal if and only if there is no $\overline{x}$ that dominates $\overline{x}^*$, i.e., there is no $\overline{x}$ such that $\forall i \in \{1, 2, \ldots, k\}$, $y_i(\overline{x}) \leq y_i(\overline{x}^*)$ and $\exists i \in \{1, 2, \ldots, k\}, y_i(\overline{x}) < y_i(\overline{x}^*)$. In words, $\overline{x}^*$ is Pareto optimal if there exists no feasible vector $\overline{x}$ which causes a reduction on some criterion without a simultaneous increase in at least another. In general, Pareto optimum usually admits a set of solutions called *non-dominated* solutions.

In multi-objective problem, our aim is to investigate a set of solutions, where each of which satisfies the objectives at an acceptable level without being dominated by any other solution. A solution is said to be *Pareto optimal* if it is

not dominated by any other solution in the solution space. The *Pareto optimal solution set* in the decision space X is denoted as the *(Pareto set)* $X^* \subseteq$ X, and we will denote its image in objective space as *Pareto front* $Y^* = f(X^*) \subseteq$ Y. With many multi-objective optimization problems, knowledge about this set helps the decision maker in choosing the best compromise solution. For most of the multi-objective problems, entire Pareto optimal set identification is practically almost impossible for its size. For many problems, especially for combinatorial optimization problems, proof of solution optimality is computationally infeasible. Therefore, a practical approach is to investigate a set of solutions or *the best-known Pareto set* for multi-objective optimization that represent the Pareto optimal set as well as possible. Therefore, the goal of the optimization problem is to find or *approximate* the Pareto set. The outcome of a MOEA is considered to be a set of mutually non-dominated solutions also called *Pareto set approximation.*

2.1 Non-dominated Sorting Genetic Algorithm-II

For most of the multi-objective problems, entire Pareto optimal set identification is practically impossible for its size. Therefore, the goal of the optimization is to find an *approximate* Pareto set. The outcome of a Multi-Objective Evolutionary Algorithm (MOEA) is considered to be a set of mutually non-dominated solutions. The Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [12], a popular MOEA method, is used here as the underlying optimization strategy. The brief algorithmic description of NSGA-II [12] is provided in algorithm 1.

Actually, in NSGA-II [12], a random population P_0 is created with N chromosome and known as the initial parent population. According to their non-domination level, they are sorted and give a rank to each one solution under the population equal to their non-domination level. At first, they create a child population Q_t of the same size as parent by using selection, crossover and mutation operations. Then combine the parent and child population and create a population of size R_t and sort according to their non-domination level. Now the next parent population P_{t+1} is created by selecting the chromosome from R_t one by one according to their level. But it is not necessary that L_1 (the population of level 1) to L_i (the last level of the selected population for P_{t+1}) be the exact size of the population. So, here a crowded comparison method in descending order is included for selecting population from level L_i to choose the best solutions needed to fill all population slots. This crowded comparison operator is used to introduce the diversity among the non-dominated solutions (called Diversity Preservation), in selection phase and also in population reduction phase.

3 Proposed Technique

We propose a novel approach that simultaneously identify the cluster of each sample and rank of each feature (gene) according to their participation to create clusters of samples. As per our knowledge, the process of simultaneous clustering

Algorithm 1 Algorithm NSGA-II

1: Create a random parent population P_0 of size N. Set t = 0.
2: Apply crossover and mutation on P_0 to create offspring population Q_0 of size N.
3: **if** The stopping criterion is satisfied **then**
4: stop and return to P_t.
5: **end if**
6: Set $R_t = P_t \bigcup Q_t$.
7: Using the fast non-dominated sorting algorithm, identify the non-dominated fronts $F_1, F_2, \cdots, F_k$ in R_t.
8: **for** $i = 1$ to k **do**
9: Calculate crowding distance of the solutions in F_i.
10: Create P_{t+1} as follows:

Case 1: If $|P_{t+1}| + |F_i| \leq N$, then set $P_{t+1} = P_{t+1} \bigcup F_i$;
Case 2: If $|P_{t+1}| + |F_i| > N$, then add the least crowded $N - |P_{t+1}|$ solutions from F_i to P_{t+1}.

