OBTAINING A RESTRICTED PARETO FRONT IN EVOLUTIONARY MULTIOBJECTIVE OPTIMIZATION

Carlos C.H. BORGES and Helio J.C. BARBOSA[1]

Abstract. Novel strategies are proposed to deal with multiobjective optimization problems solved via evolutionary computation. The goal of this work is to introduce procedures that allow for the search over a restricted Pareto front where all solutions attain acceptable objective functions values in a way that a minimum of prior knowledge about the objective functions is required.

1 Introduction

The treatment of multiobjective optimization problems (MOPs) is evolving rapidly in recent years specially by means of Evolutionary Computation (EC) which has been shown to be adequate and robust to solve this kind of problems. In contrast with Mathematical Programming (MP) techniques, EC can solve the multicriteria problem obtaining a large number of elements belonging to the Pareto front in a single run. MOPs are generally solved in procedures using the decision maker (DM) knowledge, before, during and after the search takes place in order to reach the most adequate points of the Pareto front. All MP techniques, such as the Weight Method, Goal Programming and Min-Max formulation, are DM dependent in the sense that the parameters chosen by the DM lead to one element of the Pareto set that represents the preferences of the DM. However, if the parameters are not properly chosen, the solution will not correspond to the DM expectations. The non-commensurable objectives in a MOP and the lack of prior information makes this task very difficult in real world problems.

A MOP solved by means of EC has the advantage of being less sensitive to prior DM knowledge. But this does not guarantee that good results will indeed be

[1]Laboratório Nacional de Computação Científica, MCT, Rua Getúlio Vargas 333, 25651 070, Petrópolis, RJ, Brasil, e-mail: {cchb,hcbm}@lncc.br

obtained. The number of Pareto elements obtained in an EC method is limited by the size of the population used. In complex real world problems with several objectives the lack of information about the Pareto front geometry and its extension can produce a poor or non-representative group of solutions. On the other hand, the population size required in order to obtain a good representation of the Pareto front can be prohibitive for the computational resources available for the analysis.

A hybrid method with MP and EC characteristics is developed with the introduction of thresholds, based on the DM aspirations, so that the search will be performed in a restricted domain of the objective function space.

Furthermore, a procedure is developed to obtain limit points for each objective, which serve as a basis for the definition of the thresholds, at the same time that the points of the Pareto set are being searched.

The performance of the proposed procedure is examined by means of numerical examples.

2 Multiobjective Optimization and Genetic Algorithms

In an unconstrained MOP one seeks to optimize the m components of a vector of objective functions $F(x) = (f_1(x), f_2(x), \ldots, f_m(x))$ where $x \in \Omega \subset R^n$ is the vector of design (or decision) variables which belong to the admissible set Ω. The function $F : \Omega \mapsto \Lambda$ maps solution (or design) vectors $x = (x_1, x_2, \ldots, x_n)$ to vectors $y = (y_1, y_2, \ldots, y_m)$, with $y_i = f_i(x)$, in the objective function space.

Here, the optimization problem associated with each objective will be considered to be a minimization one, without loss of generality. As the objectives are in general conflicting there is usually not a solution which minimizes all m objectives simultaneously. This motivates the concept of *dominance*: a vector $a = (a_1, a_2, \ldots, a_n)$ dominates a vector $b = (b_1, b_2, \ldots, b_n)$ if $a_i \leq b_i \, \forall i$ and there is j such that $a_j < b_j$. One then defines Pareto optimality: a solution $x \in D$ is said to be Pareto-optimal in D if and only if there is no $x' \in D$ such that its image $F(x')$ dominates $F(x)$. The Pareto-optimal set, $\mathcal{P}$, is thus the set of all $x \in D$ such that $F(x)$ is non-dominated in Λ. The Pareto front $\mathcal{P}_F$ is the image of the Pareto-optimal set in the objective function space.

Genetic algorithms (GAs) have features that make them attractive for the approximation of the Pareto set: they are population-based, require only objective function evaluations, use probabilistic transition rules which make then less prone to local optimum entrapment and allow for several types of parallel implementations.

The first application of GAs to MOPs dates back to the mid-eighties[7, 8] and numerous papers on the application of evolutionary techniques to MOPs have been published since then (see [4, 5, 11, 2]).

Although other techniques have been used for finding a suitable solution to MOPs, we are interested here in those that try to approximate the Pareto set.

The most popular techniques seem to be MOGA, due to Fonseca and Fleming[3], NPGA, due to Horn and Nafpliotis[6] and NSGA, due to Srinivas and Deb[9].

Almost all the work with GAs for MOPs involved algorithms of the so called generational type, where a large portion of the current population is replaced by newly generated individuals. Valenzuela-Rendon and Uresti-Charre[10] proposed a non-generational scheme based on the similarities observed between the tasks faced by a GA when dealing with a learning classifier system (LCS) using the Michigan approach and when dealing with a MOP. In [1] Borges and Barbosa proposed another non-generational GA for MOPs that alleviates the task of parameter setting, and obtain the Pareto front with a good distribution by the conservation of a population with the best points reached in terms of domination and distribution. Usually, in GAs the fitness function is to be maximized, i.e., the higher the fitness value, the better the candidate solution is. However, in [1], the fitness function was defined in such a way that lower values indicated better solutions. As this function is used here again to construct a new fitness function, the terminology fitness function is preserved, although one seeks to minimize it.

