Memetic NSGA – A Multi-Objective Genetic Algorithm for Classification of Microarray Data

Praveen Kumar K.
Dept of Computer Engg, NITK - Surathkal
praveenk.blr@gmail.com

Sharath S.
Dept of Computer Engg, NITK – Surathkal
gs_sharath@gmail.com

Rio D'Souza G.
Dept of Computer Engg, St Joseph Engg College, Mangalore
rio@ieee.org

K. Chandra Sekaran
Dept of Computer Engg, NITK – Surathkal
kchandrain@yahoo.co.in

Abstract

In Gene Expression studies, the identification of gene subsets responsible for classifying available samples to two or more classes is an important task. One major difficulty in identifying these gene subsets is the availability of only a few samples compared to the number of genes in the samples. Here we treat this problem as a Multi-objective optimization problem of minimizing the gene subset size and minimizing the number of misclassified samples.

We present a new elitist Non-dominated Sorting-based Genetic Algorithm (NSGA) called Memetic-NSGA which uses the concept of Memes. Memes are a group of genes which have a particular functionality at the phenotype level. We have chosen a 50 gene Leukemia dataset to evaluate our algorithm. A comparative study between Memetic-NSGA and another Non-dominated Sorting Genetic Algorithm, called NSGA-II, is presented. Memetic-NSGA is found to perform better in terms of execution time and gene-subset length identified.

1. Introduction

The working principle of Genetic Algorithms (GA) [1] is very different from that of other classical optimization techniques. In brief, the GA-based evolution starts from a set of individuals or a set of solutions that represent the functions to be optimized and proceeds from generation to generation using basic genetic operators like crossover and mutation.

Most real world search and optimization problems deal with simultaneous optimization of multiple objectives. There exist a number of algorithms for solving a multi-objective optimization problem (MOOP) [2]. However these algorithms ignore some fundamental differences between the working principles of single and multi-objective optimization algorithms [3]. In fact, many multi-objective problems involve working with many conflicting trade-off objectives, which cannot be solved using single-objective optimization [4]. These are some of the factors that motivated us to work more deeply on multi-objective optimization.

2. Multi-objective genetic algorithms

First we take a look at a few concepts related to Multi-Objective Genetic Algorithms (MOGA) [5]:

1) Pareto-Optimality: A Pareto-front is the locus formed by a set of solutions which are equally good as compared to any other solutions of that set, i.e., any solution on that front cannot be selected when compared with another, without being biased towards the other.

2) Domination: The concept of Domination is used to identify non-dominated sets (i.e., Pareto-optimal sets) of solutions for the given set of solutions. For this purpose, two solutions are compared on the basis of whether one dominates the other of not.

3) Elitism: Although the crossover and mutation are the core search operators of GA, there is a possibility of destruction of the good solutions developed so far, due to these operators. Using elitism, it is possible to pass the best solutions of a particular generation to the next generation, unaffected. This helps the evolutionary algorithm (EA) converge faster [6]. This ensures the preservation of good individuals from the old and the new populations.

0-7695-3059-1/07 $25.00 © 2007 IEEE
DOI 10.1109/ADCOM.2007.114

3. Related work

Among various multi-objective optimization algorithms, we adopt an elitist Non-dominated Sorting Genetic Algorithm (NSGA) [7] which is an effective approach that suits our purpose. The working principle of Non-dominated Sorting is described here.

3.1. Non-dominated sorting genetic algorithm

In Non-dominated Sorting, the offspring population Qt (of size N) is first created by using the parent population Pt (of size N) and the usual genetic operators such as single-point crossover and bit-wise mutation operators. Thereafter, the two populations are combined together to form Rt of size 2N. Then, depending on the multiple objective functions the fitness of each individual in the 2N population is determined, using the concept of domination and ranks are assigned with respect to their relative fitness.

