Strategies for multiobjective genetic algorithm development: Application to optimal batch plant design in process systems engineering

A. Dietz, C. Azzaro-Pantel *, L. Pibouleau, S. Domenech

Laboratoire de Génie Chimique – UMR 5503 CNRS/INPT/UPS, 5 Rue Paulin Talabot BP1301, 31106 Toulouse Cedex 1, France

Abstract

This work deals with multiobjective optimization problems using Genetic Algorithms (GA). A MultiObjective GA (MOGA) is proposed to solve multiobjective problems combining both continuous and discrete variables. This kind of problem is commonly found in chemical engineering since process design and operability involve structural and decisional choices as well as the determination of operating conditions. In this paper, a design of a basic MOGA which copes successfully with a range of typical chemical engineering optimization problems is considered and the key points of its architecture described in detail. Several performance tests are presented, based on the influence of bit ranging encoding in a chromosome. Four mathematical functions were used as a test bench. The MOGA was able to find the optimal solution for each objective function, as well as an important number of Pareto optimal solutions. Then, the results of two multiobjective case studies in batch plant design and retrofit were presented, showing the flexibility and adaptability of the MOGA to deal with various engineering problems.

Keywords: Multiobjective genetic algorithm; Multiobjective optimization; Batch plant design; Pareto sort procedure

1. Introduction

Multiobjective problems are widely found in chemical engineering, ranging from applications of process design to determination of optimal operating conditions, and involve simultaneous optimization of several incommensurable and often competing objectives. For example, if we refer to the process design, we will normally want to minimize its investment cost or net present value, but at the same time, we will want to minimize its environmental impact, maximize its safety, control and flexibility ...While in single-objective optimization, the optimal solution is usually clearly defined, this does not hold for multiobjective optimization problems. Instead of a single optimum, there is rather a set of alternative trade-offs, generally known as Pareto-optimal

* Corresponding author. Tel.: +33 5 34 61 52 72; fax: +33 5 34 61 52 53.
E-mail address: Catherine.AzzaroPantel@ensiacet.fr (C. Azzaro-Pantel).

solutions. These solutions are optimal in the wider sense that no other solutions in the search space are superior to them when all objectives are considered. Based on the so-called concept of Pareto Dominance, a decision vector a is optimal (i.e., not dominated by any other decision vector) in the sense that it cannot by improved in any objective without causing a degradation in at least one other objective. Such solutions are denoted as Pareto optimal.

Currently, much work has been carried in the area of multiobjective optimization and there are more than 20 mathematical optimization methods to deal with multiple objectives (Collette & Siarry, 2002).

Classical methods for generating the Pareto-optimal set aggregate the objectives into a single, parameterized objective function by analogy to decision making before search.

However, the parameters of this function are not set by the decision-maker, but systematically varied by the optimizer. Several optimization runs with different parameter settings are performed in order to achieve a set of solutions which approximates the Pareto-optimal set (Stefanis, Livingston, & Pistikopoulos, 1997). Basically, this procedure is independent of the underlying optimization algorithm. Some representatives of this class of techniques are the weighting method (Cohon, 1978), the constraint method (Cohon, 1978), goal programming (Steuer, 1986), and the minmax approach (Koski, 1984).

Let us only recall the basic principles of both the weighting sum, i.e., the function to optimize is the weighted sum of the set of objective functions and the ε-constraints method, i.e., one of the functions is chosen as the objective function and the others are added as constraints of the optimization problem, specially used when one of the objective function is more relevant than the others. The main reason for using this kind of techniques is that the traditional optimization tools can be implemented without any supplementary modification. For a full detailed presentation of multiobjective optimization techniques, (see Collette & Siarry, 2002).

An alternative way to treat multiobjective problems is to use Evolutionary Algorithms (EA). This kind of technique stands for a class of stochastic optimization methods that simulate the process of natural evolution (mainly genetic algorithms, evolutionary programming, and evolution strategies Bäck, Hammel, & Schwefel, 1997). All of these approaches operate on a set of candidate solutions. Using strong simplifications, this set is subsequently modified by the two basic principles of evolution: selection and variation. Selection represents the competition for resources among living beings. Some are better than others and more likely to survive and to reproduce their genetic information. In evolutionary algorithms, natural selection is simulated by a stochastic selection process. Each solution is given a chance to reproduce a certain number of times, dependent on their quality. Thereby, quality is assessed by evaluating the individuals and assigning them scalar fitness values. The other principle, variation, imitates natural capability of creating ''new'' living beings by means of recombination and mutation.

Although the underlying principles are simple, these algorithms have proven themselves as a general, robust and powerful search mechanism.

Moreover, EAs seem to be especially suited to multiobjective optimization because they are able to capture multiple Pareto-optimal solutions in a single simulation run and may exploit similarities of solutions by recombination. Some researchers suggest that multiobjective search and optimization might be a problem area where EAs do better than other blind search strategies (Fonseca & Fleming, 1995). Although this statement must be qualified with regard to the ''no free lunch'' theorems for optimization (Wolpert & Macready, 1997), up to now there are few if any alternatives to EA-based multiobjective optimization (Horn, 1997). The numerous applications and the rapidly growing interest in the area of MultiObjective Evolutionary Algorithms (MOEAs) take this fact into account. After the first pioneering studies on evolutionary multiobjective optimization appeared in the mid-1980s (Schaffer, 1985) a few different MOEA implementations were proposed in the years 1991–1994 (Kursawe, 1991; Fonseca & Fleming, 1993; Horn, Nafpliotis, & Goldberg, 1993; Srinivas & Deb, 1994). Later, these approaches (and variations of them) were successfully applied to various multiobjective optimization problems (Ishibuchi & Murata, 1996; Valenzuela-Rend'on & Uresti-Charre, 1997; Fonseca & Fleming, 1998a, 1998b; Parks & Miller, 1998) and, particularly, in the chemical engineering literature. A review on the applications of multiobjective optimization in chemical engineering (Bhaskar, Gupta, & Ray, 2000) suggests that several interesting studies have been reported on the multiobjective optimization of polymerization reactors.

This work lies in this perspective and is devoted to the treatment of optimal design problems involving mixed variables, which is common in the chemical engineering background and which is identified as a difficult

problem in the optimization community. Typically, the continuous variables represent the operating conditions while the integer ones code the structure. The computation of the fitness function is performed as usual in process systems engineering by use of a simulator (Bernal Haro, 1999; Dedieu, 2001; Dedieu, Azzaro-Pantel, Dietz, Pibouleau, & Domenech, 2002, 2003; Dietz, Azzaro-Pantel, Pibouleau, & Domenech, 2004a, Dietz, Azzaro-Pantel, Davin, Pibouleau, & Domenech,2004b, 2004). This remark is all the more important as GA are particularly applicable to unconstrained (or largely feasible) problems as the individual solutions of a population are then directly comparable in terms of their fitness.

In chemical engineering, the combinatorial aspect of the problem is sometimes important: for example, in batch plant design, the structure variables are the equipment sizes and number for each unit operation that generally take discrete values since discrete value ranges are available; as a consequence, an evolutionist algorithm is more suitable on the one hand to solve this kind of problem than the mathematical programming approach of large size problems. On the other hand, the evolutionist algorithms do not need any information about the mathematical properties (derivability, convexity etc.) of the function to optimize that sometimes are too difficult or impossible to establish; in this case, the multiobjective genetic algorithm is developed in order to be coupled with a simulation tool that only gives values for the performance criteria selected. The drawback of this kind of optimization algorithm is that the optimality of the solution is not guaranteed. Different evolutionist algorithms have been presented and used in the dedicated literature: Simulated Annealing (SA) (Kirkpatrick, Gellat, & Vecchi, 1983) Genetic Algorithms (GA) (Fonseca & Fleming, 1995) and Ants Colony (AC) (Colorni, Dorizo, & Manniezzo, 1991). GA was selected here since it has the main advantage over other methods to manipulate a population of individuals. It is therefore tempting to develop a strategy in which the population captures the whole set of compromise solutions in one single optimization run.

Although GA systems follow the same general algorithmic procedure, each GA system is designed and implemented in a slightly different way, using alternative procedures for selection, encoding or crossover. Therefore, no two systems will optimize a model using exactly the same path. Other differences which have a significant effect on the solution process are the various parameter options which are implemented in each solver – for example, some of the key parameters are the selection method, crossover rate, mutation rate, recombination scheme, elitist strategy and population size.

Among other similar general purpose tools available in the literature, the main objective is to embed in the optimization procedure, identical representation techniques for continuous and integer parameters. The major aim is at designing a general optimization framework with identical crossover or mutation techniques, independently on the coded variable, either integer or continuous. The idea is to develop a procedure that will be easily reused for the treatment of other problems, without changing the genetic operators, once the precision degree is specified as imposed by the physical nature of the variable. The only changes that may be required concern the computation of the adaptation function, which is typical of the treated problem.

The second objective of this work is to treat multiobjective problems that are particularly relevant to chemical engineering applications. The underlying idea is to develop a search procedure, leading to the Pareto non-dominated solutions, independently from the problem that is being tackled. This means that a special effort is devoted to coding and selection steps which are recognized to be crucial in a GA.

This paper is organized as follows: Section 2 presents rapidly the principles of the multiobjective genetic algorithm used through this study for multiobjective optimization purposes. Section 3 is devoted to the procedure validation with mathematical test functions. Section 4 respectively outlines the methodology for multicriteria batch plant design and displays results of the method applied to the studied batch plants. Section 5 concludes the current work and suggests new areas for investigation.

2. Development of a multiobjective framework with GA

2.1. Some considerations about conflicting behaviour among criteria

The multiobjective optimization using GA was treated in the literature by several authors (Jones, Mirrazavi, & Tamiz, 2002; Zitzler & Thiele, 1999). Among the most widespread methods, let us mention here VEGA (Vector Evaluated Genetic Algorithm) Schaffer (1985), NPGA (Niched Pareto Genetic Algorithm) (Horn et al., 1993), MOGA (Multi Objective Genetic Algorithm) Fonseca and Fleming (1993) NSGA

(Non-dominated Sorting in Genetic Algorithm) Srinivas and Deb (1994) and SPEA (Strength Pareto Evolutionary Algorithm) Zitzler and Thiele (1999). A detailed presentation can be found in Massebeuf (2000).

In Dedieu (2001), Dedieu et al. (2002), the global GA framework is adapted without including neither new procedures nor parameters. Fig. 1a shows schematically the results obtained when the monocriterion framework is applied to two optimization criteria. The GA begins with a set of solutions integrating the initial population, which then evolves towards the optimal solution for each criterion separately. The option proposed in Dedieu (2001) in order to obtain the set of Pareto optimal solution, was to apply a Pareto sort procedure over the set of solutions evaluated during the GA evolution, from the randomly generated initial population towards the final population, selecting the "good" solutions for the considered criteria.

The implementation of this framework is possible when the criteria to optimize do not have a marked conflicting behaviour, i.e., they may have an interdependent evolution towards the optimal solution; Fig. 1b shows qualitatively this kind of behaviour. The methodology presented is then simple to implement and is able to find the set of optimal Pareto solutions. When the criteria exhibit a strong conflicting behaviour, as it often happens in process engineering (i.e., processing time-product quality, investment cost-environmental impact), the above mentioned strategy may fail to find the whole set of Pareto optimal solutions. Fig. 1c shows schematically the search carried out by this method under strong conflicting behaviour criteria. Only a reduced part of the Pareto set of solutions is found around the optimal value for each optimization criterion. It is therefore necessary to propose a new genetic search based methodology that can find simultaneously "good" solutions for each criterion independently as well as a set of compromise solutions between the optimization criteria considered.

The aim of this contribution is to propose a generic multiobjective genetic algorithm able to evolve naturally towards the whole set of optimal Pareto solution. This evolution must be carried out from an initial population, which is generally randomly generated and composed of individuals not adapted to the considered criteria; it is necessary to ensure that the added mechanism allows an efficient search for the compromise zone (Fig. 1d). Two stakes are highlighted: the former concerns the simultaneous optimization of several criteria and the latter involves the search for compromise solutions for the considered criteria.

