Integrating Cross-Dominance Adaptation in Multi-Objective Memetic Algorithms

Andrea Caponio[1] and Ferrante Neri[2]

[1] Department of Electrotechnics and Electronics,
Technical University of Bari, Italy
caponio@deemail.poliba.it

[2] Department of Mathematical Information Technology,
University of Jyväskylä, Finland
neferran@cc.jyu.fi

This chapter proposes a novel adaptive memetic approach for solving multi-objective optimization problems. The proposed approach introduces the novel concept of cross-dominance and employs this concept within a novel probabilistic scheme which makes use of the Wigner distribution for performing coordination of the local search. Thus, two local searchers are integrated within an evolutionary framework which resorts to an evolutionary algorithm previously proposed in literature for solving multi-objective problems. These two local searchers are a multi-objective version of simulated annealing and a novel multi-objective implementation of the Rosenbrock algorithm.

Numerical results show that the proposed algorithm is rather promising and, for several test problems, outperforms two popular meta-heuristics present in literature. A real-world application in the field of electrical engineering, the design of a control system of an electric motor, is also shown. The application of the proposed algorithm leads to a solution which allows successful control of a direct current motor by simultaneously handling the conflicting objectives of the dynamic response.

1 Introduction

Many optimization problems in engineering and applied science, due to their nature, require the satisfaction of necessities of various kinds i.e. the desired candidate solution should perform well according to various objectives. In the vast majority of these cases, the objectives are in mutual conflict and a compromise must be accepted. More specifically, as these objectives are usually conflicting, it is not possible to find a single solution that is optimal with respect to all objectives. Which solution is the best depends on the users' utility function, i.e., how the different criteria are weighted. Unfortunately, it is usually rather difficult to formally specify user preferences before the alternatives are known. One way to solve this predicament is by searching for the whole Pareto-optimal front of solutions i.e., all solutions that can not be improved in any criterion without at least sacrificing another criterion.

For example, in control engineering, when a control system is designed, it is desirable that the speed response is very reactive to the input and, at the same time, contains no overshoot and oscillations. Under-dumped responses are usually very reactive but

C.-K. Goh, Y.-S. Ong, K.C. Tan (Eds.): Multi-Objective Memetic Alg., SCI 171, pp. 325–351.
springerlink.com

contain overshoot and oscillations in the settling process; on the contrary, over-dumped responses do not contain overshoot or oscillations but usually perform rather poorly in terms of reactivity. It is thus necessary to partially give up both the objectives and find a compromise, that is, a solution which is fairly reactive without excessive overshoot and oscillations.

The well-known scalarized approach [1] i.e. to associate a weight factor (on the basis of their importance) to each objective and then optimize the weighted sum, though in some cases rather efficient, implicitly accepts that a ranking of the importance of the objectives and the related proportion of how much each objective should be taken into account with respect to the others, is known beforehand. Moreover, the weighted sum approach has the main disadvantage that it implicitly excludes some solutions from the search since the a priori determination of the weighted value may assign a low fitness value to some solutions which could on the contrary be interesting in the application viewpoint. Therefore, in many cases it is preferable to employ a multi-objective approach [1]. Since the latter considers the objectives simultaneously and leads to a set of solutions, the user can choose by means of a decision making process the most suitable solution amongst those that are actually available.

Due to their structure, Evolutionary Algorithms (EA) have been proven to be very promising in multi-objective optimization and have been intensively used during the last two decades (see the implementation proposed in [2]). As shown in [3] and [4], Multi-objective Optimization Evolutionary Algorithms (MOEA) are very efficient in finding the Pareto-optimal or near Pareto-optimal solutions. Several algorithms have been designed for such a purpose, for example the Non-dominated Sorting Genetic Algorithm II (NSGA II) [5] and the Strength Pareto Evolutionary Algorithm-2 (SPEA-2) [6].

Memetic Algorithms (MAs) are population based meta-heuristics which combine local search components within an evolutionary framework [7]. For single-objective optimization problems MAs, if well-designed for specific applications by taking into account features of the fitness landscape, have been proven to outperform classical meta-heuristics e.g. Genetic Algorithms (GAs), Evolution Strategy (ES), Particle Swarm Optimization (PSO) etc. [8] [9]. One crucial problem in the algorithmic design of MAs is coordination among the evolutionary framework and local search and amongst the various local searchers [10]. The problem of local search coordination has been widely discussed over the years. In [7] the concept of coordination and cooperation of local searchers has been introduced, after being developed in [11]. In [10] the use of multiple local search operators having different features in order to explore the decision space under different perspectives has been proposed. In recent years, several kinds of adaptation and self-adaptation for coordinating the local search have been designed. In [12] a classification of adaptive MAs is given while a tutorial which organizes the basic concepts of MAs including the coordination of the local search is given in [13].

MAs have been recently applied to multi-objective optimization, as discussed in [14] and several Multi-Objective Memetic Algorithms (MOMA) have therefore been designed. In such design two crucial problems arise: the first is the proper definition of local search in a multi-objective environment, the second is the balance between global and local search in presence of many simultaneous objectives [15], [16]. This balance, which is strictly related to the local search coordination, is extremely difficult to be performed and, as highlighted in the empirical study reported in [17], an adaptation is so difficult to be defined that it might be preferable in several cases to employ simple time-dependant heuristic rules.

This chapter proposes an adaptation scheme based on the mutual dominance between non-dominated solutions belonging to subsequent generations uncoupled with a probabilistic criterion in order to coordinate and balance the global and local search within a MOMA. The proposed algorithm is called Cross Dominant Multi-Objective Memetic Algorithm (CDMOMA). Section 2 gives a detailed description of the algorithmic components and their interaction, Section 3 shows the behavior of the proposed algorithm in an extensive amount of test cases, Section 4 analyzes a real-world engineering problem, Section gives the conclusion of our work.

2 Cross Dominant Multi-Objective Memetic Algorithm

Let us consider a classical multi-objective optimization problem:

$$\left.\begin{array}{lll} \text{Minimize/Maximize} & f_m(x), & m = 1,2,\ldots,M \\ \text{subject to} & x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1,2,\ldots,n \end{array}\right\} \tag{1}$$

where f_m is the m^{th} single objective function, a solution x is a vector of n decision variables. Each decision variable is limited to take a value within a lower $x_i^{(L)}$ and an upper $x_i^{(U)}$ bound. These bounds define the decision space D.

In order to solve the problem in eq. (1), the CDMOMA has been designed. The CDMOMA is composed of an evolutionary framework resorting the NSGA-II and two local searchers, a multi-objective implementation of the Rosenbrock algorithm and of the Simulated Annealing respectively, adaptively coordinated by criterion based on mutual dominance amongst the individuals of two populations at two consecutive generations and a probabilistic scheme.

For the sake of completeness and better understanding of the CDMOMA the classical definitions of dominance [3] are given. Without a generality loss, the following definitions refer to the minimization of all the objective functions.

Definition 1. A solution $x^{(1)}$ is said to dominate the other solution $x^{(2)}$ ($x^{(1)} \preceq x^{(2)}$), if both conditions 1 and 2 are true:

1. The solution $x^{(1)}$ is no worse than $x^{(2)}$ in all objectives,
2. The solution $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective.

Definition 2. A solution $x^{(1)}$ is said to strictly dominate the other solution $x^{(2)}$ ($x^{(1)} \prec x^{(2)}$), if solution $x^{(1)}$ is strictly better than $x^{(2)}$ in all the M objectives.

2.1 The Evolutionary Framework

The Non-dominated Sorting Genetic Algorithm II (NSGA-II) introduced in [5], is the second and improved Version of the Non-dominated Sorting Genetic Algorithm proposed in [18], it is an elitist multi-objective evolutionary algorithm which proves to have high performance in terms of both quality and distribution of the detected non-dominated solutions.

Briefly, an initial sampling is performed pseudo-randomly within the decision space, thus generating S_{pop} individuals. At each generation, $S_{pop}/2$ parents are selected according to a binary tournament selection. For each pairwise comparison the winner is the dominating individual (for definition of dominance see [1] or [3]). If the individuals are non-dominant (to each other), the individual having a higher value in the crowding distance is selected. The crowding distance of an individual is a measure of the distance between the individual under examination and the other individuals belonging to the same set of non-dominated solutions (see [5] for details).

Then, for $S_{pop}/2$ times a pseudo-random value is generated. Each time, if this value is lower than 0.1, one individual is pseudo-randomly selected and then mutated; if, on the contrary, it is higher than 0.1, two parents are pseudo-randomly paired and undergo crossover.

Polynomial mutation [19] and simulated binary crossover [20], [21] are employed. Since mutation generates one child and the simulated binary crossover two children, an offspring population composed of a number of individuals between $S_{pop}/2$ and S_{pop} is thus generated.

