Estimating Nadir Objective Vector Quickly Using Evolutionary Approaches

Kalyanmoy Deb, Shamik Chaudhuri
Indian Institute of Technology Kanpur
Kanpur, PIN 208016, India
{deb,shamik}@iitk.ac.in
http://www.iitk.ac.in/kangal

Kaisa Miettinen
Helsinki School of Economics
PO Box 1210, FI 00101, Helsinki, Finland
kaisa.miettinen@hse.fi
http://www.mit.jyu.fi/~miettine

KanGAL Report Number 2005009

Abstract

Nadir point plays an important role in multi-objective optimization because of its importance in estimating the range of objective values corresponding to desired Pareto-optimal solutions and also in using many classical interactive optimization techniques. Since this point corresponds to the worst Pareto-optimal solution of each objective, the task of estimating the nadir point necessitates information about the whole Pareto optimal frontier and is reported to be a difficult task using classical means. In this paper, for the first time, we have proposed a couple of modifications to an existing evolutionary multi-objective optimization procedure to focus its search towards the extreme objective values front-wise. On two to 20-objective optimization problems, both proposed procedures are found to be capable of finding a near nadir point quickly and reliably. Simulation results are interesting and should encourage further studies and applications in estimating the nadir point, a process which should lead to a better interactive procedure of finding and arriving at a desired Pareto-optimal solution.

Keywords: Nadir objective vector, nadir point, multi-objective optimization, non-dominated sorting GA, evolutionary multi-objective optimization (EMO), ideal point.

1 Introduction

In a multi-objective optimization procedure, an estimation of the nadir objective vector is an important task. The nadir objective vector represents the worst value of each objective function corresponding to the entire Pareto-optimal set. Sometimes, this point is confused with the point representing the worst objective value of the entire search space, which is often an over-estimation or under-estimation of the nadir objective vector. Along with the ideal objective vector (a point constructed by the best value of each objective), the nadir objective vector is used to normalize objective functions [12], a matter often desired for an adequate functioning of a multi-objective optimization algorithm. With these two extreme values, the objective functions can be scaled so that each scaled objective takes values more or less in the same range. These scaled values can be used for optimization with different algorithms like the weighted-sum approach or the Tchebycheff metric method [12, 3]. Such a scaling procedure may help in reducing the computational cost by solving the problem faster [14]. Apart from normalizing the objective function values, nadir objective vector is also used for finding Pareto-optimal solutions by different interactive algorithms like *guess* method and others [12]. The general idea of these methods is to maximize the minimum weighted deviation from the nadir objective vector. Moreover, the knowledge of nadir and ideal objective values helps the decision-maker in adjusting her/his expectations on a realistic level by

knowing the range of each objective and can then be used to focus on a desired region. Furthermore, in visualizing Pareto-optimal solutions, the knowledge of the nadir objective vector is essential. Along with the ideal point, the nadir point will then provide the range of each objective to facilitate in visualizing the trade-off information through value paths, bar charts, petal diagrams etc. [12, 13].

Since the estimating the nadir point necessitates information about the whole Pareto optimal frontier, any procedure of estimating this point should involve finding Pareto-optimal solutions. This makes the task more difficult compared to finding the ideal point [11], which corresponds to the best objective values over the entire search space. Since evolutionary multi-objective optimization (EMO) algorithms attempt to find the entire Pareto-optimal frontier, EMO methodologies stand as viable candidates for this task. However, an estimation of the nadir objective vector need not involve finding intermediate Pareto-optimal solutions. Only extreme solutions corresponding to the Pareto-optimal set will be adequate for the task. In this paper, we have suggested two modifications to an existing EMO methodology – elitist non-dominated sorting GA or NSGA-II [4] – for emphasizing extreme Pareto-optimal solutions, thereby leading to the estimate of the nadir point. Simulation results on two to 20-objective optimization problems demonstrate that one of the two approaches – the extremized crowded NSGA-II – is capable of finding a near nadir point more quickly and reliably than the other method and the original NSGA-II.

2 Pareto-Optimal Solutions and Nadir Objective Vector

Multi-objective optimization problems involve a number of conflicting objectives (f_j, $j = 1, 2, \ldots, M$) and theoretically give rise to a set of Pareto-optimal solutions, which provide a trade-off among the objectives. In the sense of minimization of objectives, Pareto-optimal solutions can be defined as follows [12]:

Definition 1 *A decision vector $\mathbf{x}^* \in S$ is Pareto-optimal if there does not exist another decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$ for all $i = 1, 2, \ldots, M$ and $f_j(\mathbf{x}) < f_j(\mathbf{x}^*)$ for at least one index j.*

Provided that the problem is correctly specified, the final solution of a rational decision-maker is always Pareto-optimal and hence there is need to concentrate on finding Pareto-optimal solutions in a multi-objective optimization problems. We now define a nadir objective vector as follows:

