Coevolutionary Multi-Objective Optimization using Clustering Techniques

Margarita Reyes Sierra and Carlos A. Coello Coello

CINVESTAV-IPN (Evolutionary Computation Group)
Electrical Eng. Department, Computer Science Dept.
Av. IPN No. 2508, Col. San Pedro Zacatenco, México D.F. 07300, MÉXICO

Abstract. We propose a new version of a multiobjective coevolutionary algorithm. The main idea of the proposed approach is to concentrate the search effort on promising regions that arise during the evolutionary process as a product of a clustering mechanism applied on the set of decision variables corresponding to the known Pareto front. The proposed approach is validated using several test functions taken from the specialized literature and it is compared with respect to its previous version and another approach that is representative of the state-of-the-art in evolutionary multiobjective optimization.

1 Introduction

Despite the considerable volume of research on evolutionary multiobjective optimization [1], little emphasis has been placed on certain algorithmic design aspects such as efficiency [2, 3]. Additionally, the use of coevolutionary mechanisms has been scarce in the evolutionary multiobjective optimization literature. As in our original proposal [4], the main motivation of the work reported here is precisely to take advantage of some coevolutionary concepts to design a multi-objective evolutionary algorithm (MOEA) that can be more efficient (in terms of fitness function evaluations). The main idea of the proposed algorithm is to obtain information along the evolutionary process as to focus the search in the "promising" sub-regions, and then to use a subpopulation for each of these subregions. At each generation, these different subpopulations "cooperate" and "compete" among themselves and from these different processes we obtain a single Pareto front. The size of each subpopulation is adjusted based on their contribution to the current Pareto front. The proposed approach uses the adaptive grid proposed in [3] to store the nondominated vectors obtained along the evolutionary process, enforcing a more uniform distribution of such vectors along the Pareto front. This new version of our algorithm performs a clustering analysis on the set of decision variables of the current Pareto front to find the promising regions of the search space. In this way, the number of populations needed does not exceed the total number of members on the true Pareto front.

2 Coevolution

Coevolution refers to a reciprocal evolutionary change between species that interact with each other. The relationships between the populations of two different species can

be described considering all their possible types of interactions. Such interaction can be positive or negative depending on the consequences that such interaction produces on the population. Evolutionary computation researchers have developed several coevolutionary approaches in which normally two or more species relate to each other using any of the possible relationships, mainly competitive (e.g., [5]) or cooperative (e.g., [6]). Also, in most cases, such species evolve independently through a genetic algorithm. The key issue in these coevolutionary algorithms is that the fitness of an individual in a population depends on the individuals of a different population.

3 Previous Work

Parmee and Watson [7] proposed a collaborative scheme in which they use one population to optimize each of the objective functions of a problem. The method is really created to converge to a single (ideal) trade-off solution. However, through the use of penalties the algorithm can maintain diversity in the population. These penalties relate to variability in the decision variables' values. Keerativuttitumrong et.al. [8], Tan et.al. [9] and Iorio and Li [10], proposed cooperative schemes in which one population is defined for each decision variable of the problem. In order to evaluate an individual in any population, individuals from the other populations must be selected in order to complete a solution. In [8], the evolution of each of these populations is controlled through Fonseca and Fleming's MOGA [11]. The method in [9] uses an external archive to store and update the nondominated solutions found so far and also to guide the search to the less exploded subregions of the search space. Finally, in [10] the evolution of each of the populations is controlled through the scheme of the NSGA-II [2]. After each generation, the method proposed in [10] uses a nondominated sorting over all the subpopulations of parents and offspring to determine the new parents subpopulations.

4 Description of Our Approach

As in [4], the main idea of our approach is to focus the search efforts only towards the promising regions of the search space. Such "promising" regions are determined using clustering analysis of the current Pareto front. The evolutionary process of our approach is divided in two main stages. The first stage takes place during the first quarter of the total of generations. After that, in the second stage (the rest of the generations) we perform what we call a *checkpoint* in specific moments that will be mentioned later.