In [1] the fitness function (to be minimized) is defined as the non-linear expression

$$\mathcal{F}(x^i) = (1 + \mathcal{D}(x^i))^\beta (1 + \mathcal{N}(x^i)) \tag{1}$$

where $\mathcal{D}(x^i)$ is the domination measure which expresses the state of domination of the i-th individual, x^i, with respect to the current population:

$$\mathcal{D}(x^i) = \sum_{j=1}^{p} \mathrm{nd}(x^i, x^j)$$

where p is the number of individuals in the population and $\mathrm{nd}(x^i, x^j)$ is one if element x^j dominates element x^i, and zero otherwise. The parameter β is usually set to one and $\mathcal{N}(x^i)$ is a neighbor density measure associated with element x^i given by

$$\mathcal{N}(x^i) = \sum_{j=1}^{p} \mathrm{sh}(d_{ij})$$

which is calculated according to the sharing function

$$\mathrm{sh}(d_{ij}) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma_s})^\alpha, & \text{if } d_{ij} < \sigma_s; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

where d_{ij} is the distance between elements x^i and x^j with σ_s indicating the size of the neighborhood of each element. The parameter α is generally set to one.

A standard linear rank-based selection scheme was adopted in [1] and it is preserved here. It provides a more uniform selection pressure along the evolutionary process than a fitness-based selection scheme.

3 Restricting the search

In the Weight Method the weights are used to determine the relative importance of each function in the process and balance the different magnitudes of the objectives.

In Goal Programming the DM chooses a value for each objective function that seems to be adequate and try to find a point of the Pareto front which is closest to these goals.

In this paper the goal is to use the entire GA population to search for a particular region of the Pareto front – which is specified by the user by means of adequate thresholds – instead of searching for the whole front since it contains regions where some very good values of a given objective(s) are obtained at the expense of some other(s) which attain unacceptably poor value(s).

Multiobjective evolutionary algorithms (MOEAs) which are based only in some form of dominance and density of elements for the fitness definition (such as equation (1)) have no natural way of incorporating preferences of objectives defined by the DM. This drawback will be circumvented here by the introduction of an additional function $\mathcal{L}(x)$ – which can be interpreted as a penalty applied to element x – to be combined with $\mathcal{F}(x)$ in order to define a new fitness function in the form:

$$\mathcal{F}_{\mathcal{L}}(x) = g(\mathcal{F}(x), \mathcal{L}(x)). \tag{3}$$

It will be assumed that the DM defines the constant vector $F^l = (f_1^l, f_2^l, \ldots, f_m^l)$ containing the limit values for each of the objective functions. Given the vector F^l, it is clear that the Pareto front can be divided into two sub-regions: an internal region, comprising points within the bounds, and an external region, with the remaining points/solutions.

When one is interested in the internal region, the value of each element of F^l represents the maximum (resp. minimum) acceptable value for the corresponding objective to be minimized (resp. maximized). The idea is to obtain a final population with solutions that attain the level of quality specified by the vector F^l i.e. $f_i(x^j)$ is not greater (resp. smaller) than f_i^l for all objectives $i = 1, \ldots, m$ and all solutions x^j in the population. The internal region contains solutions which attain a good compromise among the objectives. The remaining points in the Pareto front, constitute the external region.

The function $\mathcal{L}(x)$ can then be defined as zero if

$$f_k(x) \succ f_k^l \quad \forall\ k = 1, \ldots, m \tag{4}$$

where $\succ$ denotes $<$ (resp. $>$) for minimization (resp. maximization). Otherwise one has

$$\mathcal{L}(x) = \sqrt{\sum_{\substack{k=1,m \\ f_k(x) \not\succ f_k^l}} \left(1 + \frac{(f_k(x) - f_k^l)^2}{(f_k^M - f_k^m)^2}\right)} \tag{5}$$

where it is understood that the summation is taken over all the objective function values that do not obey the corresponding bound, thus introducing a penalty for that candidate solution x. In (5) f_k^M and f_k^m are respectively the best and worst values of the k-th objective function for nondominated solutions in the current population. The constant one guarantees that the result is always greater than one.

In the new fitness definition, the value of $\mathcal{L}(x)$ given by equation (5) and the value of the standard fitness (corresponding to the unconstrained search along the Pareto front) such as (1) are nonlinearly combined:

$$\mathcal{F}_{\mathcal{L}}(x) = \mathcal{F}(x)(1 + \mathcal{F}^M \mathcal{L}(x)) \tag{6}$$

where $\mathcal{F}^M$ is the best fitness (as measured by eq. (1)) in the current population. By construction, the new fitness definition (equation (6)) ensures that when performing an internal (resp. external) search, any interior (resp. exterior) point/solution has a better fitness value than any exterior (resp. interior) solution.

4 Restricting the search – without prior thresholds

The specification of the limit acceptable values that must be made by the DM requires some prior knowledge of the behavior of each objective. The solutions obtained will lay inside this restricted region but to assure that this is indeed a good region of the search space is not a trivial task. An improvement of the method is proposed in order to avoid the prior specification of those limits. The idea is to simultaneously: (i) obtain solutions in the desired region and (ii) construct the vectors F^M and F^m which contain in the k-th component, respectively, the greatest and the smallest values of the k-th objective function in the current population, thus defining the current extreme points of the Pareto front. Those vectors are used to automatically define a range of variation for each objective function. The definition of f_k^l is now given by

$$f_k^l = f_k^M - p_k(f_k^M - f_k^m), \quad k = 1, \ldots, m \tag{7}$$

where $p_k \in (0, 1)$. Larger values for p_k indicate a smaller range for the k-th objective function (see Figure 2(a)). The vectors F^M and F^m are updated every time a better value appears during the evolution. Also it should be noted that the k-th component of the vectors F^M and F^m are f_k^M and f_k^m respectively and that they are obtained from nondominated solutions. This prevents objective function values from dominated solutions far from the current non-dominated set from being used instead.

With the definition (7) the DM only needs to specify a reduction fraction ($p_k \in (0, 1)$) for each objective function range, with no need to know the actual range of each objective function.

