

---

# Application of the Plant Propagation Algorithm and NSGA-II to Multiple Objective Linear Programming

Paschal Bisong Nyiam, Abdellah Salhi

Department of Mathematical Sciences, University of Essex, Colchester, United Kingdom

## Email address:

pbnymia@essex.ac.uk (Paschal Bisong Nyiam), nyiampaschal@unical.edu.ng (Paschal Bisong Nyiam), as@essex.ac.uk (Abdellah Salhi)

## To cite this article:

Paschal Bisong Nyiam, Abdellah Salhi. Application of the Plant Propagation Algorithm and NSGA-II to Multiple Objective Linear Programming. *Mathematics and Computer Science*. Vol. 8, No. 1, 2023, pp. 19-38. doi: 10.11648/j.mcs.20230801.13

**Received:** May 3, 2022; **Accepted:** May 31, 2022; **Published:** February 10, 2023

---

**Abstract:** Multiple Objective Linear Programming (MOLP) problems are usually solved by exact methods. However, nature-inspired population based stochastic algorithms such as the plant propagation algorithm are becoming more and more prominent. This paper applies the multiple objective plant propagation algorithm (MOPPA) and nondominated sorting genetic algorithm II (NSGA-II) for the first time to MOLP and compares their outcomes with those of prominent exact methods. Computational results from a collection of 51 existing MOLP instances suggests that MOPPA compares favourably with four of the most prominent exact methods namely extended multiple objective simplex algorithm (EMSA), affine scaling interior MOLP algorithm (ASIMOLP), Benson's outer-approximation algorithm (BOA) and parametric simplex algorithm (PSA), and returns best nondominated points which are of higher quality than those returned by NSGA-II. However, the nondominated points approximated by NSGA-II are evenly distributed across the nondominated front. The methods compare well with the four exact methods especially on the large instances which the exact methods failed to solve even when given generous amounts of computation times.

**Keywords:** Multiple Objective Linear Programming, Plant Propagation Algorithm, Nondominated Sorting Genetic Algorithm II, Penalty Function Method, Best Nondominated Point

---

## 1. Introduction

We stated in [60] that multiple objective linear programming (MOLP) is a branch of Multiple Criteria Decision Making (MCDM) that seeks to optimize two or more linear objective functions subject to a set of linear constraints. MOLP has been an active area of research since the 1960s because of its

relevance in practice, [27]. Indeed, many real world decision making problems involves more than one objective function and can be formulated as MOLP problems. Consequently, MOLP has been widely applied in many fields and has become a useful tool in decision making, [65]. Formally, it can be written as

$$\begin{aligned} \min \quad & c_1^T x = f_1 \\ & \vdots \\ & c_q^T x = f_q \\ \text{subject to } & x \in X = \{x \in \mathbb{R}^n : Ax = b, b \in \mathbb{R}^m, x \geq 0\}, \end{aligned} \tag{1}$$

where  $c_1, \dots, c_q$  are  $n$ -vectors containing the coefficients of the multiple objective functions,  $A$  is an  $m \times n$  constraint matrix and  $b$  is the right hand side vector.

We also noted in [60] that in practice, MOLP is typically solved by the Decision Maker (DM) with the support of the analyst who look for a most preferred (best) solution in the feasible region  $X$ . This is because optimizing all of the

objective functions simultaneously is often not possible due to their conflicting nature, [61]. Consequently, the concept of optimality is replaced with that of efficiency.

The purpose of MOLP is to obtain either all the efficient or nondominated points or a subset of either, or a most preferred point depending on the purpose for which it is needed.

In the last six decades, a number of exact methods have been introduced for solving the problem. Some of them have proven to be effective on small and medium scale MOLP instances.

Real life MOLP problems are difficult to solve. In a worst-case situation all vertices might be efficient, meaning that the problem would be intractable as there might be exponentially many efficient vertices, [47]. It is, therefore, clear that MOLP is intractable in the worst-case. Moreover, looking at [46] which shows that listing all vertices of a polyhedron is NP-hard, one can deduce that MOLP is also NP-hard since in the worst-case scenario all vertices must be found to determine the efficient ones. Thus, exact methods are sometimes inefficient and costly, especially when the problem size is large. Instead of finding efficient and nondominated points, heuristics generally find good approximations or near efficient solutions in acceptable computational times. For this reason, they are widely used in multi-objective optimisation (MOO). Given that finding all efficient solutions of MOLP is NP-hard since it is equivalent to enumerating all vertices of the feasible region [13], it is astonishing to note that only one approximate method, namely NSGA [78], has been applied to it [14]. It is worth investigating these methods as an alternative solution approach to the problem.

This paper applies nature-inspired population based stochastic algorithms namely multiple objective strawberry plant propagation algorithm (MOPPA) [33] and nondominated sorting genetic algorithm II (NSGA-II) [19, 20] which were originally meant to solve multiple objective non-linear programming problems to solve MOLP problems using the penalty function method to handle general constraints. The paper then compares the outcomes of these two approximate methods with those of EMSA [59], ASIMOLP [5], BOA [10] and PSA [68] which are prominent exact methods. Note that EMSA [59] is an extension of the multiple objective simplex algorithm of Evans and Steuer [31] to generate the set of all nondominated points of the problem. In order to apply MOPPA and NSGA-II to MOLP, we will reformulate (1) to penalize each of the objective function for any constraint that is violated, and to achieve the comparison we shall compute a Most Preferred Nondominated Point (MPNP) whose components are as close as possible to an unattainable ideal objective point from the nondominated set returned by exact methods to compare with the Best Nondominated Point (BNP) returned by MOPPA and NSGA-II.

To the best of our knowledge, no application of MOPPA and NSGA-II to MOLP or their comparison with exact methods has been conducted before. We intend to fill this gap here.

This paper is organized as follows. Section 2 introduces MOLP and basic notation. Section 3 is a brief review of the relevant literature. Section 4 discusses heuristic approaches to multi-objective optimisation. We present the strawberry

plant in Section 5. Section 6 presents the basic plant propagation algorithm. The solution procedure is presented in Section 7. We illustrated MOPPA and NSGA-II in Section 8. Section 9 discusses the determination of MPNPs in exact methods. Section 10 presents experimental results obtained with approximate and exact methods. Summary of results is presented in Section 11. Finally, a conclusion is presented in Section 12.

## 2. Notation and Definitions

We also stated in [60] that an alternative and compact formulation of (1) is as follows

$$\begin{aligned} \min \quad & Cx \\ \text{subject to} \quad & Ax = b \\ & x \geq 0, \end{aligned} \tag{2}$$

where  $C$  is a  $q \times n$  criterion matrix consisting of the rows  $c_k^T$ ,  $k = 1, 2, \dots, q$ ,  $A$  and  $b$  are as described earlier. The feasible set in the decision space is  $X = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$  and in the objective space it is  $Y = \{Cx : x \in X\}$ . The set  $Y$  is also referred to as the image of  $X$ , [58]

A nondominated point in the objective space is the image of an efficient solution in the decision space; nondominated points form the nondominated set, [60].

An efficient solution of an MOLP problem is a solution that cannot improve any of the objective functions without deteriorating at least one of the other objectives. A weakly efficient solution is one that cannot improve all the objective functions simultaneously, [60]. Let  $\hat{x} \in X$  be a feasible solution of (2) and let  $\hat{y} = C\hat{x}$ :

1.  $\hat{x}$  is called efficient if there is no  $x \in X$  such that  $Cx \leq C\hat{x}$  and  $Cx \neq C\hat{x}$ ; correspondingly,  $\hat{y} = C\hat{x}$  is called nondominated.
2.  $\hat{x}$  is called weakly efficient if there is no  $x \in X$  such that  $Cx < C\hat{x}$ ; and  $\hat{y} = C\hat{x}$  is called weakly nondominated [26].

The set of all efficient solutions and the set of all weakly efficient solutions of (2) are denoted by  $X_E$  and  $X_{WE}$  respectively. The sets  $Y_N = \{Cx : x \in X_E\}$  and  $Y_{WN} = \{Cx : x \in X_{WE}\}$  are the nondominated and weakly nondominated sets in the objective space of (2) respectively, [60]. The nondominated faces in the objective space of a given problem constitutes the nondominated frontier and the efficient faces in the decision space of the problem constitutes the efficient frontier. The ideal objective point  $y^*$  is the minimum criterion values over the efficient set  $X_E$ . The ideal objective values are easy to obtain by simply minimizing each objective function individually over the feasible region  $X$  [2].

## 3. Literature Review

Several exact methods have been introduced in the last six decades for computing either the entire efficient decision set

$X_E$  or the nondominated set  $Y_N$  or a subset thereof, or a most preferred solution to the problem.

We stated in [60] that Eiselt and Sandblom [29] note that, Evans and Steuer [31], Philip [63], and Zeleny [90] derived generalized versions of the simplex method known as MSA. That of Philip [63] first determines if an extreme point is efficient and subsequently checks if it is the only one that exists. If not, the algorithm finds them all. This MSA approach, however, may fail at a degenerate vertex. In Philip [64], it was modified to overcome this difficulty, [60].

The MSA of Evans and Steuer [31] also generates all the efficient extreme points and unbounded efficient edges of MOLPs; see also Algorithm 7.1, on page 178 of Ehrgott [26]. The algorithm first establishes that the problem is feasible and has efficient solutions. Thereafter, it generates all of them. An LP test problem is solved to determine the pivots that lead to efficient vertices and the algorithm is implemented as a software called ADBASE in [80].

As was also noted in [60] that the MSA variant of Zeleny [90] also uses an LP test problem to determine the efficiency of extreme points. But, here, vertices are tested for efficiency after they have been obtained unlike in [31] where the test problem determines pivots leading to efficient vertices.

In [85, 87] Yu and Zeleny used the method in [90] to generate the set of all efficient solutions and presented a formal procedure for testing the efficiency of extreme points. The efficient extreme points are derived from the efficient faces, in a top-to-down search strategy. Numerical examples using problems with three objectives were used to demonstrate the effectiveness of the method. In a similar paper, Yu and Zeleny [86] applied their approach expanded in [87] to parametric linear programming. Two basic forms of the problem and two computational procedures for computing the efficient set were presented: the direct decomposition of the parametric space into subspaces associated with extreme points and the indirect algebraic method. Numerical illustration show that, the indirect algebraic approach is superior to the direct decomposition.

We also noted in [60] that Isermann [41] proposed a variant of the MSA of Evans and Steuer [31] that solves fewer LPs when determining the entering variables. The algorithm first establishes whether an efficient solution for the problem exists, and solves a test problem to determine pivots leading to efficient vertices. It was implemented as a software called EFFACET in [40].

In [35], Gal presented an MSA that computes the set of all efficient solutions and higher-dimensional faces of the problem. This method is meant to address the problem of determining efficient faces and higher dimensional faces not resolved in [31] and [63]. Here, efficient solutions are computed using a test problem. The algorithm also determines higher-dimensional efficient faces for degenerate problems which were only discussed in [41] and [90] but were not solved. The efficient faces are generated in a bottom-to-top search strategy as we noted in [60].

Steuer [79] used the MSA of Evans and Steuer [31] to solve parametric and non-parametric MOLP problems. Different

methods for obtaining an initial efficient extreme point as well as different LP test problems were also presented. Efficient extreme points are generated through the decomposition of the weight space into finite subsets that provide optimal weights corresponding to extreme point solutions, [60]. Similarly, Ehrgott [26] also used the MSA of Evans and Steuer [31] to solve MOLP problem instances with two and three objective functions, [58].

In [24] was proposed a variation on the algorithm of Evans and Steuer [31]. The authors noted that algorithms usually start from an initial efficient extreme point and moved to an adjacent one following the solution of an LP problem. The proposed method does not require the solution of any LP problem to test for the efficiency of extreme points and the feasible region needs not be bounded. The algorithm enumerates all efficient extreme points and appears to have computational advantage over other methods.

We stated in [60] that Ecker *et al.* [23] presented yet another variant of MSA. The algorithm first determines the maximal efficient faces incident to a given efficient vertex (i.e. containing the efficient vertex) and ensures that previously generated efficient faces are not regenerated. This is done following a bottom-to-top search strategy as in [35], which dramatically improves computation time. The proposed approach was illustrated with a degenerate example given in [87], to demonstrate its applicability. It was computationally more efficient than the method in [87].