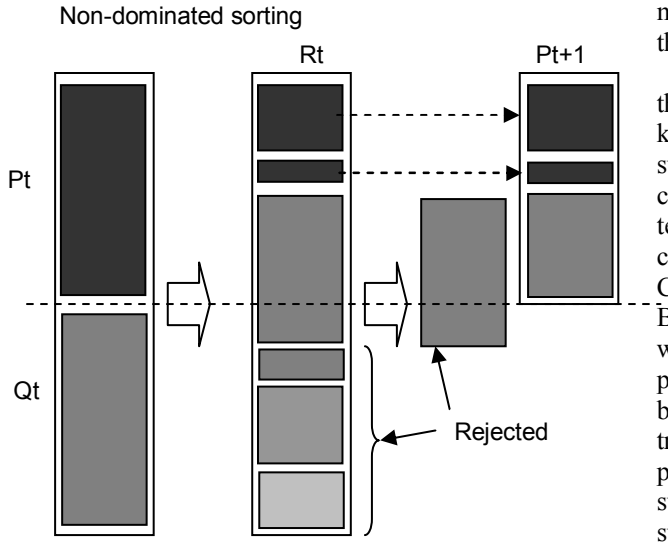

Figure 1. The non-dominated sorting procedure

With all individuals ranked, a non-dominated sorting procedure is carried out over the 2N population to divide them into different non-dominated fronts. Once the non-dominated sorting is over, the new parent population Pt+1 is created by choosing solutions of different non-dominated fronts, one at a time. The filling starts with strings from the best non-dominated front and continues with strings of second non-dominated front, and so on, as shown in Fig. 1.

Since the overall population size of Rt is 2N, not all fronts can be accommodated in the new parent population. The fronts which could not be accommodated at all are simply discarded. However, while the last allowed front is being considered, there may be more stings in it than the remaining population slots in the new population. This scenario is illustrated in Fig. 1.

To deal with this condition we simply take the first x individuals which make up the N population number and discard the rest of the solutions. This procedure is continued for a maximum of T iterations.

3.2. Classifier for identification of gene-sets

In our work we have used Leave-Out-One-Cross-Validation (LOOCV) to classify samples into class A or class B. We describe a class determination procedure [8] for classifying samples into two classes only, although modifications can be made to generalize the procedure for any number of classes.

It is a common practice to divide the available data sets into two groups one used for training purposes for generating a classifier and the other used for testing the developed classifier. The most commonly employed method to estimate the accuracy in such situations is the cross-validation approach.

In cross-validation, the training datasets (say T of them) are partitioned into k subsets, C1, C2,..., Ck (k is known as the number of cross-validation trials). Each subset is kept roughly of the same size. Then a classifier is constructed using Ti = T - Ci samples to test the accuracy on the samples in Ci. Once the classifier is constructed using Ti samples, each of the Ci samples is tested using the classifier for class A or B. Since these Ti samples are used as training samples, we can compare the classification given by the above procedure with the actual class in which the samples belong. If there is a mismatch, we increment the training sample mismatch counter τ-train by one. This procedure is repeated for all Ci samples in the i-th subset. Thereafter, this procedure is repeated for all k subsets and overall training sample mismatch counter τ-train is noted.

LOOCV is one of the most commonly used cross-validation approaches in which only one sample in the training set is withheld and classifier is constructed using the rest of the samples to predict the class of the withheld sample. Thus, in the LOOCV there are k = T subsets. The number of mismatches τ-test obtained by comparing the predicted class with the actual class of each sample is noted.

3.3. The resulting optimization problem

Here we formulate the gene subset identification task as a multi-objective optimization problem with three different conflicting objectives. Here one of the objectives of the identification task is to minimize the

number of genes used in classification while maintaining acceptable classification accuracy. Hence, the gene subset identification task can be formulated as a multi-objective optimization problem with three different conflicting objectives.

The multi-objective optimization problem formulation is as follows:

1) The first objective function - The gene subset identification task is to minimize the number genes in a subset or to minimize the gene subset size for a classifier.

2) The second objective function - Minimize the number of class prediction mismatches in the training samples. These are calculated using the LOOCV, i.e., τ-train.

3) The third objective function - To minimize the number of class prediction mismatches in the test samples, these are calculated using the classifier constructed based on all samples in the training set, i.e., τ-test.

We have used an evolutionary based technique to identify most discriminative gene subsets, a weighted voting approach is employed to predict the class of a sample based on such informative gene subsets and a set of samples with known class labels, and the LOOCV procedure is used to determine the number of mismatches in the training samples. Thereafter, the classifier is constructed using such informative gene subsets and all the training samples. Then, the performance of the classifier is estimated using the remaining samples in a test-set. The multi-objective GA (NSGA-II) described above is considered for handling the above three conflicting objectives.

4. Memetic-NSGA: a new approach

First we define memes and then distinguish Memetic Algorithms from Genetic Algorithms. Thereafter we describe the Memetic-NSGA and then the operators used to manipulate memes. Finally we present the Memetic-NSGA.

4.1. The concept of memes

Richard Dawkins coined the term meme [9], and defined it as "a unit of cultural transmission, or a unit of imitation". In other words memes are a set of similar genes which provide a specific functionality at the phenotype level.