First Stage. During the first stage, the algorithm is allowed to explore all of the search space, by using a population of individuals which are selected using Fonseca and Fleming's Pareto ranking scheme [11]. Additionally, the approach uses the adaptive grid proposed in [3]. At the end of this first stage, the algorithm analyses the current Pareto front (stored in the adaptive grid) in order to determine the promising regions of the search space. In this new version, we perform a clustering analysis on the set of values of the decision variables corresponding to the current Pareto front. The aim is to determine the promising regions of the search space (line 6, Figure 1). This analysis is performed independently for each decision variable. Once we know the clusters of the values corresponding to each one of the decision variables, we proceed to form a set

```
1. gen = 0
2. populations = 1
3. while (gen < Gmax) {
        if (gen ≥ Gmax/4)
4.          if (gen = Gmax/4, Gmax/2, 3Gmax/4 or
            ∃ x ∈ pop_zero : x ∈ current Pareto front) {
5.              check_active_populations()
6.              clustering_algorithm()
7.              construct_new_subpopulations()}
8.      for (i = 1; i ≤ populations; i++)
9.          if (population i contributes to the current Pareto front)
10.             evolve_and_compete(i)
11.     elitism()
12.     reassign_resources()
13.     gen++ }
```

Fig. 1. Pseudocode of our algorithm.

of new populations. This process is illustrated in Figure 2. A cluster is a set of values, so for each cluster of each variable, we obtain the limits that bound that cluster. Once that we know the limits of each cluster, we have a set of intervals for each variable. Then, a set of sub-regions is created in the following way. For each point in the current Pareto front, we proceed to locate the interval on each variable to which it belongs. This process gives us a region in the search space. For each point in the current Pareto front we first check if it belongs to any region already located. If the point belongs to an existing region, we continue with the next point. Otherwise, we proceed to create the corresponding region, and so on. After that, we assign a new population to each region created, i.e., those that have individuals in the current Pareto front (line 7, Figure 1). In this way, in the worst case we will have as many populations as points in the current Pareto front. Finally, we use one extra population (called *population zero*) that continues searching for good solutions on the whole search space. This population is initialized with an 80% of points of the current Pareto front and a 20% of random points (with the aim of generating intermediate points on the current Pareto front while adding diversity).

Second Stage. When reaching the end of the first stage, the algorithm consists of a certain number of populations looking each at different regions of the search space. At each generation, the evolution of all the populations takes place independently and, later on, the nondominated elements from each population are sent to the adaptive grid where they "cooperate" and "compete" in order to conform a single Pareto front. After this, we count the number of individuals that each of the populations contributed to the current Pareto front. Our algorithm is *elitist* (line 11, Figure 1), because after the first generation of the second stage, all the populations that do not provide any individual to the current Pareto front are automatically eliminated and the sizes of the other populations are properly adjusted (line 12, Figure 1). Each population is assigned or removed individuals such that its final size is proportional to its contribution to the current Pareto front. These individuals to be added or removed are randomly generated/chosen. This process is illustrated in Figure 2. Thus, populations compete with each other to get as many extra individuals as possible. Note that it is, however, possible that the sizes of the