We also noted in [60] that the MSA of Armand and Malivert [7] determines the set of efficient extreme points even for degenerate MOLPs. The approach follows a bottom-to-top search strategy and utilizes a lexicographic selection rule to choose the leaving variables which proves effective when solving degenerate problems. It was tested successfully on a number of degenerate problems. A numerical example with five objectives and eight constraints which was solved in [87] was also used to demonstrate its effectiveness. The proposed MSA was superior to that in [87].

In [58], we stated that a modification of the PSA for single objective LP to solve bounded bicriterion LP problems was presented in [69]. The approach was applied to a large mean-risk portfolio optimization problem for which the nondominated portfolios were generated.

Ehrgott *et al.* [25] presented a primal-dual simplex algorithm for bounded MOLP problems. This algorithm finds a subset of efficient solutions that are enough to generate the whole efficient frontier. The algorithm starts with a coarse partitioning of the weight space which continues in each iteration as well as solves an expensive LP in each iteration. A vertex enumeration is then performed in the last step to obtain efficient solutions. Numerical illustrations show the applicability of the algorithm.

In [68] Rudloff *et al.* introduced a PSA for the problem. The algorithm is a generalization of the algorithm in [69] and is similar to that of Ehrgott *et al.* [25]. It works for any dimension, solves bounded and unbounded problems (unlike that in [25] and [69]), and does not find all the efficient solutions just like that in [25]. Instead, it finds a solution

based on the idea of Löhne [49], i.e. a subset of efficient extreme points and directions that allows to generate the whole efficient frontier. This is the so called PSA. The algorithm does pivoting for just one leaving variable among the set of all possible leaving variables. It was compared with a version of BOA in [37] and MSA of Evans and Steuer [31] using small MOLP instances which were randomly generated with 3 and 4 objectives and up to 50 variables and constraints. The numerical results show that the proposed algorithm is superior to BOA for non-degenerate problems. However, BOA is better for highly degenerate problems. PSA was also found to be computationally more efficient than the algorithm of Evans and Steuer [31].

In [59], we presented an extension of the MSA of Evans and Steuer [31] to generate the set of all nondominated points of the problem and no redundant ones. This extended version was compared to the primal variant of BOA [10] that also computes the set of all nondominated points of the problem. Numerical results on a collection of 56 existing MOLP instances show that the total number of nondominated points returned by EMSA is the same as that returned by BOA for most of the problems considered.

Of all the MSA variants discussed so far, it was noted in [73] that, the MSA of Evans and Steuer [31] is the most popular and successful for computing all efficient extreme points of the problem.

We noted in [60] that MSA and its variants make explicit use of the vertices of the feasible region while interior-point approaches on the other hand, generate iterates in the interior of the feasible region. Various such approaches have been suggested for the problem. The difference between them depends on the methodology employed to assess the suitability of points used to derive a combined search direction along which one heads towards the next iterate, [60].

As we also stated in [60] that the first to adapt a variant of Karmarka [45] interior-point algorithm, to solve MOLP seems to be Abhyankar [1]. The author relies on the method of centers and utilizes a parameterization of ellipsoids in the  $n$ -dimensional space to approximate the efficient frontier of the problem in polynomial time.

In [4], Arbel modified and adapted a variant of Karmarka [45] algorithm resulting in the so called Affine Scaling Interior MOLP (ASIMOLP) algorithm. He used the convex combination of individual directions to derive a combined direction along which to step toward the next iterate. Specifically, the algorithm generates step direction vectors based on the objectives of the problem. The relative preference of these directions is then assessed using a utility (or preference) function to obtain the points used in combining them into a single direction vector that moves the current iterate to a new one. The process is repeated until the algorithm converges to a most preferred efficient solution after meeting some termination conditions, [60].

In [5] was proposed yet another ASIMOLP algorithm. This approach offers another means of assessing preference information to establish a combined search direction rather than using the DM's utility function. The Analytic Hierarchy

Process (AHP) developed in [70] was applied to obtain the relative preference of points used to derive a combined direction along which the next step is taken. It is based on the assessed preferences to weigh the step direction vectors for each of the objectives in order to derive a combined step direction vector. This process continues to generate search directions and new feasible points at each iteration, until the algorithm converges to a most preferred point on the efficient frontier, [60].

We noted in [60] that another ASIMOLP algorithm based on the AHP was introduced in [6]. Here, the derived preference information is applied to the projected gradients in order to obtain anchoring points and cones used in searching for a most preferred solution. The boundaries of the constraints polytope are constantly probed to make more directions available, which enables one to arrive at a most preferred solution.

Wen and Weng [82] modified the ASIMOLP algorithm in [4] in order to resolve zigzagging issues. However, the modified algorithm may not yield a most preferred efficient solution.

Lin *et al.* [48] also proposed a modification of the algorithm in [4]. They adopted the utility function trade-off method to weigh the objective functions involved and compared the modified algorithm with that in [82] and the simplex method. Numerical experiments show that their algorithm is superior. On computing efficiency, the interior point based algorithms outperform the simplex-based ones on large scale problems, [60].

In [83], Weng and Wen introduced yet another ASIMOLP based algorithm. The proposed algorithm computes a weighted sum of the different search directions involved using a utility function. These search directions are then normalized with the weights to obtain a combined direction that moves the current solution to an anchor point. Computational experiments show that the proposed algorithm is suitable for solving large scale instances.

We noted in [58] and [60] that, due to the various difficulties arising from solving MOLP problems in the decision space (such as having different efficient solutions that map onto the same point in the objective space), efforts were made to look at the possibility of solving them in the objective space. See also Dauer [16], Dauer and Liu [17], Dauer and Saleh [18] and Benson [10].

In [59] we stated that, Benson [10] presented a detailed account of decision set based methods and proposed an algorithm for generating the set of all nondominated points in the objective space. This is the so called BOA. According to him, this algorithm is the first of its kind. Computational results suggest that the objective space based methods are better than the decision space based ones. A further analysis of the objective space based method for the problem was presented in [11]. Here, the outer approximation algorithm also generates the set of all weakly nondominated points, thereby enhancing the usefulness of the algorithm as a decision aid, [60].

Benson [12] suggested yet another algorithm for solving the problem in the objective space. This time a hybrid method

that partitions the objective space into simplices that lie in each face so as to compute the set of nondominated points. This idea was earlier presented in [9]. The algorithm is quite similar to that in [10]. The difference between them is in the manner in which the nondominated vertices are found. While a vertex enumeration procedure is employed in [10], a simplicial partitioning technique is used in the latter, [60].

In [75], Shao and Ehrgott modified the algorithm of Benson [10]. While in [10], a bisection method that requires the solution of several LPs in one step is required, here, solving just one LP achieves the desired result and in the process improves computation time. Again in [76], Shao and Ehrgott introduced an approximate dual variant of the algorithm of Benson [10] for obtaining approximate nondominated points of the problem. The proposed algorithm was applied to the beam intensity optimization problem of radio therapy treatment planning for which approximate nondominated points were generated. Numerical application suggests that the method is faster than solving the primal directly.

We noted in [60] that the explicit form of the algorithm of Benson [10] as modified by Shao and Ehrgott [75] is presented in [49]. This version solves two LPs in each iteration during the process of obtaining the nondominated extreme points. In [50] Löhne presented a Matlab implementation of this modified version called BENSOLVE-1.2, for computing the entire set of nondominated points and directions (unbounded nondominated edges) of the problem.

Csirmaz [15] also modified and introduced an improved version of the algorithm of Benson [10] that solves one LP and a vertex enumeration problem in each iteration. While in Benson [10], solving two LPs to determine a unique boundary point and a supporting hyperplane of the image is required in two steps of the algorithm, here, the two steps are merged into one and solving just one LP does both tasks, thereby improving computation time. The algorithm was used to generate all the nondominated vertices of the polytope defined by a set of Shannon inequalities on four random variables so as to map their entropy region. Numerical results show the applicability of the method to medium and large instances, [60].

In [37], Hamel *et al.* presented new variants and extensions of the algorithm of Benson [10]. The primal and dual variants of the algorithm that solves only one LP problem in each iteration. Numerical testing reveal a reduction in computation time.

We noted in [58] that Löhne *et al.* [51] also extended the primal and dual variants of the algorithm of Benson [10] to approximately solve convex vector optimization problems in the objective space.

More recently Dörfler, D. *et al.* [22] presented an algorithm for approximately solving bounded vector optimization problems in the objective space. The proposed algorithm is an improvement and a modification of the algorithm in [37] where a new selection rule for vertex enumeration is presented to improve the over all efficiency of the algorithm. Numerical illustrations suggests that the new algorithm may be faster than that in [37].

Having discussed exact methods to the problem, we now

turn our attention to heuristics or approximate approaches that seek to find good approximations or near efficient or nondominated points in acceptable computational times. As stated earlier, heuristics or approximate methods have been commonly applied to nonlinear and discrete multi-objective optimisation and not so much to MOLP.

In [14], Chakraborty and Ray applied multi-objective parametric fuzzy programming and NSGA [78] to MOLP transportation problem. Here, the MOLP problem is transformed into a single objective parametric problem with interval parameters. Numerical illustration using a coal energy resource allocation problem show the applicability of the method. NSGA has been widely applied in different discrete and continuous multi-objective optimisation problems. In [8], Bagchi applied NSGA to several scheduling problems for which efficient solutions were found. For extensive applications of NSGA to chemical engineering problems, see [57]. NSGA-II [19, 20] which is an improved version of NSGA [78] and arguably the most popular in the context of nonlinear multi-objective problems has also had tremendous applications in different nonlinear multi-objective optimisation. It was successfully applied in the energy generation expansion planning problem in [44] for which the minimum investment and outage costs were approximated. Similarly, Hu *et al.* [39] also applied NSGA-II to solve a real-life combined gas and electricity network expansion planning problem in Hainan province (China) with an aim of minimizing investment, production and carbon emission costs. The problem was formulated as a bicriterion nonlinear multi-objective optimisation problem and solved using NSGA-II for which the nondominated front was determined. In [55], Massobrio *et al.* applied NSGA-II to the taxi sharing problem in order to determine the minimum cost of journey and delay time by passengers from the same location to different destinations. The problem was formulated as a bicriteria multi-objective optimisation problem and solved with NSGA-II and greedy heuristics. Numerical results show that NSGA-II outperform the greedy algorithms by achieving significant improvements in both objectives in acceptable computational time. Recently, NSGA-II was applied in the communication industry to solve the spectrum assignment problem in [54]. Here, the spectrum assignment problem was also formulated as a bicriterion multi-objective optimisation problem and solved using NSGA-II. It was observed from the results obtained that there is an improvement in throughput at the cost of spectral efficiency which offers useful guidelines to the service provider to maintain customer satisfaction in the spectrum sharing network. In [21], a modification of NSGA-II [19, 20] was presented and applied to solve the combined economic and emission dispatch problem. The authors noted, however, that NSGA-II ensures diversity along the nondominated front using the concept of crowding distance, but lateral diversity may be lost due to the lack of diversity in a particular decision variable which may push the search towards the nondominated front. The modified version is aimed at resolving this issue by incorporating controlled elitism into NSGA-II and replacing the crowding distance operator with a dynamic version which

appears to solve the problem.

Salhi and Fraga [71] presented a plant propagation algorithm (PPA) to solve the survival optimisation problem. This is the so called PPA. The algorithm emulates the way plants and in particular, the strawberry plant propagate by sending many short runners when they are in good spots and sending fewer but longer runners to explore the environment when they are in not so good spots. It was tested on a complex nonlinear process design problem and compared with the Nelder-Mead algorithm. Experimental results show the effectiveness of the proposed method and it performs significantly better than the Nelder-Mead search method. In [33], Fraga and Amusat extended the PPA in [71] to solve multi-objective nonlinear programming problems. A novel fitness function that emphasizes the end-points is introduced into the extended version and applied to the integrated energy systems design for off-grid mining operations problem for which good designs that achieve the desired objectives were approximated. Recently, Rodman *et al.* [67] applied the extended multi-objective PPA (MOPPA) to the industrial beer fermentation process in order to minimize the production time and maximize ethanol production. The problem was modelled as a bicriteria nonlinear dynamic optimisation problem and solved with MOPPA. Numerical results show the effectiveness of MOPPA in solving complex multi-objective optimisation problems. Some well-known heuristic methods to multi-objective optimisation will be discussed in the following section.

## 4. Heuristic Approaches to Multi-objective Optimisation

### 4.1. Genetic Algorithm

The Genetic Algorithm (GA) was developed by Holland in 1975, [38]. It is based on the idea of natural selection. The algorithm works with three operators which are referred to as genetic operators namely; crossover, mutation and reproduction.