We have then proposed to take into account the multiobjective aspects at the selection stage and the compromise solution search at the crossover stage. The classical selection procedure selected is the biased roulette

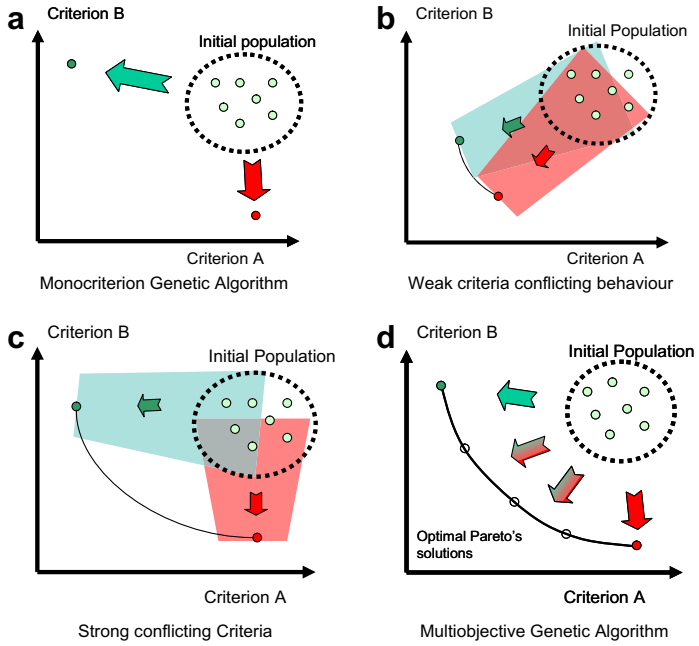


Fig. 1. Various ways to take into account multicriteria aspects with a genetic algorithm.

wheel (Goldberg, 1994): a roulette for each criterion to optimize is then implemented. An equal number of individuals for each criterion were chosen to complete the total number of individual passing from the survival procedure to the next population. The crossover procedure charged of proposing compromise solutions was not modified.

It must be pointed that the population was composed of "good" individuals for each selected criterion due to the effect of the roulette wheel and that the individuals were chosen through a random procedure. This allows "good" solutions with respect to a criterion to cross with "good" solutions with respect to another one, with a strong probability to generate a compromise between both criteria. In the case that both "good" chosen solutions correspond to the same criterion, the crossover procedure will carry out the traditional function of generating a better solution than the two previous ones. The mutation procedure is not modified and its aim is, as usual, to diversify the search and to avoid local optimum solutions.

2.2. Implementation principles

The MOGA developed in this study involves different procedures, as summarized in the flowchart presented in Fig. 2 that illustrates the cycle {Evaluation, Selection, Crossover and Mutation} which is repeated until a stop criterion is reached. After this cycle, the Pareto sort is applied.

Concerning selection and multicriteria aspects involved, it must be emphasized that for a given survival rate, the selection process is achieved via a classical roulette wheel (Goldberg, 1994) relative to each criterion.

The genetic algorithm includes the following steps (1 to 7). The objective of each separated step is quite classical for GA practitioners and does not require additional comment. What is important to note here, is the interconnection of these basic steps and the treatment of the multiobjective elements that are not systematically tackled in the same way as reported by numerous researchers. The implementation of several delicate phases is presented in detail in the following section.

Step 1: An initial population is generated randomly. This procedure guarantees a diversified initial population covering the complete space search.

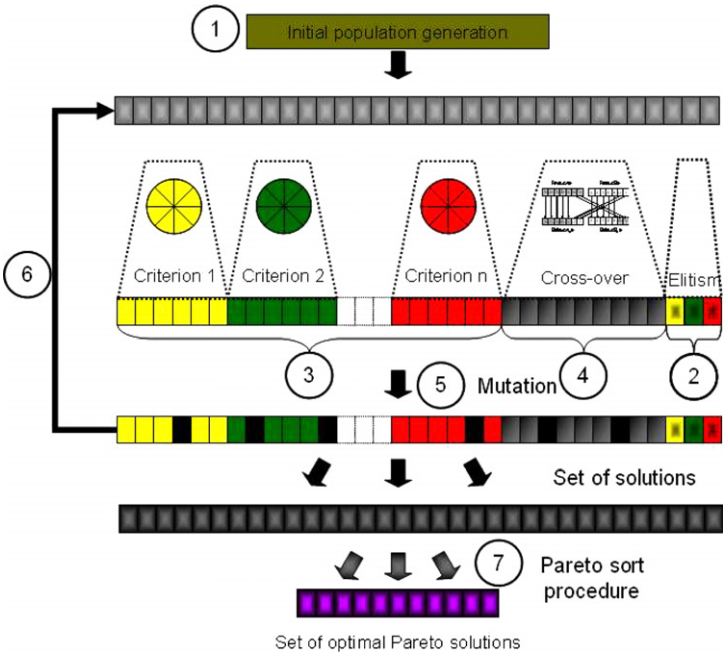


Fig. 2. Multiobjective genetic algorithm.

The following steps are performed in order to pass from the actual population (k) to the next one ($k+1$). First an intermediate population is generated in steps 2 to 4.

Step 2: An elitism procedure selects the best individual for each criterion, which is then placed in intermediate population.

Step 3: An equal number of individuals are chosen for each criterion using the biased Goldberg's roulette.

Step 4: The intermediate population is completed with individuals generated by the crossover procedure. Let us note that the individuals to whom this procedure is applied are chosen randomly from the population k. The simplest crossover procedure is used with only one cutting point.

Step 5: The mutation procedure is applied to a fixed number of individuals chosen randomly. Only one point of the chromosome is modified, changing its value from 0 to 1 or the opposite.

Step 6: The new population becomes the current one and the Steps 2 to 5 are repeated until the maximal number of generations is reached.

Step 7: A Pareto sort procedure is carried out over all the individuals evaluated over generations.

At the end of the algorithm, the set of Pareto non-dominated solutions is obtained.

3. Procedure validation with mathematical test functions

3.1. Introduction

The above mentioned methodology proposed for taking into account simultaneously conflicting criteria is first tested over mathematical test functions. These test bench functions are presented in Table 1 and served previously as a reference in Dedieu (2001) and Viennet (1997). The mathematical function tests are composed

Table 1
Mathematical test problems

Functions:	Minimum:
Problem 1:	
$f_1 = \frac{(x_1-2)^2}{2} + \frac{(x_2+1)^2}{13} + 3$	$\hat{f}_1(2;-1) = 3$
$f_2 = \frac{(x_1+x_2-3)^2}{36} + \frac{(-x_1+x_2+2)^2}{8} - 17$	$\hat{f}_2(2,5;0,5) = -17$
$f3 = \frac{(3.x_1-2.x_2-1)^2}{175} + \frac{(-x_1+2.x_2)^2}{17} - 13$	$\hat{f}_3(0,5;0,25) = -13$
$x = (x_1, x_2) \in [-4,4]^2$	
Problem 2:	
$f_1 = \frac{(x_1-2)^2}{2} + \frac{(x_2+1)^2}{13} + 3$	$\hat{f}_1(2;-1) = 3$
$f_2 = \frac{(x_1+x_2-3)^2}{36} + \frac{(-x_1+x_2+2)^2}{8} - 17$	$\hat{f}_2(2,5;0,5) = -17$
$f_3 = \frac{(3.x_1-2.x_2+4)^2}{18} + \frac{(x_1-x_2+1)^2}{27} + 15$	$\hat{f}_3(-2;-1) = 15$
$x = (x_1, x_2) \in [-4,4]^2$	
Problem 3:	
$f_1 = \frac{x_1^2}{2} + \frac{(x_2+1)^2}{13} + 3$	$\hat{f}_1(0;-1) = 3$
$f_2 = \frac{x_1^2}{2} + \frac{(2.x_2+2)^2}{15} + 1$	$\hat{f}_2(0;-1) = 1$
$f_3 = \frac{(x_1+2.x_2-1)^2}{175} + \frac{(2.x_2-x_1)^2}{27} - 13$	$\hat{f}_3(0,5;0,25) = -13$
$x = (x_1, x_2) \in [-4,4]^2$	
Problem 4:	
$f_1 = 0,5.(x_1^2 + x_2^2) + sin(x_1^2 + x_2^2)$	$\hat{f}_1(0;0) = 0$
$f_2 = \frac{(3.x_1-2.x_2+4)^2}{8} + \frac{(x_1-x_2+1)^2}{27} + 15$	$\hat{f}_2(-2;-1) = 15$
$f_3 = \frac{1}{(x_1^2+x_2^2+1)} - 1,1.exp(-x_1^2 - x_2^2)$	$\hat{f}_3(0;0) = -0,1$
$x = (x_1, x_2) \in [-4,4]^2$	

of four problems with three objective functions, each one depending on two independent variables. The choice of these mathematical functions is due to their particular behaviour, which will be further justified.

Of course, the test bench used was more extended in the complete study: for instance, we used other classical test bench functions, as recommended by several specialists (see the book of Collette & Siarry, 2002). By lack of place, here, only the test functions used in the work of Viennet (1997) are analysed here.

It must be emphasized that this preliminary study is carried out with respect to the optimization variables ($X1$ and $X2$) and not to the objective function values. The aim of this study is to analyse the MOGA search performance, especially when looking for compromise solutions in the search space, which justifies the motivation in optimization variable values. The shape and the parameters of the objective functions were set in order to obtain a conflicting behaviour. The compromise representation between the values of the objective function does not give any additional information; the main interest is the location of the solutions within the search space.

3.2. Genetic algorithm implementation

3.2.1. Variable encoding

A binary system was chosen for encoding, as it simplifies the genetic operators, crossover and mutation. The aim is to define crossover and mutation procedures that are independent of the problem tackled. Any other encoding system can always be translated in a binary encoding system. The continuous variables were discretized and encoded in a binary way by a variable change (Fig. 3). In order to simplify the encoding parameters, all the continuous variables were encoded using the same bit number (20). For each one, it was checked whether the discretization was accurate enough for the problem. An example concerning a continuous variable $X1$ is also presented.

3.2.2. Initial population creation

In a general case, each chromosome is filled randomly with binary values. Then, if it represents a solution to our problem, the chromosome is kept; otherwise, another chromosome is generated. For a particular case, a guided initialisation may be defined.

3.2.3. Survival

For a given survival rate, the selection process is achieved via a classical biased roulette wheel (Goldberg, 1994). The selection is performed and each selected individual is included into the new population.

3.2.4. Crossover operation

To complete the new population, a classical one-point crossover is performed on pairs of individuals randomly chosen in the current population.

3.2.5. Mutation operation

After selection and crossover, mutation is then applied on the resulting population, with a fixed mutation rate. The number of individuals on which the mutation procedure is carried out is equal to the integer part of

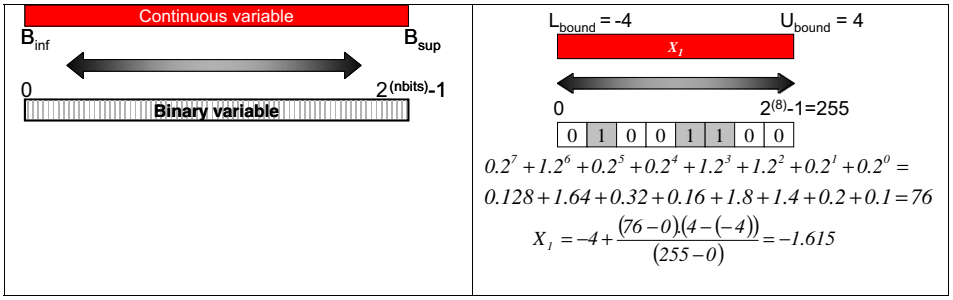


Fig. 3. Continuous variables encoding.

Table 2
Genetic algorithm parameters

Population size	120
Generation number	200
Survival rate	0, 40
Mutation rate	0, 15
Elitism	1

the value of the population size multiplied by the mutation rate. These individuals are chosen randomly among the population and then the procedure is applied.

3.2.6. Elitism

The elitism consists in keeping the best individual from the current population to the next one. Table 2 presents the GA parameters used in the preliminary study. The reference values were taken from Dedieu (2001). The GA parameters used for the mathematical functions take quite classical values. Their setting and relative value did not generate any difficulty and the results exhibit the same trends whatever the value. The experimental design framework initially performed did not need to be presented in detail.

The population size is set to the criteria number multiplied by the monocriterion population size. Given the population size and the generation number, it may be illusory to find the whole Pareto set of optimal solutions. Low values for this parameter were set in order to evaluate the algorithm performance search, allowing however to find the optimal solution for each objective function. Let us note at that level that for real GA implementations, the search space fraction will still be lower for computing time reasons.

