Parallel multi-swarm optimizer for gene selection in DNA microarrays

José García-Nieto, Enrique Alba

Abstract The execution of many computational steps per time unit typical of parallel computers offers an important benefit in reducing the computing time in real world applications. In this work, a parallel Particle Swarm Optimization (PSO) is used for gene selection of high dimensional Microarray datasets. The proposed algorithm, called PMSO, consists of running a set of independent PSOs following an island model, where a migration policy exchanges solutions with a certain frequency. A feature selection mechanism is embedded in each subalgorithm for finding small samples of informative genes amongst thousands of them. PMSO has been experimentally assessed with different population structures on four well-known cancer datasets. The contributions are twofold: our parallel approach is able to improve sequential algorithms in terms of computational time/effort (Efficiency of 85%), as well as in terms of accuracy rate, identifying specific genes that our work suggests as signifi-cant ones for an accurate classification.

Additional comparisons with several recent state the of art methods also show competitive results with improvements of over 100% in the classification rate and very few genes per subset.

Keywords Gene selection · Parallel particle swarm optimization · DNA microarrays

J. García-Nieto (✉) · E. Alba
Dept. de Lenguajes y Ciencias de la Computación, ETSI Informática, University of Málaga, Campus de Teatinos, Málaga 29071, Spain
e-mail: jnieto@lcc.uma.es

E. Alba
e-mail: eat@lcc.uma.es

1 Introduction

DNA Microarrays [29] allow scientists to simultaneously analyze thousands of genes, thus providing important insights about cells' functions, since changes in the physiology of an organism are generally associated with changes in large gene ensembles of expression patterns. The vast amount of data that is involved in a typical Microarray experiment usually require from scientists to perform a complex statistical analysis, with the goal of classifying the dataset into correct classes. The key issue in this classification is to identify significant and representative gene subsets that may be later used to predict class membership for new external samples. Furthermore, these subsets should be as small as possible in order to develop fast and low consuming processes for future class prediction. The main difficulty in Microarray classification versus other domains is the availability of a relatively small number of samples in comparison with the number of genes in each sample. In addition, expression data are highly redundant and noisy, and most genes are believed to be uninformative with respect to studied classes, as only a fraction of genes may present distinct profiles for different classes of samples.

In this context, machine learning techniques have been applied to handle large and heterogeneous datasets, since they are able to isolate the useful information by rejecting redundancies [11, 30]. Concretely, feature selection (gene selection in Biology) is often considered as a necessary preprocess step in analyzing large datasets, as this method can reduce the dimensionality of the datasets and often results in better analyses [18, 32, 33].

Feature selection for gene expression analysis in cancer prediction often uses wrapper classification methods [24] to determine a type of tumor, to reduce the number of genes to investigate in the case of a new patient, and also to assist in drug discovery and early diagnosis. Several classification algorithms could be used for wrapper methods, such as K-Nearest Neighbor (K-NN) [15] or Support Vector Machines (SVM) [13]. By defining clusters, a big reduction of the number of considered genes and an improvement of the classification accuracy can be finally achieved. The formal definition of the feature selection problem that we consider here is:

Let $F = \{f_1, \ldots, f_i, \ldots, f_n\}$ be a set of features; find a subset $F' \subseteq F$ that maximizes a scoring function $\Theta : \Gamma \to G$ such that $F' = \operatorname{argmax}_{G \subset \Gamma}\{\Theta(G)\}$; where Γ is the space of all possible feature subsets of F and G a subset of Γ.

Optimal feature selection is a complex problem proved to be NP-hard [28]. Therefore, we need efficient automated approaches to tackle it. Metaheuristics algorithms have proved to be adequate tools for this matter, since they are capable of solving the feature selection accurately and efficiently for the large dimensions needed in Biology. Evolutionary Algorithms (EAs) and, specifically, Genetic Algorithms (GAs) have been successfully used in the past to tackle the gene selection of Microarrays [6, 19–21]. All these approaches consist in using single population sequential algorithms which can achieve competitive performances (from the point of view of the quality of solution), but without considering other important aspects such as the computational effort and the time consumption.

Parallel metaheuristics have always been very popular in the literature [2], and thus there exists a large number of implementations and algorithms. Concretely, population based algorithms are naturally prone to parallelism, since most of their variation operators can be easily undertaken in parallel. However, the truly interesting observation is that the use of a structured population, i.e, a given spatial distribution of individuals, either in the form of a set of islands or a diffusion grid, is responsible for the numerical benefits as evaluated in [2, 3]. The execution of many computational steps per time unit (when in parallel machines with several CPUs or cores) offers an additional benefit in reducing the computing time of such numerically enhanced structured metaheuristics. Unfortunately, not much work has been done so far on parallel structured metaheuristics for feature selection [35], and no related approaches (to the best of our knowledge) have been developed for gene selection of Microarray datasets.

In this work, a structured PSO is used for gene selection of high dimensional Microarrays datasets. The proposed algorithm, called Parallel Multi-Swarm Optimizer (PMSO), consists in running a set of parallel subPSOs algorithms forming an island model, where a migration operation exchanges solutions between those islands with a certain frequency. A feature selection mechanism is embedded in each subalgorithm for finding small samples of informative genes amongst thousands of them. The reported solutions, codifying gene subsets, are then evaluated by means of their classification accuracy by using a SVM classifier, 10-fold cross-validation, and final testing with external test datasets. The contributions of our approach are noticeable, since the parallelization of the so called Geometric PSO [27] will be shown to improve existing algorithms in terms of computational effort and classification accuracy. Besides, the gene ensembles found by this technique are successfully interpreted in the light of independent existing results from Biology to show their actual impact.

The effectiveness of our proposal is analyzed on four well-known public datasets: Leukemia [16], Colon [9], Lymphoma [8] and Lung [17], discovering new and biologically challenging gene subsets, and identifying specific genes that our work suggests as significant ones. Comparisons with several recent state of art methods show the effectiveness of our results in terms of computational time/effort, reduction percentage and classification rate.

The remainder of this paper is organized as follows. In Sect. 2, we provide the reader with basic concepts about Microarrays technology and Geometric Particle Swarm Optimization. Section 3 gives the details of our Parallel Multi-Swarm Optimizer algorithm for feature selection. Experimental results and comparisons are presented in Sect. 4, including performance analyses and biological validation of the genes obtained. Conclusions and further work are finally given in Sect. 5.

2 Preliminaries

In this section, the basic concepts of Microarrays of DNA (application field) and Geometric PSO (involved in the solver technique) are briefly introduced.

2.1 DNA microarrays

A DNA Microarray consists of an arrayed series of thousands of DNA molecules spotted in different positions in a matrix structure [29]. These DNA molecules, that correspond to particular genes, are mixed with cellular cDNA molecules (called labeled or colored DNA) during a hybridization process. A cDNA molecule is obtained from cellular RNA or mRNA during a labeling process showing the relative expression level of each molecule. RNA molecules are isolated from a particular cell type or tissue comprising of a complex mixture of different RNA transcripts. The abundances of individual transcripts in the mixture reflect the different expression levels of the corresponding genes. This process is called hybridization, after which abundant sequences will generate strong signals, while rare sequences will generate weak signals.