This offspring population is merged to the population produced from the previous generation. Then, according to an elitist logic, S_{pop} individuals are selected for survival to the subsequent generation. The survivor selection scheme sorts individuals according to their rank i.e. divides the individuals into subsets according to their level of dominance. Thus, the subset of rank 1 is the set of non-dominated solutions, the subset of rank 2 is the set of non-dominated solutions if we remove those individuals belonging to the first subset, the subset of rank 3 is the set of non-dominated solutions after having removed the individuals of the first and second subset and so on. Within each subset, the individuals are then sorted on the basis of their crowding distance. More formally, for a given pair of individuals i and j, and indicating with i_r and i_{cd} the rank and crowding distance of i respectively, the partial order (here indicated with $\prec_n$) is defined as:

$$i \prec_n j \ IF \ (i_r < j_r) \ OR \ ((i_r = j_r) AND (i_{cd} > j_{cd}))$$

The sorting performed amongst a set of solutions by employing the formula above is called non-dominated sorting. The selected individuals compose the new population for the subsequent generation.

2.2 Local Searchers

The CDMOMA employs two local searchers within the generation loop of the NSGA-II evolutionary framework. These algorithms are a novel multi-objective implementation of the Rosenbrock algorithm and the Pareto Domination Multi-Objective Simulated Annealing (PDMOSA) proposed in [22]. In the following subsections a description of these two algorithms is given.

2.2.1 The Multi-Objective Rosenbrock Algorithm

The classical Rosenbrock Algorithm [23] is a single objective algorithm that works on a solution and attempts to improve upon it by means of a steepest descent pivot rule.

A novel implementation of the Multi-Objective Rosenbrock Algorithm (MORA) is proposed here. The MORA consists of the following. Starting from x, a trial is made in all the n orthogonal directions of the n-dimensional decision space. A trial over the i^{th} decision variable is performed by checking the value of $y = [x_1, x_2 ... x_i + stepLength, x_{i+1}, .., x_n]$ where the $stepLength$ is the step length i.e. the length of the exploratory step. When a new point y is generated, it is compared with the old one x. If the new point is not dominated by the old one we have a success. In such a case, the new point is retained ($x = y$) and the step length is multiplied by a positive factor α. If the new point is dominated by the old one we have a failure. In this case, the vector of variables is left unchanged and the step length is multiplied by a negative factor

```
i = 1;
initialize stepLength;
initialize SuccessAndFailure;
while budget condition
   generate next point y from point x:
      y_j = x_j for j = 1,...,n and j ≠ i;
      y_i = x_i · stepLength_i;
   if y is out of bounds
      f_k(y) = ∞ ∀ k = 1,...,M;
   else
      evaluate y;
   end-if
   if x ⪯ y
      stepLength_i = stepLength_i · β;
      if SuccessAndFailure_i == success
         SuccessAndFailure_i = successFailure;
      else
         SuccessAndFailure_i = failure;
      end-if
   else
      x = y;
      stepLength_i = stepLength_i · α;
      SuccessAndFailure_i = success;
   end-if
   if SuccessAndFailure_j == successFailure ∀ j = 1,...,n;
      rotate base by Gram and Schmidt procedure;
      initialize stepLength;
      initialize SuccessAndFailure;
   else-if i < n
      i = i + 1;
   else
      i = 1;
   end-if
end-while
```

Fig. 1. MORA pseudo-code

$-1 < \beta < 0$. According to Rosenbrock's suggestions $\alpha = 3$ and $\beta = -0.5$ have been set [23]. As in the single objective Rosenbrock algorithm, the process is repeated until at the least a success is followed by a failure in each direction. When such a condition is satisfied, the orthogonolization procedure of Gram and Schmidt (see [24]) is executed and the search, along the new set of directions, begins again. The algorithm is stopped when a budget condition is exceeded.

According to the given definitions of "success" and "failure", the MORA accepts a new point only when it does not decrease performance in each of the objective functions; if even one worsens, the point is discarded. Thus, the MORA handles the various objective functions without performing a scalarization.

It must be highlighted that when, during a MORA step, a solution outside the decision space is generated, the algorithm assigns an infinite value to every one of its objectives.

Fig. 1 shows the pseudo-code of the proposed MORA. It should be noted that the dominance condition is represented by the symbol $\prec$ e.g. x dominates y is expressed by $x \prec y$; analogously, x does not dominate y is expressed by $x \succeq y$ (see [3]). With reference to Fig. 1, the variable *SuccessAndFailure* is a vector of three valued flag variables which records for each of its elements $SuccessAndFailure_i$ the behavior of the algorithm during the previous two steps. More specifically, it records the value *failure* if the trial failed twice, it records the value *success* if the trial either succeeded twice or succeeded after having failed, it records the value *successFailure* if the trial failed after having succeeded. The latter condition determines the activation of the Gram and Smith procedure.

2.2.2 Pareto Domination Multi-Objective Simulated Annealing

The multi-objective simulated annealing algorithm implemented here is based on the Pareto Domination Multi-Objective Simulated Annealing PDMOSA (PDMOSA) proposed in [22]. The PDMOSA works on a solution x and an auxiliary population in order to improve upon the starting point. At each step, the current best solution is perturbed by means of a Gaussian distribution and a perturbed solution y is thus generated. For

```
while budget condition
   initialize y;
   while y is out of the bounds of the decision space
      generate y by perturbing xby means of a Gaussian distribution;
   end-while
   calculate all the single objective values of y;
   dx = number of the population individuals dominated by x;
   dy = number of the population individuals dominated by y;
   replace x with y with a probability p = e^((dx-dy)/T);
   decrease temperature T by means of an hyperbolic law;
end-while
```

Fig. 2. MOSA pseudo-code

both, current best and perturbed solution, the number of individuals of the population which are dominated by x and y respectively are calculated. In the fashion of simulated annealing the new solution y replaces x with a time-dependant probability. The temperature is decreased by means of a hyperbolic law as suggested in [25]. For the sake of clarity, the PDMOSA pseudo-code which highlights the working principles is shown in Fig. 2.

2.3 Adaptation

Definition 3. Let us consider two sets of candidate solutions, namely X and Y respectively. Without a generality loss, let's assume that the cardinality of both sets is N. By scrolling all the elements of set Y, let's enumerate the dominance occurrences with each element of set X. N^2 comparisons are thus performed. Let us assign Λ to be this number of dominance occurrences. The set Y is said to **cross-dominate** the set X **with a grade**:

$$\lambda = \frac{\Lambda}{N^2} \tag{2}$$

Fig. 3 gives a graphical representation of the concept of cross-dominance. The solid lined arrow represents the dominance of the point under examination while the dash lined arrow represents non-dominance.

This chapter proposes to use the concept of cross-dominance in order to perform an adaptive coordination of the local search. More specifically, at the end of each generation the parameter λ is calculated:

$$\lambda = \frac{\Lambda^{t+1}}{N^2} \tag{3}$$

where Λ^{t+1} is the number of dominance occurrences obtained by the comparison of the population at generation $t+1$ (which plays the role of the set Y in the definition above) with respect to the population at generation t (which plays the role of X).

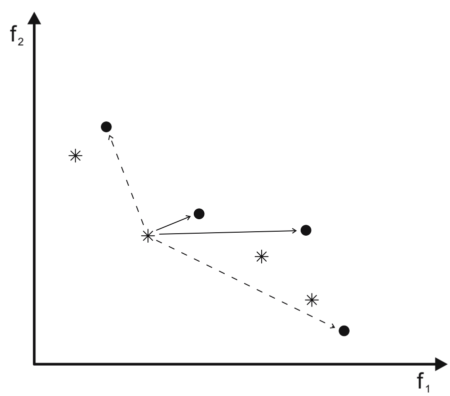

Fig. 3. Graphical Representation of the Cross-dominance

In this way, the algorithm can monitor the overall improvements of the population by means of a parameter which acquires values between 0 and 1. More specifically, if $\lambda = 1$ the algorithm is making excellent improvements and all individuals of the population at generation $t+1$ strictly dominate all individuals at generation t. On the contrary, if $\lambda = 0$ the algorithm is not leading to any improvement and the new population is equivalent to the old one in terms of dominance. It must be remarked that this adaptation index should be integrated within a fully elitist system (as the NSGA-II), thus a temporary worsening is not allowed. In addition, it should be observed that even though λ can acquire values between 0 and 1, most likely it will acquire values around zero ($\lambda = 0.05$ means that the population is still significantly better than the previous one).

The main idea is to design an adaptive system which automatically coordinates evolutionary framework and local search components by estimating algorithmic improvements, thus the necessity of the search during the optimization process.

2.4 Coordination of the Local Search

In order to perform coordination of the local search, λ is employed in a novel way. More specifically, for each local searcher, a generalized Wigner semicircle distribution is generated:

$$p(\lambda) = \frac{2}{\pi R^2}\sqrt{R^2 - (\lambda - a)^2}\frac{c}{\left(\frac{2}{\pi R}\right)} \tag{4}$$

where R is the radius of the distribution (the shape of the distribution depends on R), a determines the shift of the distribution, c is the maximum value of the distribution. For the MORA, we consider a distribution that has its maximum value equal to 0.8 for $\lambda = 0$, and that is 0 for $\lambda > 0.007$ (we consider just a half of the semi-elliptic Wigner distribution). For the PDMOSA, we consider a function that has its maximum value equal to 0.1 for $\lambda = 0.0125$, while it is 0 for $\lambda < 0.005$ and $\lambda > 0.02$. Thus, these two distributions return the probability of the local search activation dependent upon the adaptive parameter λ. Furthermore, the MORA is applied to 25 individuals while the PDMOSA is applied to 5 individuals pseudo-randomly selected respectively. Fig. 4 graphically shows the probability distribution for the CDMOMA adaptive local search coordination.