Definition 2 *An objective vector $z^{\text{nad}} = (z_1^{\text{nad}}, \ldots, z_M^{\text{nad}})^T$ constructed with worst value of objective functions in the complete Pareto-optimal set P^* is called a nadir objective vector.*

Hence, for minimization problems, $z_j^{\text{nad}} = \max_{\mathbf{x} \in P^*} f_j(\mathbf{x})$. Estimation of the nadir objective vector is, in general, a difficult task. Unlike the ideal point $(z_1^*, z_2^*, \ldots, z_M^*)^T$, which can be found by minimizing each objective individually over the feasible set S (or, $z_j^* = \min_{\mathbf{x} \in S} f_j(\mathbf{x})$), the nadir point cannot be formed by maximizing objectives individually over S.

Let us consider a two-objective minimization problem, as shown in Figure 1. To calculate the nadir objective vector if f_1 and f_2 are maximized individually, we shall obtain points A and B, respectively. These two points produce an extreme point $\mathbf{z}^w$ (termed as the 'worst objective vector' in the figure), which is not the true nadir point. A nadir point must be constructed with the worst objective values of all Pareto-optimal solutions, and not with the worst objective values of the entire search space. Thus, to find the nadir point, the Pareto-optimality of the solutions used for constructing the nadir point must be first established. This makes the task of finding the nadir point difficult.

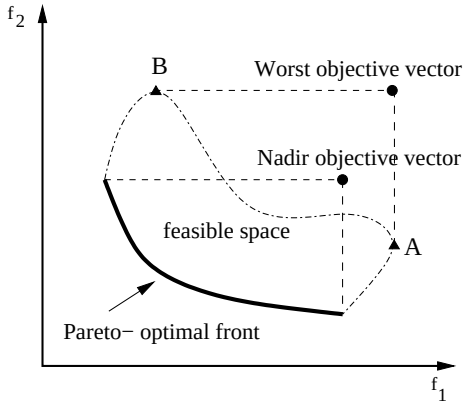

Figure 1: The nadir and worst objective vectors.

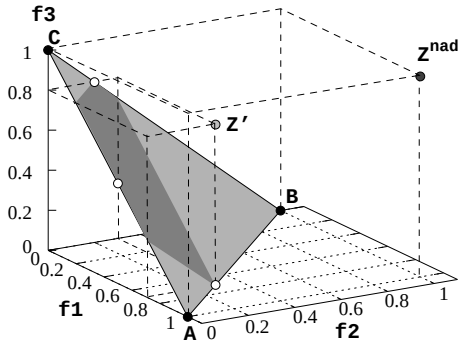

Figure 2: Payoff table method may not find the true nadir point.

3 Existing Methods

To overcome this difficulty, researchers have suggested different methods. Benayoun et al. [1] introduced the first interactive multi-objective optimization method using a nadir point (although authors did not use the term 'nadir'), which was to be found by using a payoff table method. In this method, a table is constructed where i-th row of the table represents values of all other objective functions calculated at the point where i-th objective is minimum. Thereafter, the maximum value of the i-th column can be considered as an estimate of the upper bound of the i-th objective and these maximum values together may be used to construct the nadir vector. The main difficulty of such an approach is that corresponding to the minimum solution of an objective there may exist more than one solutions having different values of other objectives, especially in problems having more than two objectives. Consider the Pareto-optimal front of a typical three-objective optimization problem shown in Figure 2. The minimum value of the first objective function is $f_1 = 0$. It can be seen from the figure that there exist a number of solutions having $f_1 = 0$ and different values of f_2 and f_3 (all solutions on the line BC). In the payoff table method, when the following three solutions are found $\mathbf{f}^{(1)} = (0, 0, 1)^T$ (point C), $\mathbf{f}^{(2)} = (1, 0, 0)^T$ (point A), and $\mathbf{f}^{(3)} = (0, 1, 0)^T$ (point B) corresponding to minimizations of f_1, f_2, and f_3, the true nadir point $z^{\text{nad}} = (1, 1, 1)^T$ can be found. However, if solutions $\mathbf{f}^{(1)} = (0, 0.2, 0.8)^T$, $\mathbf{f}^{(2)} = (0.5, 0, 0.5)^T$ and $\mathbf{f}^{(3)} = (0.7, 0.3, 0)^T$ (marked with open circles) are found corresponding minimizations of f_1, f_2, and f_3, respectively, a wrong estimate $z' = (0.7, 0.3, 0.8)^T$ of the nadir point will be made. The figure shows such a wrong nadir point represents only a portion (shown dark-shaded) of the Pareto-optimal front.