Microarrays are normally used to compare gene expression levels within a sample or look at differences in the expression of specific genes across different samples, such as a few samples of one disease or healthy and unhealthy tissues. A gene expressed only in the disease sample, for example, might represent a useful drug target. This is especially appropriate in cancer analysis, since it allows us to discriminate against tumoral tissues and normal ones. Several gene expression profiles obtained from tumors such as Leukemia, Colon, and Lung cancers have been published.

2.2 Geometric particle swarm optimization

The Particle Swarm Optimization [22] is an efficient optimization technique initially developed for continuous optimization problems. A recent version called Geometric Particle Swarm Optimization (GPSO) [27] enables us to generalize PSO to virtually any solution representation in a natural and straightforward way, extending the search to other search spaces, such as combinatorial ones.

However, many practical engineering problems are formulated as combinatorial optimization problems and specifically as binary decisions (which is the case in feature selection). Several binary versions of PSO can be found in the literature [12, 23]. Nevertheless, all these versions are *ad hoc* adaptations from the original PSO and therefore their performance is usually improvable.

The key issue in GPSO consists in using a multi-parental recombination of particles (solutions) which leads to the generalization of a *mask-based crossover* operation, proving that it respects four requirements for being a *convex combination* in a certain space. A convex combination is an affine combination of vectors where all coefficients are non-negative. When vectors represent points in the space, the set of all convex combinations constitutes the convex hull (see [27] for a complete explanation). This way, the mask-based crossover operation substitutes the classical *movement* in PSO and *position update* operations, initially proposed for continuous spaces. This property has been demonstrated for the cases of Euclidean, Manhattan and Hamming spaces in the cited work.

For Hamming spaces, which is the focus in this approach, a *three-parent mask-based crossover* (3PMBCX) is defined in a straightforward way:

Given three parents a, b and c in $\{0,1\}^n$, generate randomly a crossover mask of length n with symbols from the alphabet $\{a,b,c\}$. Build the offspring o by filling each position with the bit from the parent appearing in the crossover mask at the position.

In a convex combination, weights w_a, w_b and w_c indicate (for each position in the crossover mask) the probability of having the symbols a, b or c, respectively.

The pseudocode of the GPSO algorithm for Hamming spaces is illustrated in Algorithm 1. For a given particle i, three parents take part in the 3PMBCX operator (line 9): the current position x_i, the social best position g_i and the historical best position found h_i (of this particle).

Algorithm 1 GPSO for Hamming Spaces

```
1:  $S \leftarrow SwarmInitialization()$
2:  while not stop condition do
3:    for each particle $i$ of the swarm $S$ do
4:      $evaluate(x_i)$
5:      $update(h_i)$
6:      $update(g_i)$
7:    end for
8:    for each particle $i$ of the swarm $S$ do
9:      $x_i \leftarrow$ 3PMBCX$((x_i, w_a), (g_i, w_b), (h_i, w_c))$
10:     $mutate(x_i)$
11:   end for
12: end while
13: Output: best solution found
```

The weight values w_a, w_b and w_c indicate (for each element in the crossover mask) the probability of having values from the parents x_i, g_i or h_i, respectively. These values associated to each parent represent the *present* influence of the current position (w_a), the *social* influence of the global best position (w_b), and the *individual* influence of the historical best position found (w_c). A restriction of the geometric crossover forces w_a, w_b and w_c to be non-negative and add up to one.

In summary, the GPSO developed in this study operates as follows: In the first phase, uniform random initialization of particles is carried out by means of the *SwarmInitialization*() function (Line 1). In a second phase, after the evaluation of particles (line 4), historical and social positions are updated (lines 5 and 6, respectively). Finally, particles are "moved" by means of the 3PMBCX operator (line 9). In addition, with a certain probability, a simple bit-mutation operator (line 10) is applied in order to introduce diversity in the swarm to avoid early convergence. As evaluated in [6], the *three-parent mask-based crossover* used in GPSO makes the offspring inherit the shared selected features present in the three parents involved in the mating.

The GPSO developed in this study operates as follows: in a first phase, uniform random initialization of particles is carried out. In the second phase, after the evaluation of particles, historical and social positions are updated. Finally, particles are "moved" by means of the 3PMBCX operator. In addition, with a certain probability, a simple bit-mutation operator is applied in order to introduce diversity in the swarm to avoid early convergence. As evaluated in [6], the *three-parent mask-based crossover* used in GPSO makes the offspring inherit the shared selected features present in the three parents involved in the mating.

In this case, as defined in 3PMBCX, only one offspring is generated, which represents the new position of the current particle. Here, non-shared features are inherited by the offspring corresponding to the ith parent ($\{a, b, c\}$) of the mask. This way, we can state (and experiments confirm this), that the 3PMBCX crossover operator for Hamming spaces of GPSO is especially suitable for the feature selection problem, showing thus an implicit property of the studied algorithm.

3 PMSO for gene selection

In this section, our new approach named Parallel Multi-Swarm Optimizer (PMSO) for feature selection is presented. We first describe the parallelization model coupled to the GPSO algorithm. After that, the gene selection scheme is explained.

3.1 GPSO parallel model

In analogy with Parallel Genetic Algorithms (PGAs) [4, 31], we define our Parallel Multi-Swarm GPSO (PMSO) as a pair $\langle \mathcal{S}, \mathcal{M} \rangle$, where $\mathcal{S} = \{S_1, \ldots, S_m\}$ is a collection of m swarms (populations) and $\mathcal{M}$ is the migration policy. The main parameters of the migration policy constitute a five-tuple $\mathcal{M} = \langle \sigma, \rho, \phi_s, \phi_r, \tau \rangle$, where $\sigma \in \mathbb{N}$ (*migration gap*) denotes the number of iterations in every subswarm between two successive exchanges of particles (steps of isolated evolution), $\rho \in \mathbb{N}$ (*migration rate*) is the number of particle copies that undergo migration in each exchange; ϕ_s and ϕ_r are two functions which respectively decide how to select emigrant particles and what particles have to be replaced by incoming immigrants. The topology model is denoted by $\tau : \mathcal{S} \to 2^{\mathcal{S}}$, e.g., a unidirectional ring, in our case. Algorithm 2 shows the pseudocode of PMSO.

Two alternatives exist for ϕ_s: *random* and *best*. The first one refers to randomly selecting any particle to be migrated; the second one selects the best-known particle in the island subswarm. Concerning ϕ_r, also two strategies are considered *always* and *if_better*. The former refers to always replacing the worst particle(s) by the incoming emigrant(s); the later replaces the worst particle(s) only if it is worse than the incoming emigrant(s).

Finally, concerning the topology (τ), an unidirectional *ring* is considered where each subswarm sends (and receives) particles to (from) the two consecutive nearest subswarms in the ring. It must be noted that the subswarms proceed asynchronously, giving rise to loosely coupled contiguous epochs of computation and then communication. For this work, we have selected the asynchronous version since it usually provides a better performance than the synchronous one [5].