5 The implementation

The success of the method depends on reasonable values for the vector $F^l \in R^m$. The definition proposed here (7) depends on each pair of extreme values f_k^M and f_k^m. The major difficulty is when the search is not focused on this extreme regions of the Pareto front and good values may not be obtained. One possible solution to improve the search in those extreme regions is to initially evolve without the term $\mathcal{L}(x)$ in the fitness function. Alternatively, it can be useful to determine good values for F^M and F^m by initializing the search in the external region and then moving to the internal region during the evolution.

A pseudo-code of the restricted search procedure reads:

```
Begin
   Set variables:  internal (true or false), p_k, k = 1,...,m
   repeat
      Compute F(x^i) i = 1,...,p (population size)
      Update F^M, F^m and F^l according to the non-dominated elements
      for i=1,p
         if( f_k(x^i) ≠ f_k^M and f_k(x^i) ≠ f_k^m, ∀k ∈ {1,...,m} ) then
            if( internal ) then
               for k=1,m
                  if( f_k(x^i) ⊁ f_k^l ) Add contribution to L(x^i)
               endfor
            else
               if( f_k(x^i) ≻ f_k^l, ∀k ∈ {1,...,m} ) Compute L(x^i)
            endif
            F_L(x^i) = F(x^i) (1 + F^M L(x^i))
         endif
      endfor
      Generate offsprings ^o x^j j = 1,...,n_o (number of offsprings)
      Compute F(^o x^j), j = 1,...,n_o
      for j=1, n_o
         if( f_k(^o x^j) ⊁ f_k^M and f_k(^o x^j) ⊁ f_k^m, ∀k ∈ {1,...,m} ) Compute F_L(^o x^j)
      endfor
      Update the population (preserving solutions that generated F^M and F^m)
   until(termination criteria satisfied)
   Stop
End
```

This algorithm can be adapted for generational as well as non-generational reproductive schemes. In the generational scheme the elements that form the vector F^M and F^m should be preserved for the next generation. In a non-generational procedure these points are automatically preserved because: (i) the function $\mathcal{L}(x)$ is set to zero and (ii) in the internal case the density of elements for these points is low, leading to good values for $\mathcal{F}(x)$, while in the external case they have a low neighborhood density value since they are at the boundary of the Pareto front.

6 Numerical experiments

In order to investigate the feasibility and performance of the approach proposed here the corresponding Fortran code was implemented and some numerical experiments were performed. The results shown correspond to typical runs; similar results were obtained using other values for the seed supplied to the pseudo-random number generator. For a given test-problem, the results shown for the proposed algorithm correspond to the same seed so that comparisons can be made.

The procedure for the restricted search was implemented within the non-generational code described in [1].

6.1 Test-problem 1

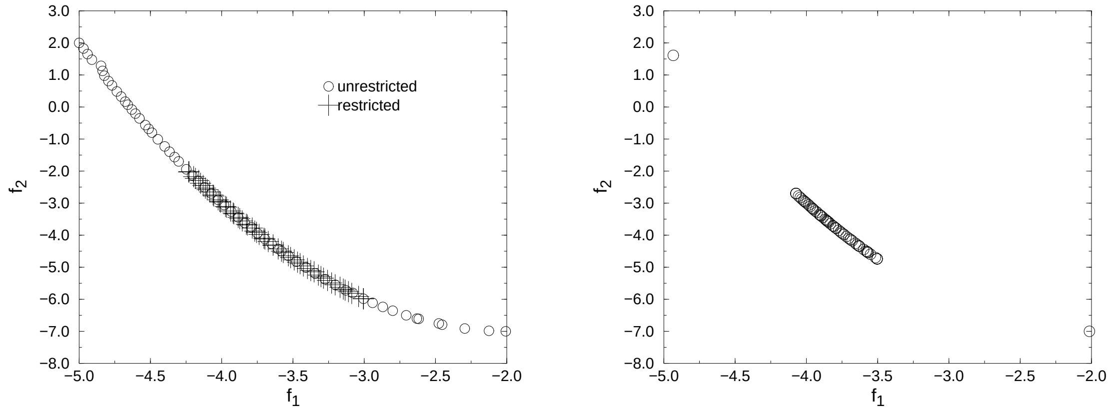
The test-problem corresponds to the most difficult one among those solved in [10]. The functions f_1 and f_2 to be minimized are given by

$$f_1 = x + y + 1 \qquad f_2 = x^2 + 2y - 1$$

In this problem f_1 and f_2 are functions of two variables $x \in [-3, 3]$ and $y \in [-3, 3]$. The parameters used in the algorithm are the same used in [10], with a chromosome length of 20 bits, two-point crossover with 0.85 probability, mutation rate equal to 0.033, a population of 60 individuals with 2000 function evaluations allowed. The value of σ_s is set to 0.2. The results are obtained using the neighborhood density computed in the design variable space.

Figure 1(a) shows the results for two cases: the unconstrained response, that is the traditional approach used in MOEAs with the elements of the population distributed all over the Pareto front, and a case with a DM restriction for each function. In the restricted case it is assumed that $f_1^l = -3.0$ and $f_2^l = -2.0$ and that the value of f_1 and f_2 should be less than those bounds. The results correspond to the expectations of the DM, with a good representation of the Pareto front in the specified region. In Figure 1(b) the strategy with no prior thresholds is introduced with the internal search active and the values of $p_1 = p_2 = 0.5$. According to the range of each objective function the result should be comprised approximately for $f_1 \leq -3.5$ and $f_2 \leq -2.7$ as the graphic shows. The extreme points of the Pareto front obtained for each direction are very similar to those obtained for the unconstrained case with no additional effort, showing that the method is adequate for this kind of search.