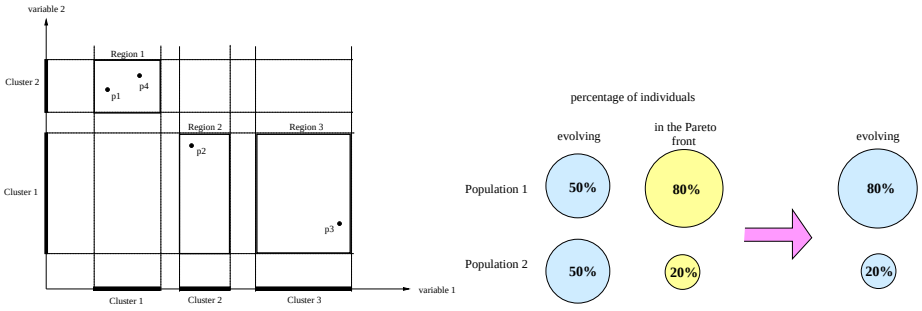

Fig. 2. (left) Mechanism used to locate the promising regions of the search space. A population will be assigned to each located promising region. (right) Resources reassignment: Each population is assigned or removed individuals such that its final size is proportional to its contribution to the current Pareto front.

populations "converge" to a constant value once their contribution to the current Pareto front no longer changes.

Checkpoint. During the second stage, we perform a *checkpoint* in specific moments of the evolutionary process (line 4, Figure 1). The checkpoint takes place as before (see Figure 1) [4], but also when the population zero includes a new individual in the adaptive grid, that is, in the current Pareto front. When the checkpoint happens, we perform a check on the current populations in order to determine how many (and which) of them can continue (i.e., those populations which continue contributing individuals to the current Pareto front, which are the "good" populations) (line 5, Figure 1). As at the end of the first stage, we perform again the clustering analysis on the set of values of the decision variables corresponding to the current Pareto front, and proceed to form a set of new populations. The non-dominated individuals from the "good" populations are kept. All the good individuals are distributed across the newly generated populations. The elitist process continues and the size of each population will be adjusted based on the same criteria as before. Note however, that we define a minimum population size and this size is enforced for all populations after each checkpoint.

Clustering Analysis. We implemented a clustering algorithm based on the nature of the *k-means* algorithm [12]. Since the *k-means* algorithm: (1) depends on the initial centroids and (2) requires the number of clusters needed, we made two modifications to overcome these drawbacks. Regarding the first disadvantage, when the distance from a given point x to its centroid is greater than the minimum distance between two centroids, the point x becomes a new centroid. To maintain the number of clusters constant, once we have selected a point to be a new centroid, we choose one of the closest centroids to be eliminated. With respect to the second disadvantage, we use the following mechanism [12]: Let x_i be a point, d_i the distance between x_i and its centroid, K the current total number of clusters and $\bar{d}_i$ the average distance between x_i and the K centroids. We create a new cluster with centroid x_i when: $|d_i - \bar{d}_i| \leq \bar{d}_i T$ where T is such that $0 < T < 1$. As big is the value of T, as big is the number of clusters created. Since the previous mechanism creates new clusters, we use a simple mechanism to also elim-

inate clusters when the corresponding centroids are very close: If the distance between two centroids is less that T times the average distance between centroids, one of them is eliminated.

Parameters Required. Our proposed approach requires the following parameters (The parameter T of the clustering algorithm used was fixed to $T = 1$): (1) Crossover rate (p_c) and mutation rate (p_m), (2) Maximum number of generations ($Gmax$), (3) Size of the initial population ($popsize_{init}$) to be used during the first stage and minimum size of the secondary population ($popsize_{sec}$) to be used during the second stage.

5 Results

To validate our approach, we performed both quantitative comparisons (adopting four metrics) and qualitative comparisons (plotting the Pareto fronts produced) with respect to the previous version of our approach (CO-MOEA) [4] and with respect to one MOEA that is representative of the state-of-the-art in the area: the Nondominated Sorting Genetic Algorithm II (NSGA-II) [2]. For our comparative study, we implemented the four following metrics:

Error Ratio (ER) [13]: This metric indicates the percentage of solutions (from the nondominated vectors found) that are not members of the true Pareto optimal set: $ER = \frac{\sum_{i=1}^{n} e_i}{n}$, where n is the number of vectors in the current set of nondominated vectors available; e_i = 0 if vector i is a member of the Pareto optimal set, and $e_i = 1$ otherwise.