Algorithm 2 PMSO Pseudocode

```
1:  DO IN PARALLEL for each $i \in \{1, \ldots, m\}$
2:  initialize($S_i$)
3:  while not stop_condition do
4:     iterate $S_i$ for $n$ steps /* GPSO evolution */
5:     for each $S_j \in \tau(S_i)$ do
6:        send $\rho$ particles in $\phi_s(S_i)$ to $S_j$
7:     end for
8:     for each $S_j$ such that $S_i \in \tau(S_j)$ do
9:        receive $\rho$ particles from $S_j$
10:       replace $\rho$ particles in $S_i$ according to $\phi_r$
11:    end for
12: end while
13: Output: best solution ever found in the multi-swarm
```

3.1.1 *Gene selection and classification scheme*

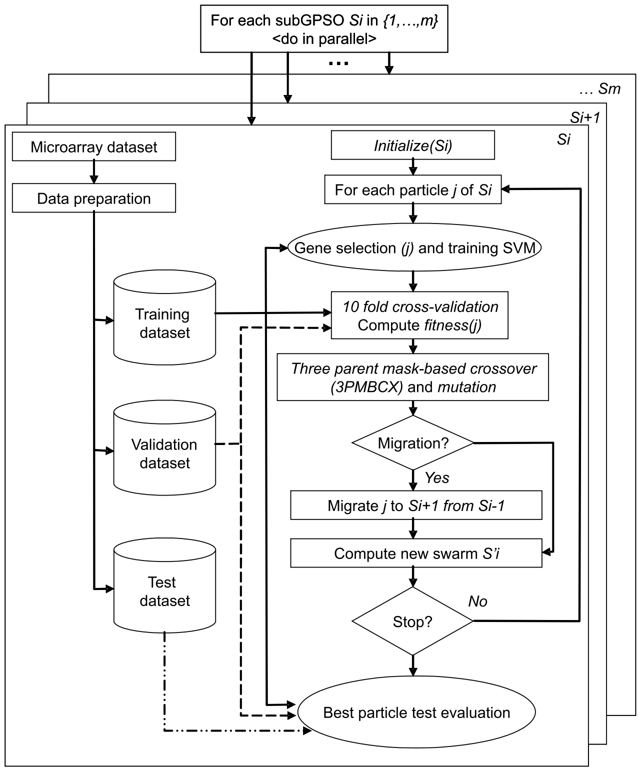
Following the basic scheme of solution encoding used in feature selection, PMSO provides a binary encoded particle (vector) where each bit represents a gene in the dataset. If a bit is 1, it means that this gene is kept in the reduced subset, while 0 indicates that the gene is not included. Therefore, the particle length is equal to the number of genes in the initial Microarray dataset.

As illustrated in Fig. 1, where a general model of our PMSO is provided, the particles of each subswarm (S_i), representing gene subsets, are evaluated by means of a SVM classifier and 10-fold cross-validation as follows: each gene subset (codified by a particle) is divided into ten subsets, nine of them constituting the training set and the remaining one used as the validation set. The SVM is trained using the training set and then the accuracy obtained (number of correct classifications by the SVM once trained) is evaluated on the validation set [13]. This evaluation is repeated ten times, each one alternating the used validation set. This method reinforces the validation process, so that the final accuracy value is the resulting average of the ten validation folds. Such a strong validation is necessary when the number of samples is low regarding the number of features, which is the case for this work. As a final evaluation, the resulting subset solution is evaluated on the external test set, thus obtaining the final accuracy (standard protocol recommended in supervised learning).

3.1.2 *Fitness function*

A fitness function is needed to guide the search by assigning to any tentative solution a quality value. Once the accuracy value and the number of genes are known, the fitness function is calculated according to:

$$f(x) = (100 - acc) + \lambda \cdot \frac{\#(genes\ in\ subset)}{\#(total\ genes)}, \qquad (1)$$



For each subGPSO *Si* in *{1,...,m}* <do in parallel>
... Sm
Si+1
Si
Microarray dataset
Data preparation
Initialize(Si)
For each particle *j* of *Si*
Gene selection (*j*) and training SVM
Training dataset
10 fold cross-validation Compute *fitness(j)*
Three parent mask-based crossover (3PMBCX) and mutation
Validation dataset
Migration?
Yes
Migrate *j* to *Si+1 from Si-1*
Compute new swarm *S'i*
Test dataset
No
Stop?
Best particle test evaluation

Fig. 1 General model of PMSO for gene selection and classification of Microarrays. Training, validation and testing mechanisms are embedded into the parallel algorithm

$$being,\ \lambda = 10^{\lfloor \log(\#(total\ genes))+1 \rfloor}. \tag{2}$$

The objective here consists of maximizing the accuracy and minimizing the number of genes. For convenience (only minimization of fitness), the first factor is presented as $(100 - acc)$ and the second one is normalized in order to control the trade off between these two factors. A constant value λ (which depends on the total number of genes) is used in this normalization. Therefore, if the number of features in the subset is high (with regards to the total number of genes in the original dataset), then the fitness function promotes the reduction of features. Otherwise, if the number of features in the subset is small, then this fitness function promotes the improvement in accuracy.

4 Experimental results

We have implemented the proposed PMSO algorithm for gene selection in C++ following the *skeleton* architecture of the MALLBA library [7]. For the SVM classifier we have used a set of classes provided by the LIBSVM [10] library for training, validation, and testing. These classes were coupled with those of the PMSO for this evaluation phase.

In this section, the experiments are described by first discussing the Microarrays datasets used, then the experimentation setup, the analysis of results and comparisons, and a global discussion.

4.1 Microarray datasets and data preprocessing

The instances used are classified into four well-known datasets taken from real-word Microarray experiments. All of them were taken from the public repository of Kent Ridge Bio-medical Dataset (http://datam.i2r.a-star.edu.sg/datasets/krbd/index.html). In particular:

- The ALL-AML Leukemia dataset consists of 72 Microarrays experiments with 7,129 gene expression levels. Two classes exist: *Acute Myeloid Leukemia* (AML) and A*cute Lymphoblastic Leukemia* (ALL). The complete dataset contains 25 AML and 47 ALL samples. The original dataset is divided into a training set of 38 samples and a test set of 34 samples.
- The Colon Tumor dataset consists of 62 Microarray experiments collected from colon-cancer patients with 2,000 gene expression levels. Among them, 40 tumor biopsies are from *tumors* and 22 (*normal*) are from healthy parts of the colon of the same patient.
- *Types of Diffuse Large B-cell Lymphoma* dataset consists of 47 tissue samples, 24 of them are from *germinal centre B-like group* while the rest 23 are *activated B-like group*. Each sample is described by 4,026 genes.
- The Lung Cancer dataset involves 181 experiments with 12,533 gene expression levels. Classification occurs between *Malignant Pleural Meso-thelioma* (MPM) and *Adenocarcinoma* (ADCA) of the lung. In tissue samples there are 31 MPM and 150 ADCA.