#### 4.1.1. Crossover

The crossover operator selects a random point which shows a position on the individual. Then, parts of two selected individuals are exchanged to generate two new individuals. This procedure is called a single-point crossover. Another type is called two-point crossover. In this variant, two random positions are selected and parts of parents are exchanged.

#### 4.1.2. Mutation

A predetermined number of individuals are mutated. This is done by changing/flipping some of the entries of an individual. This operator helps exploration of the search space.

#### 4.1.3. Reproduction

This copies good individuals into the new population as they are.

It was noted in [74] that to implement GA the following are needed:

1. *Initial Population*: A predetermined number of individuals is randomly generated to form an initial population. The basic GA starts with this population.
2. *Fitness Function*: This measure is essential for the implementation of GA. It allows to rank individual solutions in the population. It is often the objective function of the optimisation problem.
3. *Selection of Parents*: The main idea of selection is choosing individuals from the population to be parents to new individuals. The latter are expected to be better than the parents. There are different selection methods such as the Roulette Wheel and Tournament Selection, [66].

### 4.1.4. Stopping Criteria

The algorithm stops when the number of generations reaches a predetermined maximum number of generations. Another commonly used stopping criterion is the maximum number of generations without improvement in the current solutions, [38, 74].

### 4.2. Vector Evaluated Genetic Algorithm (VEGA)

VEGA is the first population-based evolutionary multi-objective genetic algorithm (MOGA) applied to multi-objective optimisation problems. It was introduced by Schaffer [72]. Here, the population is divided randomly into equal sub-populations at each iteration. Fitness values are assigned to all the solutions in a sub-population based on one of the objective functions and each objective is used to evaluate members in the population. In each sub-population, a fitness proportionate selection is done and the selected members are used for procreation. The process is repeated until convergence is achieved.

### 4.3. Multi-Objective Genetic Algorithm (MOGA)

MOGA is the first population-based evolutionary algorithm that uses the nondominated classification of the population. In MOGA, each solution is checked for its domination in the population and a rank  $i$ , equal to  $n_i$  the number of solutions that dominates solution  $i$ , is assigned to it. To ensure that diversity is achieved, the algorithm uses a sharing function model, [36].

### 4.4. Nondominated Sorting Genetic Algorithm (NSGA)

NSGA is one of the multi-objective evolutionary algorithms (MOEA) which has the capacity to find nondominated points in a single run. It was introduced by Srinivas and Deb [78]. In NSGA the population is sorted according to nondomination and classified into a number of fronts ( $F_1, F_2, \dots, F_n$ ). Using niching and nondominated sorting of solutions in every generation, the good solutions are selected for procreation. The algorithm also uses a sharing function model to ensure diversity. Its main issues are: It requires the potential user to

specify the sharing parameter, which is difficult for the user to determine the ideal value; the nondominated sorting technique is time consuming and computationally expensive; it lacks elitism, which may be important in preventing the loss of good solutions once they are found, [88].

#### 4.5. Nondominated Sorting Genetic Algorithm II (NSGA-II)

NSGA-II [19, 20] is an improved version of NSGA [78]. Though NSGA enjoyed patronage in the multi-objective evolutionary community, it was also widely criticized for the above three issues (lack of elitism, high computational cost of nondominated sorting and the requirement for specifying the sharing parameter). The NSGA-II succeeded in solving all the three issues at once by introducing a fast nondominated sorting and tournament selection using the concept of crowding

distance, [62]. In NSGA-II, in addition to the genetic operators of crossover and mutation, two new specialized multi-objective operators or mechanisms have been proposed to solve the above three issues:

1. *Nondominated Sorting*: NSGA-II employs a fast nondominated sorting that is aimed at reducing the complexity of sorting as compared to that used in NSGA.
2. *The Crowding Distance*: It is a technique to replace the sharing parameter that was needed in the old version. This approach involves ranking among members of a front those that are dominating or being dominated by each other.

These two procedures are used together with the genetic selection operators to create the population of the next generation. The pseudo-code of NSGA-II adapted from [88] is given as Algorithm 1.

---

#### Algorithm 1 Nondominated Sorting Genetic Algorithm II [88]

---

```

1: Initialization:
2:   ◇ Generate random population
3:   ◇ Evaluate objective values
4:   ◇ Assign rank (level) based on nondomination
5:   ◇ Generate child population
      - Tournament selection
      - Crossover and mutation
  For  $i = 1$  to number of generations
6:   ◇ Parent and child population are assigned rank based on nondomination
7:   ◇ Generate sets of nondominated fronts
8:   ◇ Determine the crowding distance between points on each front
9:   ◇ Select points based on crowding distance calculation and fill into the parent population until full
10:  ◇ Create next generation
11:  ◇ Tournament Selection
12:  ◇ Crossover and Mutation
13:  ◇ Evaluate Objective Values
14:  ◇ Increment generation index
  End

```

---

Among all the above mentioned MOEAs, NSGA-II is the most popular and known for its capacity to promote the quality of solutions, [43]. There are new nature-inspired population based stochastic algorithms which have shown a lot of promise on nonlinear single and multi-objective optimisation problems. One such algorithm is the so called plant propagation algorithm or PPA. It emulates the way plants and in particular the strawberry plant propagate, [71]. The details of PPA will be provided in Section 6 where it is investigated and used to solve MOLP.

Based on our extensive review of the topic, it was observed that no application of MOPPA and NSGA-II to MOLP has been carried out before, and no comparison of a MPNP chosen from exact methods with the BNP returned by MOPPA and NSGA-II has been carried out. We intend to fill this gaps here.

## 5. The Strawberry Plant

The strawberry plant (*Fragaria Xananassa*) belongs to the Rose family. The strawberry-growing industry started in Paris in the seventeenth century with the European variety. In 1714, Amedee-Francois Frezier, a mathematician and engineer, hired by Louise XIV [30] to draw maps of South America returned from Chile with some Chilean strawberry plants which give a larger fruit. Subsequent crossings with the European variety and selections led to the modern plant, [71].

Looking at mature strawberry plants, one will observe after a period of time, a concentration of younger plants around strong and well-established ones; that is the plants send many short runners as they are in good spots. Plants that are not well-established and are not looking very strong, send few but longer runners to explore the environment in search of better spots with enough water, nutrients and sunlight. These

basic principles are behind the design of PPA and subsequently MOPPA.

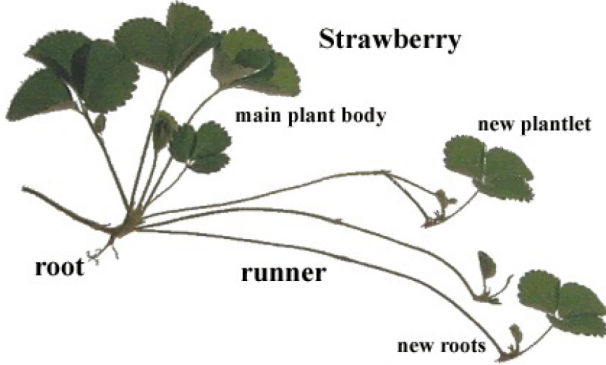


Figure 1. The strawberry plant (*Fragaria Xananassa*).

## 6. The Basic Plant Propagation Algorithm

The Plant Propagation Algorithm (PPA) introduced by Salhi and Fraga [71] emulates the way plants and in particular the strawberry plant propagate. That is, it emulates the strategy that plants deploy to survive by colonising new places which have good conditions for growth. Plants, like animals, survive by overcoming adverse conditions using often basic but effective strategies. The strawberry plant, for instance, has a survival and expansion strategy which is to send short runners to exploit the local area if the latter has good conditions (with enough water, nutrients and light), and to send long runners to explore new and more remote areas, that is to run away from a not so favourable current area (with poor water supply, nutrients and light). PPA in its multi-objective version has not been applied to MOLP, yet. We intend to do that here. MOPPA [33] is the implemented version. The mechanism of the basic PPA is described below, [74, 81].

---

### Algorithm 2 The Plant Propagation Algorithm (PPA) [71]

---

```

1: Generate a population  $P = X_i, i = 1, \dots, NP$  of plants;
2:  $g \leftarrow 1$ 
3: for  $g = 1 : g_{\max}$ 
4:   Compute  $N_i = f(X_i), \forall X_i \in P$ 
5:   Sort  $P$  in ascending order of fitness values  $N$  (for minimization);
6:   Create new population  $\phi$ 
7:   for each  $X_i, i = 1, \dots, NP$ 
8:      $r_i \leftarrow$  set of runners where both the size of the set and the distance for each runner (individually) are proportional to the fitness values  $\phi \leftarrow \phi \cup r_i$  (append to population);
10:  endfor
11:   $P \leftarrow \phi$  (new population);
12: endfor
13: Return  $P$ , (the population of solutions).
```

---

The algorithm starts with a population of plants each of which represents a solution in the search space.  $X_i$  denotes the solution represented by plant  $i$  in an  $n$ -dimensional space.  $X_i \in R^n$ , i.e.  $X_i = [x_{ij}]$ , for  $j = 1, \dots, n$  and  $x_{ij} \in R$ .  $NP$  is the population size. This iterative process stops when  $g$  the counter of generations reaches its given maximum value  $g_{\max}$ . Individuals/plants/solutions are evaluated and then ranked (sorted in ascending or descending order) according to their objective (fitness) values and whether the problem is a min or a max problem. The number of runners of a plant is proportional to its objective value and conversely, the length of each runner is inversely proportional to the objective value, [71]. For each  $X_i$ ,  $N_i \in (0, 1)$  denotes the normalized objective function value space. The number of runners for each plant to generate is

$$n_r^i = \lceil (n_{\max} N_i \beta_i) \rceil \quad (3)$$

where  $n_r^i$  shows the number of runners and  $\beta_i \in (0, 1)$  is a randomly picked number. For each plant, the minimum

number of runners is set to 1. The distance value found for each runner is denoted by  $dx_j^i$ . It is:

$$dx_j^i = 2(1 - N_i)(r - 0.5), \text{ for } j = 1, \dots, n. \quad (4)$$

where  $r \in [0, 1]$  is a randomly chosen value. Calculated distance values are used to position the new plants as follows:

$$y_{ij} = x_{ij} + (b_j - a_j) dx_j^i, \text{ for } j = 1, \dots, n. \quad (5)$$

where  $y_{ij}$  shows the position of the new plant and  $[a_j, b_j]$  are the bounds of the search space. If the bounds of the search domain are violated, the point is adjusted to be within the domain  $[a_j, b_j]$ . The new population that is created by appending the new solutions to the current population is sorted. In order to keep the number of population constant, the solutions that have lower objective value are dropped, [74]. The algorithm was originally designed for single-objective nonlinear optimisation problems. It has been successfully tested on single-objective and bicriteria



continuous optimisation problems in [71]. It has also been applied successfully to a single-objective dynamic optimisation problem in the built environment [34]. Recently, the algorithm was equipped with a new fitness function that emphasizes the end-points and extended to multi-objective nonlinear programming problems in [33]. This version was successfully applied to the integrated energy systems design for off-grid mining operations which is a bicriteria dynamic

optimisation problem. Most recently, the algorithm was also applied to the industrial beer fermentation process in [67] for which the nondominated front was successfully approximated. Given the successes of the algorithm recorded so far for single and multi-objective nonlinear programming problems, we intend to apply MOPPA to MOLP problems. The pseudo-code of MOPPA adapted from [33] is given as Algorithm 3.