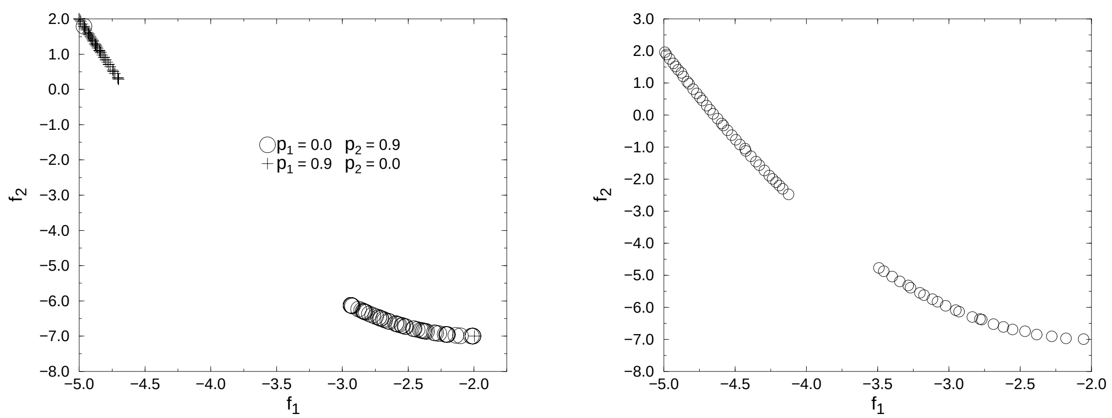
In Figure 2(a) the goal is to show extreme cases for the p_k variable setting: for one objective the value is 0.9 and for the other it is set to zero. The results obtained were as expected: only the extreme of the Pareto front for each case. The use of 1.0 and 0.0 for the p_k variable reduces the response to only the two extreme points of the Pareto front. Finally, in Figure 2(b) the internal search is not active and one sets $p_1 = p_2 = 0.5$ in order to obtain the extremes of the Pareto front. This is a good strategy to be combined with the internal search for reaching good values for the extreme points of the Pareto front. The initial evaluations can be made with the internal search not active to force a better determination of the extreme values. Then the internal search can be set to active with good extreme values guaranteed.



(a) Unconstrained and restriction case

(b) Bounds-independent search - *internal* set to true

Figure 1: Results for example 1



(a) Extreme cases for p_k

(b) Bounds-independent search - *internal* set to false

Figure 2: Results for example 1

6.2 Test-problem 2

The second test-problem is a case of optimization with 3 objective functions. The problem consists in minimizing the distance of a point in the plane to 3 given straight lines, $\overline{P_1P_2}$, $\overline{P_2P_3}$ and $\overline{P_3P_1}$ where the points have coordinates $P_1(1,6)$, $P_2(5,9)$ and $P_3(9,2)$.

The Pareto front corresponds to the interior and the boundary of a triangle in the three-dimensional objective space. The problem can also be analyzed in the design variable space, where the Pareto set is the interior and the boundary of the triangle formed by the 3 points in the x-y space. Here a population of 100 elements is used evolving during 30000 evaluations. A chromosome length of 40 bits is used with the variables x and y in the range $[-5, 15]$, two-point crossover with 0.85 probability and mutation rate equal to 0.01. For the σ_s parameter 3 values were used. When the variable *internal* is undefined $\sigma_s = 0.7$, when *internal* is set to false $\sigma_s = 0.5$ and when *internal* is set to true $\sigma_s = 0.3$. This is to make the value of σ_s more adequate to the size of the region searched. The results correspond to a typical run of the genetic algorithm.

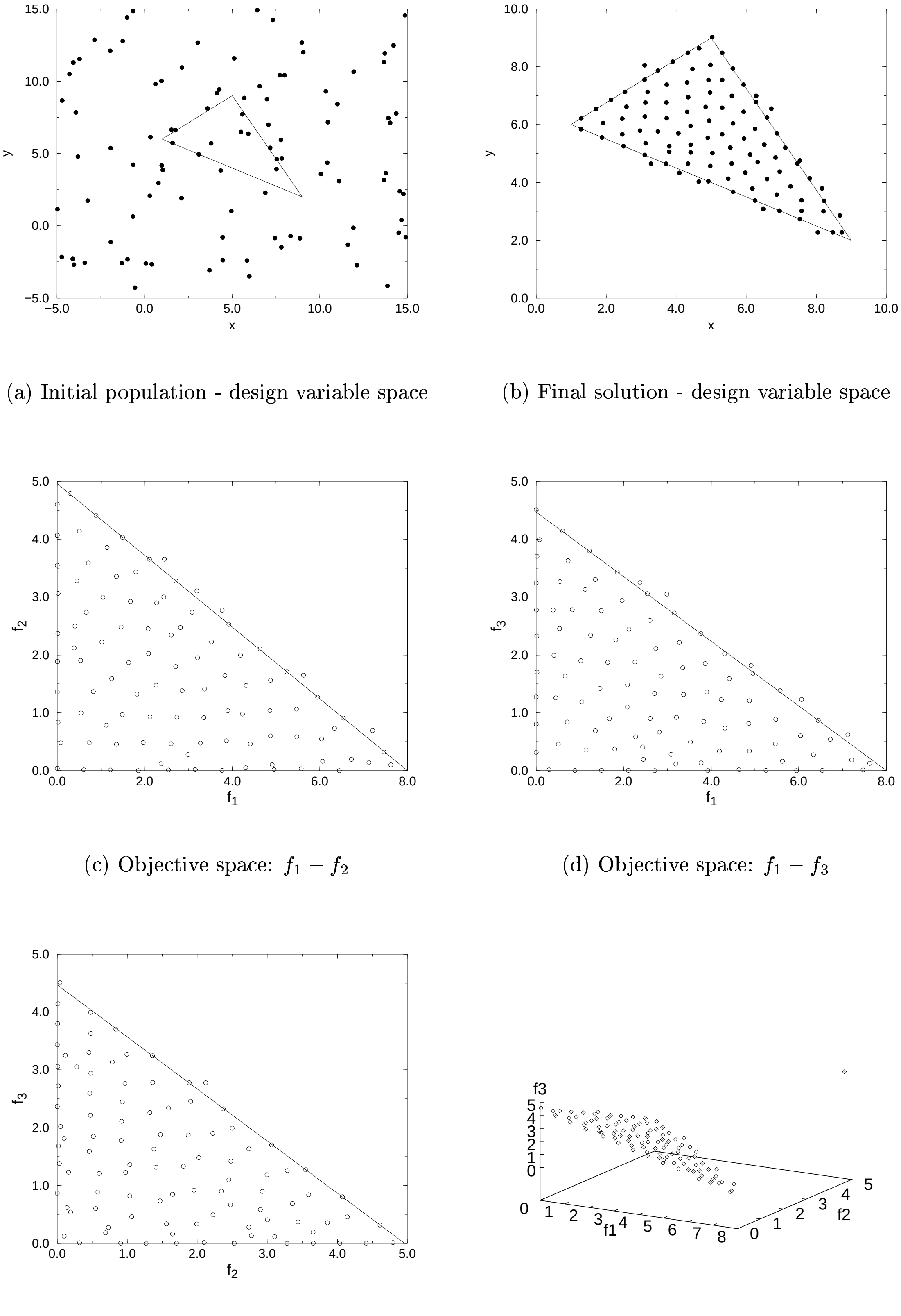
Figure 3(a) represents the initial population in an unconstrained case in the design variable space. The Figure 3(b) shows the final response in the unconstrained problem in the design variable space. Figures 3(c), 3(d) and 3(e) are the projections of the solutions obtained for the Pareto front over $f_1 - f_2$, $f_1 - f_3$ and $f_2 - f_3$ planes respectively. The solid lines in the figures represent the Pareto set and Pareto front boundaries for the design variables in Figure 3(a), 3(b) and for the objective functions values in Figures 3(c), 3(d) and 3(e). In Figure 3(f) the three-dimensional graphic of the Pareto front is plotted.