11: Use binary tournament selection based on the crowding distance to select parents from P_{t+1}.
12: Apply crossover and mutation to P_{t+1} to create offspring population Q_{t+1} of size N.
13: Set t = t + 1, and go to Step 3.
14: **end for**

and gene ranking by using multi-objective optimization is new in this area. Here we identify the cluster of the samples and rank the genes, simultaneously. A novel encoding technique is proposed here for the problem to fit into multi-objective frame work. Since, the Multi-objective Evolutionary Algorithms (MOEA) are known as the global search heuristics primarily used for optimization tasks. We use this process for our simultaneous optimization.

Here, our aim is to propose a method to simultaneously optimize the feature ranking and clustering. To optimize the task of finding the cluster and rank the feature according to their ability to create clusters by maintaining the competing constraints is an NP-complete problem. Due to this high complexity, researchers are motivated to use various approximation techniques to generate near optimal solutions. Since, the MOEA is known as the global search heuristics primarily used for optimization tasks, we use this for our simultaneous optimization. The NSGA-II(Non-dominated Sorting Genetic Algorithm-II) [12] is used here as an important MOEA to optimize the chromosome under population. The algorithm is given in algorithm 1. Also it is used as a baseline algorithm to compare with other methods. NSGA-II is computationally efficient algorithm for implementing our idea but one can use other MOEA. Another important point, the number of cluster is fixed, so the chromosome length is also fixed. Here we also present the representation and the general framework of MOEA for our simultaneous clustering and gene ranking task.

The Multi-objective Simultaneous Clustering and Feature Ranking Algorithm (MOSCFRA) is summarized as follows:

1. Initialize the chromosome under population as represented bellow.
2. Execute the NSGA-II algorithm (given in algorithm 1) with some terminating criteria to optimize the rank as well as cluster center through crossover, mutation, selection, elitism as described bellow.
3. Choose the appropriate solution from the Pareto set solutions for the problem.

The description of each component of our proposed technique are given in the subsequent subsections.

3.1 Chromosome Representation and Initial Population

A gene expression matrix is represented by rows and columns corresponding to samples (experimental biological conditions) and genes. Consider, a gene expression matrix D has d genes and s samples. The samples will be partitioned into K clusters and each cluster has a center which is represented by d dimensions. One solution is represented by one chromosome and each chromosome has $(d + (K \times d))$ bits to represent rank of each gene and K cluster center with d dimensions. The first d bit represents the weight of each gene and are used to encode the rank of the genes. The remaining bits are used for cluster centers. The one population is composed of several such chromosomes. The initial population is generated randomly.

3.2 Fitness Computation

Two validity indices, Xie-Beni(XB) [13] and Davis-Bouldin(DB) [14] are used as two objective functions to validate the generated cluster centers. Both of these objective functions are of minimization type.

The Xie-Beni index [13] is a representative index in the category of indices involving the membership values and the dataset. Consider a fuzzy partitioning of the data set $X = x_j; j = 1, \cdots, n$ with $v_i (i = 1, \cdots, K)$ the centers of each cluster and u_{ij} the membership of data point j to cluster i. The fuzzy deviation, d_{ij}, of x_j from cluster i, is defined as the distance between x_j and the center of cluster i, weighted by the fuzzy membership of data point j to cluster i. Here, the crisp version of XB index is used where membership values are either 0 or 1. i.e., the crisp version of fuzzy deviation is, $d_{ij} = ||x_j - v_i||$. For a cluster i, the sum of the squares of deviation of the data points denoted by σ_i, is called variation of cluster i. The total deviation $\pi = (\sum_{i=1}^{K} \sigma_i)$. The separation of the partitions is defined as the minimum distance between cluster centers. i.e., $D_{min} = \min_{i,j=1 \ to \ K, i \neq j} ||v_i - v_j||$

Then XB index is defined as,

$$XB = \frac{\pi}{n \times D_{min}} \tag{1}$$

where n is the number of points in the data set. It is clear that small values of XB are expected for compact and well-separated clusters.

In Davies-Bouldin(DB) [14], the similarity measure R_{ij} between the clusters C_i and C_j is defined based on a measure of dispersion of a cluster C_i and a dissimilarity measure between two clusters d_{ij}. The R_{ij} is non-negative and symmetric. i.e., $R_{ij} = (s_i + s_j)/d_{ij}$ and the value of s for each cluster is calculated as, $s_i = \frac{1}{|C_i|} \sum_{x \in C_i} ||v_i - x||$.