---

**Algorithm 3 The Multi-Objective Plant Propagation Algorithm, [33]**


---

**0: Given:**  $f(x)$ , a vector function;  $n_g$ , number of generations to perform;  $n_p$ , the propagation size;  $n_r$ , maximum number of runners to propagate.  
**1: Output:**  $z$ , vector approximation to Nondominated frontier.  
**2**     $p \leftarrow$  initial random population of size  $n_p$   
**3**    **for**  $n_g$  generations **do**  
**4**     prune population  $p$ , removing similar solutions  
**5**      $N \leftarrow$  fitness( $p$ )                       $\triangleright$  Use rank based fitness  
**6**      $\bar{p} \leftarrow \emptyset$                                $\triangleright$  Empty set  
**7:**     **for**  $i \leftarrow 0 \dots n_p$  **do**  
**8:**        $x \leftarrow$  select( $p, N$ )                       $\triangleright$  Tournament fitness based selection  
**9:**       **for** each runner to generate **do**                       $\triangleright$  Number proportional to fitness rounded up  
**10:**           $\bar{x} \leftarrow$  new solution( $x, 1 - N$ )                       $\triangleright$  Distance inversely proportional to fitness  
**11:**           $\bar{p} \leftarrow \bar{x} \cup \bar{p}$                                $\triangleright$  Add to new population  
**12:**       **endfor**  
**13:**        $p \leftarrow p \setminus x$                                $\triangleright$  Remove from old population  
**14:**       **endfor**  
**15:**        $p \leftarrow \bar{p} \cup \text{Nondominated}(p)$                        $\triangleright$  New population with elitism  
**16:**    **endfor**  
**17:**     $z \leftarrow \text{Nondominated}(p)$

---

## 7. Solution Procedure

In order to apply MOPPA to MOLP, we use the penalty function method [56] to handle the constraints. The penalty function method is the most popular constraint handling technique in evolutionary algorithms and many other optimisation frameworks [56, 84]. It penalizes each objective

or fitness function by reducing its fitness values in proportion to the degree of constraint violation [77]. In other words, a penalty term is added to each of the objective function penalizing the function values that are not in the feasible region. To use this method, MOLP problem (1) is reformulated as follows:

$$\begin{aligned}
 \min \quad & f_1(x) + Kp_1(x) \\
 & \vdots \\
 & f_q(x) + Kp_q(x) \\
 \text{subject to } & x \in X = \{[a, b]^n \subset \mathbb{R}^n : g_j(x) \leq 0, j = 1, \dots, m\},
 \end{aligned} \tag{6}$$

where  $X$  is the search space or feasible region which is described by box constraints,  $a$  and  $b$  are the lower and upper bounds on all variables, the scalar quantity  $K$  is a constant which is called the penalty parameter and the function  $p_k(x)$ ,  $k = 1, \dots, q$  is the penalty function. Equation (6) is now our new MOLP penalty program.

The penalty function  $p_k(x)$  satisfies the following

1.  $p_k(x) = 0$ , if  $g_j(x) \leq 0$
2.  $p_k(x) > 0$ , if  $g_j(x) \not\leq 0$ ,

that is to say, the penalty function is zero if no violation of the constraint occurs and is positive if a constraint is violated; the penalty parameter term would be added to the objective function such that the solution is pushed back towards the feasible region. A large penalty value prevents searching the

infeasible region and enables the method to converge to a feasible solution quickly, [84].

## 8. Illustration of MOPPA

We consider the following MOLP adapted from [42].

We implemented the MOLP penalty program 6 in Matlab and applied a Matlab implementation of MOPPA which can be found in [32] to solve Problem 7. With  $x_1 \in [0, 7]$ ,  $x_2 \in [0, 5]$  as variable bounds, a population size of 50, maximum number of runners 5 and the number of generations to perform at 200 are chosen. The nondominated front approximated by the algorithm is shown in Figure 1.

$$\begin{aligned}
\min f_1 &= -x_1 \\
\min f_2 &= -x_2 \\
\text{Subject to} \quad & 6x_1 + 10x_2 \leq 60 \\
& x_1 \leq 7 \\
& x_2 \leq 5 \\
& x_1, x_2 \geq 0
\end{aligned} \tag{7}$$

**Table 1.** Nondominated Points and their corresponding fitness values.

f1	f2	fitness
-6.98	-1.8	0.96
-1.64	-5.00	0.95
-1.79	-4.89	0.93
-7.00	-1.79	0.91
-6.76	-1.94	0.90
-2.50	-4.46	0.88
-6.87	-1.83	0.86
-6.32	-2.13	0.84
-3.17	-3.92	0.82
-6.61	-2.01	0.81
-2.73	-4.24	0.79
-3.26	-3.81	0.77
-1.93	-4.79	0.75
-1.97	-4.77	0.73
-3.42	-3.80	0.71
-1.68	-4.96	0.69
-6.86	-1.85	0.67
-1.93	-4.82	0.65
-3.11	-4.07	0.63
-6.08	-2.19	0.61
-6.77	-1.90	0.59
-3.78	-3.59	0.57
-2.42	-4.51	0.55
-4.10	-3.29	0.53
-1.74	-4.90	0.51
-2.00	-4.77	0.49
-6.87	-1.84	0.47
-6.36	-2.12	0.45

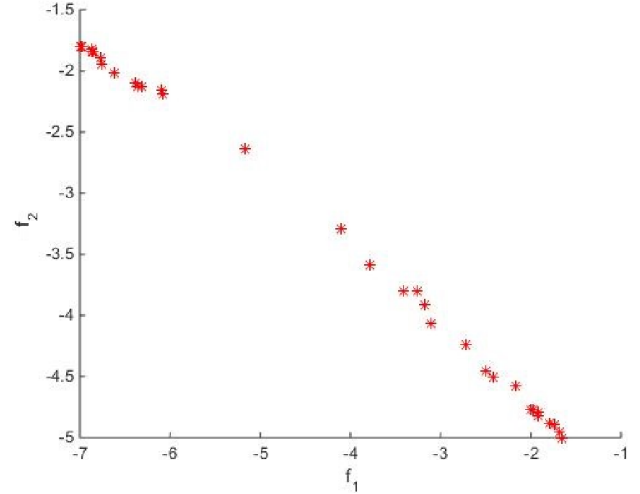
The nondominated points and their corresponding fitness values are shown in Table 1. These are sorted in decreasing order from top to bottom of the table according to the rank based fitness function incorporated in the algorithm.

We are interested in the nondominated point with the best fitness value which will serve as the Best Nondominated Point (BNP). From Table 1, the BNP is  $f^1 = (-6.98, -1.80)^T$  with a fitness value of 0.96, where  $f^1 = (f_1^1, f_2^1)^T \in Y_N$ . Note that when solving Problem 7 with exact methods, the MPNP was found to be  $f^1 = (-7.0, -1.80)^T$  as would be seen in Section 8.

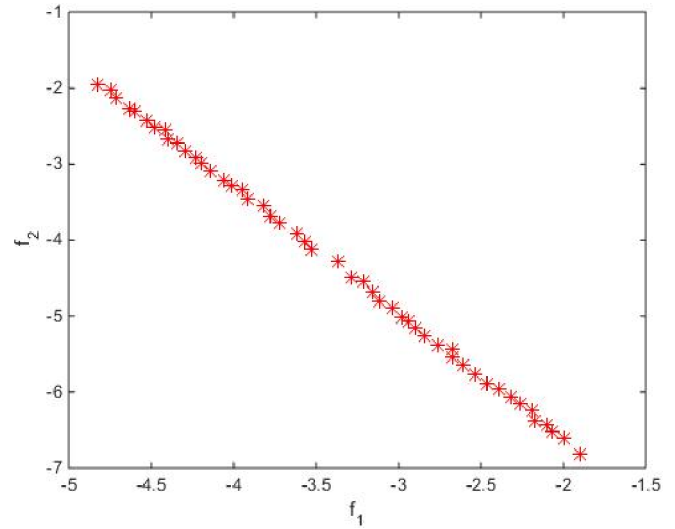
In Figure 2, it can be seen that MOPPA is able to approximate the nondominated frontier using the penalty function method. The nondominated points are not evenly distributed on the frontier but tend to concentrate more towards

the end-points of the front.

We have also solved Problem 7 using NSGA-II [19, 20] which is another approximate multi-objective evolutionary algorithm with the same settings as in Equation (6). We use the same variable bounds with a population size of 50, 200 generations, mutation rate of 0.02 and crossover rate of 0.8. The nondominated frontier as approximated by NSGA-II is shown in Figure 3.



**Figure 2.** Nondominated frontier approximated by MOPPA.



**Figure 3.** Nondominated frontier approximated by NSGA-II.

The nondominated points approximated by NSGA-II are in Table 2. Each of these points is assigned a rank or fitness value according to its domination level and sorted in descending and sorted in descending order from top to bottom of the table according to their crowding distances. The end-points which are fitter than other points are assigned an infinite distance value. From Table 2, the end-point  $f^1 = (-6.82, -1.90)^T$  which is the first to be listed and therefore highest in ranking of the two end-points, is selected as the BNP.

**Table 2.** Nondominated Points and their corresponding Crowding distances.

f1	f2	Crowding Distance
-6.82	-1.9	Inf
-1.96	-4.82	Inf
-4.28	-3.36	0.1582
-4.12	-3.53	0.1232
-6.60	-20.0	0.1171
-4.49	-3.29	0.1066
-3.78	-3.72	0.1032
-3.91	-3.62	0.1014
-5.77	-2.54	0.0974
-5.64	-2.61	0.0953
-5.26	-2.84	0.0943
-5.39	-2.76	0.0933
-3.55	-3.82	0.0925
-4.88	-3.03	0.0917
-5.88	-2.47	0.0916
-3.10	-4.14	0.0912
-3.45	-3.92	0.0905
-5.97	-2.39	0.0894
-2.13	-4.72	0.0863
-4.55	-3.21	0.0842

In Figure 3, it can be seen that NSGA-II did not only approximate the nondominated frontier, but also distribute the points evenly on the nondominated front for the problem.

In terms of the quality of nondominated points approximated by these two algorithms, it can be seen in Tables 1 and 2 that the points returned by MOPPA are of higher quality than those returned by NSGA-II. This can easily be seen that the BNP  $f^1 = (-6.98, -1.80)^T$  selected from Table 1 is of higher quality and closer to the MPNP  $f^1 = (-7.0, -1.80)^T$  determined from the exact methods in Section 8 than the BNP  $f^1 = (-6.82, -1.90)^T$  selected from Table 2.

For comparison purposes, we will compare the BNPs returned by these two approximate methods with the MPNPs returned by the exact methods EMSA[59], BOA [10], PSA [68] and ASIMOLP [5].

We also solved Problem 7 of Section 7.1 using Matlab implementations of exact methods EMSA, BOA, PSA and ASIMOLP. The nondominated points found for EMSA, BOA, PSA are  $f^1 = (-7.0, -1.80)^T$  and  $f^2 = (-1.6, -5.0)^T$  where  $f^1$  and  $f^2 \in Y_N$ , while ASIMOLP returns  $f^1 = (-3.6418, -3.7913)^T$  as the MPNP.

## 9. Determination of Most Preferred Nondominated Points in Exact Methods

As we noted in [58] that to determine the MPNP, we employed the technique of Compromise Programming (CP) introduced by [89] and compute the ideal objective point which would serve as a reference point in each case. CP

is a mathematical programming method that is based on the notion of distance of a most preferred solution from the ideal point  $y^*$  [91]. CP can be used to find the best nondominated point by determining the minimum distance to the ideal point [92]. Ehrgott and Tenfelde-Podehl [28] note that the ideal point is an essential component of CP, and the idea is to find a nondominated point which is as close as possible to it. This is a point in the objective space whose components are the optimal values of the objective functions when they are individually optimized, [3]. It was also noted in [91] that the ideal point serves as a rationale directing and facilitating human choice and decision making. To find the ideal point, we simply solve  $q$  single objective problems

$$\begin{aligned} \min \quad & c_k^T x, \quad k = 1, 2, \dots, q \\ \text{subject to} \quad & x \in X. \end{aligned} \quad (8)$$

We note here that, the ideal point itself is not an element of the nondominated set ( $y^* \notin Y_N$ ). Otherwise, this would mean that the objective functions are not conflicting, but it always exists in the objective space. Its corresponding point in the decision space may not exist, [3].

For our numerical illustration above (problem 7 of Section 7.1), solving each of the objective function individually over the feasible region  $X$  yields the ideal objective point  $y^* = (-7.0, -5.0)^T$ . Clearly  $y^* \notin Y_N$  where  $Y_N = \{(-7.0, -1.80)^T, (-1.6, -5.0)^T\}$ , [58].

Having computed the ideal objective point  $y^*$ , we now determine the minimum distance of each nondominated point  $\hat{y}$  from it by finding

$$\min \{ \|\hat{y}_1 - y^*\|, \|\hat{y}_2 - y^*\|, \dots, \|\hat{y}_n - y^*\| \}$$

where  $\hat{y}_i \in Y_N$  has already been found either by EMSA, BOA and PSA,  $\|\cdot\|$  is the Euclidean norm on  $\mathbb{R}^q$  and  $y^*$  is the ideal objective point.