It is well-known that the parameters of the GA may have a significant influence on convergence (see for instance a study on batch plant design Bernal Haro, 1999). It was shown typically that a low survival rate and/or a high mutation rate give a slower convergence, a high survival rate and/or low mutation rate increase the risk of local optimum solution. Since a multiobjective optimization is considered here, a new interest focus is the solution location in the space search. The first results which will be further presented and commented in detail show that solutions are ranged in the search spaces forming lines. This problem is of course not found in monocriterion optimization because the search path is never analysed. This behaviour is due to the effect of mutation and crossover that modifies only one place in the chromosome and, consequently, only one of the optimization variables. The mutation rate could be increased or the survival rate decreased but this would affect MOGA search performance. One solution, used in most of the cases, is to define adapted crossover and mutation procedures to the problem, but it may penalise the generality of the GA, so we have chosen to modify the chromosome, yet keeping the same encoding: we only change the order in which the bits of the original encoding are placed in the chromosome. In this work, a new variable ranging is proposed in order to avoid this result behaviour.

The studied case considers two optimization variables encoded using a binary system. In this work, three different variables locations in the chromosome are proposed: sequentially, altering bits and randomly positioned bits (Fig. 4).

Traditionally, after encoding, the variables are located sequentially in the chromosome (Fig. 4, case A). Two other variable ranging options are considered: the former, which consists in altering one bit of each variable (Fig. 4, case B) and the latter, which consists in placing the bits of each variable randomly in the chromosome (Fig. 4, case C).

3.3. Influence of the variable location in the chromosome

For each problem, ten optimization runs were carried out. At the end of each run, the Pareto sort procedure was implemented, thus obtaining the non-dominated solutions corresponding to each run. At the end of ten optimization runs, a Pareto sort procedure is implemented again over the set of optimal solution obtained at each GA implementation. These results will constitute the proposed solution to the optimization problem, i.e., the set of non-dominated Pareto optimal solutions. The next part is dedicated to the results analysis of the multiobjective optimization problems proposed.

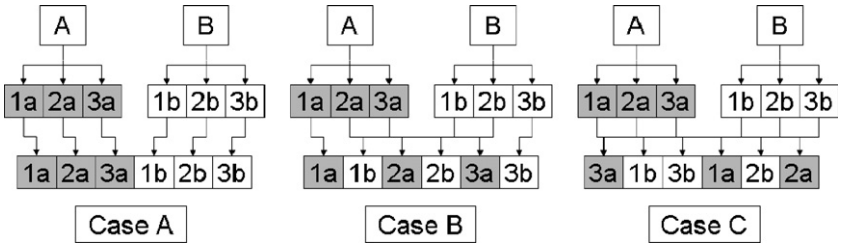


Fig. 4. Variable ranging in the chromosome.

In this section, for briefness reasons, only the figures corresponding to the first multiobjective problem are presented, and the results corresponding to the three other multiobjective optimization problems are presented in the Appendix.

First, the influence of the variables location in the chromosome is analysed. Fig. 5 shows the results for one optimization run for each variable location. The optimization run presented was chosen because it clearly presents the features of the results obtained for each kind of variable location, once the optimal value for each objective function is obtained.

In Fig. 5a, the typical feature corresponding to the traditional ranging of variables in the chromosome is shown. The alignment of points is strongly marked, both vertically ($X2$ variable) and horizontally ($X1$ variable). This behaviour comes from the choice of only one point for the crossover procedure. The crossover procedure applied to two individuals only modifies one of the variables each time and, consequently, several solutions obtained after the Pareto sorting procedure have the same value for one of the variables, mainly those that correspond to the optimal solutions for one criterion.

The encoding choice of alternating the bits of each variable guarantees (except in the case of the first and the last cutting point where only one bit is exchanged between the two chromosomes) the modification of two variables when the crossover procedure is applied. Fig. 5b shows typical results obtained with this type of ranging. The same behaviour as above mentioned (alignment of points) is obtained, but it is less marked in this case and the points are better distributed. The alignments are due to the non-uniform weight of bits in binary encoding. Although the two variables are modified at the same time, the exchanged part has less weight that the preserved part. Once the optimal solution is found, the search takes place mainly around these values.

Observing this behaviour, a random bit ranging in the chromosome is considered. It guarantees, on the one hand, to modify the two variables simultaneously, as in the previous case, and on the other hand, to exchange different weighted bits. The results obtained are presented in Fig. 5c. The solution distribution in the search space is more uniform than in the two previous cases and, moreover, the alignment is considerably reduced.

3.4. Result comparison with different bit ranging encoding strategies

After ten optimization runs for each problem with the corresponding encoding method, a final Pareto sorting procedure is carried out over the Pareto optimal solutions obtained at each run. The set of solutions obtained after the sorting procedure is considered as the solution proposed by the methodology to the multiobjective optimization problem.

Fig. 6 shows the results obtained, after the final Pareto sorting procedure, for the different types of variable ranging in the chromosome, i.e., A, B and C type, respectively, for the first problem. The results obtained for the three other examples are presented in the Appendix.

The three figures (a, b and c) display the obtained solutions, which are of equivalent quality for the three types of variable ranging in the chromosome.

3.5. Choice of the selecting compromise solution method

As it was mentioned previously, the selected concept for multiobjective optimization follows Pareto's ideas (Pareto, 1896). This widespread concept is presented below and compared qualitatively to other techniques. Lets us recall that a solution is considered Pareto optimal, if there does not exist any other feasible solution

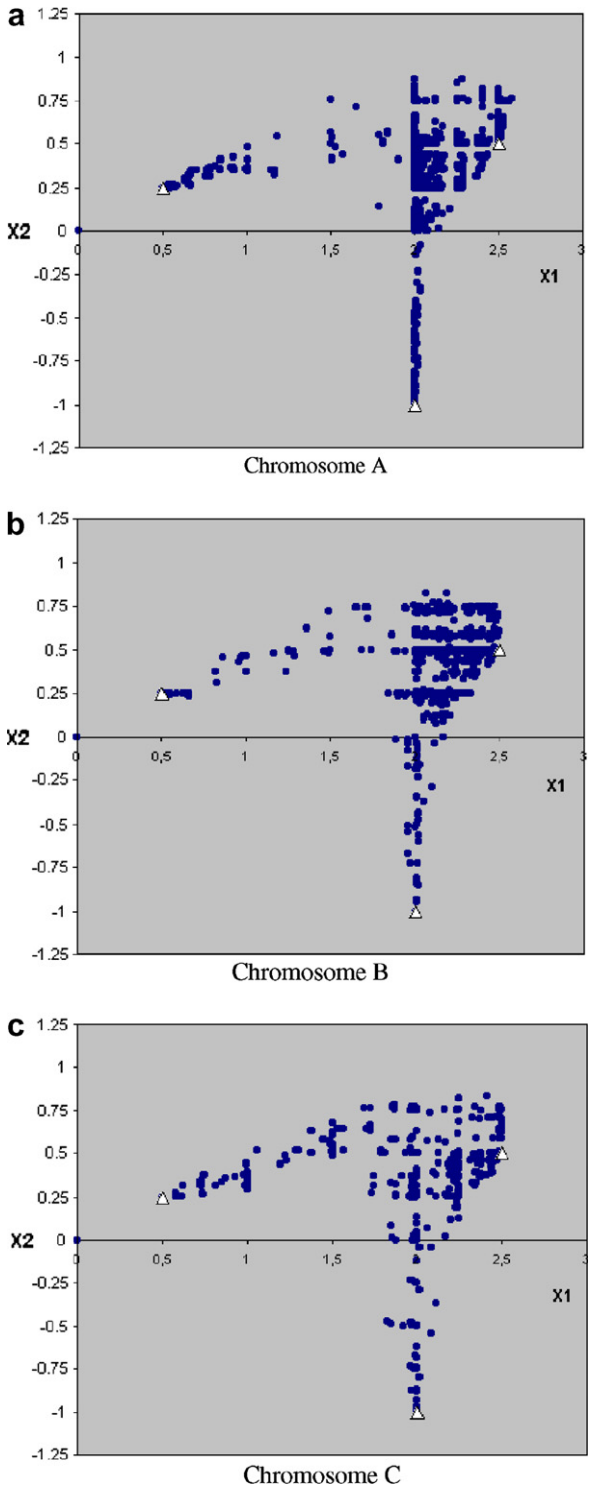


Fig. 5. Typical results, problem 1, chromosomes A, B and C.

that decreases a criterion without increasing at the same time at least another criterion. As pointed out in introduction section, the weighted sum is a largely used technique that transforms the multiobjective optimization problem in a monocriterion problem and was implemented here for illustration purposes.

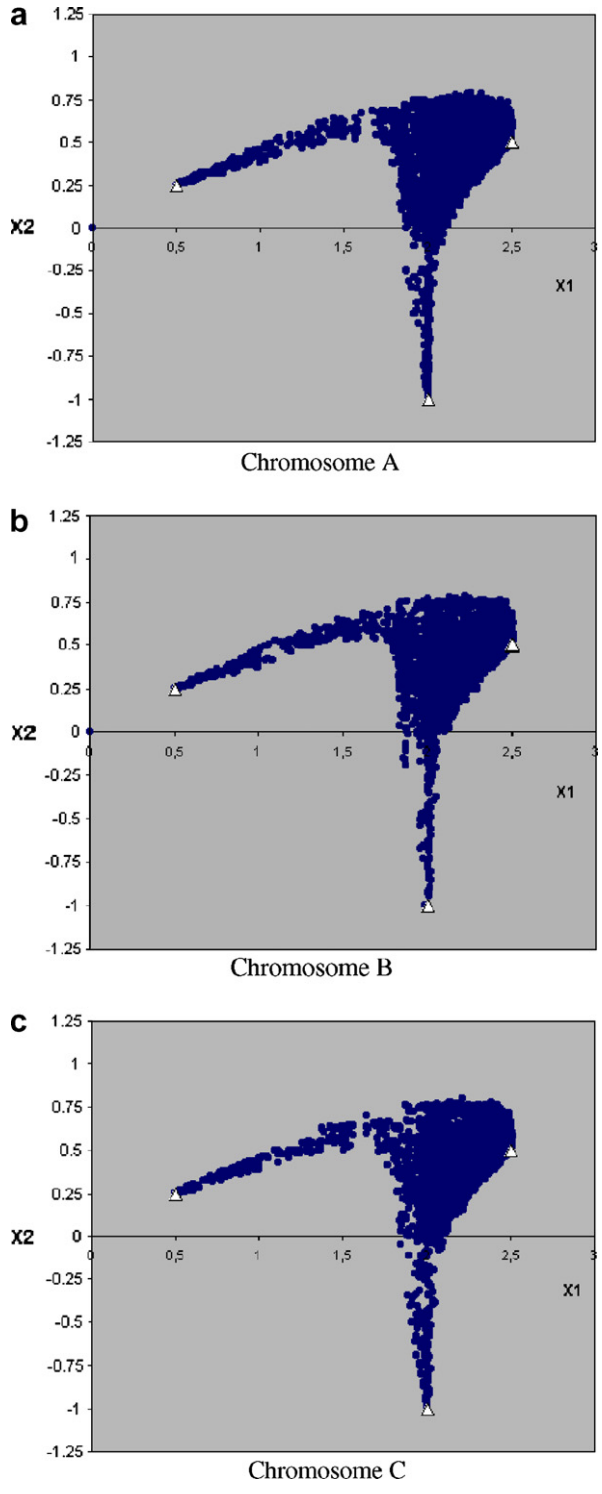


Fig. 6. Problem 1, Chromosomes a, b and c.

Fig. 7 presents the results obtained with the weighted sum method on the previous Pareto zone (A). Series B corresponds to the optimum of each function solely considered. The other series correspond to

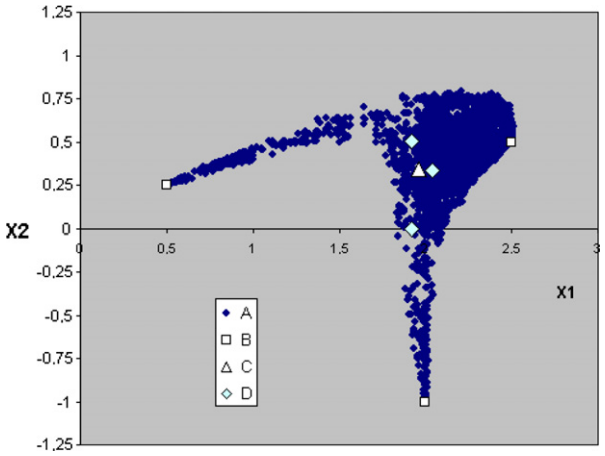


Fig. 7. Solutions obtained with the weighted sum method.

two different cases: the former with identical values for the three weighting coefficients (C) and the latter involves a coefficient that is twice the value of the two other ones (D).