Now we assume a restriction value for each objective. The values correspond to half the maximum possible distance in the Pareto front for each line. So $f_1 = 4.000$, $f_2 = 2.480$ and $f_3 = 2.236$. Figure 4(a) shows the results when the variable *internal* is set to true in the design variable space. This means that the admissible region has values less or equal than the limits adopted, that is $f_1 \leq 4.000$, $f_2 \leq 2.480$ and $f_3 \leq 2.236$. The restricted Pareto region is indicated by the area of the solid line triangle. Figures 4(b), 4(c) and 4(d) are the projections in the planes of objective functions. The same is shown in Figures 5(a), 5(b), 5(c) and 5(d) when the variable *internal* is set to false. The Pareto front is defined by the region outside of the solid-line triangle and inside the dashed-line triangle.

Figures 6(a) and 6(b) are the tri-dimensional objective functions graphic for *internal* set to true and *internal* set to false respectively.

The prior bounds independent search is applied with $p_1 = p_2 = p_3 = 0.5$ for *internal* set to true and the results can be seen in Figures 7(a), 7(b), 7(c) and 7(d). The Pareto region should be the same of the previous case (prior bounds are given). The analysis of the results shows that the extreme values are not well determined causing a translation in the Pareto region obtained. In Figures 8(a), 8(b), 8(c) and 8(d) the results are obtained with the variable *internal* set to false.

In the Figures 9(a), 9(b), 9(c) and 9(d) another strategy is used to define the bounds in the case where *internal* is set to true. Now, in the initial evaluations (one third of the total), the variable *internal* is set to false. Then the value of *internal* is inverted to true and the remaining of the evolutionary process is dedicated to the search in the interior of the triangles but using better extreme values. Comparing Figures 7 with Figures 9 it is clear that this strategy produced better results.