Then the DB index is defined as

$$DB_n = \frac{1}{n} \sum_{i=1}^{n} R_i \tag{2}$$

$$R_i = \max_{i,j=1 \ to \ n, i \neq j} R_{ij} \tag{3}$$

Another important idea, *Weighted Distance Method*, is used in our algorithm for computing the validity index. We give a weight to each gene and rank them according to their weight. The weight is also used to calculate the distance between two samples. In our algorithm, we use the Euclidean Distance in weighted form as the distance measure. The equation of Weighted Euclidean Distance is:

$$D(x, y) = \sqrt{\sum_{l=1}^{d} w_l^2 (x_l - y_l)^2} \tag{4}$$

For each chromosome, first we assign the sample in each cluster center present in the chromosome based on nearest center criterion. After assigning the samples, we update the cluster centers according to their sample values by taking the means. The new cluster centers are used to update the chromosome.

3.3 Crossover

In this algorithm, each chromosome in the population has two parts, the gene weight part and the cluster center part. The Uniform Crossover is used for the feature part of the chromosome and Single Point Crossover is used in the cluster center part of the chromosome. In both cases, the same C_p (Crossover Probability) value is used.

After crossover, a pair of parent chromosomes generates a pair of offspring chromosomes. So, the parent population generates the same size of offspring population. This offspring population is used in the mutation process.

3.4 Mutation

Here, a very small mutation probability (M_p) is used. Each time, if mutation is possible the actual value of the mutated bit is replaced by a random value. The range of the random value is between [0,1], since our data sets are normalized. The same technique is used for both part of the chromosome, i.e., gene weight and cluster center part.

3.5 Selection, Elitism and Termination

In our method, we use binary tournament selection with crowded and rank comparison method [12]. After successful completion of the crossover and mutation operation of a generation, the child population is combined with the parent population of that generation. From this combined population, the non-dominated chromosomes are selected and a new population of the same size is created for the next generation. This property of NSGA-II is called the *Elitism*. This technique ensures faster convergence of the process by keeping track of the best solutions generated so far. The NSGA-II has been executed for a fixed number of generations. This fixed number is supplied by the user for terminating the process. After terminating, the process gives a set of non-dominated solutions in the last generation.

3.6 Final Solution Selection

The final solution from the last non-dominated solution set is selected through the CP index and the R index. Both indices are described in the next section. For artificial data, the maximum value of CP index and R index of the solutions are selected but in case of real life micro-array gene expression cancer data, only maximum value of CP index is used. Our approach for simultaneous clustering and gene ranking is unsupervised but the process which is used here for selecting the best solution from the non-dominated set is supervised process. Rank of each genes in a chromosome is evaluated from the first d bits. The highest rank is given to that gene whose weight value is maximum.

The Multiobjective Simultaneous Clustering and Feature Ranking Algorithm (MOSCFRA) is summarized in algorithm 2.

Algorithm 2 Algorithm MOSCFRA

1. Initialize the population of chromosomes.
2. Execute the NSGA-II algorithm to optimize the ranks as well as cluster center.
3. Choose the appropriate solution from the Pareto set solutions for the problem.

4 Experiments and Results

In this section, we present the experimental design procedure and the results of the method with small discussion. For this task, two artificial data sets and two real data sets are used to measure the performance of our proposed method. Two performance measures, CP Index and R Index, are used for this purpose. After that, we compare the performance of the proposed method with several other important methods in this area.

4.1 Experimental Design

Here, we have given the information about the datasets of both real and artificial. Then, the required steps are given for the preprocessing of data sets.

Artificial Datasets
For our experiments, we create two artificial datasets viz., Arda25_30_3 and Arda50_75_5. Arda25_30_3 have 25 genes and 30 samples with 3 classes and Arda50_75_5 have 50 genes and 75 samples with 5 classes. In both the data sets, the genes are artificially generated so that they have different abilities in distinguishing the sample clusters.