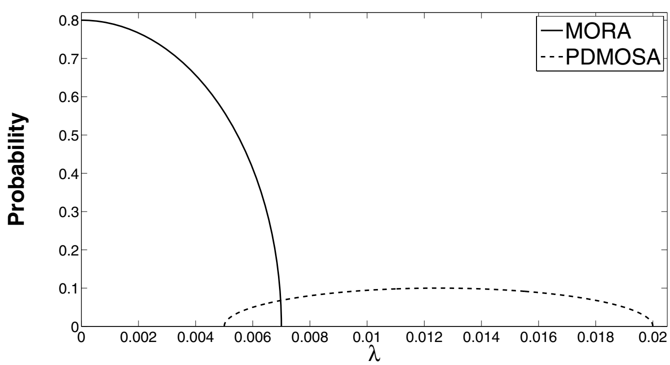

Fig. 4. Graphical Representation of the Probabilistic Scheme for the Local Search Coordination

```
generate initial population pseudo-randomly;
compute the fitness values of the individuals of the initial population;
perform the non-dominated sorting;
while budget condition
   execute NSGA-II generation;
   apply the cross-dominance procedure and compute λ = Λ^{t+1}/N^2;
   compute p(λ);
   generate pseudo-randomly ε ∈ [0,1]
   if ε < p(λ)
      execute PDMOSA on 5 individuals pseudo-randomly selected,
      for 3000 fitness evaluations;
      replace the 5 individuals with the results of the PDMOSA;
   end-if
   if ε < p(λ)
      execute RA on 25 individuals pseudo-randomly selected,
      for 1000 fitness evaluations;
   end-if
end-while
```

Fig. 5. CDMOMA pseudo-code

The two employed local searchers have clearly different structures in terms of pivot rule and neighborhood generating function. It should be noted that the MORA is a steepest descent local searcher which explores the neighborhood of a promising solution. On the contrary, the PDMOSA employs a simulated annealing logic which attempts to achieve a global property during the exploration of the decision space. According to our algorithmic philosophy, a decrease of the parameter λ during the optimization process corresponds to a settlement of the population over a set of non-dominated solutions. These solutions will most likely be better spread out by the evolutionary framework without any improvement, in terms of quality, of the detected solutions. In such conditions, the PDMOSA has the role of providing a new perspective into the search and hopefully detects new non-dominated solutions in still unexplored areas of the decision space. When, notwithstanding this action (by the PDMOSA) the populations' new generation seems to have negligible improvements, the MORA attempts to further improve the solutions by exploring their neighborhood. In other words, the CDMOMA attempts, at first, to generate a set of non-dominated solutions by the NSGA-II, then combines the actions of the evolutionary components and local searchers for improving the performance of the non-dominated set and eventually employs the local search to further improve the solutions and the NSGA-II to assure a good spread to the set.

Fig 5 shows the pseudo-code of the proposed CDMOMA.

3 Numerical Results

The CDMOMA has been tested on eight popular test problems: FON (from Fonseca and Fleming's study [26]), POL(from Poloni's study [27]), KUR (from Kursawe's study

[28]) and five ZDT problems (from Zitzler, Deb and Tiel) selected from [29] formulated according to the study in [30].

Table 1 lists all the test problems under examination and the related details.

Table 1. Test Problems

Prob.	n	Bounds	Objective functions	Solutions	Comments
ZDT1	30	$[0,1]$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}}\right]$ $g(x) = 1 + 9\frac{\sum_{i=2}^{n} x_i}{n-1}$	$x_1 \in [0,1]$, $x_i = 0$, $i = 2,..,n$	convex
ZDT2	30	$[0,1]$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \left(\frac{x_1}{g(x)}\right)^2\right]$ $g(x) = 1 + 9\frac{\sum_{i=2}^{n} x_i}{n-1}$	$x_1 \in [0,1]$, $x_i = 0$, $i = 2,..,n$	nonconvex
ZDT3	30	$[0,1]$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ $g(x) = 1 + 9\frac{\sum_{i=2}^{n} x_i}{n-1}$	$x_1 \in [0,1]$, $x_i = 0$, $i = 2,..,n$	convex, disconnected
ZDT4	10	$x_1 \in [0,1]$ $x_i \in [-5,5]$ $i = 2,..,n$	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{\frac{x_1}{g(x)}}\right]$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10\cos(4\pi x_i)\right]$	$x_1 \in [0,1]$, $x_i = 0$, $i = 2,..,n$	nonconvex
ZDT6	10	$[0,1]$	$f_1(x) = 1 - \exp(-4x_1)\sin^6(4\pi x_1)$ $f_2(x) = g(x)\left[1 - \left(\frac{f_1(x)}{g(x)}\right)^2\right]$ $g(x) = 1 + 9\left[\frac{\sum_{i=2}^{n} x_i}{n-1}\right]^{0.25}$	$x_1 \in [0,1]$, $x_i = 0$, $i = 2,..,n$	nonconvex, nonuniformly spread
FON	3	$[-4,4]$	$f_1(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_1(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$	$x_1 = x_2 =$ $= x_3 \in$ $\left[\frac{-1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right]$	nonconvex
POL	2	$[-\pi,\pi]$	$f_1(x) = \left[1 + (A_1 - B_1)^2 + (A_2 - B_2)^2\right]$ $f_2(x) = \left[(x_1+3)^2 + (x_2+1)^2\right]$ $A_1 = 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2$ $A_2 = 1.5\sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2$ $B_1 = 0.5\sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2$ $B_2 = 1.5\sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2$		nonconvex, disconnected
KUR	3	$[-5,5]$	$f_1(x) = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right)$ $f_2(x) = \sum_{i=1}^{n}\left(\lvert x_i\rvert^{0.8} + 5\sin x_i^3\right)$		nonconvex

The CDMOMA performance has been compared with the SPEA-2 [6] and the NSGA-II [5]. The three algorithms have been executed with a population size of 150 individuals, with a total budget of 800000 fitness evaluations. For each test problem 50 initial populations have been pseudo-randomly sampled within the respective decision space. For each of these 50 populations the three algorithms have been independently run. Thus, for each test problem each algorithm has been run 50 times. Fig.'s 6, 7, 8, 9, 10, 11, 12, 13 show the results obtained on selected runs.

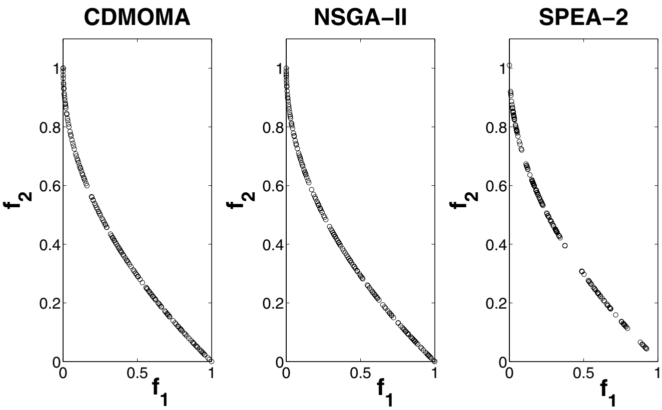

Fig. 6. ZDT1, selected solutions

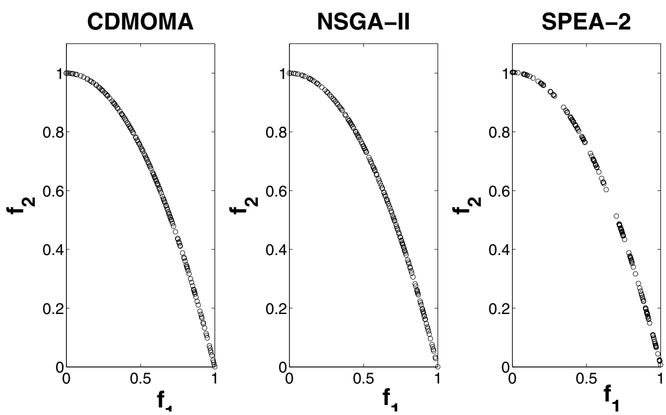

Fig. 7. ZDT2, selected solutions

Numerical results on selected runs qualitatively show that the CDMOMA is able to detect very good sets of non-dominated solutions in terms of fitness values and spreading.

In order to also give a graphical representation of the average algorithmic performance, for the single run of each algorithm, the final population has been sorted on the basis of the first objective function. For each algorithm, the sorted objective function values are averaged over each objective. Fig.'s 14, 15, 16, 17, 18, 19, 20, 21 show the average algorithmic performance.