Using the nondominated points  $f^1$  and  $f^2$  returned by EMSA, BOA and PSA for problem 7 yields

$$\|f^1 - y^*\| = 3.2 \quad \text{and} \quad \|f^2 - y^*\| = 5.4.$$

Since, the relative distance of  $f^1$  from the ideal point  $y^*$  is 3.2 which is the smallest of the two, it therefore means that  $f^1 = (-7.0, -1.80)^T$  is the closest of the two nondominated points to the ideal point  $y^* = (-7.0, -5.0)^T$ . Hence,  $f^1$  is selected as the DM's most preferred nondominated point.

Next, we measure the distance of the nondominated point  $f^1 = (-3.6418, -3.7913)^T$  returned by ASIMOLP for the same problem from the ideal point  $y^* = (-7.0, -5.0)^T$ , as was done with those returned by EMSA, BOA and PSA. It turned out that, the distance

$$\|f^1 - y^*\| = 3.5691$$

is bigger than 3.2 which was the closest when measuring the points returned by EMSA, BOA and PSA, thereby making the nondominated points returned by EMSA, BOA and PSA closer to the ideal point and of higher quality.

We have used this method to choose the MPNP from the nondominated sets returned by EMSA, BOA and PSA for comparison with those returned by MOPPA and NSGA-II. There is no selection of a MPNP in ASIMOLP as the algorithm computes a most preferred efficient solution and also returns the corresponding most preferred nondominated point which is also used for the comparison, [60].

## 10. Experimental Results

In this section, we provide numerical results to compare the quality of the BNP returned by MOPPA and NSGA-II which find approximate points to the MPNP computed by exact methods. Since heuristic approaches are best tested on problems for which the solutions are known, [20]. Table 4 shows the numerical results for a collection of 51 existing problems from the literature ranging from small to medium and realistic MOLP instances. Problem 1 is taken from Ehrgott [26]. Problems 2 to 10 are from Zeleny [91]. Problems 11 to 21 are test problems from the interactive MOLP explorer (iMOLPe) of Alves *et al.* [3]. Problems 22 to 47 are taken from Steuer [79]. Problem 48 is a test problem in Bensolve-2.0 of Löhne and Weißing [53]. Finally, Problems 49 to 51 are from MOPLIB [52] which stands for Multi-Objective Problem Library.

Note that problems 1 to 47 are non-degenerate. Problem 48 is such that the constraint matrix is dense with an identity matrix of order  $n$  as its criterion matrix where  $n$  is the number of variables in the problem. The RHS vector is such that all the components are zeros except for a one (1) at the beginning as the only non-zero element. In Problem 49, the constraint matrix is sparse, the criterion matrix is dense and all the elements in the RHS vector are ones. Problem 50 is such that the constraint and criterion matrices are sparse while the components of the RHS vector are all zeros except for a one (1) at the centre as the only non-zero entry. Finally, Problem 51 is such that the constraint and criterion matrices are sparse while

the components of the RHS vector are all zeros except for a ninety (90) at the end as the only non-zero entry. These larger instances are very challenging, numerically ill-posed and have difficult structures.

All algorithms were implemented in Matlab and executed on an Intel Core i5-2500 CPU at 3.30GHz with 16.0GB RAM. In all tests,  $m$  is the number of constraints,  $n$  the number of variables and  $q$  the number of objectives. We used the MPNPs computed from the nondominated set returned by EMSA, BOA, PSA and ASIMOLP as illustrated in Section 8 to compare with the BNP returned by MOPPA and NSGA-II.

As can be seen from Table 4, MOPPA compared favourably in terms of the quality of the BNP it returns. The algorithm returns BNPs which are of higher quality and closer to those returned by exact methods than NSGA-II, which confirms what was reported in [33] that the objective values obtained with NSGA-II are of lower quality than those obtained with MOPPA.

Of particular interest is Problems 3, 9, 11 and 23 where the points returned are exactly the same with those of the exact methods. In terms of diversity (spread) between the two approximated methods, as can be seen from Section 7.1, NSGA-II returns nondominated points that are more uniformly or evenly distributed than the nondominated front of MOPPA whose points tend to concentrate more towards the end-points of the front.

We also observed in Table 4 that some of the exact methods could not produce results for some of the numerically ill-posed and highly challenging test problems considered due to one reason or the other as stated in the table. The approximate methods on the other hand, were able to solve all these difficult instances approximately.

## 11. Summary of Results

In this section, we present the summary of experimental results discussed in the previous section in Table 3.

Table 3. Summary of experimental results.

Algorithms	Criteria for Evaluation	
	Diversity (Spread)	Quality of BNP returned
MOPPA	The nondominated points are not evenly distributed on the front but concentrate more towards the end-points	Return high quality BNP than NSGA-II which is closer to those returned by the exact methods
NSGA-II	The nondominated points are uniformly distributed on the nondominated front	The quality of BNP is not as good as that returned by MOPPA

Table 4. Comparative results for individual problem.

Prob.	Origin	Algorithm			EMSA	ASIMOLP	BOA	PSA	NSGA-II	MOPPA
		n	m	q	MPNP	MPNP	MPNP	MPNP	BNP	BNP
1	Ehrgott 2006	3	3	3	f1 = -2.00	f1 = -1.74	f1 = -2.00	f1 = -2.00	f1 = -0.35	f1 = -0.70
					f2 = 10.00	f2 = 5.56	f2 = 10.00	f2 = 10.00	f2 = 3.27	f2 = 7.12
					f3 = -5.00	f3 = -2.75	f3 = -5.00	f3 = -5.00	f3 = -1.98	f3 = -3.56
2	Zeleny 1982	2	2	2	f1 = -25000	f1 = -30626	f1 = -25000	f1 = -25000	f1 = -22302	f1 = -24880
					f2 = -66000	f2 = -64132	f2 = -66000	f2 = -66667	f2 = -34750	f2 = -37320

Table 5. Comparative results for individual problem.

Prob.	Algorithm			q	EMSA	ASIMOLP	BOA	PSA	NSGA-II	MOPPA
	Origin	n	m		MPNP	MPNP	MPNP	MPNP	BNP	BNP
3	"	2	4	2	f1 = -9.00 f2 = -15.00	f1 = 4.00 f2 = -18.42	f1 = -9.00 f2 = -15.00	f1 = -9.00 f2 = -15.00	f1 = -9.18 f2 = -11.21	f1 = -9.00 f2 = -15.00
4	"	2	4	3	f1 = -3.00 f2 = -7.50 f3 = -9.00	f1 = -3.50 f2 = -2.74 f3 = 4.89	f1 = -3.00 f2 = -7.50 f3 = -9.00	f1 = -3.00 f2 = -7.50 f3 = -9.00	f1 = -3.10 f2 = -2.21 f3 = -4.13	f1 = -3.10 f2 = -2.50 f3 = -4.40
5	"	2	6	2	f1 = -24.00 f2 = -16.00	f1 = -21.29 f2 = -17.29	f1 = -24.00 f2 = -16.00	f1 = -24.00 f2 = -16.00	f1 = -14.61 f2 = -9.32	f1 = -21.00 f2 = -17.00
6	"	3	3	3	f1 = 3.00 f2 = -6.00 f3 = -12.00	f1 = 1.33 f2 = -6.20 f3 = -9.68	f1 = 3.00 f2 = -6.00 f3 = -12.00	f1 = 3.00 f2 = -6.00 f3 = -12.00	f1 = -0.17 f2 = -3.11 f3 = -3.97	f1 = 0.00 f2 = -6.00 f3 = -3.00
7	"	5	3	3	f1 = 0.00 f2 = -4.00 f3 = -24.00	f1 = -1.38 f2 = -8.77 f3 = -10.04	f1 = 0.00 f2 = -4.00 f3 = -24.00	f1 = 0.00 f2 = -4.00 f3 = -23.62	f1 = -1.40 f2 = -3.00 f3 = -5.08	f1 = -0.10 f2 = -4.00 f3 = -9.00
8	"	5	2	2	f1 = -52.00 f2 = -52.00	f1 = -4.11 f2 = -29.30	f1 = -52.00 f2 = -52.00	f1 = -52.00 f2 = -52.00	f1 = -17.15 f2 = -29.72	f1 = -51.00 f2 = 54.00
9	"	6	4	2	f1 = 0.00 f2 = 0.00	f1 = -0.02 f2 = 0.00	f1 = 0.00 f2 = 0.00	f1 = 0.00 f2 = 0.00	f1 = -0.03 f2 = -0.03	f1 = 0.00 f2 = 0.00
10	"	7	4	3	f1 = -48.00 f2 = -32.00 f3 = 16.00	f1 = -7.65 f2 = -13.80 f3 = -7.75	f1 = -48.00 f2 = -32.00 f3 = 16.00	f1 = -16.00 f2 = 0.00 f3 = -16.00	f1 = -10.85 f2 = -11.66 f3 = -2.52	f1 = -13.90 f2 = -10.76 f3 = 2.65
11	iMOLPe	2	3	2	f1 = -21.00 f2 = -7.00	f1 = -11.87 f2 = -17.22	f1 = -21.00 f2 = -7.00	f1 = -21.00 f2 = -7.00	f1 = -20.02 f2 = -8.44	f1 = -21.00 f2 = -7.00
12	"	3	3	4	f1 = -10.00 f2 = -20.00 f3 = -100.00 f4 = -10.00	f1 = -5.59 f2 = -18.62 f3 = -34.83 f4 = -42.23	f1 = -10.00 f2 = -20.00 f3 = -100.00 f4 = -10.00	f1 = -10.00 f2 = -20.00 f3 = -100.00 f4 = -10.00	f1 = -8.86 f2 = -17.15 f3 = -20.95 f4 = -12.55	f1 = -12.00 f2 = -19.00 f3 = -35.00 f4 = -12.00
13	"	3	5	3	f1 = -21.00 f2 = -4.50 f3 = -4.00	f1 = -10.48 f2 = -3.62 f3 = -2.14	f1 = -21.00 f2 = -4.50 f3 = -4.00	f1 = -21.00 f2 = -4.50 f3 = -4.00	f1 = -14.40 f2 = -4.57 f3 = -3.83	f1 = -16.50 f2 = -4.50 f3 = -5.00
14	"	3	3	3	f1 = -2.66 f2 = -2.00 f3 = -0.33	f1 = -1.10 f2 = -1.22 f3 = -1.57	f1 = -2.66 f2 = -2.00 f3 = -0.33	f1 = -2.66 f2 = -2.00 f3 = -0.33	f1 = -2.22 f2 = -1.07 f3 = -0.40	f1 = -2.00 f2 = -1.50 f3 = -1.50
15	"	4	3	3	f1 = -48.50 f2 = -19.50 f3 = -37.00	f1 = -35.80 f2 = -43.97 f3 = -29.82	f1 = -48.50 f2 = -19.50 f3 = -37.00	f1 = -48.50 f2 = -19.50 f3 = -37.00	f1 = -30.37 f2 = -21.88 f3 = -35.70	f1 = -35.00 f2 = -30.00 f3 = -35.00
16	"	4	2	3	f1 = -20.00 f2 = -80.00 f3 = -40.00	f1 = -31.71 f2 = -49.12 f3 = -69.69	f1 = -20.00 f2 = -80.00 f3 = -40.00	f1 = -20.00 f2 = -80.00 f3 = -40.00	f1 = -20.48 f2 = -24.56 f3 = -27.11	f1 = -35.00 f2 = -30.00 f3 = -35.00
17	"	4	4	3	f1 = -40.00 f2 = -50.00 f3 = -10.00	f1 = -32.22 f2 = -32.50 f3 = -36.27	f1 = -40.00 f2 = -50.00 f3 = -10.00	f1 = -40.00 f2 = -50.00 f3 = -10.00	f1 = -21.90 f2 = -20.58 f3 = -28.34	f1 = -35.00 f2 = -30.00 f3 = -35.00
18	"	3	3	3	f1 = 0.00 f2 = -2.00 f3 = -4.00	f1 = -1.12 f2 = -2.14 f3 = -2.63	f1 = 0.00 f2 = -2.00 f3 = -4.00	f1 = 0.00 f2 = -2.00 f3 = -4.00	f1 = -1.93 f2 = -1.22 f3 = -1.87	f1 = -2.00 f2 = -1.92 f3 = -2.00
19	"	15	10	2	f1 = -363.82 f2 = -33.70	f1 = -137.09 f2 = -198.96	f1 = -363.82 f2 = -33.70	f1 = -229.18 f2 = -35.31	f1 = -73.53 f2 = -31.14	f1 = -143.05 f2 = -32.11
20	"	15	10	3	f1 = -363.82 f2 = -33.70 f3 = -136.71	f1 = -107.15 f2 = -169.94 f3 = -166.26	f1 = -343.50 f2 = -42.43 f3 = -158.75	f1 = -134.17 f2 = -32.88 f3 = -135.82	f1 = -74.85 f2 = -36.96 f3 = -55.35	f1 = -133.77 f2 = -45.05 f3 = -76.54
21	"	10	5	3	f1 = 226.40 f2 = -501.86 f3 = -351.14	f1 = 59.42 f2 = -357.21 f3 = -356.48	f1 = 226.40 f2 = -501.86 f3 = -351.14	f1 = 223.09 f2 = -496.23 f3 = -246.64	f1 = -111.85 f2 = -47.52 f3 = -58.96	f1 = -164.65 f2 = -47.33 f3 = -58.37
22	Steuer 1986	5	5	2	f1 = -10.00 f2 = -3.00	f1 = -6.30 f2 = -6.90	f1 = -10.00 f2 = -3.00	f1 = -10.00 f2 = -3.00	f1 = -6.36 f2 = -3.33	f1 = -6.50 f2 = -3.50