Real Life Datasets
From several publicly available real life cancer datasets, two bench mark datasets, viz., Brain tumor and Lung tumor data sets, available at http://algorithmics.molgen.mpg.-de/Static/Supplements/CompCancer/datasets.htm, are used for our experiments. The descriptions and their pre-processing are given here.
Brain tumor: This data set contains 42 tissue samples divided in 5 clusters (primitive neuroectodermal tumours (PNETs) (8 samples), atypical teratoid/rhabdoid tumours (Rhab) (10 samples), malignant gliomas (Mglio) (10 samples), medulloblastomas (MD) (10 samples) and normal tissues (Ncer) (4 samples)). There are total 1379 genes in the data set. Depending on the maximum variation of genes across the sample, the numbers of genes are reduced to 100. Therefore, after pre-processing the data size is 42×100.
Lung tumor: Using oligonucleotide microarrays, mRNA expression levels corresponding transcript sequences in 186 lung tumor samples and 17 normal lung tissues (NL) has been analyzed. The lung tumors included adenocarcinoma (AD) (139 samples), small-cell lung cancer (SCLC) (6 samples), pulmonary carcinoids (COID) (20 samples) and squamous cell lung carcinomas (SQ) (21 samples). The number of genes in the data set is 1543. Here also the same maximum variations of genes across the samples are used as a preprocessing step. After pre-processing, the size is reduced to 203×100.

Both the artificial and real datasets are normalized along the column. So, the value of all the data ranges from 0 to 1.

Parameter Settings
Our experiments are used to measure the quality of our proposed method for identifying the cluster and rank of genes. To compare with the different methods along these lines, we therefore performed the experiments with 100 generations. In each case, twenty trial runs were performed on each expression datasets and the average of the best solution of each run is given in the result. The number of clusters parameter is fixed for particular datasets, the number of clusters for Arda25_30_3 is 3 for instance, and set to 5 for other artificial & real datasets. The crossover rate is 0.8, mutation rate is 0.01 and population size is 50.

Performance Measures

The performance of the algorithm is measured in terms of both clustering and gene ranking ability. These are measured in terms of CP index and a newly defined R index. Only for artificial data sets, these indices calculation are possible, since class label and rank of the features are known. But in the case of real life data sets, since no rank information is available for the genes, the performance of clustering ability is calculated based on the CP index only.

Percentage of CP Index (correctly Classified Pairs) has been used to find the quality of the clustering results. CP index is used to compare a clustering solution with it actual clustering present in the data set. Say for a gene expression data set, the true clustering is C based on domain knowledge and c is a clustering result given by any clustering algorithm. Also assume that, s, d and t be the same, different and total number of pairs that belong to clusters in C and c, respectively. The percentage of CP index is defined as:

$$CP(C, c) = \frac{s + d}{t} \times 100 \tag{5}$$

From the above equation, we can say, higher value of CP means better clustering solution given by the algorithm. So, for CP(C,C) = 100%.

To find the quality of the ranking for a solution, a newly defined index, R index (Rank index) is used. In R index, we compare the generated ranking with true ranking. For this, we first sort the genes according to their ranks in both true and generated rankings. Thereafter, for first g genes, $g = 1, \ldots, d$, the intersection and union of the genes between true set and generated set are calculated and we divide the number of genes in intersection with that in union.

The plot of the corresponding R index is called the R plot. Since the maximum value of R index is 1, the R curve of better solution in the R plot will be nearer to 1.

Competitive Methods

The performance of MOSCFRA is determined by comparing the algorithm with its single objective counter parts that minimizes the objective function DB × XB (SOSCFRA_DX), only DB (SOSCFRA_DB) and only XB (SOSCFRA_XB). All the parameters are exactly same as that of multi-objective method MOSCFRA. Except these, two partitional clustering methods, viz., K-means clustering, fuzzy C-means (FCM) clustering, and five hierarchical clustering methods, viz., single linkage (HICSIL), average linkage (HICAL), complete linkage (HICCOL), centroid linkage (HICCEL), ward linkage (HICWAL) are used.

4.2 *Result and Discussion*

In table 1 and 2, the average value of the CP index over 20 runs on each artificial data set and real life gene expression data set are given, respectively. In brackets, the standard deviation of CP index is also shown. Moreover, higher value of CP index and lower value of standard deviation in all artificial data sets indicate that each time MOSCFRA outperforms other algorithms in terms of clustering.

The values of performance index generated from the other algorithms are inferior to those generated from MOSCFRA. Because, these methods find cluster from the data set by considering the same weight of the features which affect the clustering results. Through this, we can show the significance of the importance of feature ranking. By comparing all the results generated from all the data sets, it is clear that MOSCFRA technique gives the best clustering performance for these data sets.