Table 1 summarizes the original organization of training and testing samples of the four used Microarrays. The test sets of Leukemia and Lung were taken from the original repositories provided by the authors. In Colon and Lymphoma, only training sets are available in the original repositories, and for this reason, new test and training sets have been here generated for these two datasets by randomly (uniformly) extracting samples from the original one as stated in Table 1. These datasets were selected because of their different dimensions and gene organizations, constituting a heterogeneous test-bed to better support our conclusions.

Expression levels of training and test sets were normalized separately in order to scale their intensities, thus enabling a fair comparison between the different datasets. Therefore, for each attribute a_j (gene) with $j \in \{1 \cdots \#attributes\}$, and for each sample x_k (expression) with $k \in \{1 \cdots \#samples\}$, a scaling operation to $[-1, 1]$ was performed resulting $a'_j(x_k)$ by using the following equation (as LIBSVM recommends):

$$a'_j(x_k) = 2 \cdot \frac{a_j(x_k) - min_j}{max_j - min_j} - 1, \tag{3}$$

Table 1 Usage details of the four Microarray Datasets

Dataset	#Genes	Clases	#Train	#Test	#Total
Colon	2,000	Cancer	20	20	40
		Normal	11	11	22
Lymp.	4,026	Ac B-like	17	6	23
		Ce B-like	19	5	24
Leuk.	7,129	AML	11	14	25
		ALL	27	20	47
Lung	12,533	MPM	16	15	31
		ADCA	16	134	150

where $\max_j$ and $\min_j$ correspond to the maximum and minimum gene expression values for attribute a_j. Reductions of genes by removing them according to thresholds were not made previously, and so we make the task of correct classification harder by leaving even clearly non-functional genes for the algorithm to remove. Uninformative genes to the classifier could nevertheless be informative ones to the metaheuristic algorithm, since bad solutions are quite important to avoid guiding the search towards low quality regions of the search space.

4.2 Experimental setting

All experiments were carried out using a cluster of PCs with Linux O.S. (Suse 9.0 with kernel 2.4.19) and a Pentium IV 2.8 GHz processor, with 1 GB of RAM. The PMSO algorithm was independently executed 30 times on the Microarray datasets in order to have statistically meaningful conclusions. Each one of these PMSO executions performed 500 iterations.

In the parameters setup, an optimal configuration of the SVM classifier is crucial, since it influences the training effectiveness. Therefore, the main Kernel (RBF) parameters γ and C coefficient [13], were systematically optimized (as recommended in LIBSVM [10]) in a preprocess phase using grid-search with cross-validation. Basically, this consists in identifying the best combination of both parameters in a rank of bounded values (for example, $C = 2^{-5}, 2^{-5}, \dots, 2^{-15}, \gamma = 2^{-15}, 2^{-13}, \dots, 2^{3}$). Figure 2 plots the traces of the different grid-search parameters for each training dataset. The resulting parameters are:

- Leukemia: $C = 8$ and $\gamma = 0.0001220703125$ (accuracy $= 92.0\%$)
- Colon: $C = 128$ and $\gamma = 0.0001220703125$ (accuracy $= 82.5\%$)
- Lymphoma: $C = 8$ and $\gamma = 0.000030517557$ (accuracy $= 94.0\%$)
- Lung: $C = 2$ and $\gamma = 0.0001220703125$ (accuracy $= 93.0\%$).

These parameters were set using the SVM classifier independently of the PMSO, in order to obtain an accuracy as high as possible (as shown in parenthesis). The parameters of PMSO were set using the previously tuned SVM classifier. Finally, after the PSMO executions, a testing process was made on each resulted subset. In this process, the parameters of SVM were tuned using the grid-search method for the resulting subsets and test sets separately.

A comparison of prediction quality on all features versus the same quality obtained on selected by PMSO features sets is made in the next section.

Both sets of PMSO parameters, the ones defining the distributed model and those specific to GPSO, were set according to a preliminary study. As a result of this study, our PMSO has been run with the best configuration using 160 particles organized in 1, 2, 4 and 8 subswarms (with 160, 80, 40 and 20 particles each subswarm/processor), thus constituting four different configurations regarding the number of subswarms. In these configurations, we have considered $\sigma = 100$ and $\rho = 1$. The selection/replacement strategies choose respectively the *best* particle to be sent and replaces *if_better*. The migration topology is a *unidirectional ring*. Finally, concerning the GPSO parameters, we have used similar present, historic, and social weights $w_1 = w_2 = w_3 = 0.33$. The probability of performing mutation is 0.01.

4.3 Performance analysis

Table 2 shows the results obtained by PMSO, out of 30 independent runs, using the four different configurations (number of swarms in column 2). As a robustness indicator, column 3 denotes the number of executions in which the amount of genes (#Genes) in the resultant subset is lower than 5 (very good result from a biological point of view). Following the standard methodology when comparing classification rates, the average and standard deviation of the obtained accuracies (Accuracy 1) and the number of genes are reported in columns 4 and 5, respectively. Column 6 shows the reduction percentage[1] of each computed subset regarding the original datasets. In the last column, the accuracy percentage (Accuracy 2) of the tuned stand-alone SVM on each complete dataset (before reduction) is presented.

Several observations can be made from Table 2. First of all, the accuracy rate and the number of genes obtained by PMSO with 8 swarms (8-Swarm PMSO) is the best in all the cases for each dataset. This confirms that our parallel approach is clearly the way to go if a high accuracy and computational effort are needed, which is the case in actual labs. Significant statistical differences were found in these results regarding Colon, Lymphoma and Lung datasets. These differences in distributions (30 independent runs) were statistically assessed by using the following procedure: first a

[1] $Reduction = 100 - \frac{\#(genes\ in\ subset)}{\#(total\ genes\ in\ dataset)}$.