Prob.	Algorithm			q	EMSA	ASIMOLP	BOA	PSA	NSGA-II	MOPPA
	Origin	n	m		MPNP	MPNP	MPNP	MPNP	BNP	BNP
23	"	4	4	3	f1 = 3.42 f2 = -10.28 f3 = -3.42	f1 = -3.79 f2 = 11.38 f3 = -2.96	f1 = 3.42 f2 = -10.28 f3 = -3.42	f1 = 3.42 f2 = -10.28 f3 = -3.42	f1 = -3.09 f2 = 9.25 f3 = -2.74	f1 = -3.50 f2 = 10.50 f3 = -3.40
24	"	5	5	4	f1 = 1.02 f2 = -25.46 f3 = 24.44 f4 = -28.32	f1 = 2.28 f2 = -22.58 f3 = 20.30 f4 = -25.47	f1 = 1.02 f2 = -25.46 f3 = 24.44 f4 = -28.32	f1 = 1.02 f2 = -25.46 f3 = 24.44 f4 = -28.32	f1 = 1.98 f2 = -2.73 f3 = 10.75 f4 = -5.32	f1 = 1.20 f2 = -3.81 f3 = 2.61 f4 = -4.55
25	"	10	8	4	f1 = 106.29 f2 = -462.13 f3 = 175.57 f4 = -33.41	f1 = 80.00 f2 = -54.36 f3 = -163.73 f4 = -23.82	f1 = 106.29 f2 = -462.13 f3 = 175.57 f4 = -33.41	f1 = 183.36 f2 = -424.26 f3 = 117.29 f4 = -4.03	f1 = 13.22 f2 = -36.00 f3 = 6.07 f4 = -1.80	f1 = -54.50 f2 = -20.64 f3 = 46.59 f4 = 20.97
26	"	5	4	3	f1 = -52.07 f2 = 31.50 f3 = -17.35	f1 = -4.44 f2 = -13.17 f3 = -14.37	f1 = -52.07 f2 = 31.50 f3 = -17.35	f1 = -52.07 f2 = 31.50 f3 = -17.35	f1 = -21.00 f2 = 4.69 f3 = -13.87	f1 = -21.00 f2 = 8.79 f3 = -14.79
27	"	6	8	4	f1 = -6.94 f2 = -5.38 f3 = 6.83 f4 = -9.16	f1 = -6.69 f2 = -2.25 f3 = 6.77 f4 = -8.83	f1 = -6.94 f2 = -5.38 f3 = 6.83 f4 = -9.16	f1 = -6.94 f2 = -5.38 f3 = 6.83 f4 = -9.16	f1 = -2.90 f2 = -2.29 f3 = 1.12 f4 = -3.94	f1 = -2.50 f2 = -4.27 f3 = 5.74 f4 = -4.41
28	"	7	6	4	f1 = -31.53 f2 = -26.48 f3 = -26.57 f4 = -0.34	f1 = -25.90 f2 = -23.94 f3 = -19.06 f4 = -8.62	f1 = -31.53 f2 = -26.48 f3 = -26.57 f4 = -0.34	f1 = -31.29 f2 = -30.08 f3 = -26.33 f4 = -0.82	f1 = -17.42 f2 = -12.29 f3 = -9.58 f4 = -7.61	f1 = -20.02 f2 = -16.41 f3 = -14.00 f4 = -4.41
29	"	7	6	4	f1 = 26.80 f2 = -37.73 f3 = -24.33 f4 = -59.60	f1 = 4.03 f2 = -29.03 f3 = -18.07 f4 = -28.17	f1 = 26.80 f2 = -37.73 f3 = -24.33 f4 = -59.60	f1 = 26.80 f2 = -37.73 f3 = -24.33 f4 = -59.60	f1 = -4.05 f2 = -6.97 f3 = -4.92 f4 = -6.13	f1 = -4.79 f2 = -7.34 f3 = -6.11 f4 = -10.55
30	"	8	8	6	f1 = -74.00 f2 = -107.50 f3 = -41.25 f4 = -27.25 f5 = -9.00 f6 = -30.75	f1 = -15.46 f2 = -38.73 f3 = -43.30 f4 = -30.95 f5 = -8.30 f6 = -26.72	f1 = -74.00 f2 = -107.50 f3 = -41.25 f4 = -27.25 f5 = -9.00 f6 = -30.75	f1 = -77.00 f2 = -52.00 f3 = -16.00 f4 = -52.40 f5 = 26.00 f6 = -20.00	f1 = -12.11 f2 = -15.40 f3 = -3.90 f4 = -4.79 f5 = 4.37 f6 = -4.10	f1 = -14.41 f2 = -22.72 f3 = -10.93 f4 = -13.79 f5 = -5.36 f6 = -19.30
31	"	8	8	3	f1 = -36.57 f2 = -22.28 f3 = -14.00	f1 = -32.03 f2 = -20.03 f3 = -17.73	f1 = -36.57 f2 = -22.28 f3 = -14.00	f1 = -36.00 f2 = -23.00 f3 = -15.00	f1 = -9.53 f2 = -3.97 f3 = -4.32	f1 = -10.43 f2 = -4.02 f3 = -3.05
32	"	8	8	3	f1 = -14.03 f2 = -18.00 f3 = -4.93	f1 = -8.77 f2 = -10.56 f3 = -5.13	f1 = -14.03 f2 = -18.00 f3 = -4.93	f1 = -6.50 f2 = -11.00 f3 = -7.50	f1 = -3.85 f2 = -3.32 f3 = -2.79	f1 = -6.30 f2 = -3.82 f3 = -3.06
33	"	5	5	4	f1 = -21.50 f2 = -39.25 f3 = -16.25 f4 = 27.00	f1 = -20.83 f2 = -21.78 f3 = -16.05 f4 = 14.45	f1 = -21.50 f2 = -39.25 f3 = -16.25 f4 = 27.00	f1 = -8.00 f2 = -23.87 f3 = -7.62 f4 = 27.00	f1 = -3.44 f2 = -15.76 f3 = -4.97 f4 = 5.97	f1 = -17.00 f2 = -13.00 f3 = -5.00 f4 = 8.00
34	"	6	6	3	f1 = -12.65 f2 = 0.00 f3 = -30.15	f1 = 12.69 f2 = -3.21 f3 = -28.39	f1 = -12.65 f2 = 0.00 f3 = -30.15	f1 = 13.62 f2 = -9.75 f3 = -26.25	f1 = -4.81 f2 = 2.06 f3 = 3.25	f1 = -6.00 f2 = 4.00 f3 = 2.00
35	"	5	5	4	f1 = -14.66 f2 = -21.06 f3 = 35.73 f4 = -16.00	f1 = -6.33 f2 = -14.44 f3 = 20.77 f4 = -15.63	f1 = -14.66 f2 = -21.06 f3 = 35.73 f4 = -16.00	f1 = -14.00 f2 = 0.00 f3 = 27.00 f4 = 0.00	f1 = -6.64 f2 = -11.42 f3 = 18.07 f4 = -8.71	f1 = -9.00 f2 = -11.21 f3 = 20.21 f4 = -9.00
36	"	10	10	4	f1 = 46.50 f2 = 19.21 f3 = -27.07 f4 = -27.07	f1 = 50.69 f2 = 18.98 f3 = -23.38 f4 = -23.85	f1 = 46.50 f2 = 19.21 f3 = -27.07 f4 = -27.07	f1 = 46.50 f2 = 19.21 f3 = -27.07 f4 = -27.07	f1 = -1.46 f2 = -2.51 f3 = -4.09 f4 = -3.50	f1 = 12.98 f2 = 8.84 f3 = -4.04 f4 = -8.09

Prob.	Algorithm			q	EMSA	ASIMOLP	BOA	PSA	NSGA-II	MOPPA
	Origin	n	m		MPNP	MPNP	MPNP	MPNP	BNP	BNP
37	"	8	8	3	f1 = -14.48 f2 = -4.74 f3 = 6.93	f1 = -2.46 f2 = -3.22 f3 = -1.93	f1 = -14.48 f2 = -4.74 f3 = 6.93	f1 = -14.48 f2 = -4.74 f3 = 6.93	f1 = -1.13 f2 = -2.12 f3 = 0.09	f1 = -2.06 f2 = -4.68 f3 = 4.02
38	"	6	7	4	f1 = -2.61 f2 = -12.63 f3 = 9.70 f4 = 2.37	f1 = -1.80 f2 = -4.00 f3 = 2.78 f4 = -2.07	f1 = -2.61 f2 = -12.63 f3 = 9.70 f4 = 2.37	+	f1 = -2.07 f2 = -0.27 f3 = 0.45 f4 = -0.90	f1 = -2.60 f2 = -2.44 f3 = 1.66 f4 = -1.04
39	"	12	16	4	*	f1 = -5.09 f2 = -9.83 f3 = -9.53 f4 = -6.18	f1 = -5.25 f2 = -14.25 f3 = -8.25 f4 = -1.00	f1 = -5.13 f2 = -3.38 f3 = 1.83 f4 = -1.18	f1 = -2.99 f2 = -2.02 f3 = -2.57 f4 = -1.98	f1 = -5.75 f2 = -2.19 f3 = -3.05 f4 = -1.90
40	"	10	14	5	*	f1 = -1.07 f2 = -3.83 f3 = -5.53 f4 = -16.87 f5 = -8.42	f1 = -5.16 f2 = -2.79 f3 = -4.38 f4 = -18.70 f5 = -9.69	f1 = -18.00 f2 = 70.60 f3 = -3.20 f4 = 8.00 f5 = -4.30	f1 = -2.20 f2 = -3.53 f3 = -1.72 f4 = -5.09 f5 = -1.50	f1 = -4.15 f2 = -2.86 f3 = -3.14 f4 = -5.36 f5 = -1.73
41	"	7	6	3	f1 = -29.40 f2 = -65.30 f3 = -39.30	f1 = -10.74 f2 = -32.20 f3 = -24.39	f1 = -29.40 f2 = -65.30 f3 = -39.30	f1 = -29.40 f2 = -65.30 f3 = -39.30	f1 = -26.09 f2 = -51.35 f3 = -31.97	f1 = -30.00 f2 = -52.00 f3 = -24.00
42	"	7	7	3	f1 = -62.18 f2 = -93.50 f3 = -52.00	f1 = -47.39 f2 = -86.99 f3 = -54.78	f1 = -62.18 f2 = -93.50 f3 = -52.00	f1 = -62.18 f2 = -93.50 f3 = -52.00	f1 = -36.80 f2 = -40.01 f3 = -14.05	f1 = -34.50 f2 = -50.50 f3 = -22.00
43	"	6	6	4	f1 = -37.50 f2 = -11.25 f3 = -7.50 f4 = -20.25	f1 = -10.40 f2 = -6.52 f3 = -5.91 f4 = -0.34	f1 = -37.50 f2 = -11.25 f3 = -7.50 f4 = -20.25	f1 = -37.50 f2 = -11.25 f3 = -7.50 f4 = -20.25	f1 = -11.04 f2 = -2.47 f3 = -6.11 f4 = 0.13	f1 = -17.58 f2 = -3.04 f3 = -0.24 f4 = -10.86
44	"	6	6	4	f1 = 34.50 f2 = -7.50 f3 = -56.00 f4 = -31.50	f1 = 28.39 f2 = -6.38 f3 = -45.43 f4 = -25.83	f1 = 34.50 f2 = -7.50 f3 = -56.00 f4 = -31.50	f1 = 34.50 f2 = -7.50 f3 = -56.00 f4 = -31.50	f1 = 6.02 f2 = -6.95 f3 = -9.14 f4 = -3.30	f1 = 10.00 f2 = -7.00 f3 = -13.00 f4 = -6.50
45	"	10	14	5	*	f1 = 3.35 f2 = -3.18 f3 = -2.48 f4 = -2.50 f5 = 2.10	f1 = 1.03 f2 = -2.19 f3 = 2.01 f4 = -8.13 f5 = 7.22	+	f1 = 2.62 f2 = -2.73 f3 = 0.27 f4 = -2.18 f5 = 2.69	f1 = 3.03 f2 = -2.89 f3 = -0.16 f4 = -2.64 f5 = 2.55
46	"	10	14	5	*	f1 = 2.35 f2 = 0.73 f3 = -11.72 f4 = -1.90 f5 = -10.33	f1 = -4.9 f2 = -3.42 f3 = -4.38 f4 = -18.91 f5 = -9.27	f1 = -14.93 f2 = -5.57 f3 = -2.83 f4 = -16.28 f5 = -6.13	f1 = 1.73 f2 = -3.43 f3 = -1.32 f4 = 3.35 f5 = -1.26	f1 = 2.04 f2 = -4.01 f3 = -1.86 f4 = 4.04 f5 = -1.34
47	"	7	7	3	f1 = -3.83 f2 = -76.46 f3 = -49.57	f1 = -6.82 f2 = -68.93 f3 = -25.83	f1 = -3.83 f2 = -76.46 f3 = -49.57	f1 = -3.83 f2 = -76.46 f3 = -49.57	f1 = -1.83 f2 = -29.69 f3 = -19.70	f1 = -2.00 f2 = -30.00 f3 = -20.00
48	Bensolve 2.0	5	31	5	*	f1 = 0.00 f2 = 0.00 f3 = 0.00 f4 = 0.00 f5 = 0.01	f1 = 0.00 f2 = -1.00 f3 = 0.00 f4 = 0.00 f5 = -2.00	f1 = 0.00 f2 = 0.00 f3 = 0.00 f4 = 0.00 f5 = 0.00	f1 = -0.17 f2 = -0.17 f3 = -0.17 f4 = -0.17 f5 = -0.17	f1 = -0.20 f2 = -0.20 f3 = -0.20 f4 = -0.20 f5 = -0.20
49	MOPLIB	100	20	3	f1 = -168.00 f2 = -124.00 f3 = -143.00	f1 = -61.18 f2 = -73.93 f3 = -96.38	f1 = -168.00 f2 = -124.00 f3 = -143.00	f1 = -168.00 f2 = -124.00 f3 = -143.00	f1 = -100.12 f2 = -99.09 f3 = -109.69	f1 = -121.82 f2 = -120.09 f3 = -126.84
50	"	30	21	12	f1=5.0E-12, f2=5.0E-12 f3=5.0E-12, f4=5.0E-12	-	f1=5.0E-12, f2=5.0E-12 f3=5.0E-12, f4=5.0E-12	f1 = 0, f2 = 0 f3 = 0, f4 = 0	f1=8.0E+15, f2=8.0E+15 f3=8.0E+15, f4=8.0E+15	f1=8.0E+15, f2=8.0E+15 f3=8.0E+15, f4=8.0E+15