(a) Initial population - design variable space

(b) Final solution - design variable space

(c) Objective space: $f_1 - f_2$

(d) Objective space: $f_1 - f_3$

(e) Objective space: $f_2 - f_3$

(f) Pareto front

Figure 3: Unconstrained case

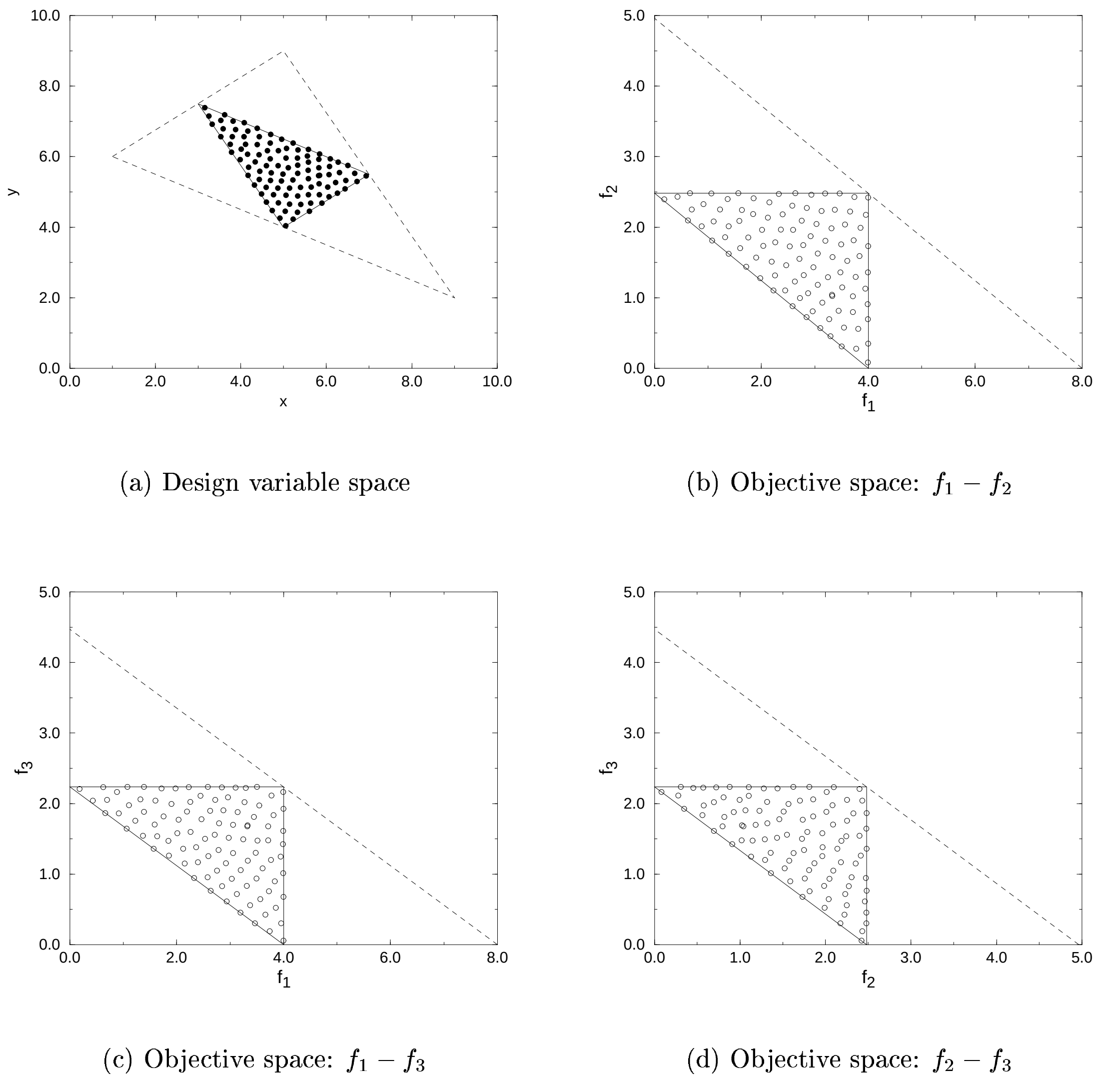
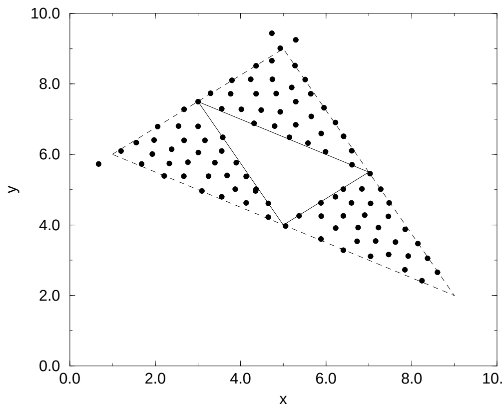
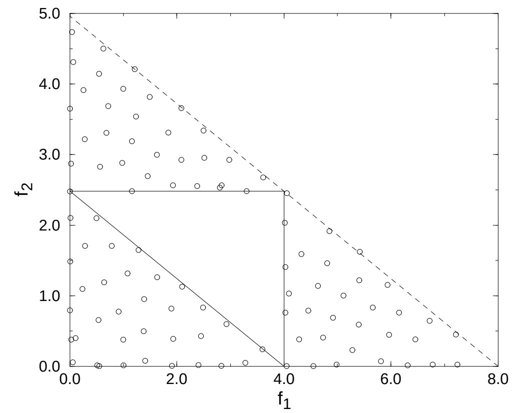


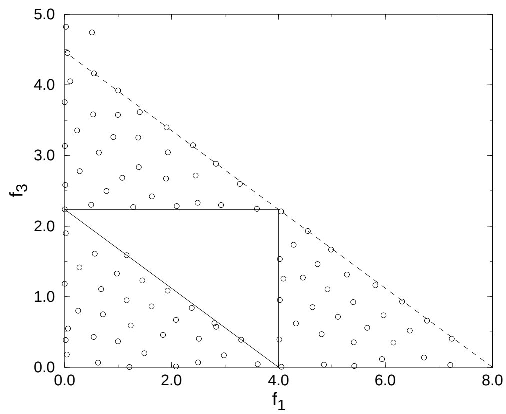
Figure 4: Restriction case - *internal* set to true