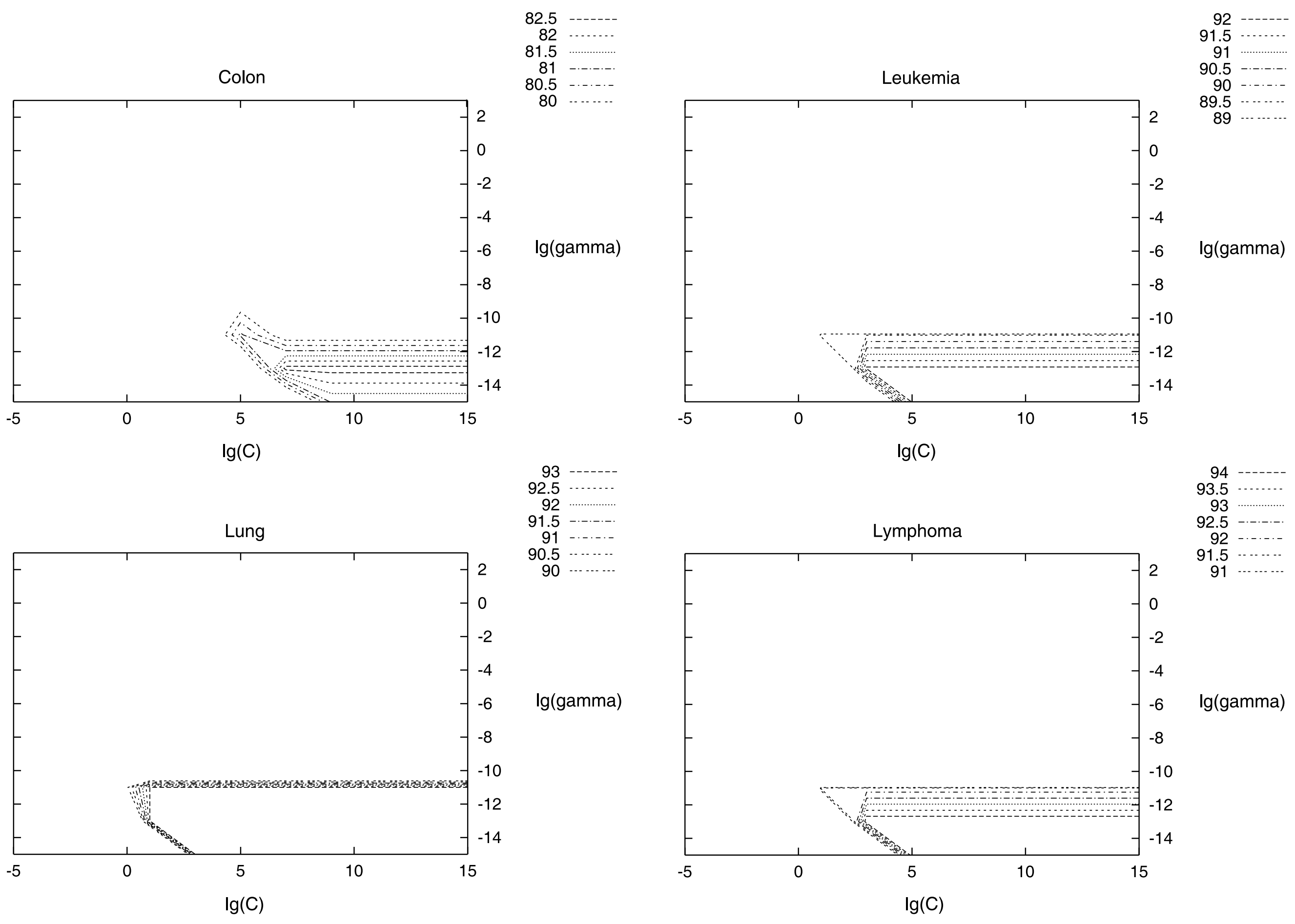


Fig. 2 Plots of the different grid-search parameters evaluated in SVM for each one of the datasets

Table 2 Results of PMSO using 1, 2, 4, and 8 subswarms configurations. The columns denotes the number of subswarms, the hit rate (expressed as subsets with less than 5 genes), the accuracy rate, the number of genes, and the reduction percentage

Datasets	Swarms	(#Genes < 5)	Accuracy 1 (%)	#Genes	Reduction (%)	Accuracy 2 (%)
Colon	1	30	79.98±5.61	2.40±0.71	88	82.5
	2	30	82.04±5.24	2.23±0.49	89	
	4	30	85.53±3.61	2.06±0.24	90	
	8	30	**85.55±4.06**	**2.00±0.00**	**90**	
Lymphoma	1	22	96.94±3.32	4.06±1.15	90	92.0
	2	24	96.75±2.87	3.60±1.08	92	
	4	28	97.12±2.72	3.23±0.76	92	
	8	30	**97.87±2.56**	**2.86±0.49**	**93**	
Leukemia	1	30	98.00±2.21	3.00±0.74	96	94.0
	2	24	98.00±2.60	4.00±0.92	95	
	4	28	98.00±2.54	3.00±0.88	96	
	8	28	**98.00±1.85**	**3.00±0.77**	**96**	
Lung	1	30	96.00±3.69	3.00±0.63	98	93.0
	2	30	97.39±3.13	2.63±0.70	98	
	4	30	97.00±3.17	2.66±0.64	98	
	8	30	**97.39±1.99**	**2.26 ± 0.44**	**99**	

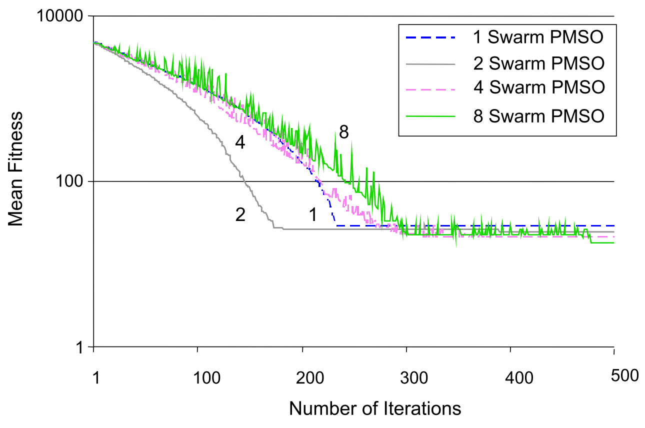


Fig. 3 Mean performance out of 30 executions of the PMSO in four different swarm distributions (1, 2, 4, and 8 subswarms). The mean fitness (in logarithmic scale) is plotted versus the number of iterations

Kolmogorov Smirnov test was performed in order to check whether the variables were normal or not and the Levene test to check the homoskedasticity of samples (equality of variances). If they were normal with equal variances, an ANOVA I test was performed, otherwise we performed a Kruskal-Wallis test. A level of significance of 95% ($\alpha = 0.05$) was always applied in order to check if there were statistically significant differences. After that, we did a multiple comparison test.

Secondly, in comparison with the accuracy percentage (Accuracy 2) of the SVM on each complete dataset, our results show better accuracy percentages in 14 out of 16 configurations. This is a true improvement since the reduction of features in all the cases with regards to the original datasets is around 90% in our model while SVM is using all genes. Specifically, our PMSO with 4 and 8 swarms always obtains better accuracies (with subsets lower than 5 genes) than the stand-alone SVM on the complete datasets.

When analyzing the internal behavior of these configurations of PMSO, we can clearly see that the ones distributed with more subswarms (4 and 8) show a smoother convergence than the others (i.e., diversity is better preserved). We suspect that the migration mechanism in PMSO introduces also a high diversity which helps the initial exploration in all configurations. This behavior is clearly observable in Fig. 3 where the mean performances through the evolution steps of 30 independent runs of each PMSO are plotted. In this figure, the lines represent the mean of the fitness obtained by the subswarms at each iteration. For this reason, the lines corresponding to PMSO with 4 and 8 subswarms show peaks that represent migrations injecting diversity in the receptor swarm, hence altering the averages of the global evolution fitness. This beneficial behavior is only slightly observed in PMSO with 2 subswarms, and non-existent in PMSO with 1 subswarm.