This method drives to a unique solution of the problem that does not represent the whole set of compromise solutions. To get it, it would be necessary to implement several optimization runs while modifying the relation between the weighting coefficients. This method may be very penalising from a computing time point of view.

It is interesting to compare the results with an approach that consists in choosing a solution of compromise from values of the optimization variables and not from the objective function values. Fig. 8 shows this comparison for the four problems treated (Fig. 8a–d).

For problems 1 and 2, where the function optimal values are located at different positions in the search space (Fig. 8a and b), a compromise solution chosen as the barycentre of the triangle defined by these three points, places this solution outside Pareto optimal solution zone. For problems 3 and 4 (Fig. 8c and d), two of the objective functions have their optimal value at the same point. The possible compromise solutions chosen on the tie line that joins the two points representing the optimal solutions for both criteria are located outside Pareto optimal solutions zone.

This analysis reinforces the idea of using a Pareto sort at the end of the procedure, since a set of solutions that is representative of Pareto optimal solution zone can be reached with a low number of function evaluations.

3.6. Comparative study on the performances of variable ranging method

From the results in different optimization runs, the performance of each type of variable location has been analysed through two criteria:

- average number of generations necessary to reach the optimal value for each criterion;
- number of optimal solutions obtained after final Pareto sort procedure.

Table 3 presents the average of generations, over 10 optimization runs, necessary to reach optimal values for each optimization criterion, for problems P1, P2 and P3. In the case where the optimal value has not been found by the genetic algorithm, the maximal number of generations was considered since the obtained value is very close to the optimum. Problem P4 was not analyzed, since it presents some numeric difficulties in the neighbourhood of the optimal value that is zero.

Even if it could be quite simplistic to analyse the performances of each variable ranging method through only one value, i.e., the average, the aim of this study is to obtain the qualitative trend. For problems P1 and P2, types A and B are clearly more efficient. For problem P3, the three types have a similar performance. Globally, type C gives the highest number of failures.

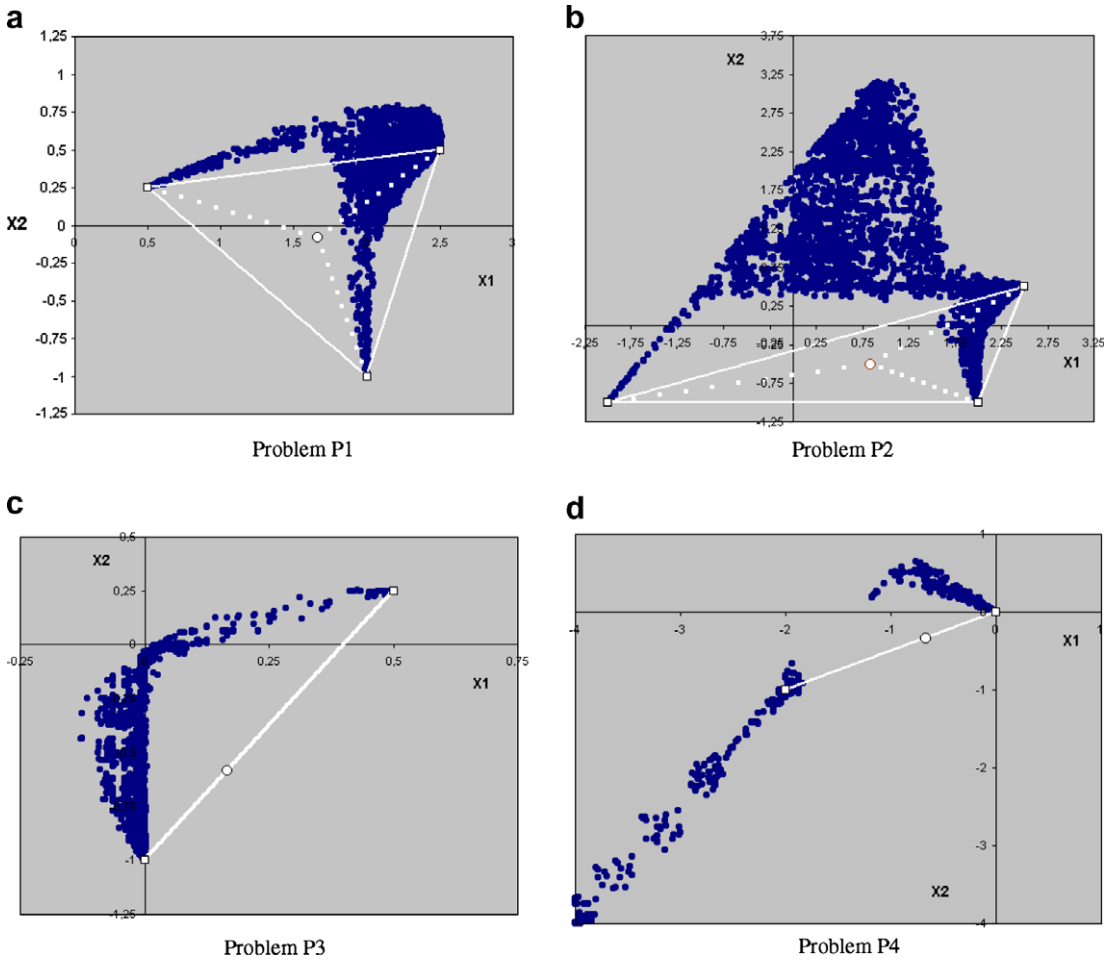


Fig. 8. Problems 1 to 4, Barycentre versus Pareto.

Table 3
Genetic algorithm performance

Problem	Average criterion 1	Average criterion 2	Average criterion 3	Average
P1 – Type A	41	42	61	48
P1 – Type B	39	41	47	42
P1 – Type C	50	51	145	82
P2 – Type A	67	43	65	58
P2 – Type B	88	40	55	61
P2 – Type C	97	67	81	82
P3 – Type A	88	48	56	64
P3 – Type B	88	46	39	58
P3 – Type C	100	38	44	61

This drawback is due to the different bit weights in the chromosome. In case A, only one of the variables in the chromosome is modified and the bits of this variable are placed by decreasing order of weight. For type B, the two variables are modified but the bits are always arranged in increasing order (or decreasing). This type of ranging enables a faster convergence for continuous and convex functions. Type C, placing bits randomly in the chromosome, drives to a radical change of variable values, thus disturbing convergence while allowing the search for compromise solutions, as it is shown in Figs. 5c, 6c and 7.

Table 4
Pareto optimal solution number

	Problem 1	Problem 2	Problem 3	Problem 4
Type A	22,043	19,214	2636	2847
Type B	14,620	14,082	3736	2494
Type C	10,508	9350	3056	2624

Since the quality of the obtained solutions after Pareto's final sort seems similar for the three types of variable ranging in the chromosome, it is interesting to compare the number of compromise solutions obtained for each case, as presented in Table 4. For problems P1 and P2, type A exhibits better performances, since it leads to a higher number of compromise solutions; in the same way, B type is better than C type. For problems P3 and P4, equivalent performances are obtained for the three types of ranging.

The features observed with problem P3 for the criterion relative to the number of necessary generations to find the optimal value and those found in problems P3 and P4 for the criterion dedicated to the number of compromise solutions can be assigned to the fact that two of the objective functions have their optimal value at the same point, transforming the problem for these two criteria to a pseudo-monocriterion problem. In what follows, the A type ranging coding was finally adopted.

4. Application to multiobjective batch plant design or retrofit

This section is dedicated to the presentation of the MOGA implementation in multicriteria batch plant design or retrofit applications.

4.1. Multiproduct batch plant design for proteins production

This example deals with the multicriteria cost-environment design of multiproduct batch plants, where the design variables are the size of the equipment items as well as the operating conditions. The case study is a multiproduct batch plant for the production of four recombinant proteins (Dietz et al., 2004a, 2004b, Dietz, Azzaro-Pantel, Pibouleau, & Domenech, 2004c; Asenjo, Montagna, Vecchietti, Iribarren, & Pinto, 2000; Montagna, Vecchietti, Iribarren, Pinto, & Asenjom, 2000; Pinto, Montagna, Vecchietti, Iribarren, & Asenjo, 2001). This is a multiproduct batch plant, with four products to be manufactured by fermentation and eight treatment stages. Fig. 9 shows the flowsheet of the multiproduct batch plant as well as batch plant effluents considered in this study.

The cost criterion considered is classically based on investment minimization, ICost, involving investment cost for equipment and storage vessels, computed by (Eq. (1)):

$$\mathrm{ICost} = \sum_{i=1}^{N_{\mathrm{OP}}} \sum_{j=1}^{N_{\mathrm{EQ}}} (A_i + B_i V_{ij}^{C_i}) + \sum_{k=1}^{N_{\mathrm{SK}}} (A_k + B_k V_{sk}^{C_{ki}}) \tag{1}$$

In Eq. (1), NOP is the number of operations, NEQi is the number of equipment items (for operation i), NSV is the number of storage vessels, A_i, B_i and C_i are the cost coefficients for operation i, A_s, B_s and C_s are the cost coefficients for storage vessels, V_{ij} is the volume of equipment ij and V_{sk} is the volume of storage vessel k.

Concerning the environmental impact, the quantity and the quality of the process effluents depend only on the operating conditions having an influence on mass balance at each treatment stage. The global index of each Environmental Impact (EI) criterion is defined as weighted sum respect to the production of each product index (Eq. (2)).

$$I_k = \frac{I_k^{\mathrm{ins}}.P_{\mathrm{ins}} + I_k^{v}.P_{\mathrm{vac}} + I_k^{\mathrm{chy}}.P_{\mathrm{chy}} + I_k^{\mathrm{pro}}.P_{\mathrm{pro}}}{P_{\mathrm{ins}} + P_{\mathrm{vac}} + P_{\mathrm{chy}} + P_{\mathrm{pro}}} \tag{2}$$

Where I_k is the pollution global index, I_k^i is the k pollution index of i product defined as the amount of the pollutant k [kg] by amount of i manufactured [kg] and P_i is the total production of the i product.

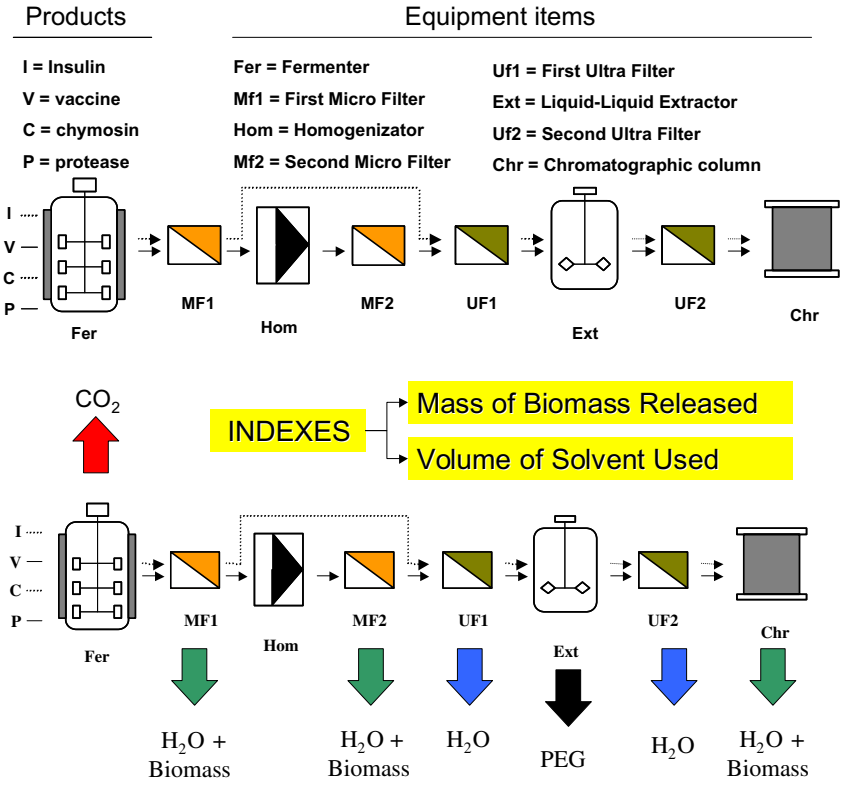


Fig. 9. Multiproduct batch plant for proteins production.