Memes have, as their fundamental property, evolution via natural selection in a way very similar to Charles Darwin's ideas concerning biological evolution, on the premise that replication, mutation, survival and competition influence them. Thus memes provide a framework for a theory of cultural evolution [10], analogous to the theory of biological evolution based on genes.

Dawkins introduced the term after writing that evolution depended not on the particular chemical basis of genetics, but only on the existence of a self-replicating unit of transmission in the case of biological evolution, the gene. For Dawkins, the meme exemplifies another self-replicating unit, and most importantly, one which he thought would prove useful in explaining human behavior and cultural evolution. This analogy suggests that the definition of a meme should refer to the physical structure, or abstract code representing that structure, representing a real idea as observed in-situ. Genes do not depend upon their transfer for their current existence; they only need to have a definite and unique physical structure. One might appropriately extend the analogy to the concept of a meme.

4.2. Memetic algorithms

A Memetic Algorithm is a population-based approach for heuristic search in optimization problems. These are shown to be orders of magnitude faster than traditional genetic algorithms for some problem domains. Genetic Algorithms are not well suited for fine-tuning structures which are close to optimal solutions.

Incorporation of local improvement operators into the recombination step of a genetic algorithm is essential if a competitive genetic algorithm is desired. Memetic Algorithms (MA) [11] are evolutionary algorithms that apply a separate local search process to refine individuals (i.e., improve their fitness). Combining global and local search is a strategy used by many successful global optimization approaches, and MAs have in fact been recognized as a powerful algorithmic paradigm for evolutionary computing.

4.3. Memetic-NSGA

Taking inspiration from the concepts of memes and evolution discussed above, we present the Memetic Non-dominated Sorting Genetic Algorithm or Memetic-NSGA. The underlying principle behind the use of memes in our algorithm is based on the following line of thought [12].

The complete Memetic-NSGA is given below:

BEGIN

While generation count is not reached

Begin Loop

- Combine parent Pt and offspring population Qt to obtain population Rt of size 2N.

- Perform Non-dominated Sort on Rt and assign ranks to each pareto front with fitness Fi.
- Starting from Pareto front with fitness F1, add each pareto front Fi to the new paren population Pt+1 until a complete front F cannot be included.
- From the current pareto front Fi, add individual members to new parent populatio Pt+1 until it reaches the size N.
- Apply the Find-Meme operator to th members of new parent population Pt+1.
- Apply the Meme-fitness factor, Mff to each meme member of new parent population Pt+1 and increment their fitness value.
- Apply selection, crossover and mutation t new parent population Pt+1 and obtain th new offspring population Qt+1.
- Increment generation count.

End Loop

END.

4.4. Operators in memetic-NSGA

1) Find-Meme operator: This operator finds the set of individuals in a given population which share the longest set of similar genes. Each of these set of individuals represents a meme. Once the memes are identified, each of the meme group is represented by one individual among them, called the leader. The individual of the highest rank in the meme group is selected as the leader.

By selecting groups which contain similar set of genes we are indirectly dividing the population into sets of individuals which share some common characteristic property or functionality or phenotype nature. This sort of division results in the localized nature of the search technique.

2) Meme-fitness factor: Memes that consists of higher rank individuals are more fit than the ones that consist of lower rank individuals. Therefore in the selection process we explicitly increase the fitness value of the individuals by a meme-fitness factor depending on the fitness value of all the individuals in the meme. The meme-fitness factor, Mff is defined as:

$$M_{ff} = (\sum_{i} f_i)^{-1}$$

Where fi is the fitness value of the i-th individual in the meme. After this calculation, the fitness value of each individual in the meme is incremented by the meme-fitness factor.

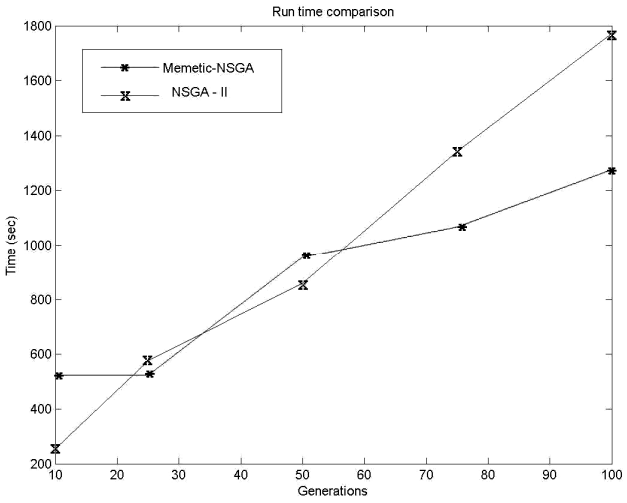

Figure 2. Execution time plotted over generations

Table 1. Execution time versus generations

Generations		25	50	75	100
Execution Time (sec)	NSGA II	575	861	1342	1773
	Memetic-NSGA	524	956	1067	1276