Numerical results indicate that the CDMOMA seems to have a promising behavior with most of the problems under examination. In particular, in the case of the POL, the CDMOMA has a performance comparable to that of the NSGA-II and better than the SPEA-2; in the case of the FON, KUR, ZDT1, and ZDT3, the CDMOMA seems to be significantly more efficient than the SPEA-2 in detecting a good set of solutions

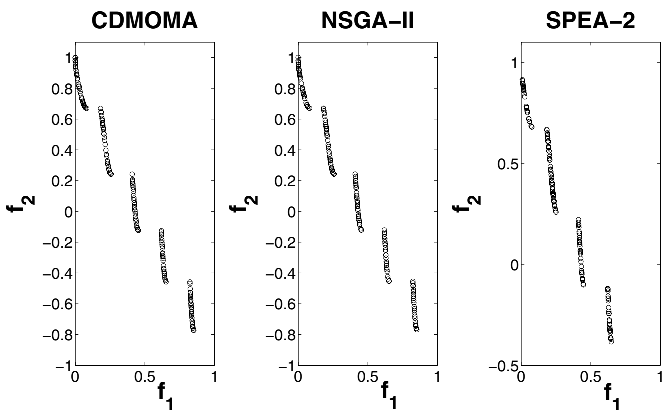

Fig. 8. ZDT3, selected solutions

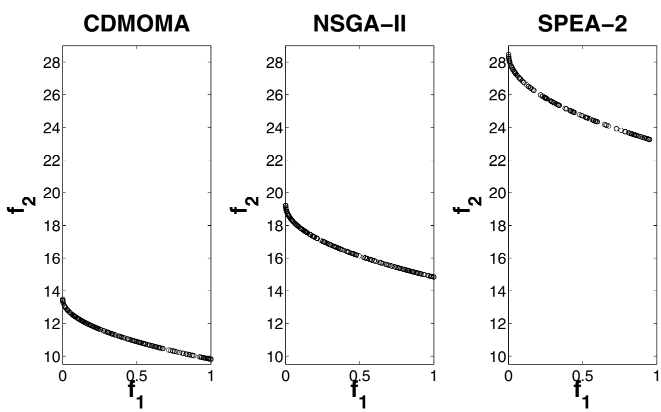

Fig. 9. ZDT4, selected solutions

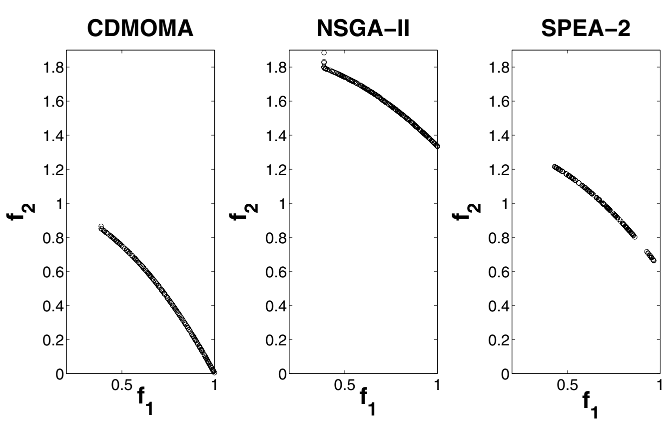

Fig. 10. ZDT6, selected solutions

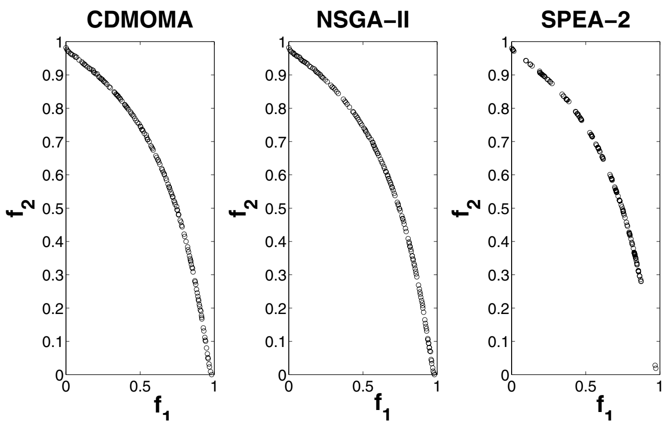

Fig. 11. FON, selected solutions

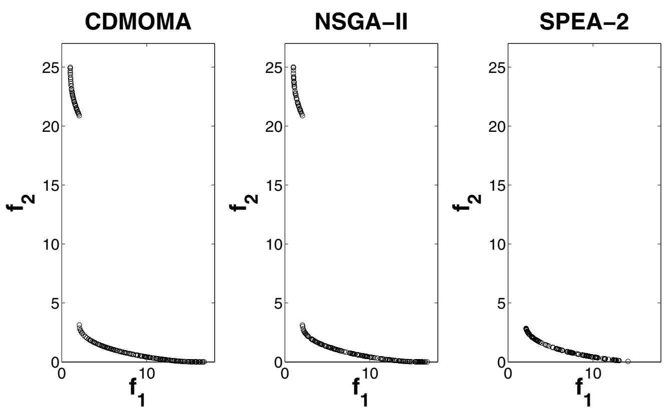

Fig. 12. POL, selected solutions

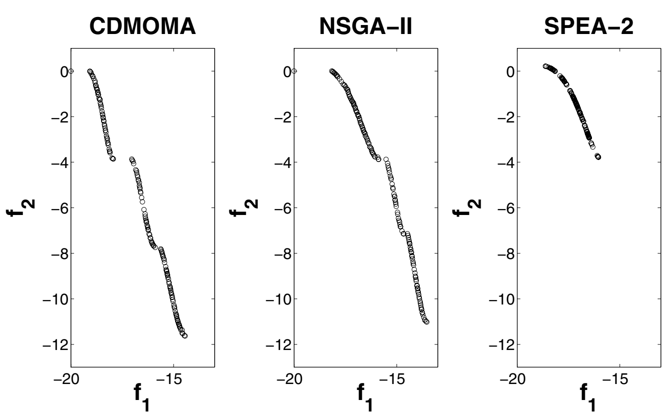

Fig. 13. KUR, selected solutions

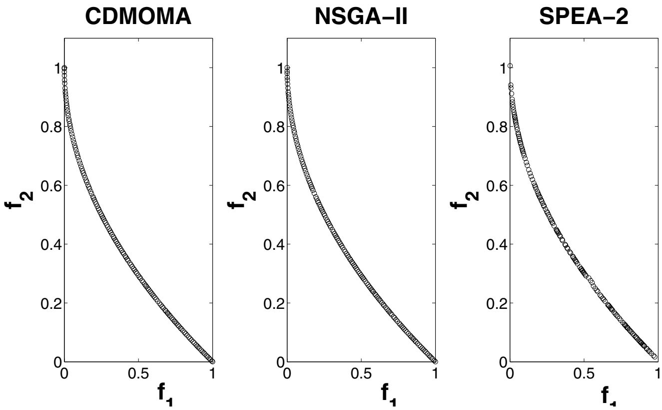

Fig. 14. ZDT1, average performance

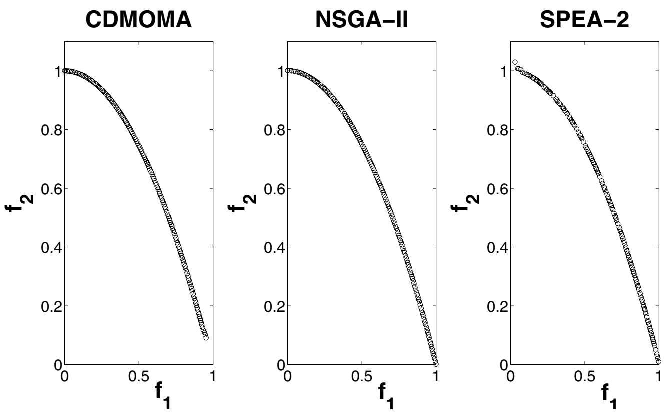

Fig. 15. ZDT2, average performance

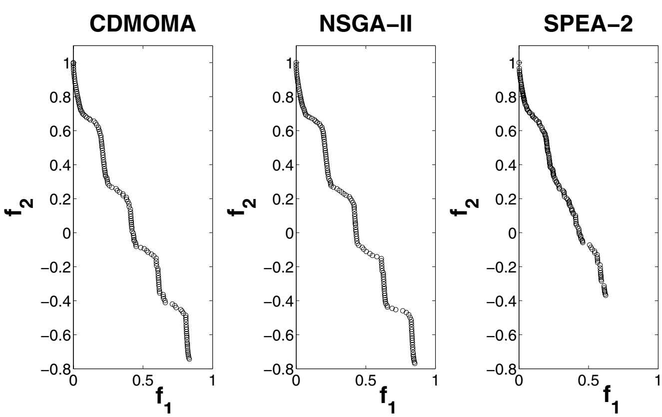

Fig. 16. ZDT3, average performance

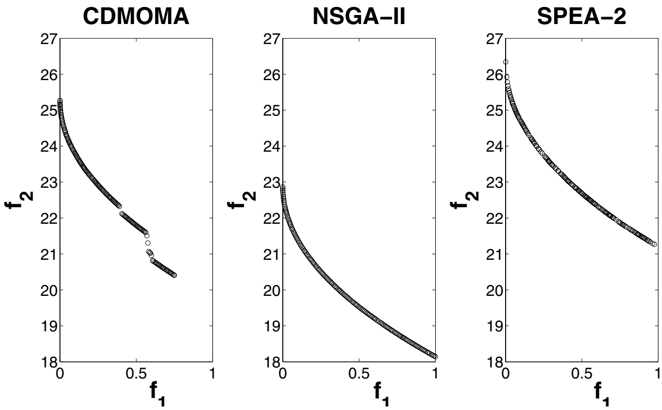

Fig. 17. ZDT4, average performance

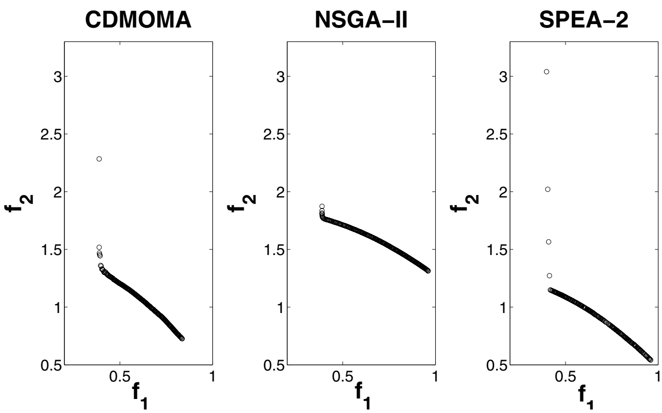

Fig. 18. ZDT6, average performance

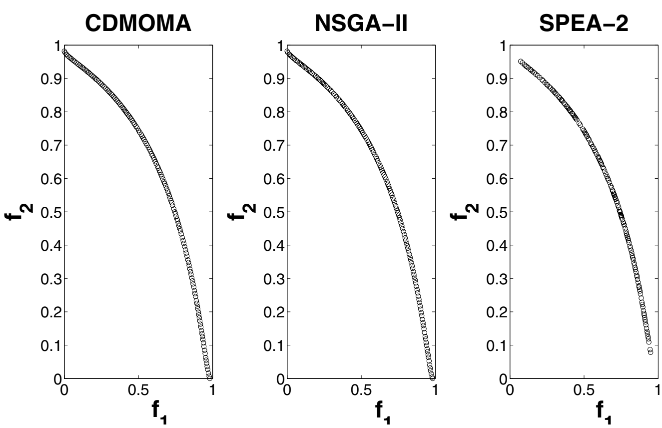

Fig. 19. FON, average performance

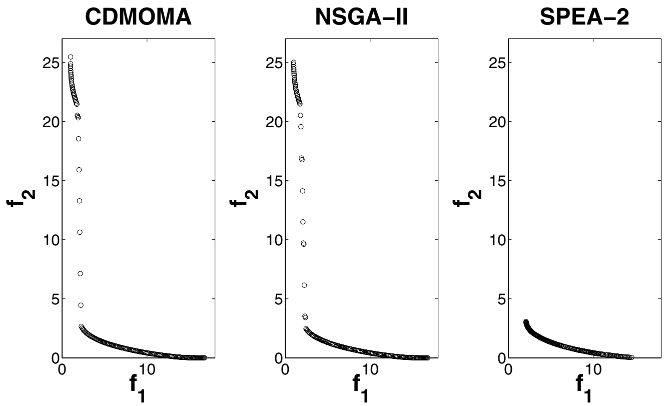

Fig. 20. POL, average performance

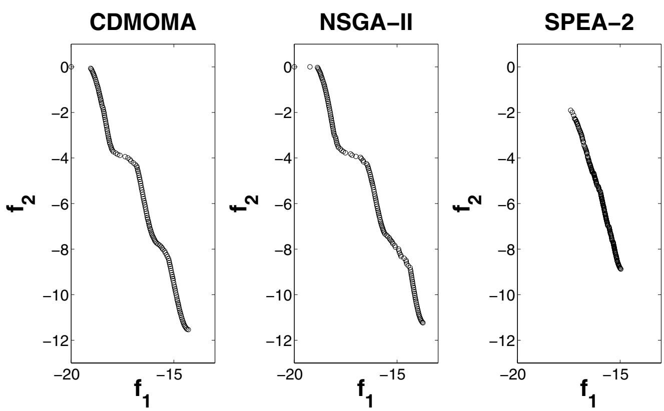

Fig. 21. KUR, average performance

and slightly more efficient than the NSGA-II; in the case of of ZDT2, the CDMOMA behaves slightly worse than the NSGA-II and comparably to the SPEA-2; in the case of the ZDT6, the CDMOMA seems to behave better than the NSGA-II but worse than the SPEA-2; in the case of the ZDT4, the CDMOMA is definitely worse than the NSGA-II and globally comparable to the SPEA-2. Thus, it can be stated that, except in the case of the ZDT4 test problem, the CDMOMA detects on average a set of solutions with high performance and good spreading features.

In order to have a more quantitative comparison by means of the performance measures, Υ and Δ (see [5] and [31]) has been carried out. The first metric Υ measures the extent of convergence to a known set of Pareto-optimal solutions. First, a set of 500 uniformly spaced solutions from the true Pareto-optimal front is detected. For each solution obtained with an algorithm, the minimum Euclidean distance it has from the 500 chosen solutions on the true Pareto-optimal front is computed. The average of these

Table 2. Υ values

	NSGA-II		CDMOMA		SPEA-2	
	$\bar{\Upsilon}$	σ^2_Υ	$\bar{\Upsilon}$	σ^2_Υ	$\bar{\Upsilon}$	σ^2_Υ
ZDT1	0.0012	$8.1314 \cdot 10^{-9}$	0.0011	$9.7175 \cdot 10^{-9}$	0.0111	$8.6124 \cdot 10^{-6}$