In a general way, the optimization problem can be presented as follows:

$$\begin{aligned} &\min f_1(y) \\ &\min f_2(x) \\ &\text{s.t.} g(x,y) \leqslant H \end{aligned}$$

where, f_1 represents the investment cost and f_2 the environmental impact. Vector $x = [x_1, x_2, \ldots, x_n]$ represents the operating conditions and $y = [y_1, y_2, \ldots, y_n]$ refers to batch plant configuration.

The investment cost, f_1, is function of the batch plant configuration, y. The environmental impact, f_2, is a function of the operating conditions, x subjected to the production constraints. The x (respectively y) vector involves continuous (respectively discrete) variables. In this paper, the environmental impact criterion was split into two objective functions (mass of biomass released, amount of solvent used) but always respecting the below formulation. The tricriteria formulation was treated elsewhere (Dietz, Azzaro-Pantel, Pibouleau, & Domenech, 2006).

The problem involves 18 continuous variables and 26 discrete variables, thus giving a total number of 44 variables to optimize.

Two production policies were kept for optimal batch plant design purposes (Dietz et al., 2004c): in the monoproduct campaign policy (denoted A), all the batches of a product are treated before processing a different product; in the multiproduct campaign mode (denoted B), one batch of each product is treated successively, following a specific order (Dietz et al., 2004c).

4.1.1. Competing results on monocriterion optimization

It must be recalled here that this example serves previously as a test bench for comparison of monocriterion optimization techniques when considering the investment criterion. Literature analysis reveals that a

significant investigation effort has been carried out to develop efficient and robust optimization methods. But if they prove to be well-fitted to the particular case they consider, the performance of these techniques is not constant over all the problems. Actually, method efficiency for a particular example is hardly predictable.

Actually, method efficiency for a particular example is hardly predictable, and the only certainty we have is expressed by the so-called *No Free Lunch* theory: there is no method that outdoes all the other ones for any considered problem. This feature generates a common lack of explanation concerning the use of a method for the solution of a particular example, and usually, no relevant justification for its choice is given *a priori*.

This lack of justification for the use of an optimization method was the issue of a previous study. The objective is then to propose some guidelines that may be useful for the choice of an appropriate optimization technique. Obviously, the quoted *No Free Lunch* theory prevents from drawing any general conclusions, which could be extended to any class of problems. Batch plant design problems based on Mixed-Integer Non-Linear Programming (MINLP) formulation provides indeed a good application aid to evaluate several methods efficiency.

A thorough study comparing the performance of three optimisation methods for a few instances of some specific batch plant design problem formulated as equation-oriented methods, provided some guidelines on the most appropriate methods: without going further into detail, it was proven that the performances of a Branch & Bound technique and a Genetic Algorithm were approximately equivalent on this example and better than those obtained with an Outer Approximation algorithm (Ponsich, Azzaro-Pantel, Domenech, & Pibouleau, 2007). This confirms that the multicriteria analysis was still performed with the GA.

4.1.2. Bicriteria optimization solvent amount used- biomass released

Table 5 displays the parameters of the genetic algorithm used for multicriteria batch plant design. In this work, the generation number was fixed as twice the population size. The global survival rate is relatively low as compared to standard values for optimization of test mathematical functions (Dedieu, Pibouleau, Azzaro-Pantel, & Domenech, 2003). Moreover, a high mutation rate was set. Although a systematic study was not carried out to find these values, they were chosen from several preliminary tests and agree with previous works (Dedieu et al., 2003) where similar problems were treated.

The MOGA presented in this work was first used to demonstrate that the two EI criteria considered (respectively, the total biomass quantity and the solvent volume) present conflicting goals (Fig. 10). Very similar results were obtained at each optimization run, so only the results after the final Pareto sort procedure are presented in Fig. 10. Moreover, it must be noted that slight differences are obtained between both production policies because the environmental impact depends only on the mass balance that is function of the continuous variables. The MOGA was able to find the optimal solution obtained by monocriterion optimization (Dietz et al., 2004c) at each implementation as well as a ''complete'' set of compromise solutions.

4.1.3. Bicriteria optimization solvent amount used- investment cost

The same approach was also applied to the cost – environment criteria. First, the amount of solvent used and the investment cost were considered.

Figs. 11 and 12 show the results obtained at each optimization run for both A and B production policies, performed with an identical parameter set to guarantee the stochastic nature of the GA. In this case, the results are not superposed as it was the case for the bicriteria optimization biomass – solvent, which shows the necessity of carrying out several optimization runs for the same problem. It can be seen that each optimi-

Table 5
Genetic algorithm parameters

	Bicriteria solvent-biomass	Bicriteria cost-solvent	Bicriteria cost-biomass
Population size	300	450	450
Generation number	600	900	900
Survival rate	0.5	0.5	0.5
Mutation rate	0.4	0.4	0.4
Elitism by criterion	1	1	1

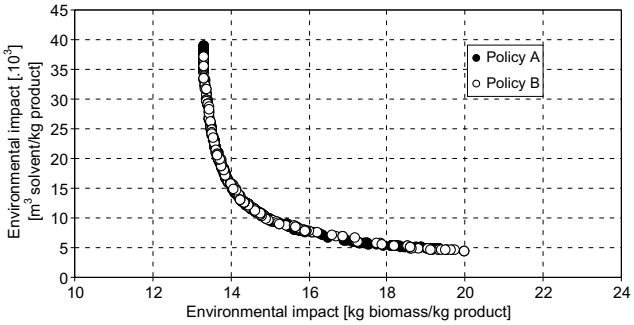


Fig. 10. Pareto's optimal solutions for biomass released used – investment cost Policy A.

zation run is directed to a part of the search region. Run 2 in Fig. 11 and run 3 in Fig. 12 are oriented towards the environmental criterion, while run 3 in Fig. 11 and run 2 in Fig. 12 are oriented towards the economical criterion. Run 1 for both Figs. 11 and 12 is placed in the compromise zone between both criteria. Let us note that Pareto's zone carried out over the set of all solutions is constituted of sparse points, since the adaptation function related to the investment cost takes discrete values.

In order to evaluate GA search performance, Table 6 presents the best solution obtained at each optimization run for each criteria considered as well as the best solution obtained with a monocriterion approach. Even though the methodology was not able to find the best solution, the values are relatively near (around 5% more expensive for the investment cost criterion). It must be noted that in the monocriterion optimization (see Dietz et al., 2004c), the best value was obtained only once and, in the other cases, the solutions were around 2–3% more expensive, which justifies the results when several criteria are taken into account simultaneously. The number of solutions obtained in each optimization run was around of 25. It is important to note that in each case, the solutions are uniformly distributed in the search space; this means that there is no preferential search region in the multicriteria search as shown in Fig. 11.

It is also interesting to see where the results are placed with respect to the criterion not optimized here, in this case the amount of biomass released. Table 7 presents the range of values for this criterion for both production policies.

The ranges have the same order of magnitude for both policies. Moreover, the minimal value of the range is close to the best value obtained in monocriterion optimization (see Fig. 10) value which allows predicting less conflict between investment cost and biomass released criteria.

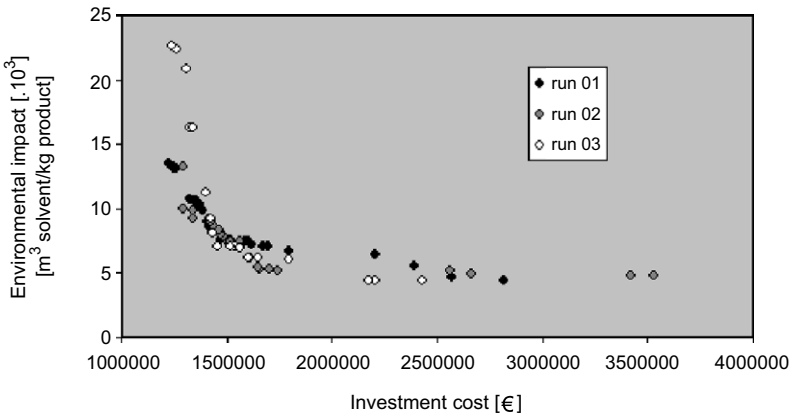


Fig. 11. Pareto's optimal solutions for solvent used – investment cost, Policy A (monoproduct).

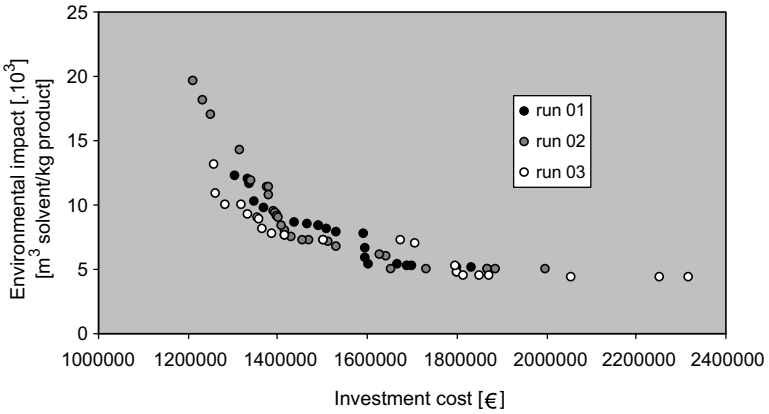


Fig. 12. Pareto's optimal solutions for solvent used – investment cost, Policy B (multiproduct).

Table 6
Bicriteria cost-solvent optimization results

	Production policy A			Production policy B		
	Cost	Solvent [$\cdot 10^{-3}$]	Solutions	Cost	Solvent [$\cdot 10^{-3}$]	Solutions
Run 1	1,221,890	4.44	32	1,303,730	5.07	21
Run 2	1,290,490	4.78	23	1,211,100	5.01	28
Run 3	1,238,050	4.41	23	1,257,200	4.41	23
Best	1,140,990	4.39	–	1,139,100	4.39	–

Table 7
Values range for the not considered criterion

	Biomass for cost-solvent		Solvent for cost-biomass	
	Minimum	Maximum	Minimum	Maximum
Policy A	14.39	20.37	$36.02 \cdot 10^{-3}$	$40.6 \cdot 10^{-3}$
Policy B	14.26	22.89	$34.9 \cdot 10^{-3}$	$41.9 \cdot 10^{-3}$

4.1.4. Bicriteria optimization investment cost-biomass released

The last bicriteria optimization considers the investment cost and biomass released. As for the previous case, three optimization runs were carried out for each production policy. In Fig. 13 relative to policy A, the same typical results as obtained previously are presented. When considering policy B (see Fig. 14), quite dispersed results are obtained between the different optimization runs. An additional optimization run was carried out with an increased search space. The generation number was increased from 450 to 600 and the same ratio between population size and generation number is preserved, that gives 1200 generations.

The results of the additional optimization run have a similar quality to the two best previous implementations. It is important to note that an increase in search space does not lead to more satisfying results, showing that GA parameters lead to a search space large enough for the problem considered. The gap between the different implementations must be related to the used scale. The weak compromise between the two criteria, cost-biomass, justifies the visual difference, that is less important in percentage.

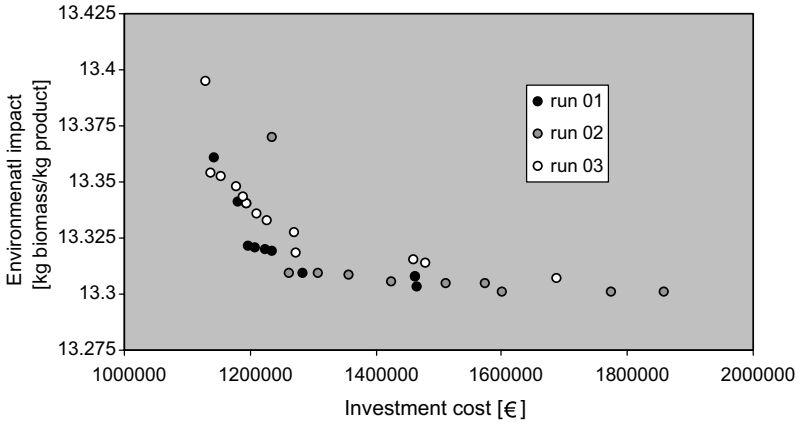


Fig. 13. Pareto's optimal solutions for biomass released – investment cost, Policy A.