Generational Distance (GD) [14]: The concept of generational distance was introduced as a way of estimating how far are the elements in the Pareto front produced by our algorithm from those in the true Pareto front of the problem: $GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n}$ where n is the number of nondominated vectors found by the algorithm being analyzed and d_i is the Euclidean distance (measured in objective space) between each of these and the nearest member of the true Pareto front.

Spacing (SP) [15]: This metric was proposed as a way of measuring the range (distance) variance of neighboring vectors in the Pareto front known: $SP = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\bar{d} - d_i)^2}$ where $d_i = min_j(\sum_{k=1}^{m} |f_k^i - f_k^j|)$, $i, j = 1, ..., n$, m is the number of objectives, $\bar{d}$ is the mean of all d_i, and n is the number of vectors in the Pareto front found by the algorithm being evaluated.

Two Set Coverage (SC) [16]: Consider X', X'' as two sets of objective vectors. SC is defined as the mapping of the order pair (X', X'') to the interval $[0, 1]$: $SC(X', X'') \triangleq |\{a'' \epsilon X''; \exists a' \epsilon X' : a' \preceq a''\}|/|X''|$. If all points in X' dominate or are equal to all points in X'', then by definition $SC = 1$. $SC = 0$ implies the opposite. In general, $SC(X', X'')$ and $SC(X'', X')$ both have to be considered due to set intersections not being empty.

For each of the test functions shown below, we performed 30 runs per algorithm and a total of 10,000 evaluations. The parameters for NSGA-II were $popsize$=100 and 100 generations. All the algorithms used a bit mutation probability (p_m) equal to $1/codesize$ and a crossover probability (p_c) equal 0.8. The Pareto fronts that we will show correspond to the median of the 30 runs with respect to the ER metric. Regarding

constraint-handling, we used the original scheme provided in the case of the NSGA-II. However, since our algorithm (both versions) does not have such a mechanism, we implemented a simple penalty function over the value of objective functions of each infeasible individual.

Test Function 1 Minimize $f_1(x_1, x_2) = x_1, f_2(x_1, x_2) = g(x_1, x_2)h(x_1, x_2)$
subject to: $g(x_1, x_2) = 11 + x_2^2 - 10cos(2\pi x_2)$

$$h(x_1, x_2) = \begin{cases} 1 - \sqrt{\frac{f_1(x_1,x_2)}{g(x_1,x_2)}} & f_1(x_1, x_2) \leq g(x_1, x_2) \\ 0 & \text{otherwise} \end{cases}$$

$0.0 \leq x_1 \leq 1.0, -30.0 \leq x_2 \leq 30.0$

In this example, our approach used: $popsize_{init} = 100$, $popsize_{rec} = 30$ (38 gen).

Test Function 2 Minimize $f_1(x) = \sum_{i=1}^{2}(-10e^{-0.2*\sqrt{x_i^2+x_{i+1}^2}})$,
$f_2(x) = \sum_{i=1}^{3}(|x_i|^{0.8} + 5sin(x_i^3))$ subject to: $-5.0 \leq x_1, x_2, x_3 \leq 5.0$

In this case, our approach used: $popsize_{init} = 100$, $popsize_{rec} = 30$ (40 gen).

Test Function 3 Minimize $f_1(x_1, x_2) = x_1, f_2(x_1, x_2) = x_2$
subject to: $g_1(x_1, x_2) = -x_1^2 - x_2^2 + 1 + 0.1cos(16arctan\frac{x_1}{x_2}) \leq 0$
$g_2(x_1, x_2) = (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2 - \frac{1}{2} \leq 0, 0.0 \leq x_1, x_2 \leq \pi$

In this example, our approach used: $popsize_{init} = 100$, $popsize_{rec} = 30$ (40 gen).

Table 1 shows the values of the metrics for each of the MOEAs compared.