5. Results

We carried out a comparative study between NSGA-II and Memetic-NSGA. Both the algorithms were run in similar conditions on the 50-gene Leukemia dataset of Golub et. al. [13]. The results obtained were analyzed over various parameters like run time, train errors, test errors and classifier length. A population of 50 individuals was considered for all the runs. Tests were carried out on a Pentium-IV, 2.0 GHz machine running Fedora Core 3.0 with 512 MB RAM.

5.1. Execution Time

Fig. 2 shows the variation of execution time with generations and Table 1 gives the corresponding values. It is evident from the graph that Memetic-NSGA is faster than NSGA-II as the number of generations increase. Since genetic algorithms are run for hundreds of generations, time is an important factor to be considered. There is a 30% improvement in the

execution time of Memetic-NSGA over NSGA-II for the 100 generations shown.

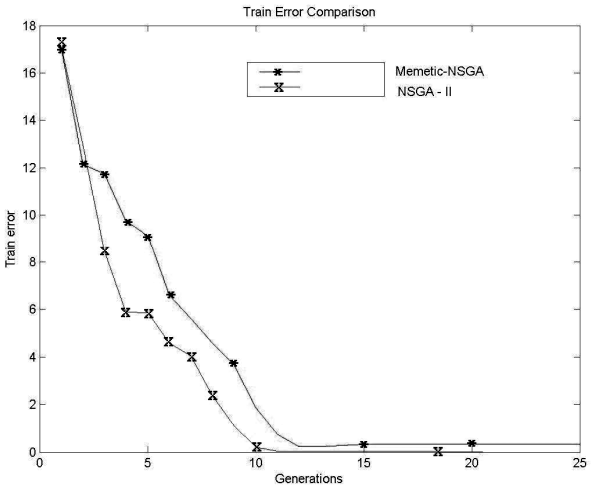

Figure 3. Train error plotted over generations

Table 2. Train error versus generations

Generations		5	10	15	20	25
Train Error	NSGA II	5.86	0.22	0.04	0.02	0.00
	Memetic -NSGA	9.12	1.86	0.32	0.32	0.32

5.2. Train error

Fig. 3 shows the variation of Train Error with generations and Table 2 gives the corresponding values. This is the first objective of the multi-objective optimization problem at hand. Here NSGA II converges faster to the ideal value of zero train errors, however both the algorithms achieve the ideal value.

5.3. Test error

Fig. 4 shows the variation of Test Error with generations and Table 3 gives the corresponding values. This is the second objective of the multi-objective optimization problem. Here too NSGA II converges faster to zero test errors but Memetic-NSGA achieves the same a few generations later.

5.4. Gene subset length

Fig. 5 shows the variation of Gene Subset (classifier) length with generations and Table 4 gives the corresponding values. A major improvement is seen in Memetic-NSGA with a result of gene subset lengths of 2 and 3 where as NSGA II achieves lengths of only 4 and 5. Therefore a 50% improvement is achieved in this objective.

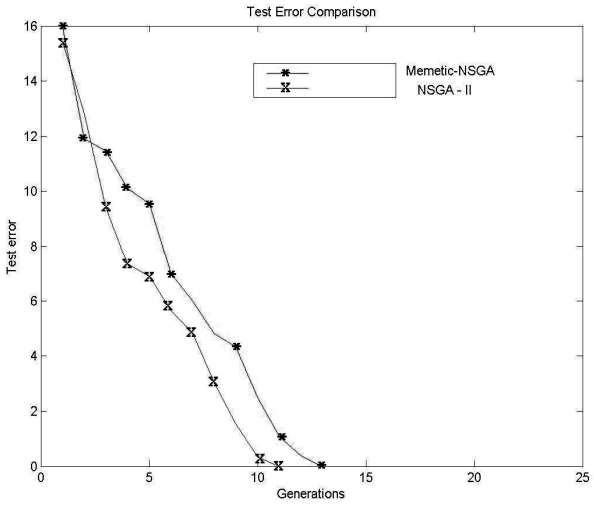

Figure 4. Test error plotted over generations

Table 3. Test error versus generations

Generations		5	10	15	20	25
Test Error	NSGA II	6.88	0.34	0.00	0.00	0.00
	Memetic -NSGA	9.56	2.48	0.00	0.00	0.00

6. Conclusion

In this work we have proposed a new Multi-objective Evolutionary Algorithm called Memetic-NSGA which is based on the concept of Memes. We started off our work by implementing the existing NSGA-II algorithm and analyzing the results generated by it. After that we decided to introduce localized Memetic algorithms to obtain better results.