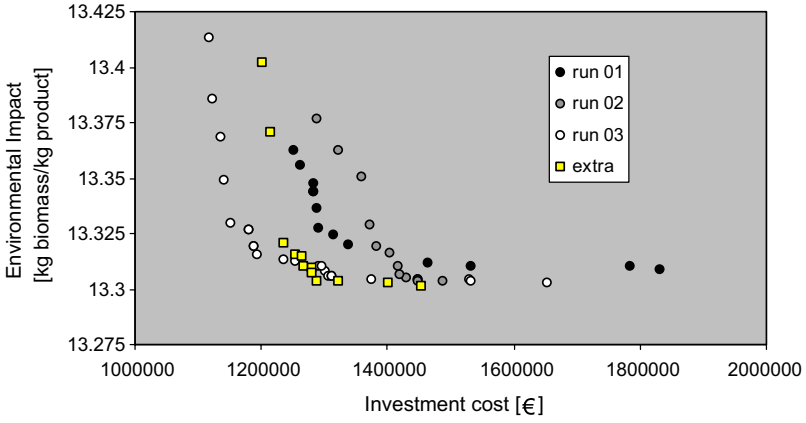


Fig. 14. Pareto's optimal solutions for biomass released – investment cost, Policy B.

Table 8
Bicriteria cost-biomass optimization results

	Production policy A			Production policy B		
	Cost	Biomass	Solutions	Cost	Biomass	Solutions
Run 1	1,143,080	13.303	10	1,252,280	13.307	15
Run 2	1,235,340	13.30	10	1,289,530	13.303	13
Run 3	1,129,290	13.305	15	1,116,950	13.302	22
Best	1,140,990	13.3	–	1,139,100	13.305	–

Table 8 presents the best solution obtained for each optimization run for each criterion considered as well as the best solution obtained with a monocriterion approach. As for the criterion referring to the amount of biomass released, the best value is obtained at each optimization run, as it was the case for the amount of solvent in the previous bicriteria optimization. The number of solutions is slightly inferior to the previous results. This can be explained by the lower antagonism between the biomass and the cost criteria.

Table 7 also presents the range of values for the criterion not considered, i.e., the amount of solvent used. These values are distant from the best values, which reminds the antagonism of this criterion with the others considered as objective functions.

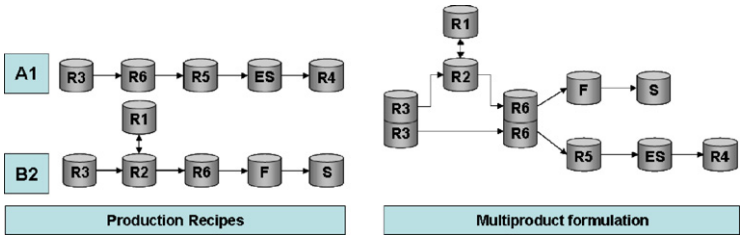


Fig. 15. Production recipes and multiproduct batch plant.

The results obtained show the typical compromise between cost and each environmental index, since the conflicting behaviour between each pair of criteria (investment cost, solvent used and biomass released). The MOGA was able to find good solutions compared to the best solution by monocriterion optimization, as well as an important number of Pareto optimal solutions well distributed in the search space.

4.2. Multiproduct batch plant design/retrofit for fine chemicals production

The case study is an industrial multiproduct batch plant producing four fine chemical products. For confidentiality reasons, only two of them are considered here. A simplified formulation was obtained from the detailed information about the production recipes in order to be implemented in the discrete event simulator used for evaluation of the objective functions. Fig. 15 presents the treatment stages for manufacturing the two products, respectively A1 and B2. Each recipe contains five treatment steps. Only two of them (R3 and R6) are shared by the two products considered.

4.2.1. Bicriteria optimization investment cost-production capacity

The bicriteria cost of investment – production batch plant design was carried out, with a classical formulation involving an imposed production as a constraint. The production based criterion is taken into account through the necessary time to achieve the desired production. If the production is manufactured in half time horizon, the total batch plant production in the complete time horizon is therefore twice the initial production. The investment cost is a criterion that must be minimized while the production must be maximized.

MOGA parameters used for bicriteria cost-production design are presented in Table 9.

Three optimization runs were carried out for both production policies. The obtained results are presented in Figs. 16 and 17 for A and B policies, respectively. A similar quality of results is obtained at each optimization run for both policies, showing the good performance of the optimization procedure.

4.2.2. Batch plant retrofit

In the retrofit application, the current configuration is now supposed to adapt to a new market demand that is 2.5 times the actual demand for both products, A1 and B2. The same approach was used and Fig. 18 shows schematically the equipment items that have to be added to the actual configuration for the two production policies considered.

In the previous configurations, the added equipment items are not in all cases of the same size as the existing ones. However, it is desirable, for flexibility and maintenance reasons, that all the equipment items of the stage be identical, which drives us to a compromise with the investment cost. In order to explore the compromise configuration which respect these two criteria, investment cost and different equipment items, the bicriteria retrofitting is carried out.

Three optimization runs were carried out. For policy A, the results obtained are illustrated in Fig. 19, and the corresponding configurations are presented in Table 10. It is interesting to visualize the dominated Pareto solutions obtained in the second optimization run (S6, S7, S8 of Table 10) which correspond to a local optimum where the main investment is carried out over the A1 production chain instead of the B2 production chain that corresponds to the optimal solution. Given the few solutions obtained, it is advised to keep them

Table 9
Genetic algorithm parameters

Population size	400
Generation number	1000
Survival rate	0.5
Mutation rate	0.4
Elitism by criterion	1

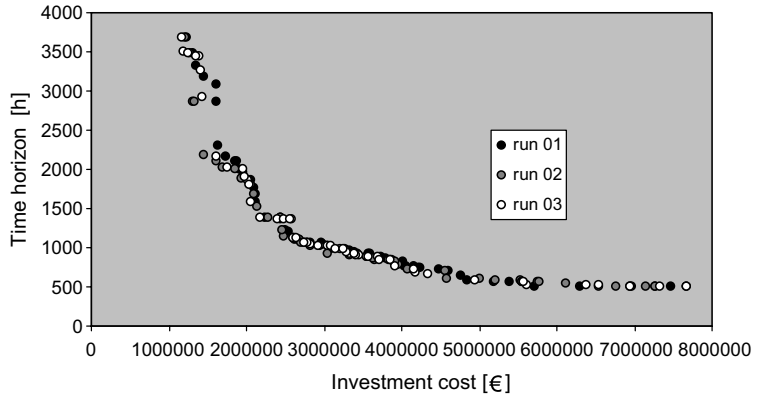


Fig. 16. Bicriteria investment cost-production capacity batch plant design, Policy A.

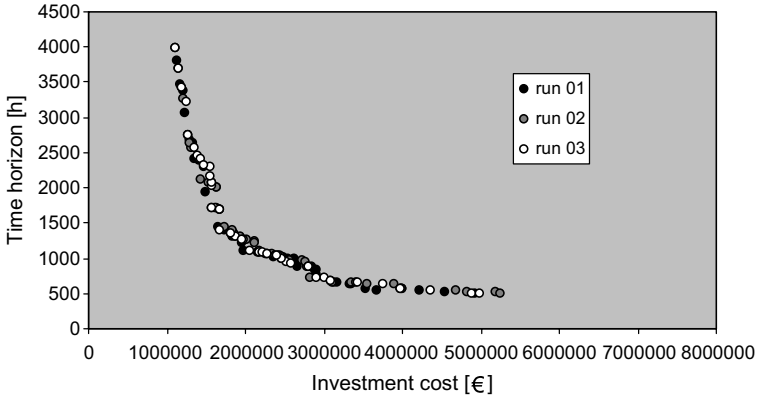


Fig. 17. Bicriteria investment cost-production capacity batch plant design, Policy B.

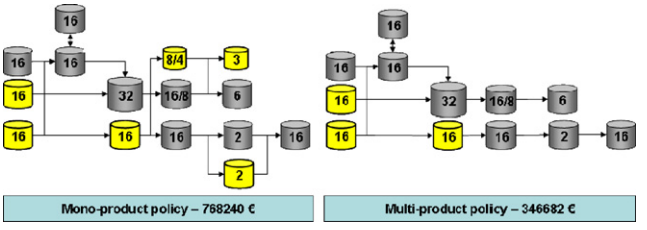


Fig. 18. Monocriterion retrofitting results.

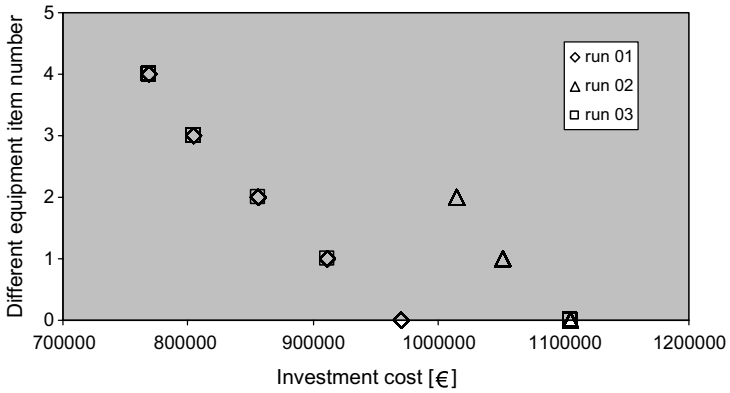


Fig. 19. Policy A multicriteria retrofitting results.

Table 10
Multicriteria retrofit results

Process information									
	Criteria	R3	R6	R5	ES	R4	R21	F	S
S1	768240/4	2–16	1–16	0	1–2	0	0	1–8/4	1–3
S2	804157/3	2–16	1–16	0	1–2	0	0	1–16/4	1–3
S3	856303/2	2–16	1–16	0	1–2	0	0	1–16/8	1–3
S4	910743/1	2–16	1–32	0	1–2	0	0	1–16/8	1–3
S5	970561/0	2–16	1–32	0	1–2	0	0	1–16/8	1–6
S6	1014890/2	2–16	1–16	1-16	2–2	1–8	0	0	0
S7	1050810/1	2–16	1–16	1-16	2–2	1–16	0	0	0
S8	1105250/0	2–16	1–32	1-16	2–2	1–16	0	0	0

for the final decision procedure, because they could be interesting for other criteria, such as maximal production capacity. Concerning policy B, the two expected solutions where found (see Table 10).

5. Conclusions and perspectives

In this paper, a multiobjective optimization procedure based on genetic search was presented. This optimization framework was developed in order to tackle multiobjective problems independently of its nature. For that, the MOGA uses a binary encoding method and the simplest mutation and crossover procedures.

The MOGA was first tested over mathematical test functions and was able to find the optimal solution for each objective function as well as a complete set of solutions in the Pareto optimal zone. An additional study was carried out over the variables ranging in the chromosome. Three configurations were proposed, the advantages and drawbacks of each ranging method were analysed and explained.

Two case studies concerning multicriteria batch plant design and retrofit were presented. Several implementations were carried out in which the MOGA was able, as for the mathematical test functions, to find good solution for each criterion separately as well as a set of solutions representing the Pareto optimal solutions zone.

Parallel works were also performed on a comparative study between different mono-objective algorithmic choices for treating a same problem (see the study of Ponsich et al., 2007) that show that GAs confirm their ability to treat engineering optimization problems.

It is important to note that optimization was performed without any preference information, which means that the Pareto-optimal set consists of all solutions according to any rational decision-maker. Here, the search

for an optimal set of solutions is separated from the final decision. The decision-maker is presented with a set of solutions from which he has to choose, and the hypothesis is that when the trade-off between the objectives is visible it would be easier to choose. However, this might not hold as the number of objectives increases and visualization becomes harder. This is an interesting field for further research: a decision making tool, taking into account various weights on criteria, reflecting the preferences of the decision's maker, may be integrated to the current framework in order to rank the obtained solutions.

Concerning the optimization framework, it could be adapted for taking into account fuzzy information, allowing fuzzy criteria optimization.