ZDT2	0.0014	$3.9939 \cdot 10^{-6}$	0.0008	$1.0723 \cdot 10^{-7}$	0.0136	$1.3993 \cdot 10^{-5}$
ZDT3	0.0014	$4.7059 \cdot 10^{-9}$	0.0013	$2.8252 \cdot 10^{-7}$	0.0128	$2.3767 \cdot 10^{-5}$
ZDT4	19.1313	$4.1036 \cdot 10^{1}$	21.7563	$1.0508 \cdot 10^{2}$	23.3591	$5.5884 \cdot 10^{1}$
ZDT6	0.8279	$1.2301 \cdot 10^{-1}$	0.4678	$5.7394 \cdot 10^{-1}$	0.4748	$5.1439 \cdot 10^{-2}$
FON	0.0061	$4.5006 \cdot 10^{-8}$	0.0061	$3.4269 \cdot 10^{-8}$	0.0071	$1.7740 \cdot 10^{-7}$

distances is used as the first metric Υ. In other words, Υ known also as the convergence metric, is a measurement of deviation of the detected set of solutions from the true Pareto-optimal front. Thus, it can be concluded that if $\Upsilon \approx 0$ algorithm is efficient. It should be remarked that this metric can be employed only when the true set of Pareto-optimal solutions is known. Thus, it is obvious that this metric cannot be used for any arbitrary problem.

The second metric Δ measures the extent of spread achieved among the obtained solutions, since one of the goals in multi-objective optimization is to acquire a set of solutions that spans the entire Pareto-optimal region. In order to compute Δ, the Euclidean distance d_i (in the multi-dimensional codomain) between consecutive solutions (with respect to the sorting according to one arbitrary objective function) in the obtained non-dominated set of solutions is calculated. The average of these distances $\bar{d}$ is then calculated. Then, if the true Pareto-optimal front is known, the Euclidean distances d_f and d_l between the extreme solutions and the boundary solutions of the obtained non-dominated set are calculated. The non-uniformity metric Δ is given by:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \overline{d} \right|}{d_f + d_l + (N-1)\,\overline{d}}, \tag{5}$$

where N is the cardinality of the non-dominated set. If the true Pareto-optimal front is not known, d_f and d_l are ignored by imposing $d_f = d_l = 0$. Formula (5) is thus modified:

$$\Delta = \frac{\sum_{i=1}^{N-1} \left| d_i - \overline{d} \right|}{(N-1)\,\overline{d}}. \tag{6}$$

For further details see [5] and [31]. Since a high spreading, in the non-dominated set of solutions, is desired, $\Delta \approx 0$ characterizes a good set of solutions.

For each algorithm and each test problem, the average values $\bar{\Upsilon}$ and $\bar{\Delta}$ have been calculated over the 50 runs available. A graphical representation of the $\bar{\Upsilon}$ and $\bar{\Delta}$ values are given in Fig. 22 and 23 respectively. Since the calculation of Υ requires the a priori knowledge of the actual Pareto front which is unknown for POL and KUR test problems, the Υ values related to these two problems are missing in the following analysis.