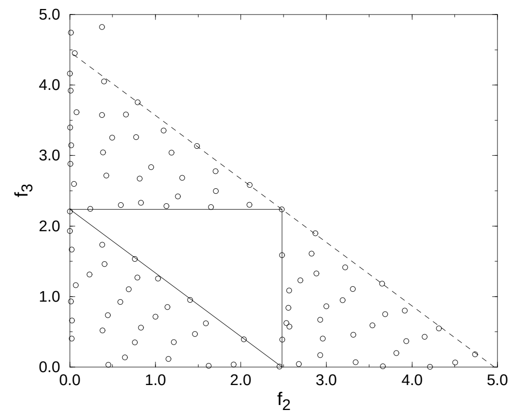
(a) Design variable space



(b) Objective space: $f_1 - f_2$

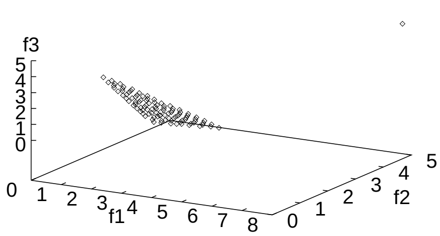


(c) Objective space: $f_1 - f_3$

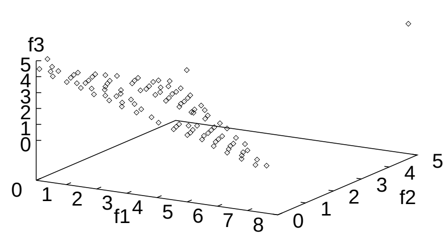


(d) Objective space: $f_2 - f_3$

Figure 5: Restriction case - *internal* set to false

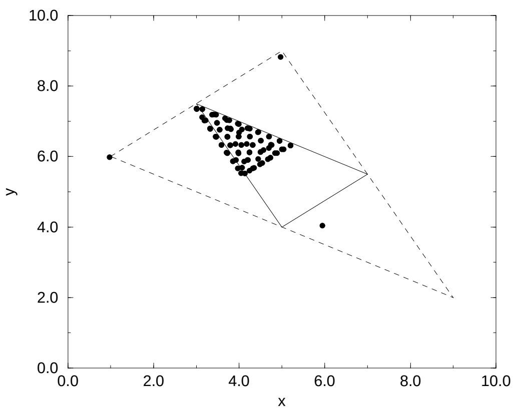


(a) Variable *internal* set to true

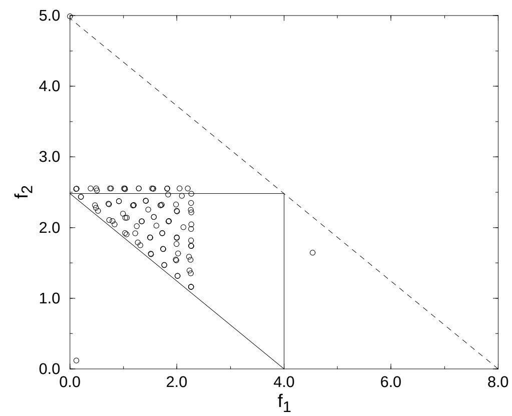


(b) Variable *internal* set to false

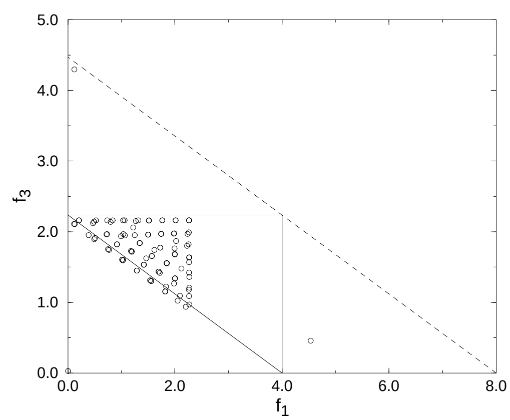
Figure 6: Pareto front



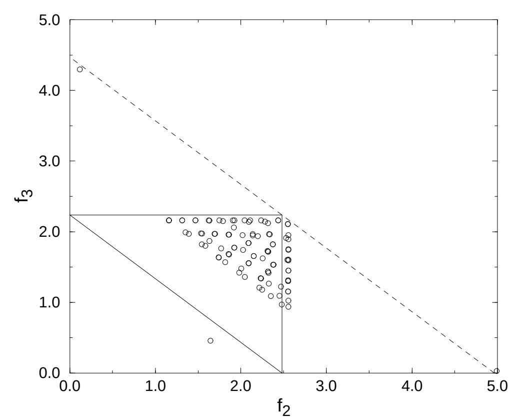
(a) Design variable space



(b) Objective space: $f_1 - f_2$

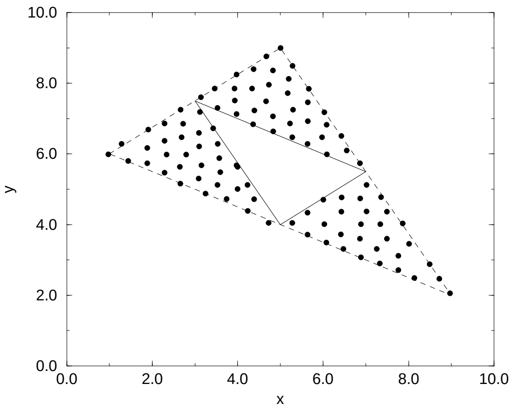


(c) Objective space: $f_1 - f_3$

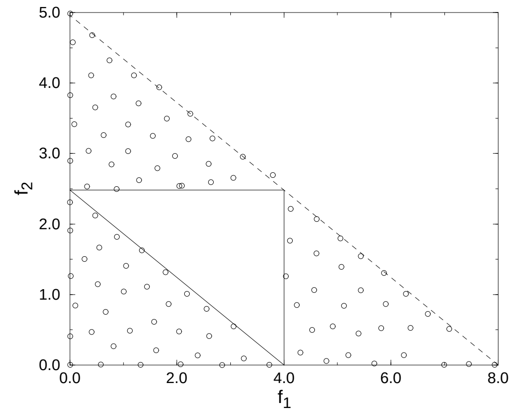


(d) Objective space: $f_2 - f_3$

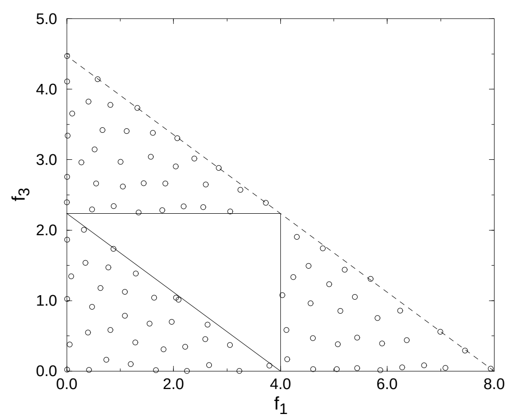
Figure 7: Bounds independent search - *internal* set to true