6 Discussion of Results

As we can see in Table 1, in the first function the new version of our approach (CO-MOEA2) is clearly better than the previous version (CO-MOEA), with respect to all the metrics. On the other hand, although the results of the NSGA-II are better on average than the results of CO-MOEA2, the SC metric indicates that the Pareto fronts obtained by both algorithms are on average almost of the same quality.

As in the first function, in the second function CO-MOEA2 is better than CO-MOEA. However, in this case the results of the CO-MOEA with respect to the SP metric are weakly better than the results of CO-MOEA2. In this function, the results of CO-MOEA2 are better on average than the results of the NSGA-II, except for the SP metric. However, as in the first function, the SC metric indicates almost the same quality on the results of CO-MOEA2 and NSGA-II.

In the case of the third function, we can see that CO-MOEA and CO-MOEA2 obtained very similar results. CO-MOEA has better results only with respect to the SP metric. On the other hand, in this function the NSGA-II has better results on average only with respect to the ER metric. With respect to the SC metric, the NSGA-II obtained the best results, followed by CO-MOEA2 and CO-MOEA.

In general, from Table 1 and Figures 3 and 4, we can conclude that in the first two functions the new version of our approach has clearly improved the original version, and obtained very competitive results with respect to the NSGA-II. In the third function, CO-MOEA2 has obtained the same quality on the results than the CO-MOEA and NSGA-II. Finally, we can conclude that CO-MOEA2 needs to improve the results obtained with respect to the distribution (SP metric). We consider this as part of our future work.

Test Function 1						
		CO-MOEA	CO-MOEA2	NSGA-II	Two Set Coverage	
	best	0.54	0.02	0.00	X	SC(X,CO-MOEA)
	median	0.83	0.10	0.07	CO-MOEA	0.00
ER	worst	1.00	0.44	0.47	CO-MOEA2	0.67
	average	0.83	0.15	**0.13**	NSGA-II	0.83
	std. dev.	0.1223	0.1112	0.1289	Average	75%
	best	0.0004	0.0001	0.0047	X	SC(X,CO-MOEA2)
	median	0.7018	0.0040	0.0056	CO-MOEA	0.00
GD	worst	20.237	0.0910	0.0061	CO-MOEA2	0.00
	average	2.0042	0.0159	**0.0055**	NSGA-II	0.02
	std. dev.	3.9452	0.0249	0.0004	Average	1%
	best	0.0098	0.0045	0.0064	X	SC(X,NSGA-II)
	median	2.2077	0.0090	0.0073	CO-MOEA	0.00
SP	worst	47.351	0.9069	0.0084	CO-MOEA2	0.00
	average	4.8611	0.1344	**0.0073**	NSGA-II	0.00
	std. dev.	9.1779	0.2491	0.0006	Average	**0%**
Test Function 2						
		CO-MOEA	CO-MOEA2	NSGA-II	Two Set Coverage	
	best	0.61	0.12	0.16	X	SC(X,CO-MOEA)
	median	0.72	0.23	0.27	CO-MOEA	0.00
ER	worst	0.83	0.35	0.37	CO-MOEA2	0.66
	average	0.72	**0.24**	0.28	NSGA-II	0.62
	std. dev.	0.0557	0.0578	0.0578	Average	64%
	best	0.0299	0.0028	0.0032	X	SC(X,CO-MOEA2)
	median	0.0311	0.0032	0.0036	CO-MOEA	0.02
GD	worst	0.0332	0.0038	0.0044	CO-MOEA2	0.00
	average	0.0313	**0.0032**	0.0037	NSGA-II	0.07
	std. dev.	0.0008	0.0002	0.0004	Average	**5%**
	best	0.0387	0.0519	0.0450	X	SC(X,NSGA-II)
	median	0.0980	0.1100	0.0553	CO-MOEA	0.02
SP	worst	0.1282	0.1534	0.1060	CO-MOEA2	0.07
	average	0.0808	0.1069	**0.0606**	NSGA-II	0.00
	std. dev.	0.0288	0.0306	0.0156	Average	**5%**
Test Function 3						
		CO-MOEA	CO-MOEA2	NSGA-II	Two Set Coverage	
	best	0.09	0.05	0.01	X	SC(X,CO-MOEA)
	median	0.15	0.16	0.08	CO-MOEA	0.00
ER	worst	0.25	0.29	0.17	CO-MOEA2	0.25
	average	0.14	0.15	**0.08**	NSGA-II	0.21
	std. dev.	0.0404	0.0461	0.0339	Average	23%
	best	0.0009	0.0009	0.0008	X	SC(X,CO-MOEA2)
	median	0.0014	0.0012	0.0013	CO-MOEA	0.23
GD	worst	0.0015	0.0015	0.0016	CO-MOEA2	0.00
	average	0.0014	**0.0012**	**0.0012**	NSGA-II	0.17
	std. dev.	0.0001	0.0001	0.0002	Average	20%
	best	0.0051	0.0047	0.0065	X	SC(X,NSGA-II)
	median	0.0064	0.0085	0.0099	CO-MOEA	0.13
SP	worst	0.0077	0.0185	0.0155	CO-MOEA2	0.14
	average	0.0064	**0.0092**	0.0101	NSGA-II	0.00
	std. dev.	0.0007	0.0027	0.0022	Average	**14%**