Since the actual ranks of the features are available for the artificial data sets and it is absent in case of real data sets, we can compute the R index as described above only for the two artificial data sets. The Figure 1 shows the R plot of the R index for the highest CP index of all the runs on each algorithm (MOSCFRA, SOSCFRA_DX, SOSCFRA_DB, SOSCFRA_XB) for the two artificial data sets, respectively. Since the other algorithms does not generate ranks of the features, they are not shown in the figures. From these figures, we can say that the proposed multiobjective algorithm produces the good ranking result.

Another important thing is that, when the DB index and XB index are merged into our single objective counter part, SOSCFRA_DX, it gives the same result as given in the SOSCFRA_DB. So, from this result, we can say the DB index is affecting the goodness of XB index and also DB index is not a good index in such cases.

	CP index for Artificial Data Sets.	
Algorithm	Arda25_30_3	Arda50_75_5
MOSCFRA	**100.00(±0.000)**	**86.6036(±3.428)**
SOSCFRA_DX	31.0345(±0.000)	18.9189(±0.000)
SOSCFRA_DB	31.0315(±0.000)	18.9189(±0.000)
SOSCFRA_XB	76.5977(±0.925)	84.8649(±0.680)
K-means	96.0115(±9.800)	84.3532(±2.371)
FCM	95.6322(±0.000)	80.5045(±0.388)
HICSIL	35.8621(±0.000)	25.3333(±0.000)
HICCOL	71.7241(±0.000)	81.6577(±0.000)
HICAL	91.7241(±0.000)	81.5495(±0.000)
HICCEL	77.2414(±0.000)	66.8468(±0.000)
HICWAL	91.7241(±0.000)	85.2973(±0.000)

Table 1. Experimental Result on Artificial Data Sets.

From the brain tumor genes, the most frequently ranked top ten genes that are responsible for that clustering through our proposed MOSCFRA algorithm are: S81957_at, D38500_at, K02268_at, X64072_s_at, M58297_at, J04132_at, M93119_at, J04444_at, L36847_at, HG3141-HT3317_f_at.

From the lung tumor genes, the most frequently ranked top ten genes that are responsible for that clustering through our proposed MOSCFRA algorithm

Algorithm	CP index for Real Life Data Sets.	
	Brain Tumor	Lung Tumor
MOSCFRA	**82.0209(±8.515)**	**78.4193(±3.618)**
SOSCFRA_DX	19.6283(±0.000)	49.4659(±0.000)
SOSCFRA_DB	19.6283(±0.000)	49.4659(±0.000)
SOSCFRA_XB	81.9698(±0.878)	76.7605(±0.555)
K-means	73.8850(±11.458)	65.5229(±6.531)
FCM	69.1347(±4.047)	60.4251(±4.052)
HICSIL	30.3136(±0.000)	56.5381(±0.000)
HICCOL	44.0186(±0.000)	71.6919(±0.000)
HICAL	30.3136(±0.000)	57.4111(±0.000)
HICCEL	30.3136(±0.000)	57.4111(±0.000)
HICWAL	67.0151(±0.000)	77.1237(±0.000)

Table 2. Experimental Result on Real Life Data Sets.

are: 39022_at, 939_at, 32251_at, 33373_at, 37849_at, 40195_at, 32034_at, 40647_at, 33273_f_at, 34335_at.

5 Conclusion and Future Scope

In this work, we have described a new algorithm for simultaneous clustering and gene ranking. Finding the rank corresponding to the weight is an important task in clustering as well as in the data analysis. In this work, we address the problem of unsupervised gene ranking and unsupervised clustering. Here we have used one general multiobjective framework (NSGA-II) for simultaneous clustering and gene ranking of gene expression dataset. A novel encoding technique is developed for our problem and XB and DB index are used as optimization criteria which are minimized simultaneously. The performance is demonstrated on two artificial data sets as well as two real-life data sets.

As a scope of future work, the algorithm can be extended for unknown number of clusters. Also, other important multiobjective algorithms can be applied and more statistical comparison method can be used. Furthermore, choice of objective functions and selection of final solution from Pareto optimal set need closer look. The authors are working in these directions.