Table 3 Mean time of execution in microseconds (ms) performed by our 8-Swarm PMSO running on 1, 2, 4 and 8 processors

Dataset	$\overline{T_1}$	$\overline{T_2}$	$\overline{T_4}$	$\overline{T_8}$
Colon	2.69E+09	1.42E+09	7.60E+08	5.28E+08
Leukemia	5.14E+09	2.83E+09	1.52E+09	1.05E+09
Lymphoma	3.05E+09	1.69E+09	8.80E+08	4.74E+08
Lung	7.64E+09	4.19E+09	2.38E+09	1.39E+09

4.4 Speedup analysis

One of the most important parameters for measuring the efficiency of a parallel algorithm is the *Speedup* (*Sp*). The standard formula of the speedup is represented in (4), where m is the number or processors used, $\overline{T_1}$ is the mean time of execution of all the subswarms of the algorithm in 1 processor, and $\overline{T_m}$ is the mean time of execution of the swarms in parallel on m processors.

$$Sp = \frac{\overline{T_1}}{\overline{T_m}}. \tag{4}$$

Table 3 shows the mean time of execution in microseconds (ms) performed by our 8-Swarm PMSO running on 1, 2, 4 and 8 processors. The most time consuming execution corresponds to $\overline{T_1}$ in Lung dataset which takes about 2.12 hours (7.64E+09 ms). This time is reduced down to 23.16 minutes (1.39E+09 ms) when using 8 processors ($\overline{T_8}$) with the same dataset. Specifically, the 8-Swarm PMSO running on 8 processors obtains a reduction in the computational time of 69.8% when dealing with Lung. This is a clear improvement concerning the time consumption since Lung is the larger dataset we have tackled.

More precisely, in order to work with a measure proportional to the number of processors employed we calculated the speedup using the execution times obtained. For this, we followed the standard methodology described in [1].

A linear (ideal) speedup is obtained when $Sp = m$, and hence, in the execution of an algorithm with linear speedup, doubling the number of processors means doubling the speed. Figure 4 shows a graphical representation of the speedup performed by our 8-Swarm PMSO algorithm executed in 1, 2, 4 and 8 parallel processors. In this graphic, the linear speedup is represented by a dotted line. The remaining lines represent the speedup of the 8-Swarm PMSO on Leukemia, Colon, Lymphoma, and Lung datasets.

As we can observe, all the speedup values are close to the linear one, being all of them higher than 5 when running in 8 processors. This way, the mean efficiency[2] reported is $E = 70\%$ for all datasets which is a very good value for facing still larger problems in the future. The best

[2]Mean efficiency $E = \frac{Sp}{m} \times 100(\%)$.

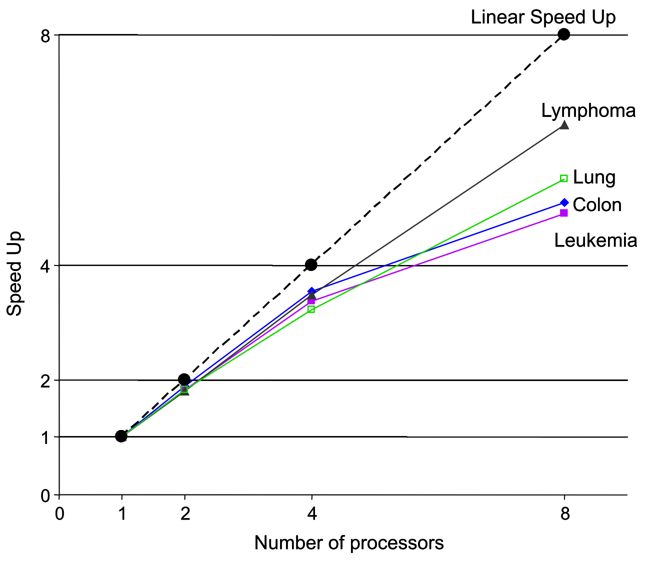


Fig. 4 Linear (ideal) speedup versus actual speedup of the 8-Swarm PMSO executed in 1, 2, 4 and 8 processors

efficiency is obtained when running in 4 processors being $E = 85\%$. In short, our PMSO provides an efficient outcome even in the presence of such a high dimensionality of the solutions (from 2,000 to 12,533 genes), stating a low overhead of communications and thus being a globally scalable technique.

4.5 PMSO versus parallel island genetic algorithm

In this section we compare the results obtained by our PMSO operating with 8 subswarms (8-Swarm PMSO), against a Parallel Island Genetic Algorithm (PGA) [2] also operating with 8 subpopulations. This way, we look to make a comparison not only versus the standard SVM but also versus other parallel techniques popular in the literature.

The PGA has been configured as follows: the whole population consists of 160 individuals (codifying gene subsets) organized in 8 subpopulations with 20 individuals each one. We have considered $\sigma = 100$ and $\rho = 1$ (explained in Sect. 4.2). The selection/replacement strategies choose, respectively, the *best* individual to be sent and replaces *if_better*. The migration topology is also a *unidirectional ring*. Finally, concerning the genetic algorithm parameters we have used a generational strategy of reproduction (e.g. number of offspring equal to number of parents), two-point crossover (probability of crossover 0.9), and simple bit-flip mutation (probability of mutation 0.01) as applied to GPSO in Sect. 3. The evaluation task in PGA has followed the same procedure as for PMSO. The evaluation task in PGA has followed the previously explained parameter setting of SVM for classification, 10-fold cross-validation, and fitness function (1). A final testing of the resultant subsets has been also accomplished.

Table 4 Comparison of PMSO versus PGA, both configured with 8 islands

Dataset	Alg.	(#Genes ≤ 5)	Accu. (%)	#Genes
Colon	PMSO	30	**85.55±4.06**	**2.00±0.00**
	PGA	30	81.45±4.22	2.10±0.30
Lymp.	PMSO	30	**97.87±2.56**	**2.86±0.49**
	PGA	30	94.44±3.72	3.60±1.28
Leuk.	PMSO	28	98.00±1.85	**3.00±0.77**
	PGA	30	98.55±1.55	**3.15±0.85**
Lung	PMSO	30	**97.39±1.99**	2.26±0.44
	PGA	30	93.43±4.86	**2.00±0.00**

The PGA was implemented in C++ using the MALLBA library. For the experiments, we have used the same pool of machines as explained in Sect. 4.2. The PGA was executed 30 times, each one of these executions performing 500 iterations.

Table 4 shows the results obtained by PGA together with those obtained by PMSO for each dataset. Column 3 denotes the number of executions in which the amount of genes in the resulted subset is smaller than 5. The columns 4 and 5 indicate the average and standard deviation of the accuracy rate and the number of genes in the final subsets, respectively. In bold we mark the most accurate results. As we can observe, PMSO obtains higher percentages of accuracy than PGA in Colon, Lymphoma and Lung datasets. These differences in accuracy have been statistically assessed by using the same procedure explained in Sect. 4.3. For Leukemia dataset, the differences in the results of PGA and PMSO are statistically negligible. Concerning the number of genes in the resulting subsets, PMSO obtains the smaller subsets in Colon, Lymphoma, and Leukemia. Only in Lung dataset, PMSO obtained slightly larger subsets than PGA which always obtained subsets with 2 genes. In conclusion, we can state that PMSO performs better than PGA in the scope of the problem and 3 of the 4 datasets studied here.