Prob.	Origin	Algorithm			EMSA	ASIMOLP	BOA	PSA	NSGA-II	MOPPA
		n	m	q	MPNP	MPNP	MPNP	MPNP	BNP	BNP
51	"	218	28	27	f5=5.0E-12,		f5=5.0E-12,	f5 = 0,	f5=8.0E+15,	f5=8.0E+15,
					f6=5.0E-12		f6=5.0E-12	f6 = 0	f6=8.0E+15	f6=8.0E+15
					f7=5.0E-12,		f7=5.0E-12,	f7 = 0,	f7=8.0E+15,	f7=8.0E+15,
					f8=5.0E-12		f8=5.0E-12	f8 = 0	f8=8.0E+15	f8=8.0E+15
					f9=5.0E-12,		f9=5.0E-12,	f9 = 0,	f9=8.0E+15,	f9=8.0E+15,
					f10=5.0E-12		f10=5.0E-12	f10 = 0	f10=8.0E+15	f10=8.0E+15
					f11=5.0E-12,		f11=5.0E-12,	f11 = 0,	f11=8.0E+15,	f11=8.0E+15,
					f12=-5.5E-11		f12=-5.5E-11	f12 = 0	f12=8.0E+15	f12=8.0E+15
					x	f1=0.52,	x	f1=-360.00,	f1=2.0E+15,	f1=2.0E+15,
						f2=0.01,	x	f2=0.00,	f2=2.0E+15,	f2=2.0E+15,
						f3=6.95,		f3=0.00,	f3=2.0E+15,	f3=2.0E+15,
						f4=2.14		f4=90.00	f4=2.0E+15,	f4=2.0E+15,
						f5=4.94,		f5=180.00,	f5=2.0E+15,	f5=2.0E+15,
						f6=-5.53,		f6=180.00,	f6=2.0E+15	f6=2.0E+15
						f7=-9.71,		f7=180.00,	f7=2.0E+15,	f7=2.0E+15,
						f8=2.23,		f8=180.00,	f8=2.0E+15,	f8=2.0E+15,
						f9=-0.31,		f9=270.00,	f9=2.0E+15,	f9=2.0E+15,
						f10=2.67		f10=0.00,	f10=2.0E+15,	f10=2.0E+15,
						f11=-3.19,		f11=360.00,	f11=2.0E+15,	f11=2.0E+15,
						f12=-5.55		f12=90.00	f12=2.0E+15	f12=2.0E+15
						f13=-5.42,		f13=180.00,	f13=2.0E+15,	f13=2.0E+15,
						f14=-4.46,		f14 =0.00,	f14=2.0E+15,	f14=2.0E+15,
						f15=-4.35,		f15=90.00,	f15=2.0E+15,	f15=2.0E+15,
						f16=6.01		f16=90.00	f16=2.0E+15,	f16=2.0E+15,
						f17=-3.36,		f17=0.00,	f17=2.0E+15,	f17=2.0E+15,
						f18=1.71,		f18=-90.00,	f18=2.0E+15,	f18=2.0E+15,
						f19=-8.01,		f19=90.00,	f19=2.0E+15,	f19=2.0E+15,
						f20=8.90,		f20=-90.00,	f20=2.0E+15	f20=2.0E+15
						f21=8.01,		f21=-90.00,	f21=2.0E+15,	f21=2.0E+15,
						f22=-5.35		f22=90.00	f22=2.0E+15,	f22=2.0E+15,
						f23=5.35,		f23=-90.00,	f23=2.0E+15,	f23=2.0E+15,
						f24=5.35,		f24=-90.00,	f24=2.0E+15	f24=2.0E+15
						f25=5.35,		f25=-90.00,	f25=2.0E+15,	f25=2.0E+15,
						f26=-5.37,		f26=90.00,	f26=2.0E+15,	f26=2.0E+15,
						f27=-4.35		f27=0.00	f27=2.0E+15	f27=2.0E+15

- (x) Out of memory
- (-) No initial starting solution
- (\*) Aborted after 3 days of running time
- (+) The image is the whole region, implying that none of the vertices is nondominated

## 12. Conclusion

We have applied two heuristic approaches for the first time to MOLP. One, namely NSGA-II is well established and popular heuristic for continuous and discrete multi-objective optimisation. The other, MOPPA, is fairly recent addition to nature-inspired algorithms which has shown a lot of promise on continuous multi-objective optimisation, and continuous and discrete single objective optimisation. Our experimental investigation using Matlab implementations of both approaches applied to an extensive and representation set

of MOLP instances has shown that the methods found on the whole good nondominated fronts. That of NSGA-II is more uniformly spread while the BNP's returned by MOPPA tend to be of better quality. The methods compare well with the exact ones especially on the large instances which the exact methods failed to solve even when given generous amounts of computation times. Constraints have been handled using a penalty function approach.



## Acknowledgements

We are grateful to ESRC, Grant ES/L011859/1, for partially funding this research.

## References

- [1] SS Abhyankar, TL Morin, and T Trafalis. Efficient faces of polytopes: Interior point algorithms, parametrization of algebraic varieties, and multiple objective optimization. *Contemporary Mathematics*, 114: 319-341, 1990.
- [2] Maria João Alves and João Paulo Costa. An exact method for computing the nadir values in multiple objective linear programming. *European Journal of Operational Research*, 198 (2): 637-646, 2009.
- [3] Maria João Alves, Carlos Henggeler Antunes, and João Clímaco. Interactive MOLP explorer: a graphical-based computational tool for teaching and decision support in multi-objective linear programming models. *Computer Applications in Engineering Education*, 23 (2): 314-326, 2015.
- [4] Ami Arbel. An interior multiobjective linear programming algorithm. *Computers and Operations Research*, 20 (7): 723-735, 1993.
- [5] Ami Arbel. A weighted-gradient approach to multi-objective linear programming problems using the analytic hierarchy process. *Mathematical and computer modelling*, 17 (4): 27-39, 1993.
- [6] Ami Arbel. Anchoring points and cones of opportunities in interior multiobjective linear programming. *Journal of the Operational Research Society*, 45 (1): 83-96, 1994.
- [7] Paul Armand and Christian Malivert. Determination of the efficient set in multiobjective linear programming. *Journal of Optimization Theory and Applications*, 70 (3): 467-489, 1991.
- [8] Tapan P Bagchi. *Multiobjective scheduling by genetic algorithms*. Springer Science and Business Media, 1999.
- [9] Vu Thien Ban. A finite algorithm for minimizing a concave function under linear constraints and its applications. In *Proceedings of IFIP Working Conference on Recent Advances in System Modelling and Optimization*, 1983.
- [10] Harold P Benson. An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. *Journal of Global Optimization*, 13 (1): 1-24, 1998.
- [11] Herold P Benson. Further analysis of an outcome set-based algorithm for multiple objective linear programming. *Journal of Optimization Theory and Applications*, 97 (1): 1-10, 1998.
- [12] Herold P Benson. Hybrid approach for solving multiple-objective linear programs in outcome space. *Journal of Optimization Theory and Applications*, 98 (1): 17-35, 1998.
- [13] Victor Blanco, Justo Puerto, and Safae El Haj Ben Ali. A semidefinite programming approach for solving multiobjective linear programming. *Journal of Global Optimization*, 58 (3): 465-480, 2014.
- [14] M Chakraborty and Ananya Ray. Parametric approach and genetic algorithm for multi objective linear programming with imprecise parameters. *Opsearch*, 47 (1): 73-92, 2010.
- [15] László Csirmaz. Using multiobjective optimization to map the entropy region. *Computational Optimization and Applications*, 63 (1): 45-67, 2013.
- [16] Jerald P Dauer. Analysis of the objective space in multiple objective linear programming. *Journal of Mathematical Analysis and Applications*, 126 (2): 579-593, 1987.
- [17] Jerald P Dauer and Yi-Hsin Liu. Solving multiple objective linear programs in objective space. *European Journal of Operational Research*, 46 (3): 350-357, 1990.
- [18] Jerald P Dauer and OA Saleh. Constructing the set of efficient objective values in multiple objective linear programs. *European Journal of Operational Research*, 46 (3): 358-365, 1990.
- [19] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGAII. In *International Conference on Parallel Problem Solving from Nature*, pages 849-858. Springer, 2000.
- [20] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6 (2): 182-197, 2002.
- [21] S Dhanalakshmi, S Kannan, K Mahadevan, and S Baskar. Application of modified NSGA-II algorithm to combined economic and emission dispatch problem. *International Journal of Electrical Power and Energy Systems*, 33 (4): 992-1002, 2011.
- [22] Daniel Dörfler, Andreas Löhne, Christopher Schneider, and Benjamin WeiBing. A benson-type algorithm for bounded convex vector optimization problems with vertex selection. *Optimization Methods and Software*, pages 1-21, 2021.
- [23] JG Ecker, Nancy Shoemaker Hegner, and IA Kouada. Generating all maximal efficient faces for multiple objective linear programs. *Journal of Optimization Theory and Applications*, 30 (3): 353-381, 1980.