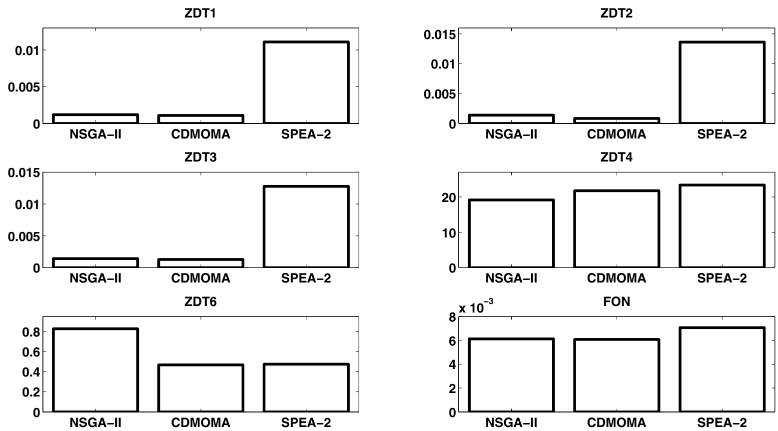

Fig. 22. $\bar{\Upsilon}$ values

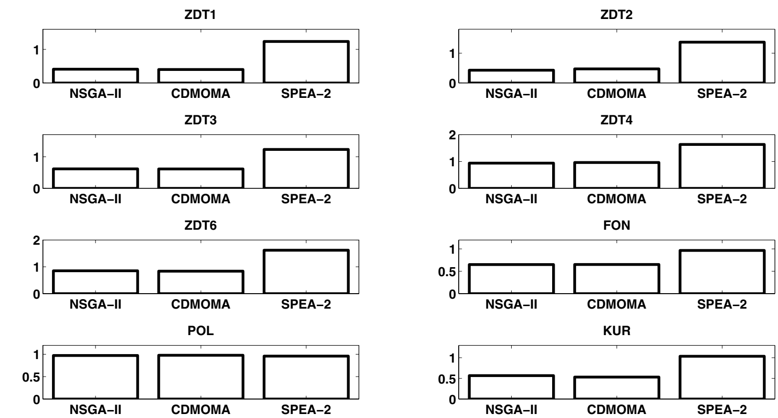

Fig. 23. $\bar{\Delta}$ values

Table 2 lists average and variance values of Υ and Table 3 lists average and variance values of Δ.

The quantitative analysis of the results shows that, regarding the convergence property of the algorithms, the CDMOMA seems to have a very promising capability of detecting a set of solutions which is similar to the true Pareto-optimal front. In particular, Fig. 22 and Table 2 show that the CDMOMA obtained the best convergence metric Υ for all the available test problems except the ZDT4. In the latter case, the CDMOMA still performs better than the SPEA-2. Regarding spreading of the solutions within the set, the CDMOMA is also rather promising. Results in Fig. 23 and Table 3 show that the CDMOMA has a better performance than the SPEA-2 (except for the POL) and comparable to the NSGA-II. This finding was somehow expectable since the CDMOMA

Table 3. Δ values

	NSGA-II		CDMOMA		SPEA-2	
	$\bar{\Delta}$	σ^2_{Δ}	$\bar{\Delta}$	σ^2_{Δ}	$\bar{\Delta}$	σ^2_{Δ}
ZDT1	0.4108	$5.6038 \cdot 10^{-4}$	0.4003	$3.5590 \cdot 10^{-4}$	1.2347	$2.8580 \cdot 10^{-3}$
ZDT2	0.42834	$6.0626 \cdot 10^{-4}$	0.4723	$7.5048 \cdot 10^{-3}$	1.3670	$2.9583 \cdot 10^{-3}$
ZDT3	0.6147	$4.1059 \cdot 10^{-4}$	0.6126	$8.1888 \cdot 10^{-4}$	1.2306	$2.8785 \cdot 10^{-3}$
ZDT4	0.9395	$2.4384 \cdot 10^{-4}$	0.9620	$1.6705 \cdot 10^{-3}$	1.6331	$9.9375 \cdot 10^{-3}$
ZDT6	0.8521	$3.6873 \cdot 10^{-3}$	0.8359	$1.1110 \cdot 10^{-1}$	1.6178	$9.1970 \cdot 10^{-3}$
FON	0.6491	$1.7710 \cdot 10^{-4}$	0.6520	$2.7579 \cdot 10^{-4}$	0.9660	$1.0791 \cdot 10^{-3}$
POL	0.9722	$2.5309 \cdot 10^{-4}$	0.9775	$6.3631 \cdot 10^{-4}$	0.9583	$2.7690 \cdot 10^{-3}$
KUR	0.5659	$3.7139 \cdot 10^{-3}$	0.5313	$2.4347 \cdot 10^{-3}$	1.0343	$2.5964 \cdot 10^{-2}$

employs the the NSGA-II logic in its evolutionary framework and thus both algorithms have the same sorting structure, this being an algorithmic component that heavily affects the spreading in the population.

In conclusion the results from the set of benchmark problems allow the authors to state that the proposed CDMOMA is a rather promising algorithm for multi-objective optimization problems. According to our interpretation, employment of the local search algorithms allows an improvement upon the evolutionary framework (NSGA-II) in detection of a non-dominated set which performs highly in terms of fitness values. On the other hand, the evolutionary framework guarantees an efficient spreading of the solutions. The proposed adaptation seems, also, to be efficient in the coordination of local search components. Finally, the cross-dominance criterion defined in this chapter is an instrument for comparing two sets of solutions and thus monitor the algorithmic improvements. This information can be generally useful since it can be employed as a feedback in the design of an adaptive algorithm for multi-objective optimization problems.

4 Real World Application: Design of a DC Motor Speed Controller

Nowadays most motion actuators are set up with electric motors since they offer high performance in terms of power density, efficiency, compactness and lightness. On the other hand, in order to have satisfactory functioning of the motor, an effective control is needed. Basically, an efficient motor control can be achieved either by applying a complex and expensive control system (see [32], [33], [34], [35]) or by using a simple and cheap control system, e.g. Proportional Integral (PI) based [36], which requires a design often very difficult to perform. In the latter case, the control design of an electric motor consists of detecting those system parameters that ensure a good system response in terms of speed and current behavior. This leads to a multi-objective optimization problem too complex for analytical solution [37]. Moreover, the application of classical design strategy [38], [39], [40] likely leads to unsatisfactory results. Thus, during recent years, interest in computational intelligence techniques has increased (see [41], [42] and [43]).

This chapter attempts to apply the CDMOMA to the control design of the Direct Current (DC) Motor whose electrical and mechanical features are shown in Table 4.

Table 4. DC Motor Nameplate

Parameter	Value
Armature resistance	$2.13\ \Omega$
Armature induction	$0.0094\ H$
Moment of inertia	$2.4e^{-6}\ Kg \cdot m^2$
Rated armature voltage	$12\ V$
Rated armature current	$1.2\ A$
Rated load torque	$0.0213\ Nm$
Rated speed	$400\ rad/s$

Fig. 24 shows the block diagram of the control scheme.

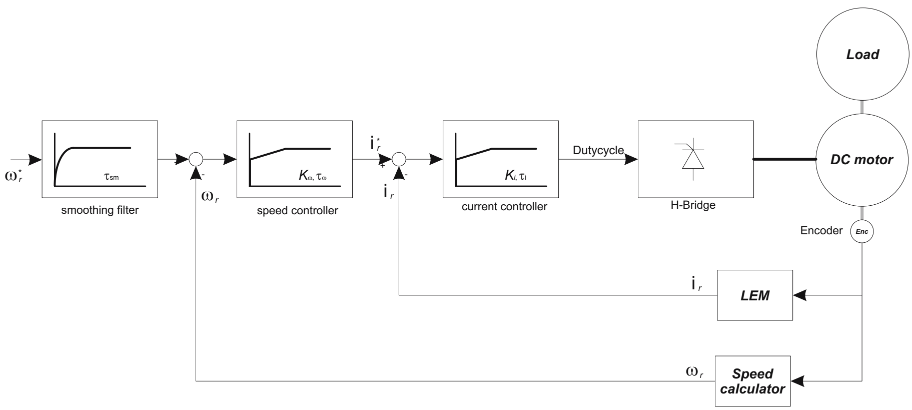

Fig. 24. Block diagram of a DC motor control

The control scheme is based on dynamic equations of the motor:

$$v_a = R_a \cdot i_a + L_a \cdot \frac{di_a}{dt} + e \tag{7}$$

$$v_f = R_f \cdot i_f + L_f \cdot \frac{di_f}{dt} \tag{8}$$

$$e = K\Phi \cdot \omega \tag{9}$$

$$T = K\Phi \cdot i_a \tag{10}$$

$$J \cdot \frac{d\omega}{dt} = T - T_r \tag{11}$$

where v_a is the voltage applied to the armature circuit, v_f is the voltage applied to the excitation circuit, R_a, R_f, L_a, L_f, i_a and i_f are the resistance, inductance and current for the armature and the excitement circuits respectively, T and T_r are the electromagnetic and load torque respectively, $K\Phi$ is the torque constant, ω is the rotor speed, J is the moment of inertia and e is the voltage generated by the rotor of the electric machine while rotating.