4.6 PMSO versus other approaches

In our aim of providing a thorough assessment of our results, in this section we additionally compare the results obtained by our PMSO operating with 8 subswarms (8-S PMSO) with six other approaches found in the literature. We must note that this study is very heterogeneous because other authors do not offer all the needed information and the algorithms are quite varied; hence, an exhaustive comparison can not be made. However, a simple comparison with other reported results is still useful.

The values are reported in Table 5, in terms of the average of the accuracy and the number of genes in the final subsets. In order to provide a more understandable comparison, these results are separated according to intervals of different numbers of genes (3 last columns). The accuracy reported by the 8-Swarm PMSO correspond to subsets achieved with 20, 10, and ≤ 5 genes (#G). We saved the subsets of genes generated with the specified lengths in each independent execution. As we can see, for subsets with more than 10 genes, the accuracy rate reported by our 8-Swarm PMSO is the best in 3 out of 4 datasets beating all the existing algorithms. In them, the only worse accuracy value was found for the Leukemia dataset [20] but our algorithm is finding solutions with a lower number of genes. With the smaller subsets ($\leq$ 5 genes), our approach reports the highest accuracy in Lymphoma, Leukemia and Lung. For Colon, Liu et al. [26] reported slightly higher accuracies than 8-Swarm PMSO, although with a larger number of genes. Finally, with subsets of between 5 and 10 genes, our approach is the best in Lung, but with lower accuracy than in Huerta et al. [20] and in Hernandez et al. [19] in Colon. Therefore, we can claim that PMSO shows a competitive performance in comparison with state of the art algorithms.

Table 5 Comparison with six other works. Columns indicate the average of the accuracy and the number of genes in the final subsets. The most accurate results are in boldface. Cells with unavailable values are marked with "–"

Dataset	Author	Accuracy		
		($\#G \leq 5$)	($5 < \#G \leq 10$)	($\#G > 10$)
Colon	[19]	–	91.20(8)	–
	[20]	–	**99.41(10)**	–
	[21]	–	–	94.12(37)
	[26]	**93.55**(4)	–	–
	8-S PMSO	87.61(**3**)	88.70(10)	**94.22(20)**
Lymp.	[19]	93.31(5)	–	–
	[25]	–	–	90.00(13)
	8-S PMSO	**97.87(3)**	**96.48(10)**	**98.70(20)**
Leuk.	[19]	91.50(3)	–	–
	[20]	–	–	**100**(25)
	[26]	87.55(4)	–	–
	8-S PMSO	**98.00(3)**	**96.31(10)**	98.15(**20**)
Lung	[6]	99.00(4)	–	–
	[26]	–	98.34(**6**)	–
	8-S PMSO	**99.38(2)**	**99.47**(10)	**100(20)**

4.7 Biological analysis of the results

In this section we provide a biological analysis of the computed gene subsets. Similar biological studies have been carried out in relevant papers in the past [14, 34]. Then we show the broad impact of using PMSO, able to compute biological ensembles of genes that have been also suggested in the domain (e.g. [16] and [8]).

In Fig. 5, a graphical distribution of the most frequently obtained genes in 30 independent executions of the 8-Swarm PMSO are reported. We have used the Leukemia dataset, since it is the one commonly studied by other related works in the literature. From this distribution, a brief selection of the 11 most overlapped genes (the ones in bold with frequency $\geq$3 in Fig. 5) out of all the computed subsets, are described in Table 6. We can remark that all of these genes were also reported in the list of the 30 most important genes (selected from 7,129 ones in Leukemia) suggested in Golub et al. [16]. Hence, we arrange the genes by means of the rank assigned in the Golub et al. [16] list (column 1 in the referenced table). This way, the gene U50136_rna1_at that we obtained with frequency 7 was ranked in [16] in eighth position. Moreover, the first ranked gene (X95735_at Zyxin) in [16], that we obtained with frequency 5, is the only gene that is capable of discerning between AML and ALL samples in just one split.

The genes reported in Fig. 5 (in boldface) were also selected as the most informative genes in recent specialized works. Concretely, in [14] a Monte Carlo method was used for feature selection and supervised classification on Leukemia and Lymphoma datasets. In [34] a Nearest Shrunken Centroid (NSC) was proposed to classify the Leukemia dataset. Both works considered the genes X95735_at and M23197_at as the most important ones, which matches our main results.

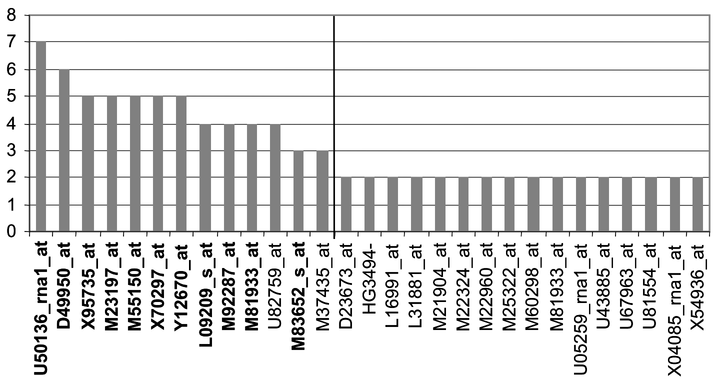


Fig. 5 Distribution of the most frequently obtained genes (in 30 independent runs) by our 8-S PMSO on the Leukemia dataset

Table 6 Top 11 genes ranked by Golub et al. which were also obtained with PMSO on the Leukemia dataset

Rank	Index	Accession	Gene description
1	4847	X95735_at	Zyxin
5	1834	M23197_at	CD33 antigen (differentiation antigen)
6	2020	M55150_at	FAH Fumarylacetoacetate
8	3320	U50136_rna1_at	Leukotriene C4 synthase (LTC4S) gene
15	4499	X70297_at	CHRNA7 Cholinergic receptor, nicotinic, alpha polypeptide 7
14	2267	M81933_at	CDC25A Cell division cycle 25A
16	5039	Y12670_at	LEPR Leptin receptor
18	6376	M83652_s_at	PFC Properdin P factor, complement
20	6041	L09209_s_at	APLP2 Amyloid beta (A4) precursor-like protein 2
24	2354	M92287_at	CCND3 Cyclin D3
28	461	D49950_at	Liver mRNA for interferon-gamma inducing factor(IGIF)

Concerning the Lymphoma dataset, three of the most frequently selected genes by 8-Swarm PMSO are: G1622X, G1618X and G2399X. These genes were also reported in the list of the 30 most important genes (selected from 4,026 ones in Lymphoma) suggested in Alizadeh et al. [8].