Table 1. Comparison of results between the previous version of our approach (denoted by CO-MOEA [4]), the new version (CO-MOEA2) and the NSGA-II [2], for all the test functions.

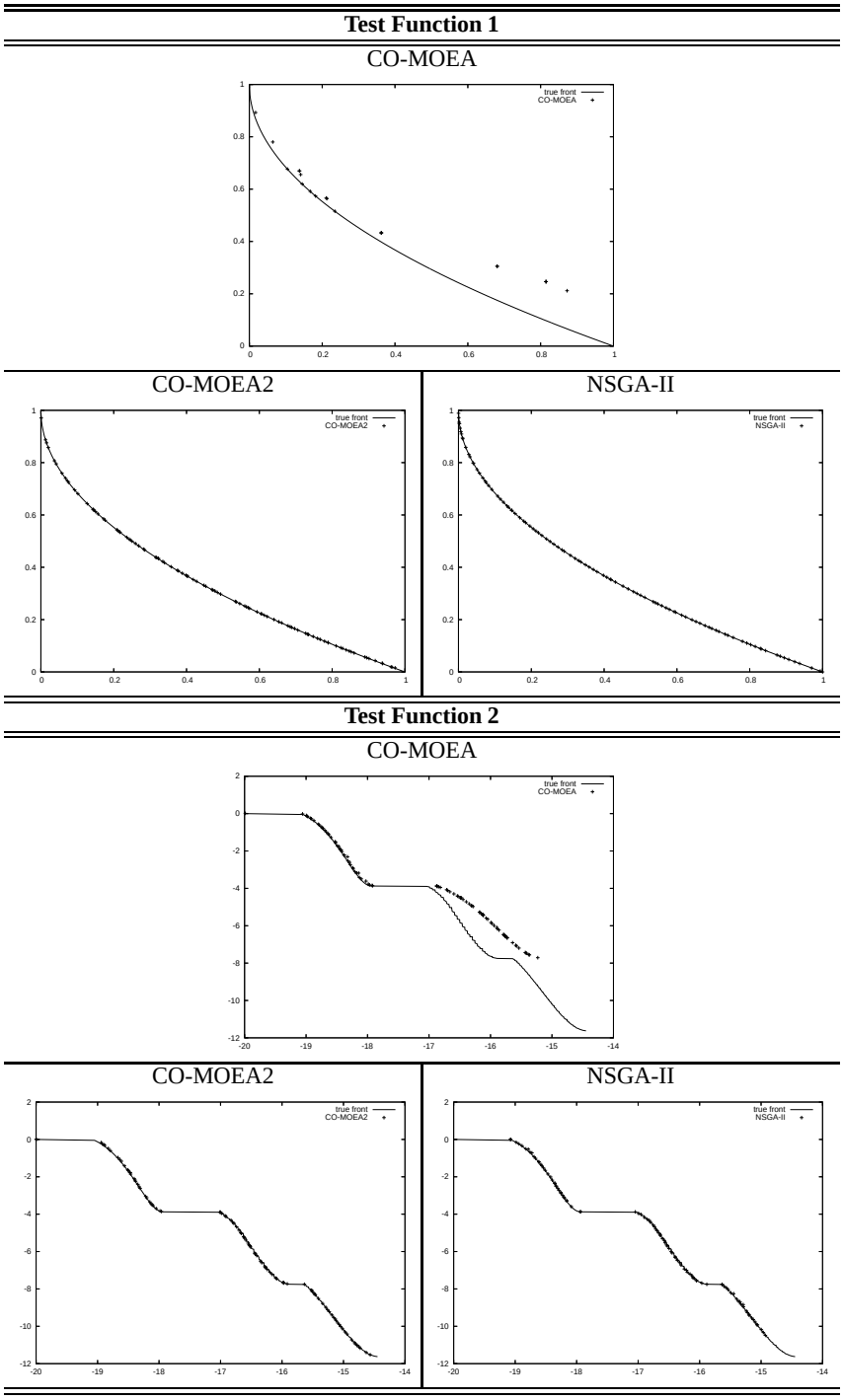

Fig. 3. Pareto fronts obtained by the previous version of our