Appendix

a–Problem P2 (see Figs. 20 and 21)

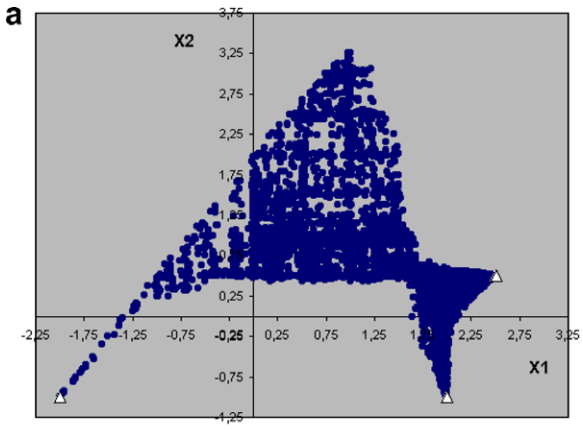
$$f_1 = \frac{(x_1-2)^2}{2} + \frac{(x_2+1)^2}{13} + 3, \quad \text{Minimum} : \hat{f}_1(2;-1) = 3$$
$$f_2 = \frac{(x_1+x_2-3)^2}{36} + \frac{(-x_1+x_2+2)^2}{8} - 17, \quad \text{Minimum} : \hat{f}_2(2,5;0,5) = -17$$
$$f_3 = \frac{(3.x_1-2.x_2+4)^2}{18} + \frac{(x_1-x_2+1)^2}{27} + 15, \quad \text{Minimum} : \hat{f}_3(-2;-1) = 15$$
$$x = (x_1, x_2) \in [-4,4]^2$$

b–Problem P3 (see Figs. 22 and 23)

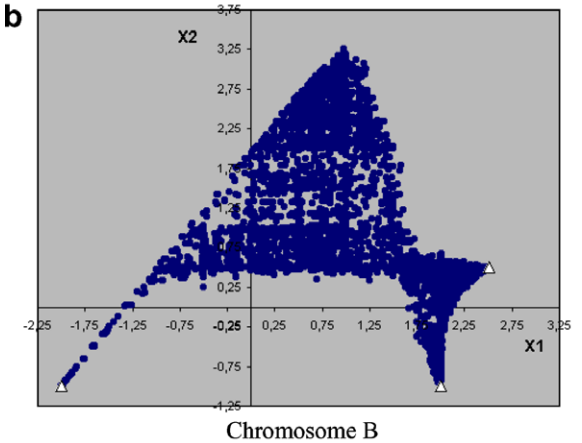
$$f_1 = \frac{(x_1-2)^2}{2} + \frac{(x_2+1)^2}{13} + 3, \quad \text{Minimum} : \hat{f}_1(2;-1) = 3$$
$$f_2 = \frac{(x_1+x_2-3)^2}{36} + \frac{(-x_1+x_2+2)^2}{8} - 17, \text{Minimum} : \hat{f}_2(2,5;0,5) = -17$$
$$f3 = \frac{(3.x_1-2.x_2-1)^2}{175} + \frac{(-x_1+2.x_2)^2}{17} - 13, \quad \text{Minimum} : \hat{f}_3(0,5;0,25) = -13$$
$$x = (x_1, x_2) \in [-4,4]^2$$

c–Problem P4 (see Figs. 24 and 25)

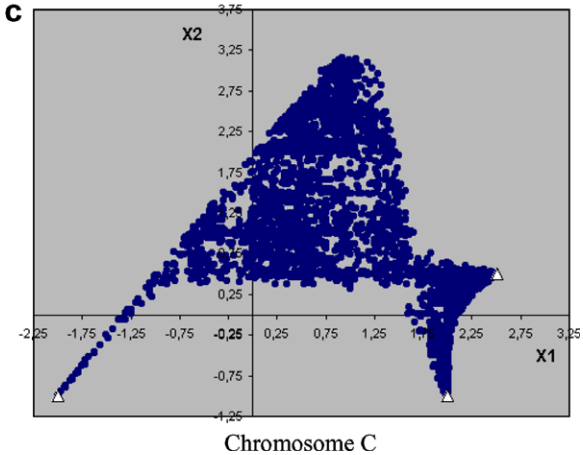
$$f_1 = 0,5.(x_1^2+x_2^2) + \sin(x_1^2+x_2^2), \quad \text{Minimum} : \hat{f}_1(0;0) = 0$$
$$f_2 = \frac{(3.x_1-2.x_2+4)^2}{8} + \frac{(x_1-x_2+1)^2}{27} + 15, \text{Minimum} : \hat{f}_2(-2;-1) = 15$$
$$f_3 = \frac{1}{(x_1^2+x_2^2+1)} - 1,1\exp(-x_1^2-x_2^2), \quad \text{Minimum} : \hat{f}_3(0;0) = -0,1$$
$$x = (x_1, x_2) \in [-4,4]^2$$



Chromosome A

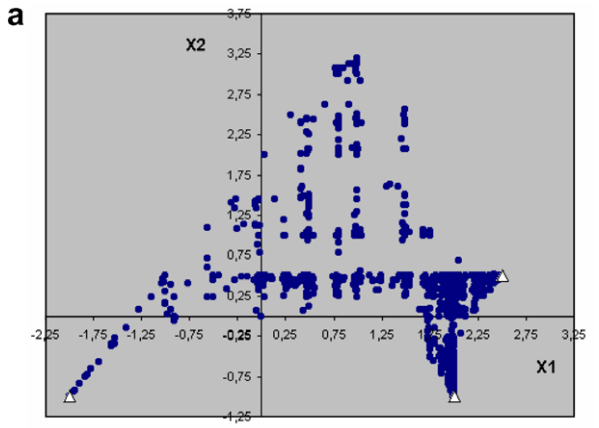


Chromosome B

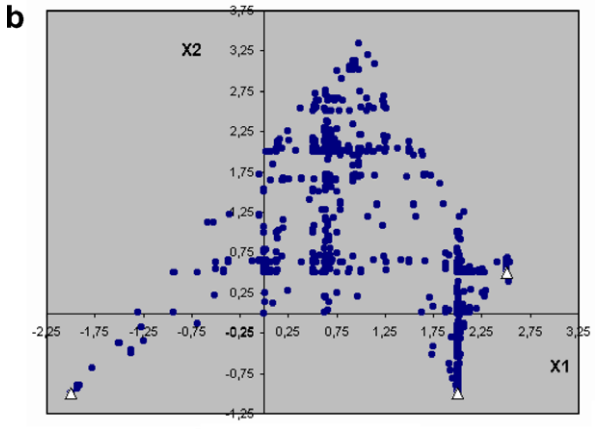


Chromosome C

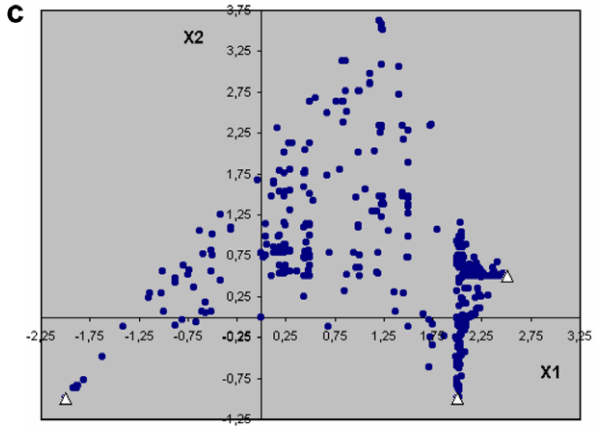
Fig. 20. Problem 2, chromosomes A, B and C.



Chromosome A

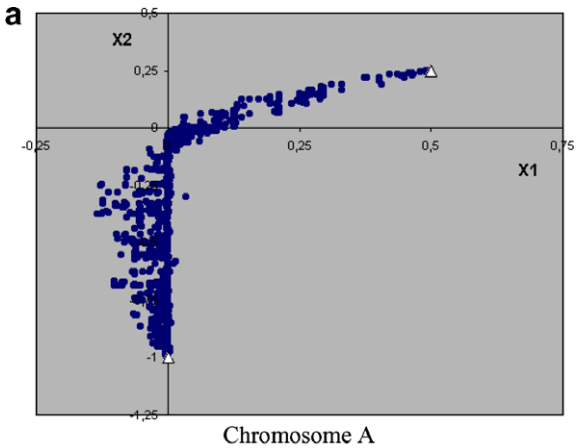


Chromosome B

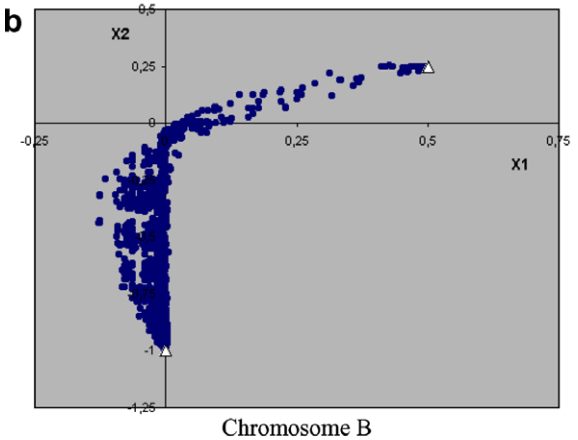


Chromosome C

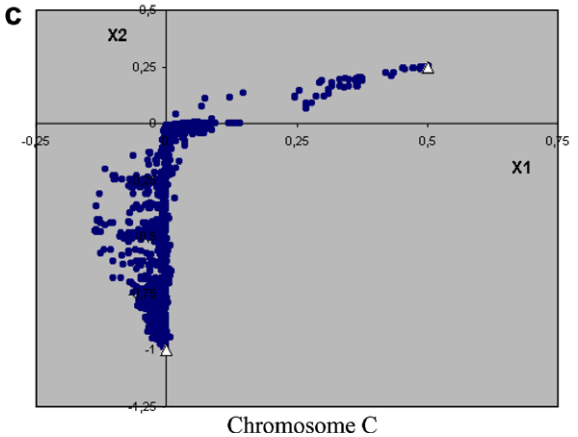
Fig. 21. Typical results, Problem 2, chromosomes A, B and C.



Chromosome A

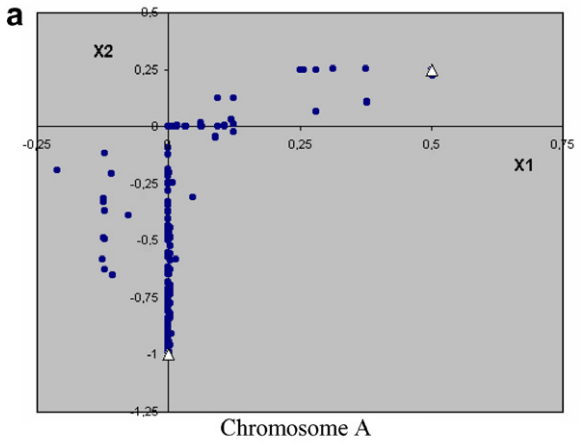


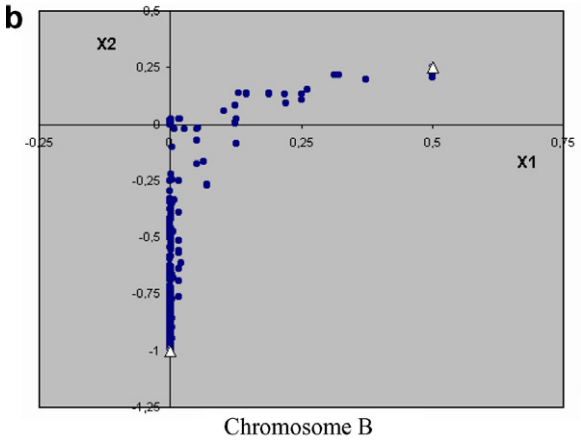
Chromosome B



Chromosome C

Fig. 22. Problem 3, chromosomes A, B and C.





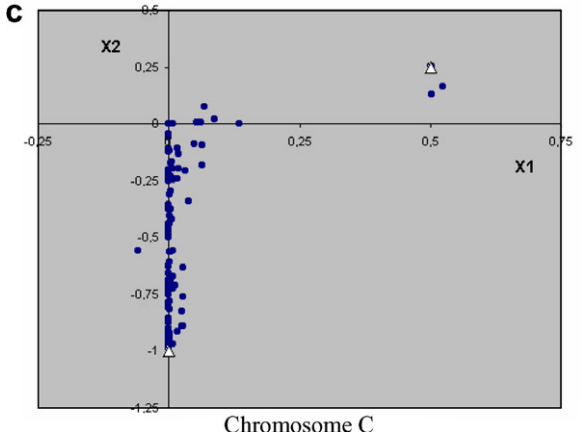


Fig. 23. Typical results, Problem 3, chromosomes A, B and C.