The DC motor control system is composed of two PI controllers. The first is used to control current and the second speed. The PIs transfer functions of the current and the speed controls are respectively $K_{pi} + \frac{K_{ii}}{s}$ and $K_{p\omega} + \frac{K_{i\omega}}{s}$. The speed reference is pre-filtered through a smoothing filter to reduce overshoot and the current required by the control in response to a speed step. The transfer function of the smoothing filter is $\frac{1}{(1+\tau_s m)}$.

With reference to Fig. 24, the control design consists of determining the parameters K_{pi}, K_{ii}, $K_{p\omega}$, $K_{i\omega}$ and τ_{sm} which guarantee very small values in rise and settling time, steady state error and overshoots. The decision space $H \subset \Re^5$ is a five dimensional hyper-rectangle given by the Cartesian product constructed around solution x_0 obtained by applying the classical symmetrical optimum (SO) criterion to design the speed regulator and the absolute value optimum (AVO) criterion to design the current regulator [44]. The lower and upper bounds of each interval have been set according to the following equations:

$$x_{lb}(i) = 10^{-6} \cdot x_0(i) \tag{12}$$

$$x_{ub}(i) = 3 \cdot x_0(i) \tag{13}$$

In order to evaluate the performance of each candidate solution, the four speed and load torque step training test shown in Fig. 25 is simulated by means of Matlab/Simulink as a discrete time control drive in order to realistically emulate an industrial digital drive. The control design of the DC Motor consists of determining a solution

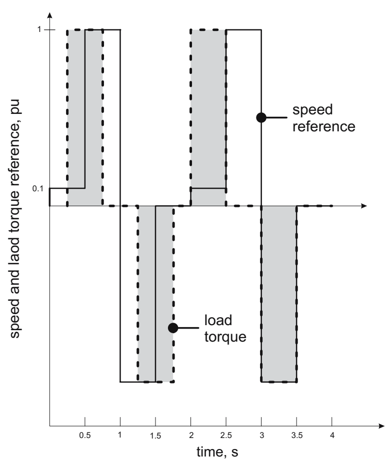

Fig. 25. Training test is a combination of speed commands and load torque

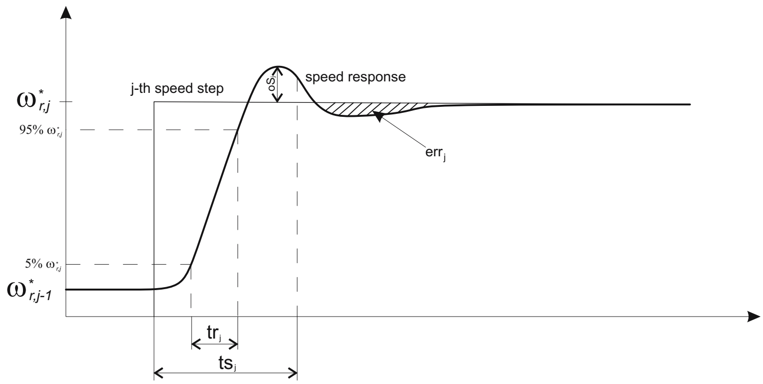

Fig. 26. j_{th} speed step of the training and values for objective function evaluation

$x = [K_{pi}, K_{ii}, K_{p\omega}, K_{i\omega}, \tau_{sm}]$ which satisfies the following multi-objective optimization problem:

$$\left.\begin{array}{ll} \text{Minimize} & \sum_{j=1}^{4} oS_j,\ \sum_{j=1}^{4} tr_j,\ \sum_{j=1}^{4} ts_j,\ \sum_{j=1}^{4} err_j \\ \text{Within} & \qquad\qquad H \end{array}\right\} \tag{14}$$

where oS_j is the overshoot, tr_j the rise time, ts_j the settling time and err_j the sum of the absolute values of the speed error in settling condition during the j_{th} trial step.

Fig. 26 illustrates oS_j, tr_j, ts_j and err_j for the generic j_{th} step of the training test. Finally, it must be remarked that, since each fitness evaluation requires a computationally expensive simulation test (8 s for each evaluation, see [45]), the problem is very demanding in terms of computational overhead.

The CDMOMA has been applied and its performance compared with the SPEA-2 and the NSGA-II. For each algorithm, 25 runs have been performed with a population size equal to 40. The average and variance values of Δ are listed in Table 5. The values related to Υ are obviously missing since the actual Pareto is unknown.

The results in Table 5 show that for the problem under study, the SPEA-2 seems to have slightly better performance than the other algorithms in terms of spreading of the solutions.

In order to detect the most suitable control design the following decision making process has been implemented. For each algorithm, all the final populations related to the 25 runs have been merged. At first, all the individuals having an error $\sum_{j=1}^{4} err_j$ above a threshold value (200 rad) are discarded. This condition means that during the entire

Table 5. Δ values for the DC Motor Control Design

NSGA-II		CDMOMA		SPEA-2	
$\bar{\Delta}$	σ^2_Δ	$\bar{\Delta}$	σ^2_Δ	$\bar{\Delta}$	σ^2_Δ
0.8951	$2.5601 \cdot 10^{-2}$	0.8375	$1.4176 \cdot 10^{-2}$	0.6858	$1.5762 \cdot 10^{-2}$

Table 6. Single objective values after the decision making

	$\sum_{j=1}^{4} oS_j$	$\sum_{j=1}^{4} tr_j$	$\sum_{j=1}^{4} ts_j$	$\sum_{j=1}^{4} err_j$
NSGA-II	8	0.1780	0.2640	8.3330
CDMOMA	10	0.1750	0.2610	7.9150
SPEA-2	145	0.2250	0.2920	19.4240

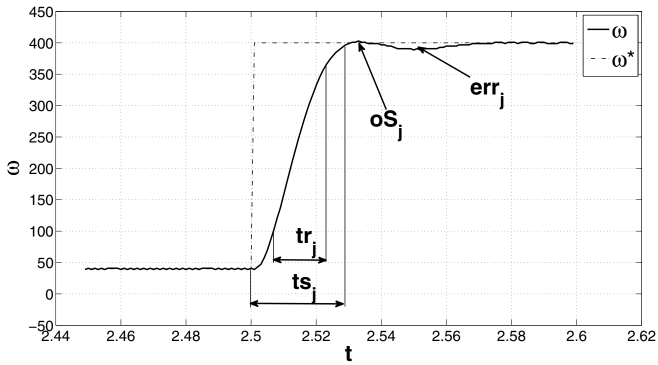

Fig. 27. Zoom detail of the speed response of NSGA-II solution

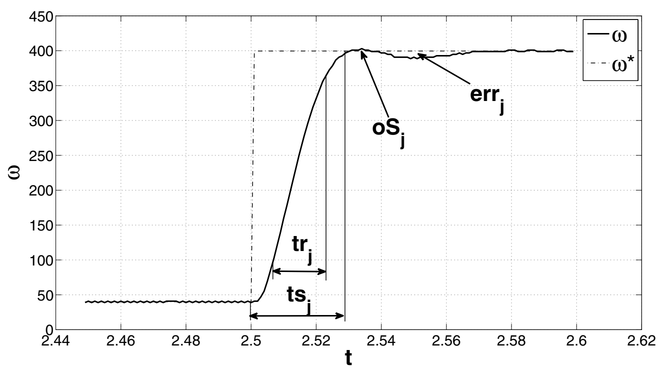

Fig. 28. Zoom detail of the speed response of CDMOMA solution

training test, the overall deviation (the sum of all the deviations) of the rotor position from the reference axis should not be more than 200 rad. Amongst the remaining solutions, all the individuals having a settling time $\sum_{j=1}^{4} ts_j$ above 0.35 s are discarded; amongst the remaining solutions, all the individuals having a rise time $\sum_{j=1}^{4} tr_j$ above 0.2

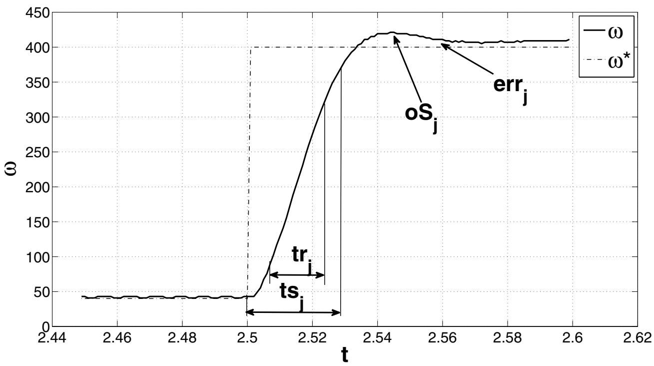

Fig. 29. Zoom detail of the speed response of SPEA-2 solution

s are discarded; amongst the remaining solutions, the solution having a minimal value in the overshoot $\sum_{j=1}^{4} oS_j$ is eventually selected.

The single objective values given by solutions obtained at the end of the decision making process are listed in Table 6.

It can be noticed that the solution returned by the SPEA-2 is dominated by the solutions returned by both NSGA-II and CDMOMA. The performance of the NSGA-II and CDMOMA solutions are, on the contrary, rather similar; both the algorithms seem to have high performance for this problem.