approach (CO-MOEA), the new version (CO-MOEA2) and the NSGA-II [2], for test functions 1 and 2.

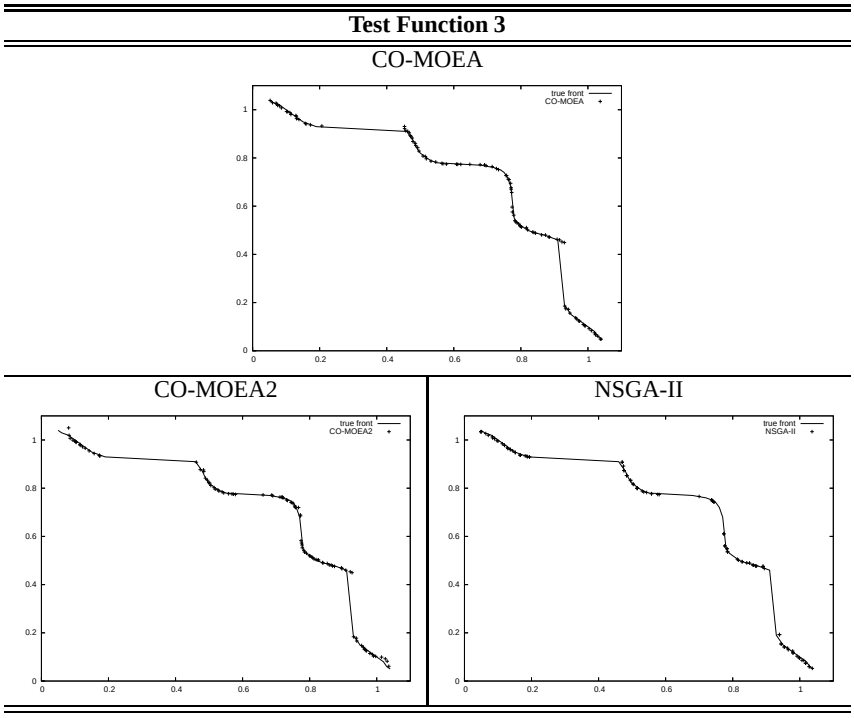

Fig. 4. Pareto fronts obtained by the previous version of our approach (CO-MOEA), the new version (CO-MOEA2) and the NSGA-II [2], for test function 3.