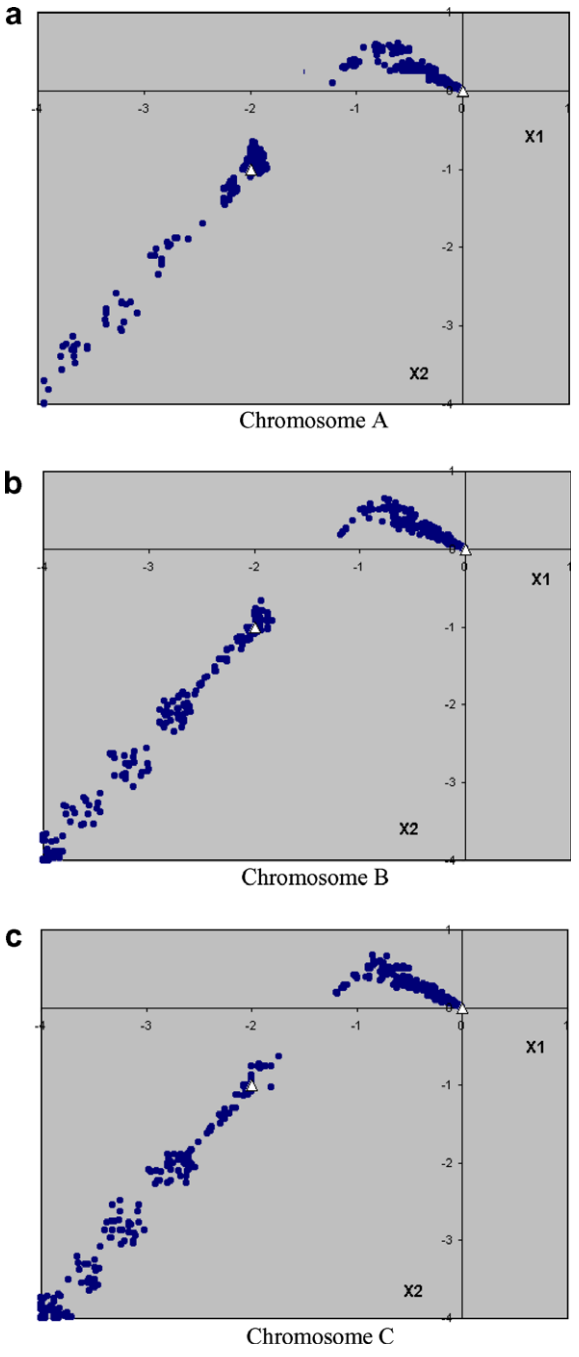


Fig. 24. Problem 4, Chromosomes A, B and C.

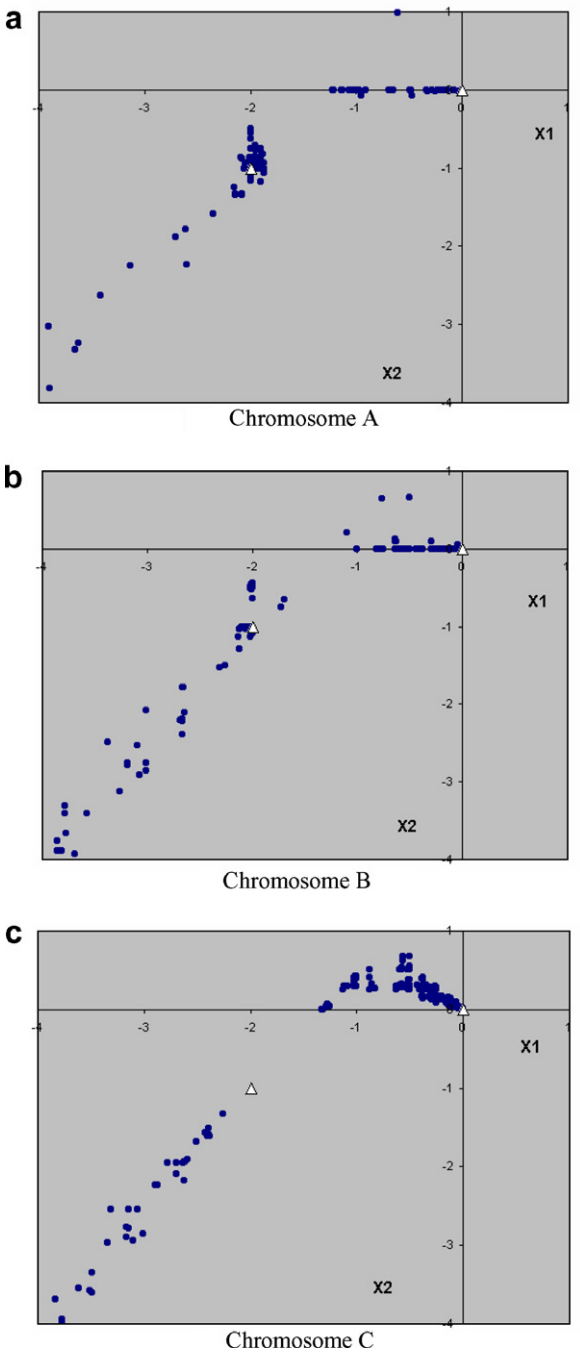


Fig. 25. Typical results, Problem 4, chromosomes A, B and C.

References

Asenjo, J. A., Montagna, J. M., Vecchietti, A. R., Iribarren, O. A., & Pinto, J. M. (2000). Strategies for the simultaneous optimization of the structure and the process variables of a protein production plant. *Computers & Chemical Engineering, 24*, 2277–2290.

Bäck, T., Hammel, U., & Schwefel, H.-P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation, 1*(1), 3–17.

Bernal Haro, L., (1999). Conception d'ateliers discontinus multi-objectifs de chimie fine par un Algorithme Génétique. Thèse de doctorat, INP ENSIGC Toulouse, France.

Bhaskar, V., Gupta, S. K., & Ray, A. K. (2000). Applications of multiobjective optimization in chemical engineering. *Reviews in Chemical Engineering, 16*(1).

Cohon, J. L. (1978). *Multiobjective programming and planning*. New York: Academic Press.

Collette, Y., Siarry, P., (2002). Optimization multiobjectif, Eyrolles. ISBN: 2-212-11168-1.

Colorni, A., Dorizo, M., & Manniezzo, V. (1991). Distributed optimization by ant colonies. In *Proceedings European Conference on Artificial Life ECAL 91* (pp. 134). New York: Elsevier Ed.

Dedieu, S., (2001). Algorithme génétique multicritère: conception et remodelage d'ateliers de chimie fine. Thèse de doctorat, INP Toulouse, France.

Dedieu, S., Azzaro-Pantel, C., Dietz, A., Pibouleau, L., Domenech, S., (2002). Procédures d'aide a la décision multicritère par algorithme génétique pour la conception d'ateliers discontinus de chimie fine. SIMO 2002 Système d'Information Modélisation, Optimization Commande en Génie des Procédés, 24–25 Octobre, Toulouse (France).

Dedieu, S., Pibouleau, L., Azzaro-Pantel, C., & Domenech, S. (2003). Design and retrofit of multiobjective batch plants via a multicriteria genetic algorithm. *Computers and Chemical Engineering, 27*, 1723–1740.

Dietz, A., Azzaro-Pantel, C., Pibouleau, L., & Domenech, S. (2006). Multiobjective optimization for multiproduct batch plant design under economic and environmental considerations. *Computers & Chemical Engineering, 30*(4), 599–613, 15 February.

Dietz, A., Azzaro-Pantel, C., Pibouleau, L., Domenech, S., (2004). Optimal design of batch plants under economic and ecological considerations: Application to a biochemical batch plant. In International Conference of Computational Methods in Sciences and Engineering 2004 (ICCMSE 2004), Greece. (19–23 November).

Dietz, A., Azzaro-Pantel, C., Davin, A., Pibouleau, L., Domenech, S., (2004). A framework for multiproduct batch plant design with environmental consideration: Application To protein production 9th International Symposium on Computer Applications in Biotechnology (CAB9) Nancy, France (28–31 March).

Dietz, A., Azzaro-Pantel, C., Pibouleau, L., Domenech, S., (2004). Integrating environmental impact minimization into batch plant design: Application To protein production ESCAPE-14: European Symposium on Computer Aided Process Engineering Lisbon, Portugal. (16–19 May).

Fonseca, C., & Fleming, P. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalisation. *Proceedings of the 5th International Conference on Genetic Algorithms*, 416–423.

Fonseca, C., & Fleming, P. (1995). An overview of evolutionary algorithms in multi-objective optimization. *Evolutionary Computation, 3*(1), 1–16.

Fonseca, C. M., & Fleming, P. J. (1998a). Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part I: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics, 28*(1), 26–37.

Fonseca, C. M., & Fleming, P. J. (1998b). Multiobjective optimization and multiple constraint handling with evolutionary algorithms—Part II: Application example. *IEEE Transactions on Systems, Man, and Cybernetics, 28*(1), 38–47.

Goldberg, D. E. (1994). *Genetic and evolutionary algorithms come of age*. ACM Press, ISSN:0001-0782.

Horn, J. (1997). F1.9 multicriteria decision making. In T. Bäck, D. B. Fogel, & Z. Michalewicz (Eds.), *Handbook of evolutionary computation*. Bristol (UK): Institute of Physics Publishing.

Horn, J., Nafpliotis, N., Goldberg, D. (1993) A niched Pareto genetic algorithm for multi-objective optimization. In Proceedings of the 1st IEEE Conference on Evolutionary Computation, 1, 82–87.

Ishibuchi, H., & Murata, T. (1996). Multi-objective genetic local search algorithm. In *Proceedings of 1996 IEEE International Conference on Evolutionary Computation (ICEC'96)* (pp. 119–124). Piscataway, NJ: IEE.

Jones, D., Mirrazavi, S., & Tamiz, M. (2002). Multi-objective meta-heuristics: An overview of the current state-of –the-art. *European Journal of Operational Research, 137*, 1–9.

Kirkpatrick, S., Gellat, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680.

Koski, J. (1984). Multicriterion optimization in structural design. In E. Atrek, R. H. Gallagher, K. M. Ragsdell, & O. C. Zienkiewicz (Eds.), *New directions in optimum structural design* (pp. 483–503). Wiley.

Kursawe, F. (1991). A variant of evolution strategies for vector optimization. In H.-P. Schwefel & R. Männer (Eds.), *Parallel problem solving from nature – Proceedings of the 1st Workshop PPSN* (pp. 193–197). Berlin: Springer.

Massebeuf, S., (2000). Optimisation multicritère de procédés discontinus d'homopolymérisation et decopolymérisationen emulsion. Thèse de doctorat, INP Lorraine, France.

Montagna, J. M., Vecchietti, A. R., Iribarren, O. A., Pinto, J. M., & Asenjom, J. A. (2000). Optimal design of protein production plants with time and size factor process models. *Biotechnology Progress, 16*, 228–237.

Pareto, V. (1896). Cours d'économie politique. Rouge, Lausanne, Switzerland.

Parks, G. T., & Miller, I. (1998). Selective breeding in a multiobjective genetic algorithm. In A. E. Eiben, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Fifth International Conference on Parallel Problem Solving from Nature (PPSN-V)* (pp. 250–259). Berlin, Germany: Springer.

Pinto, J. M., Montagna, J. M., Vecchietti, A. R., Iribarren, O. A., & Asenjo, J. A. (2001). Process performance models in the optimization of multiproduct protein production plants. *Biotechnology and Bioengineering, 74*(6), 451–465.

Ponsich, A., Azzaro-Pantel, C., Domenech, S., & Pibouleau, L. (2007). Some guidelines for genetic algorithm implementation in MINLP batch plant design problems. In Z. Michalewicz & P. Siarry (Eds.), *Advances in metaheuristics for hard optimization. Natural Computing Series*. Springer.

Schaffer, J.D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In J.J. Grefenstette (Ed.), Proceedings of an International Conference on Genetic Algorithms and Their Applications. Pittsburgh, PA, pp. 93–100. sponsored by Texas Instruments and U.S. Navy Center for Applied Research in Artificial Intelligence (NCARAI).

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithm. *Evolutionary Computation, 2*(3), 221–248.

Stefanis, S. K., Livingston, A. G., & Pistikopoulos, E. N. (1997). Environmental impact considerations in the optimal design and scheduling of batch processes. *Computers and Chemical Engineering, 21*(10), 1073–1094.

Steuer, R. E. (1986). *Multiple criteria optimization: Theory, computation, and application*. New York: Wiley.

Valenzuela-Rend'on, M., & Uresti-Charre, E. (1997). A non-generational genetic algorithm for multiobjective optimization. In T. B̈ack (Ed.), *Proceedings of the Seventh International Conference on Genetic Algorithms* (pp. 658–665). San Francisco, California: Morgan Kaufman.

Viennet, R. (1997). Nouvel outil de planification expérimentale pour l'optimisation multicritère des procédés. Thèse de doctorat, INP Lorraine, France.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82. April.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257–271.