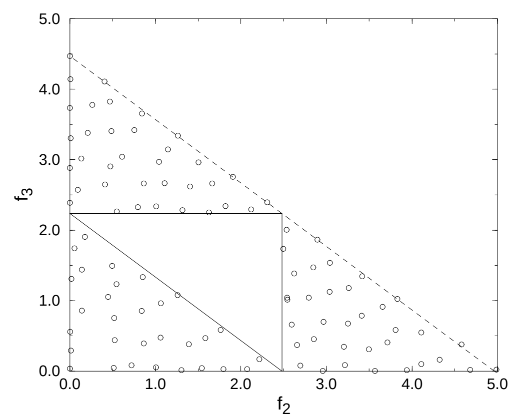
(a) Design variable space



(b) Objective space: $f_1 - f_2$

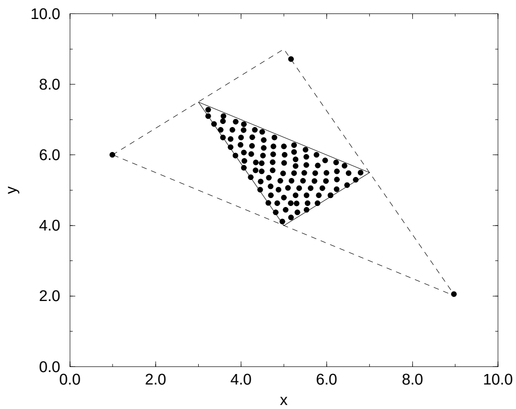


(c) Objective space: $f_1 - f_3$

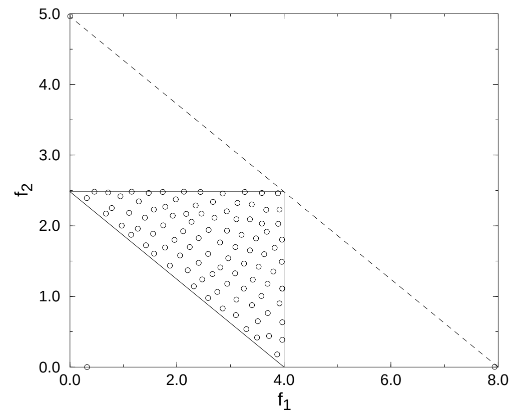


(d) Objective space: $f_2 - f_3$

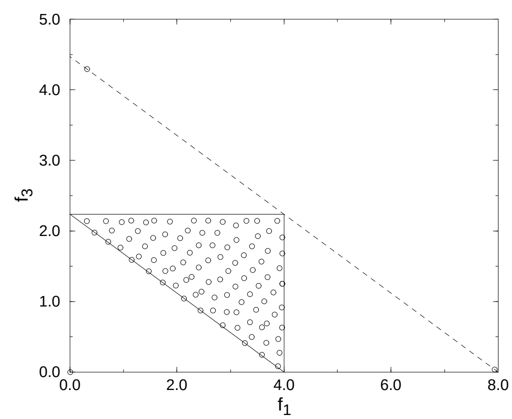
Figure 8: Bounds independent search - *internal* set to false



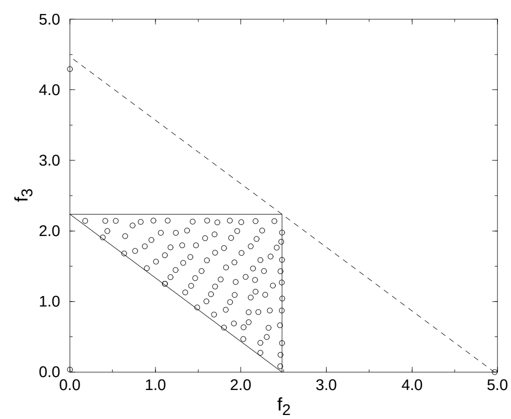
(a) Design variable space



(b) Objective space: $f_1 - f_2$



(c) Objective space: $f_1 - f_3$



(d) Objective space: $f_2 - f_3$

Figure 9: A different strategy for bounds independent search with *internal* set to true

7 Conclusions

In this work a technique for guiding the search and consequently the response in terms of the Pareto front obtained for a multiobjective problem solved via an evolutionary algorithm is proposed. The method uses the whole population to search in a restricted region of the objective function space where all objective functions attain acceptable values. This is an important step to encourage users to model real world problems as true MOPs since conveniently located solutions in the objective function space can be obtained in a robust way. Numerical experiments performed so far are promising. Tests in more complex situations with several objectives and real world problems constitute the next steps.

Additional ideas can be developed such as dividing among parallel processors the task of searching in different regions of the objective space, with the final response being the union of all sets found.

Acknowledgments. The authors would like to thank the referee for his/hers comments which lead to significant improvements in the text.

References

[1] C.C.H. Borges and H.J.C. Barbosa. A non-generational genetic algorithm for multiobjective optimization. In *2000 Congress on Evolutionary Computation*, pages 172–179, San Diego, CA, USA, July 2000.

[2] C.A.C. Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, 1999.

[3] C.M. Fonseca and P.J. Fleming. Genetic algorithm for multiobjective optimization: formulation, discussion and generalization. In *Proceedings of the 5th Int. Conf. on Genetic Algorithms*, pages 416–423, Sao Mateo, CA, 1993.

[4] C.M. Fonseca and P.J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[5] J. Horn. *Multicriterion Decision Making*, volume 1, pages F1.9:1 – F1.9:15. IOP Publishing Ltd. and Oxford University Press, 1997.

[6] J. Horn and N. Nafpliotis. Multiobjective optimization using the niched Pareto genetic algorithm. Technical Report 93005, University of Illinois, Illinois Genetic Algorithms Lab., 1993.

[7] J.D. Schaffer. *Some Experiments in Machine Learning using Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, Nashville, TN, 1984.

[8] J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st Int. Conf. on Genetic Algorithms*, pages 93–100, 1985.

[9] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1995.

[10] M. Valenzuela-Rendón and E. Uresti-Charre. A non-generational genetic algorithm for multiobjective optimization. In *Proceedings of the 7th Int. Conf. on Genetic Algorithms*, pages 658–665, Sao Mateo, CA, 1997.

[11] D.A. Van Veldhuizen and G.B. Lamont. Multiobjective Evolutionary Algorithm Research: A History and Analysis. Technical Report TR-98-03, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 1998.