Moreover, genes selected from Leukemia and from Lymphoma in this work have been validated by means of the GO system (the Gene Ontology http://www.geneontology.org/), finding in all of them associations from Human proteins. Therefore, the selection of validated genes, also discovered in specialized publications in this area, leads us to claim the great ability of our PMSO for selecting informative genes in actual DNA Microarrays.

5 Conclusions

In this work, a Parallel Multi-Swarm Optimizer (PMSO) algorithm is proposed for the first time for gene selection of high dimensional Microarray datasets. PMSO has proven to be both efficient and accurate. It is able to improve sequential algorithms, in terms of computational effort (Efficiency of 85%). It was experimentally assessed with different population structures on four well-known cancer datasets, identifying specific genes that our work suggests as significant ones. Concretely, with regard to the Leukemia and Lymphoma datasets, we could confirm that the most frequently PMSO reported genes are also the most relevant genes suggested in the original publications (Golub et al. and Alizadeh et al., respectively) concerning these Microarrays. Comparisons with several recent state of the art methods also show competitive results close to 100% classification rate and very few genes per subset (4, 5 genes) obtained in 458 out of 480 (30×16) independent executions.

As for future work, we are interested in developing and testing several combinations of other metaheuristics with classification methods in order to discover still unseen and better subsets of genes using specific Microarray datasets. In this sense, the utilization of ensemble classifiers could contribute notably to the DNA Microarrays analysis.

Acknowledgements Authors acknowledge funds from the Spanish Government and European FEDER under contract TIN2008-06491-C04-01 (M* project, http://mstar.lcc.uma.es) and CICE, Junta de Andalucía under contract P07-TIC-03044 (DIRICOM http://diricom.lcc.uma.es). José García-Nieto is supported by grant BES-2009-018767 from the MICINN.

References

1. Alba E (2002) Parallel evolutionary algorithms can achieve super-linear performance. Inf Process Lett 82(1):7–13
2. Alba E (2005) Parallel metaheuristics: a new class of algorithms. Wiley series on parallel and distributed computing. Wiley, New York
3. Alba E, Dorronsoro B (2008) Cellular genetic algorithms. Springer, Berlin
4. Alba E, Luque G (2005) Parallel metaheuristics. A new class of algorithms. In: Measuring the performance of parallel metaheuristics. Wiley series on parallel and distributed computing. Wiley, New York, pp 43–62. Chap 2
5. Alba E, Troya JM (2001) Analyzing synchronous and asynchronous parallel distributed genetic algorithms. Future Gener Comput Syst 17(4):451–465
6. Alba E, García-Nieto J, Jourdan L, Talbi E-G (2007) Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms. In: IEEE congress on evolutionary computation CEC-07, Singapore, Sep 2007, pp 284–290
7. Alba E, Luque G, García-Nieto J, Ordonez G, Leguizamón G (2007) MALLBA: a software library to design efficient optimisation algorithms. Int J Innov Comput Appl 1(1):74–85
8. Alizadeh A.A (2000) Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature 11:403–503
9. Alon U, Barkai N, Notterman D, Gish K, Ybarra S, Mack D, Levine AJ (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc Natl Acad Sci USA 96:6745–6750
10. Chang C-C, Lin C-J (2002) LIBSVM: a library for support vector machines
11. Cho S, Won H (2007) Cancer classification using ensemble of neural networks with multiple significant gene subsets. Appl Intell 26:243–250

12. Clerc M (2005) Binary particle swarm optimisers: Toolbox, derivations, and mathematical insights
13. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
14. Draminski M, Rada-Iglesias A, Enroth S, Wadelius C, Koronacki J, Komorowski J (2008) Monte Carlo feature selection for supervised classification. Bioinformatics 24(1):110–117
15. Fix E, Hodges JL (1951) Nonparametric discrimination: consistency properties. Technical report, 4, US Air Force School of Aviation Medicine, R Field, TX
16. Golub R, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA, Bloomfield CD, Lander ES (1999) Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286:531–537
17. Gordon GJ, Jensen RV, Hsiao L-L, Gullans SR, Blumenstock JE, Ramaswamy S, Richards WG (2002) Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer Res 62:4963–4967
18. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Mach Learn 46(1–3):389–422
19. Hernandez J, Duval B, Hao J-K (2007) A genetic embedded approach for gene selection and classification of microarray data. In: Marchiori E et al (eds) LNCS of EvoBio, pp 90–101
20. Huerta EB, Duval B, Hao J-K (2006) A Hybrid GA SVM approach for gene selection and classification of microarray data. In: Rothlauf F et al (eds) LNCS of EvoWorkshops, vol 3907. Springer, Berlin, pp 34–44
21. Juliusdottir T, Keedwell E, Corne D, Narayanan A (2005) Two-phase EA/K-NN for feature selection and classification in cancer microarray datasets. In: Comp int in bioinformatics and computational biology, pp 1–8
22. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proc of the IEEE international conference on neural networks, vol 4, pp 1942–1948
23. Kennedy J, Eberhart R (1997) A discrete binary version of the particle swarm algorithm. In: Proceedings of the IEEE international conference on systems, man and cybernetics, vol 5, pp 4104–4109
24. Kohavi J, John GH (1998) The wrapper approach. In: Feature selection for knowledge discovery and data mining, pp 33–50
25. Liu J, Iba H (2002) Selecting informative genes using a multiobjective evolutionary algorithm. In: Proceedings of the IEEE congress on evolutionary computation, CEC'02, May 2002, vol 1, pp 297–302
26. Liu B, Cui Q, Jiang T, Ma S (2004) A combinational feature selection and ensemble neural network method for classification of gene expression data. BMC Bioinform 5:136–148
27. Moraglio A, Di Chio C, Poli R (2007) Geometric particle swarm optimization. In: 10th European conference on genetic programming (EuroGP 2007). Lecture notes in computer science, vol 4445. Springer, Berlin
28. Narendra M, Fukunaga K (1977) A branch and bound algorithm for feature subset selection. IEEE Trans Comput 26:917–922
29. Pease AC, Solas D, Sullivan E, Cronin M, Holmes CP, Fodor S (1994) Light-generated oligonucleotide arrays for rapid dna sequence analysis. In: Proc natl acad sci, vol 96., pp 5022–5026
30. Romdhane L, Shili H, Ayeb B (2010) Mining microarray gene expression data with unsupervised possibilistic clustering and proximity graphs. Appl Intell 33:220–231
31. Salto C, Alba E (In press) Designing heterogeneous distributed GAs by efficiently self-adapting the migration period. Appl Intell (Online first). doi:10.1007/s10489-011-0297-9
32. Verma B, Hassan SZ (2010) Hybrid ensemble approach for classification. Appl Intell 34(2):258–278
33. Vinh L, Lee S, Park Y, dÁuriol B (In press) A novel feature selection method based on normalized mutual information. Appl Intell (Online first). doi:10.1007/s10489-011-0315-y
34. Wang S, Zhu J (2007) Improved centroids estimation for the nearest shrunken centroid classifier. Bioinformatics 32(2):972–979
35. Zhu H, Jiao L, Pan J (2006) Multi-population genetic algorithm for feature selection. In: ICNC (2), pp 480–487