References

1. R. Sharan, Click and expander: a system for clustering and visualizing gene expression data. Bioinformatics, vol. 19, p. 17871799, 2003.
2. A. Ben-Dor and et al, Clustering gene expression patterns. J. Comput. Biol., vol. 6, p. 281297, 1999.
3. R. O. DUDA and P. E. HART, Pattern Classification and Scene Analysis. New York, NY.: John Wiley and Sons, Inc., 1973.

4. S. Theodoridis and K. Koutroubas., Pattern Recognition. Academic Press., 1999.
5. A. Jain, M. N. Murty, and P. Flyn, Data clustering: A review. ACM Computing Surveys, vol. 31, no. 3, pp. 264323, 1999.
6. I. Guyon and A. Elissee, An introduction to variable and feature selection. Journal of Machine Learning Research, vol. 3, pp. 11571182, 2003.
7. C. Zhang, X. Lu, and X. Zhang, Significance of gene ranking for classification of microarray samples, IEEE/ACM TRANS. ON COM. BIO. AND BIOINF., vol. 3, no. 3, pp. 312320, JULY-SEPTEMBER 2006.
8. K. Jong, J. Mary, A. Cornuejols, E. Marchiori, and M. Sebag, Ensemble feature ranking, in Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD), ser. Lecture Notes In Computer Science; Vol. 3202, 2004, pp. 267278.
9. F. E. Streib, M. Dehmer, J. Liu, and M. Muhlhuser, A systems approach to gene ranking from dna microarray data of cervical cancer, in WORLD ACADEMY OF SCI., ENGG. AND TECH. vol.-8, OCTOBER 2005.
10. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, Learning to rank using gradient descent. ICML 2005.
11. Y. Freund and R. E. Schapire., Experiments with a new boosting algorithm, in Proceedings of the Thirteenth International Conference on Machine Learning, 1996, pp. 148156.
12. K. Deb, A. Pratap, S. Agrawal, and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput., vol. 6, pp. 182197, 2002.
13. X. Xie and G. Beni., A validity measure for fuzzy clustering. IEEE Trans. on P.A.M.I., vol. 13, no. 4, pp. 841846, 1991.
14. D. Davies and D. Bouldin., A cluster separation measure. IEEE Trans. on P.A.M.I., vol. 1, no. 2, pp. 224227, 1979

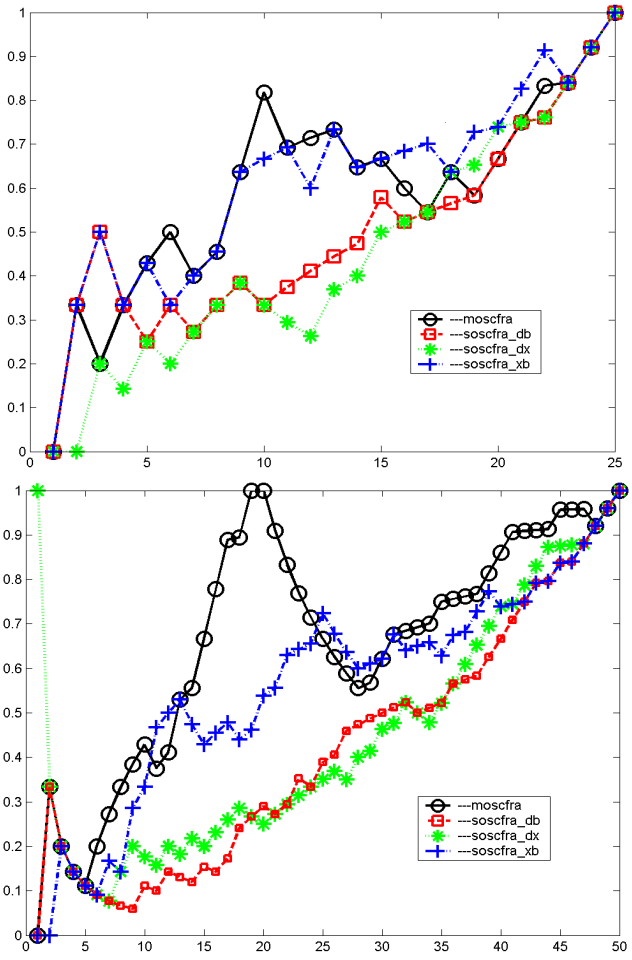

Fig. 1. R plot for the artificial data sets: (above) Arda25_30_3 data, (below) Arda50_75_5 data