For the sake of clarity, a zoom detail of the speed response which graphically highlights the difference in performance is shown in Fig. 27, 28 and 29 for the NSGA-II, the CDMOMA and the SPEA-2 respectively.

5 Conclusion

This chapter proposes the Cross-Dominant Multi-Objective Memetic Algorithm (CDMOMA), which is a memetic algorithm composed of the NSGA-II as an evolutionary framework and two local searchers adaptively integrated within the framework. The adaptation is based on a criterion which attempts to coordinate the local search by monitoring improvements in the set of non-dominated solutions. Novel contributions of this chapter are: the implementation proposed here for the Multi-Objective Rosenbrock Algorithm, the concept of Cross-Dominance and its employment within a Memetic Framework, and the probabilistic scheme based on the Wigner semicircle distribution.

The CDMOMA seems very promising in several test cases by either reaching the theoretical Pareto or outperforming the popular NSGA-II and SPEA-2. In only one test case (ZDT4) out of eight test problems the CDMOMA failed in detecting a good set of solutions. Numerical results related to the real-world problem analyzed here seem also to conclude that the CDMOMA can be a promising approach.

A further improvement in the proposed approach will be in the detection of tailored local search components for some specific applications, in the design of efficient local searchers which take into account the spreading property of the locally improved solution with respect to other individuals of the population and, finally, to propose a modification of the evolutionary framework in order to enhance its robustness over multiple runs.

Acknowledgement

The Authors wish to express their deep gratitude to Anna V. Kononova for the helpful discussions and precious suggestions.

References

1. Miettinen, K.: Nonlinear Multiobjective Optimization. Springer, Berlin (1998)
2. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: Proc. of 1st International Conference on Genetic Algorithms and Their Applications, pp. 93–100 (1985)
3. Deb, K.: Multi-objective Optimization using Evolutionary Algorithms, pp. 147–149. John Wiley and Sons LTD, Chichester (2001)
4. Coello, C.A.C., Veldhuizen, D.A.V., Lamont, G.B.: Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers, New York (2002)
5. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolutionary Computation 6(2), 182–197 (2002)
6. Zitzler, E., Laumanns, M., Thiele, L.: SPEA-2: Improving the Strength Pareto Evolutionary Algorithm. Tech. Rep. 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland (2001)
7. Moscato, P.: On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Tech. Rep. 790, Caltech Concurrent Computation Program (1989)
8. Neri, F., Toivanen, J., Mäkinen, R.A.E.: An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV. In: Applied Intelligence, vol. 27, pp. 219–235. Springer, Heidelberg (2007) (special issue on Computational Intelligence in Medicine and Biology)
9. Neri, F., Toivanen, J., Cascella, G.L., Ong, Y.S.: An adaptive multimeme algorithm for designing HIV multidrug therapies. IEEE/ACM Transactions on Computational Biology and Bioinformatics, Special Issue on Computational Intelligence Approaches in Computational Biology and Bioinformatics 4(2), 264–278 (2007)
10. Krasnogor, N.: Toward robust memetic algorithms. In: Hart, W.E., Krasnogor, N., Smith, J.E. (eds.) Recent Advances in Memetic Algorithms, pp. 185–207. Springer, Berlin (2004)
11. Ong, Y.S., Keane, A.J.: Meta-lamarkian learning in memetic algorithms. IEEE Transactions on Evolutionary Computation 8(2), 99–110 (2004)
12. Ong, Y.S., Lim, M.H., Zhu, N., Wong, K.W.: Classification of adaptive memetic algorithms: A comparative study. IEEE Transactions On Systems, Man and Cybernetics - Part B 36(1), 141–152 (2006)
13. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: model, taxonomy, and design issues. IEEE Transactions on Evolutionary Computation 9, 474–488 (2005)
14. Knowles, J., Corne, D.: Memetic algorithms for multiobjective optimization: issues, methods and prospects. In: Hart, W.E., Krasnogor, N., Smith, J.E. (eds.) Recent Advances in Memetic Algorithms, pp. 313–352. Springer, Heidelberg (2004)
15. Ishibuchi, H., Yoshida, T., Murata, T.: Balance between genetic search and local search in hybrid evolutionary multi-criterion optimization algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, pp. 1301–1308 (July 2002)

16. Ishibuchi, H., Yoshida, T., Murata, T.: Balance between genetic search and local search in memetic algorithms for multiobjective permutation flow shop scheduling. IEEE Transactions on Evolutionary Computation 7, 204–223 (2003)
17. Ishibuchi, H., Hitotsuyanagi, Y., Nojima, Y.: An empirical study on the specification of the local search application probability in multiobjective memetic algorithms. In: Proc. of the IEEE Congress on Evolutionary Computation, pp. 2788–2795 (September 2007)
18. Srinivas, N., Deb, K.: Multi-objective function optimization using non-dominated sorting genetic algorithms. Evolutionary Computation 2, 221–248 (1995)
19. Deb, K., Goyal, M.: A combined genetic adaptive search (GeneAS) for engineering design
20. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex SYstems 9(2), 115–148 (1995)
21. Deb, K., Kumar, A.: Real-coded genetic algorithms with simulated binary crossover: Studies on multi-modal and multi-objective problems. Complex SYstems 9(6), 431–454 (1995)
22. Suman, B.: Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem. Computers and Chemical Engineering 28
23. Rosenbrock, H.H.: An automatic method for findong the greatest or least value of a function. The Computer Journal 3(3), 175–184 (1960)
24. Birkho, G., Lane, S.M.: A Survey of Modern Algebra. Macmillan, Basingstoke (1953)
25. Szu, H., Hartley, R.: Fast simulated annealing. Physiscs Letters A 122, 157–162 (1987)
26. Fonseca, C.M., Fleming, P.J.: Multiobjective optimization and multiple constraint handling with evolutionary algorithms part-ii aplication example. IEEE Transctions on Systems,Man and Cyberntics, PartA, Systems and Humans, 38–47 (1998)
27. Poloni, C.: Hybrid GA for multi-objective aerodynamic shape optimization. In: Winter, G., Periaux, J., Galan, M., Cuesta, P. (eds.) Gentic algorithms in egineering and computer science, pp. 397–414. Wiley, Chichester (1997)
28. Kursawe, F.: A variant of evolution strategies for vector optimization. In: Schwefel, H.-P., Mäner, R. (eds.) Parallel Problem Solving from Nature, vol. I. Springer, Berlin
29. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms:empirical results. Evolutionary Computaion Journal 8, 173–195 (2000)
30. Multi-objective genetic algorithms:problem difficulties and construction of test functions. Evolutionary Computaion Journal 7, 205–230 (1999)
31. Deb, K., Agrawal, S., Pratab, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M., et al. (eds.) Proceedings of the Parallel Problem Solving from Nature VI Conference. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000)
32. Åström, K.J., Wittenmark, B.: Adaptive Control, 2nd edn. Prentice Hall, Englewood Cliffs (1994)
33. Slotine, J.-J., Li, W.: Applied Nonlinear Control. Prentice Hall, Englewood Cliffs (1990)
34. Khorrami, H.M.F., Krishnamurthy, P.: Modeling and Adaptive Nonlinear Control of Electric Motors. Springer, Heidelberg (2003)
35. Jain, L.C., de Silva, C.W.: Intelligent Adap.tive Control: Industrial Applications. CRC, Boca Raton (1998)
36. Åström, K.J., Hägglund, T.: The future of PID control. Control Engineering Practice 9(13), 1163–1175 (2001)
37. Panagopoulos, H., Åström, K.J., Hägglund, T.: Design of PID controllers based on constrained optimisation. IEE Proceedings - Control Theory & Applications 149, 32–40 (2002)
38. Dury, W.: Control Techniques Drives & Controls Handbook. Institution Electrical Engineers (2001)
39. Leonhard, W.: Control of Electrical Drives, 2nd edn. Springer, Heidelberg (2001)
40. Krishnan, R.: Electronic Motor Drives: Modeling, Analysis and Control. Prentice-Hall, Englewood Cliffs (2001)

41. Zanchetta, P., Sumner, M., Cupertino, F., Marinelli, M., Mininno, E.: On-line and off-line control design in power electronics and drives using genetic algorithms. In: Proceedings of the IEEE Industry Applications Conference, pp. 864–871 (2004)
42. Cascella, G.L., Salvatore, N., Sumner, M., Salvatore, L.: On-line simplex-genetic algorithm for self-commissioning of electric drives. In: Proceedings 11th EPE, pp. 277–283 (2005)
43. Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for on-line and off-line control design of pmsm drives. IEEE Transactions on System Man and Cybernetics-part B, special issue on Memetic Algorithms 37(1), 28–41 (2007)
44. Szklarski, K.J.L., Horodecki, A.: Electric Drive Systems Dynamics. Elsevier Science Ltd., Amsterdam (1990)
45. del Toro Garcia, X., Neri, F., Cascella, G.L., Salvatore, N.: A surrogate assisted hooke-jeeves algorithm to optimize the control system of a pmsm drive. In: Proceedings of IEEE International Symposium on Industrial Electronics, vol. 1, pp. 347–352 (2006)