- [24] JG Ecker and IA Kouada. Finding all efficient extreme points for multiple objective linear programs. *Mathematical Programming*, 14 (1): 249-261, 1978.
- [25] M Ehrgott, J Puerto, and AM Rodriguez-Chia. Primal-dual simplex method for multiobjective linear programming. *Journal of optimization theory and applications*, 134 (3): 483-497, 2007.
- [26] Matthias Ehrgott. *Multicriteria optimization*. Springer Science and Business Media, 2006.
- [27] Matthias Ehrgott, Andreas Löhne, and Lizhen Shao. A dual variant of Benson's outer approximation algorithm for multiple objective linear programming. *Journal of Global Optimization*, 52 (4): 757-778, 2012.
- [28] Matthias Ehrgott and Dagmar Tenfelde-Podehl. Computation of ideal and nadir values and implications for their use in mcdm methods. *European Journal of Operational Research*, 151 (1): 119-139, 2003.
- [29] Horst A Eiselt and C-L Sandblom. *Linear programming and its applications*. Springer Science and Business Media, 2007.
- [30] Philippe Erlanger. Louis XIV. Weidenfeld and Nicolson, 1970.
- [31] J Po Evans and RE Steuer. A revised simplex method for linear multiple objective programs. *Mathematical Programming*, 5 (1): 54-72, 1973.
- [32] Eric S Fraga. <http://www.ucl.ac.uk/ucecesf/strawberry.html#orgec5771e>. 2018.
- [33] Eric S Fraga and Oluwamayowa Amusat. Understanding the impact of constraints: a rank based fitness function for evolutionary methods. In *Advances in Stochastic and Deterministic Global Optimization*, pages 243-254. Springer, 2016.
- [34] Eric S Fraga, Abdellah Salhi, Di Zhang, and Lazaros G Papageorgiou. Optimisation as a tool for gaining insight: An application to the built environment. *Journal of Algorithms and Computational Technology*, 9 (1): 13-26, 2015.
- [35] Tomas Gal. A general method for determining the set of all efficient solutions to a linear vector maximum problem. *European Journal of Operational Research*, 1 (5): 307-322, 1977.
- [36] Ashish Ghosh and Mrinal Kanti Das. Non-dominated rank based sorting genetic algorithms. *Fundamenta Informaticae*, 83 (3): 231-252, 2008.
- [37] Andreas H Hamel, Andreas Löhne, and Birgit Rudloff. Benson type algorithms for linear vector optimization and applications. *Journal of Global Optimization*, 59 (4): 811-836, 2014.
- [38] John H Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. The University of Michigan Press, 1975.
- [39] Yuan Hu, Zhaohong Bie, Tao Ding, and Yanling Lin. An NSGA-II based multiobjective optimization for combined gas and electricity network expansion planning. *Applied energy*, 167: 280-293, 2016.
- [40] H Isermann and G Naujoks. Operating manual for the EFFACET multiple objective linear programming package. Fakultät fuer Wirtschaftswissenschaften, University of Bielefeld, Bielefeld, Germany, 1984.
- [41] Heinz Isermann. The enumeration of the set of all efficient solutions for a linear multiple objective program. *Journal of the Operational Research Society*, 28 (3): 711-725, 1977.
- [42] HV Junior and Marcos Pereira Estellita Lins. A win-win approach to multiple objective linear programming problems. *Journal of the Operational Research Society*, 60 (5): 728-733, 2009.
- [43] Deb Kalyanmoy. *Multi objective optimization using evolutionary algorithms*. John Wiley and Sons, 2001.
- [44] S Kannan, S Baskar, James D McCalley, and P Murugan. Application of NSGA-II algorithm to generation expansion planning. *IEEE Transactions on Power systems*, 24 (1): 454-461, 2009.
- [45] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302-311. ACM, 1984.
- [46] Leonid Khachiyan, Endre Boros, Konrad Borys, Khaled Elbassioni, and Vladimir Gurvich. Generating all vertices of a polyhedron is hard. *Discrete and Computational Geometry*, 39 (1-3): 174-190, 2008.
- [47] K-H Küfer. On the asymptotic average number of efficient vertices in multiple objective linear programming. *Journal of Complexity*, 14 (3): 333-377, 1998.
- [48] C Lin, C Chen, and P Chen. On the modified interior point algorithm for solving multi-objective linear programming problems. *International Journal of Information and Management Sciences*, 17 (1): 107, 2006.
- [49] Andreas Löhne. *Vector optimization with infimum and supremum*. Springer Science and Business Media, 2011.
- [50] Andreas Löhne. Bensolve: VLP solver, version 1.2, [www.bensolve.org](http://www.bensolve.org). 2012.

- [51] Andreas Löhne, Birgit Rudloff, and Firdevs Ulus. Primal and dual approximation algorithms for convex vector optimization problems. *Journal of Global Optimization*, 60 (4): 713-736, 2014.
- [52] Andreas Löhne and Sebastian Schenker. MOPLIB: Multi-Objective Problem Library, <http://moplib.uni-jena.de>. Accessed 13 March 2017, 2015.
- [53] Andreas Löhne and Benjamin WeiBing. Bensolve: VLP solver, version 2.0.x, [www.bensolve.org](http://www.bensolve.org). 2015.
- [54] Anabel Martinez-Vargas, Josué Domínguez-Guerrero, ángel G Andrade, Roberto Sepúlveda, and Oscar Montiel-Ross. Application of NSGA-II algorithm to the spectrum assignment problem in spectrum sharing networks. *Applied Soft Computing*, 39: 188-198, 2016.
- [55] Renzo Massobrio, G Fagúndez, and S Nesmachnow. Multiobjective taxi sharing optimization using the NSGA-II evolutionary algorithm. In 11th Metaheuristic International Conference, 2015.
- [56] Zbigniew Michalewicz and Marc Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary computation*, 4 (1): 1-32, 1996.
- [57] Anjana D Nandasana, Ajay Kumar Ray, and Santosh K Gupta. Applications of the non-dominated sorting genetic algorithm (NSGA) in chemical reaction engineering. *International Journal of Chemical and Reactor Engineering*, 1: 1018, 2003.
- [58] Paschal B Nyiam and Abdellah Salhi. A comparative study of two key algorithms in multiple objective linear programming. *Journal of Algorithms and Computational Technology*, 13: 1748302619870424, 2019.
- [59] Paschal B Nyiam and Abdellah Salhi. A comparison of benson outer approximation algorithm with an extended version of multiobjective simplex algorithm. *Advances in Operations Research*, 2021.
- [60] Paschal B Nyiam and Abdellah Salhi. On the simplex, interior point and objective space approaches to multiple objective linear programming. *Journal of Algorithms and Computational Technology*, 15: 17483026211008414, 2021.
- [61] P Pandian and M Jayalakshmi. Determining efficient solutions to multiple objective linear programming problems. *Applied Mathematical Sciences*, 7 (26): 1275-1282, 2013.
- [62] Yan Pei and Jia Hao. Non-dominated sorting and crowding distance based multiobjective chaotic evolution. In *International Conference in Swarm Intelligence*, pages 15-22. Springer, 2017.
- [63] Johan Philip. Algorithms for the vector maximization problem. *Mathematical Programming*, 2 (1): 207-229, 1972.
- [64] Johan Philip. Vector maximization at a degenerate vertex. *Mathematical Programming*, 13 (1): 357-359, 1977.
- [65] L Pourkarimi, MA Yaghoobi, and M Mashinchi. Determining maximal efficient faces in multiobjective linear programming problem. *Journal of Mathematical Analysis and Applications*, 354 (1): 234-248, 2009.
- [66] Noraini Mohd Razali, John Geraghty, et al. Genetic algorithm performance with different selection strategies in solving TSP. In *Proceedings of the world congress on engineering*, volume 2, pages 1134-1139. International Association of Engineers Hong Kong, 2011.
- [67] Alistair D Rodman, Eric S Fraga, and Dimitrios Gerogiorgis. On the application of a nature-inspired stochastic evolutionary algorithm to constrained multi-objective beer fermentation optimisation. *Computers and Chemical Engineering*, 108: 448-459, 2018.
- [68] Birgit Rudloff, Firdevs Ulus, and Robert Vanderbei. A parametric simplex algorithm for linear vector optimization problems. *Mathematical Programming*, pages 1-30, 2015.
- [69] Andrzej Ruszczyński and Robert J Vanderbei. Frontiers of stochastically nondominated portfolios. *Econometrica*, pages 1287-1297, 2003.
- [70] Thomas L Saaty. The analytic hierarchy process: planning, priority setting, resources allocation. New York: McGraw, 1980.
- [71] Abdellah Salhi and Eric S Fraga. Nature-inspired optimisation approaches and the new plant propagation algorithm. In *Proceedings of the International Conference on Numerical Analysis and Optimisation (ICeMATH'11)*, Yogyakarta, Indonesia, 2011.
- [72] J David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, Pittsburgh, PA, USA, July 1985, pages 93-100, 1985.
- [73] Murray Schechter and Ralph E Steuer. A correction to the connectedness of the Evans-Steuer algorithm of multiple objective linear programming. *Foundations of Computing and Decision Sciences*, 30 (4): 351-360, 2005.
- [74] Birsen İ Selamoğlu and Abdellah Salhi. The plant propagation algorithm for discrete optimisation: The case of the travelling salesman problem. In *Nature-inspired computation in engineering*, pages 43-61. Springer, 2016.

- [75] Lizhen Shao and Matthias Ehrgott. Approximately solving multiobjective linear programmes in objective space and an application in radiotherapy treatment planning. *Mathematical Methods of Operations Research*, 68 (2): 257-276, 2008.
- [76] Lizhen Shao and Matthias Ehrgott. Approximating the nondominated set of an molp by approximately solving its dual problem. *Mathematical Methods of Operations Research*, 68 (3): 469-492, 2008.
- [77] Alice E Smith and David W Coit. Constraint handling techniques: a  $\tilde{A}\tilde{T}$  penalty functions. *Handbook of evolutionary computation*, pages 5-2, 1997.
- [78] N Srinivas and K Deb. Multi-objective function optimisation using non-dominated sorting genetic algorithm. *Evolutionary Comp*, 2 (3): 221-248, 1995.
- [79] Ralph E Steuer. Multiple criteria optimization: theory, computation, and applications. Wiley, 1986.
- [80] Ralph E Steuer. Adbase: A multiple objective linear programming solver for all efficient extreme points and all unbounded efficient edges. Terry college of Business, University of Georgia, Athens, 2003.
- [81] Muhammad Sulaiman, Abdellah Salhi, Birsan Irem Selamoglu, and Omar Bahaaldin Kirikchi. A plant propagation algorithm for constrained engineering optimisation problems. *Mathematical Problems in Engineering*, 2014.
- [82] Ue-Pyng Wen and Wei-Tai Weng. An interior algorithm for solving multiobjective linear programming problem. Institute for Operations Research and the Management Sciences International Meeting: Tel Aviv - Israel, 1998.
- [83] Wei-Tai Weng and Ue-Pyng Wen. An interior point algorithm for solving linear optimization over the efficient set problems. *Journal of the Chinese Institute of Industrial Engineers*, 18 (3): 21-30, 2001.
- [84] Özgür Yeniay. Penalty function methods for constrained optimization with genetic algorithms. *Mathematical and computational Applications*, 10 (1): 45-56, 2005.
- [85] PL Yu and M Zeleny. The techniques of linear multiobjective programming. *Revue française d'Automatique, d'Informatique et de Recherche Opérationnelle. Recherche Opérationnelle*, 8 (3): 51-71, 1974.
- [86] PL Yu and M Zeleny. Linear multiparametric programming by multicriteria simplex method. *Management Science*, 23 (2): 159-170, 1976.
- [87] PL Yu and Milan Zeleny. The set of all nondominated solutions in linear cases and a multicriteria simplex method. *Journal of Mathematical Analysis and Applications*, 49 (2): 430-468, 1975.
- [88] Tey Jing Yuen and Rahizar Ramli. Comparision of computational efficiency of MOEAnD and NSGA-II for passive vehicle suspension optimization. *ECMS*, 2010: 219-225, 2010.
- [89] Milan Zeleny. Compromise programming. In Cochrane JL, Zeleny M (eds), *Multiple criteria decision making*, pages 262-301. University of South Carolina Press, Columbia, SC, 1973.
- [90] Milan Zeleny. *Linear multiobjective programming*, volume 95. Springer-Verlag, 1974.
- [91] Milan Zeleny. *Multiple criteria decision making*. McGraw-Hill New York, 1982.
- [92] WH Zhang. A compromise programming method using multibounds formulation and dual approach for multicriteria structural optimization. *International journal for numerical methods in engineering*, 58 (4): 661-678, 2003.