7 Conclusions and Future Work

We have presented a new version of a coevolutionary multi-objective evolutionary algorithm whose main idea is to detect the most "promising" sub-regions of the search space and focus the search on them. With this aim, the proposed algorithm applies a clustering algorithm on the set of decision variables of the known Pareto front. The proposed approach was validated using several test functions taken from the specialized literature. Our comparative study showed that the proposed approach has improved its original version and that it is very competitive with respect to another algorithm that is representative of the state-of-the-art in the area.

Since we are proposing a coevolutionary scheme that uses MOGA as its search engine, we consider as an interesting idea to use another multiobjective algorithm in order to improve the obtained results. This is a very important advantage of coevolutionary schemes. On the other hand, we also plan to replace the adaptive grid with another more efficient mechanism in order to improve our distribution results. Finally, we need to test our approach with high-dimensional functions so that we can study possible scalability difficulties.

Acknowledgments

The first author acknowledges support from CONACyT through a scholarship to pursue graduate studies at CINVESTAV-IPN. The second author acknowledges support from CONACyT project number 45683.

References

1. Coello Coello, C.A., Van Veldhuizen, D.A., Lamont, G.B.: Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers, New York (2002)
2. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. IEEE Transactions on Evolutionary Computation **6** (2002) 182–197
3. Knowles, J.D., Corne, D.W.: Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation **8** (2000) 149–172
4. Coello Coello, C.A., Reyes Sierra, M.: A Coevolutionary Multi-Objective Evolutionary Algorithm. In: Proceedings of 2003 CEC. Volume 1., IEEE Press (2003) 482–489
5. Paredis, J.: Coevolutionary algorithms. In Bäck, T., Fogel, D.B., Michalewicz, Z., eds.: The Handbook of Evolutionary Computation, 1st supplement. Institute of Physics Publishing and Oxford University Press (1998) 225–238
6. Potter, M., Jong., K.D.: A cooperative coevolutionary approach to function optimization. In: Proceedings from PPSN V, Springer-Verlag (1994) 530–539
7. Parmee, I.C., Watson, A.H.: Preliminary Airframe Design Using Co-Evolutionary Multiobjective Genetic Algorithms. In: Proceedings of GECCO'99. Volume 2., Morgan Kaufmann (1999) 1657–1665
8. Keerativuttitumrong, N., Chaiyaratana, N., Varavithya, V.: Multi-objective Co-operative Coevolutionary Genetic Algorithm. In et al., J.M.G., ed.: Proceedings of PPSN VII, Springer-Verlag (2002) 288–297
9. Tan, K., Chew, Y., Lee, T., Yang, Y.: A cooperative coevolutionary algorithm for multiobjective optimization. In: IEEE International Conference on Systems, Man and Cybernetics. Volume 1., IEEE Press (2003) 390–395
10. Iorio, A., Li, X.: A cooperative coevolutionary multiobjective algorithm using nondominated sorting. In: Proceedings of GECCO, LNCS 3102, Springer-Verlag (2004) 537–548
11. Fonseca, C.M., Fleming, P.J.: Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In: Proceedings of the Fifth International Conference on Genetic Algorithms, Morgan Kauffman Publishers (1993) 416–423
12. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, New Jersey (1988)
13. Van Veldhuizen, D.A.: Multiobjective Evolutionary Algorithms: Classifi cations, Analyses, and New Innovations. PhD thesis, Department of Electrical and Computer Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio (1999)
14. Van Veldhuizen, D.A., Lamont, G.B.: On Measuring Multiobjective Evolutionary Algorithm Performance. In: 2000 CEC. Volume 1., IEEE Service Center (2000) 204–211
15. Schott, J.R.: Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts (1995)
16. Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Evolutionary Computation **8** (2000) 173–195