We have retained the concept of Non-dominated sorting from NSGA II, and introduced new operators

like Find-meme and Meme-fitness factor. We have also retained the LOOCV validation method to classify samples.

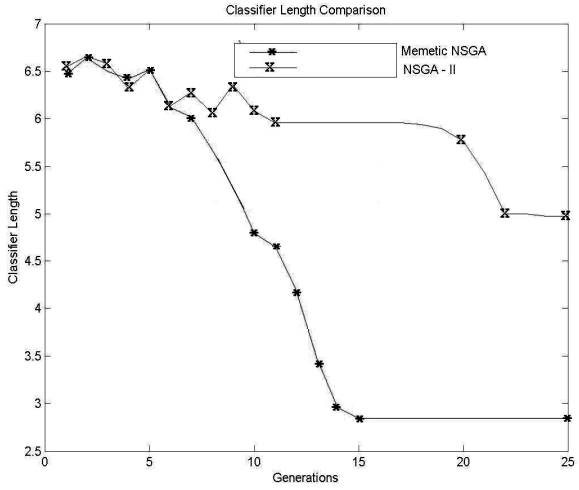

Figure 5. Classifier length plotted over generations

Table 4. Classifier length versus generations

Generations		5	10	15	20	25
Class-ifier Length	NSGA II	6.52	6.08	5.96	5.76	4.98
	Memetic -NSGA	6.52	4.80	2.84	2.84	2.84

A comparative study between NSGA II and Memetic-NSGA was carried out and simulation results were gathered and analyzed. The results were very encouraging with Memetic-NSGA giving smaller gene classifiers and also the run time of the algorithm was found to be better than NSGA-II for larger number of generations.

Future work on these lines would be to test the effectiveness of Memetic-NSGA to deal with various other multi-objective optimization problems.

7. References

[1] D.E. Goldberg, *Genetic Algorithms for Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[2] J. Andersson, "A Survey of Multiobjective Optimization in Engineering Design," *Technical report LiTH-IKP-R-1097*, Dept of Mechanical Engg, Linkping University, Sweden, 2000, pp. 34.

[3] C.A. Brizuela and E. Gutiérrez, "Multi-objective Go With the Winners Algorithm: A Preliminary Study," *Evolutionary Multi-Criterion Optimization, Third Int'l. Conf., EMO 2005*, Guanajuato, Mexico, March 9-11, 2005.

[4] D.A. Van Veldhuizen and G.B. Lamont, "Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-art," *Evol. Comp.*, Vol. 8, No. 2, 2000, pp. 125-147.

[5] K. Deb, *Multi-objective Optimization using Evolutionary Algorithms,* Wiley, Chichester, UK, 2001.

[6] L. Costa and P. Oliveira, "An Evolution Strategy for Multiobjective Optimization," In D. B. Fogel et al., editors, *CEC'02*, volume 1, IEEE, 2002, pp. 97-102.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II," *IEEE Trans. Evol. Comp.*, vol. 6, no. 2, Apr. 2002, pp. 182-197.

[8] K. Deb and A.R. Reddy, "Classification of Two-class Cancer Data Reliably Using Evolutionary Algorithms," *Publ. of Kanpur Genetic Algorithms Lab.*, *India*, KanGAL Report No. 2003001, 2003.

[9] R. Dawkins, *The Selfish Gene*, Oxford University Press, 1976.

[10] C.A.C. Coello and R.L. Becerra, "Evolutionary Multiobjective Optimization Using a Cultural Algorithm," *Swarm Intelligence Symposium*, 2003, SIS 03, Proc. of the, IEEE, 24-26 April 2003, pp. 6 – 13.

[11] S. Areibi, M. Moussa, and H. Abdullah, "A Comparison of Genetic/Memetic Algorithms and Other Heuristic Search Techniques," *2001 Int'l. Conf. on Artificial Intelligence IC-AI 2001*, Las Vegas, 2001.

[12] H. A. Abbass, "A Memetic Pareto Evolutionary Approach to Artificial Neural Networks," In M. Stumptner, D. Corbett, and M. Brooks, editors, *Proc. of the 14th Australian Joint Conf. on Artificial Intelligence* (AI'01), Springer-Verlag, Berlin, 2001, pp. 1-12.

[13] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander, "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," *Science*, vol. 286, 1999, pp. 531-537.