Later, Iserman and Steuer [10] demonstrated the difficulties of finding a nadir point even for liner problems and emphasized the need of using a method better than the payoff table method. Dessouky et al. [7] suggested three heuristic methods and Korhonen et al. [11] suggested another heuristic method for this purpose. Let us point out that all these methods suggested have been developed for multi-objective linear problems where all objectives and constraints are linear functions of the variables.

Ehrgott and Tenfelde-Podehl [8] proposed an algorithm based on subproblems, that is, to find the nadir point for an M-objective problem, Pareto-optimal solutions of lower-dimensional problems are used. Such a requirement may make the algorithm computationally impractical beyond three objectives. Moreover, authors did not suggest how to extend the idea to nonlinear problems. It must be emphasized that although the determination of the nadir point depends on finding the worst objective values in the set of Pareto-optimal solutions, even for linear problems, this is a difficult task.

4 Evolutionary Approach

Because the determination of the nadir point is associated with the Pareto-optimal solutions, thereby hinting that a determination of a set of Pareto-optimal solutions will facilitate the estimation of the nadir point. For the past decade or so, evolutionary multi-objective optimization (EMO) algorithms have been gaining popularity because of their ability to find multiple, widespread, Pareto-optimal solutions simultaneously in a single simulation run [3, 2]. Since they aim at finding a set of Pareto-optimal solutions, an EMO procedure can be an ideal way to find the nadir objective vector. However, the naive procedure of first finding a Pareto-optimal set and then determining the nadir objective vector from the set seems to cause a dilemma. Recall that the main purpose of the nadir objective vector is to normalize the objectives so an interactive multi-objective optimization algorithm can be used to find a desired Pareto-optimal solution. However, if an EMO is used to find a representative Pareto-optimal set, there is no major reason for finding the nadir point. In an interactive sense [5], an application of an EMO procedure can provide an idea of the nature of the Pareto-optimal set and subsequent applications of an EMO procedure to a desired (or preferred) region of the Pareto-optimal set may be equivalent to the nadir point based interactive methods.

However, the EMO procedures use the concept of Pareto-dominance in their working principles and have been criticized for the ineffectiveness for handling a large number of objectives (more than five objectives or so). To represent a high-dimensional Pareto-optimal front requires an exponentially large number of points [3]. This causes the EMO procedures to be inadequate to find the complete Pareto-optimal front in the first place. Thus, for handling a large number of objectives, it may be not advantageous to use the two-pronged EMO procedure of finding the Pareto-optimal front and focusing in a desired region. Instead, the EMO procedure may be used to achieve both tasks simultaneously in one application in which the main focus is to distribute the population into those regions of the Pareto-optima set which will construct the nadir point correctly. For the three-objective minimization problem of Figure 2, the proposed EMO procedure should distribute its population members near regions A, B, and C, instead of on the entire Pareto-optimal set so that the true nadir point can be found quickly. In the following section, we describe two EMO procedures for this purpose.

5 Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II)

Both EMO procedures proposed here use a specific procedure – elitist non-dominated sorting genetic algorithm (NSGA-II) [4] – which was developed by the first author and his students at Kanpur Genetic Algorithms Laboratory (KanGAL) in 2000 and has lately received a great deal of attention due to its simplicity, computationally fast approach and availability of source codes. Here, we provide a brief description of this population-based procedure.

5.1 NSGA-II

Like in a genetic algorithm (GA), NSGA-II starts with a population of N_{pop} random solutions. In the t-th iteration of NSGA-II, the offspring population Q_t is first created by using the parent population P_t and the usual genetic operators – reproduction, recombination, and mutation [9]. Thereafter, both populations are combined together to form R_t of size $2N_{\text{pop}}$. Then, a non-dominated sorting procedure [3] is applied to classify the entire population R_t into a number of hierarchical non-dominated fronts. Figure 3 shows a schematic of one iteration of NSGA-II. Fronts are marked as 1,2,... in the decreasing level of non-domination of solutions. A property of such fronts is that a solution lying on a better non-dominated front (say $\mathcal{F}$) does not get dominated by any solution of a front worse than $\mathcal{F}$. Once the non-dominated sorting of the set R_t is over, the

new population is filled with solutions of different non-dominated fronts, one at a time. The filling starts with the best non-dominated front (marked as front 1) and continues with solutions of the second non-dominated front, followed by the third non-dominated front, and so on. Since the overall population size of R_t is $2N_{\text{pop}}$, not all fronts may be accommodated in N_{pop} slots available in the new population. This scenario is shown in Figure 3. All fronts which could not be accommodated (such as, fronts 4, 5 and 6 in the figure) are simply deleted. When the last allowed front (front 3 in the figure) is being considered, there may exist more solutions in the front than the remaining slots in the new population. Instead of arbitrarily discarding some members from the last front, the solutions which will make the *diversity* of the selected solutions the largest are chosen. In this step,

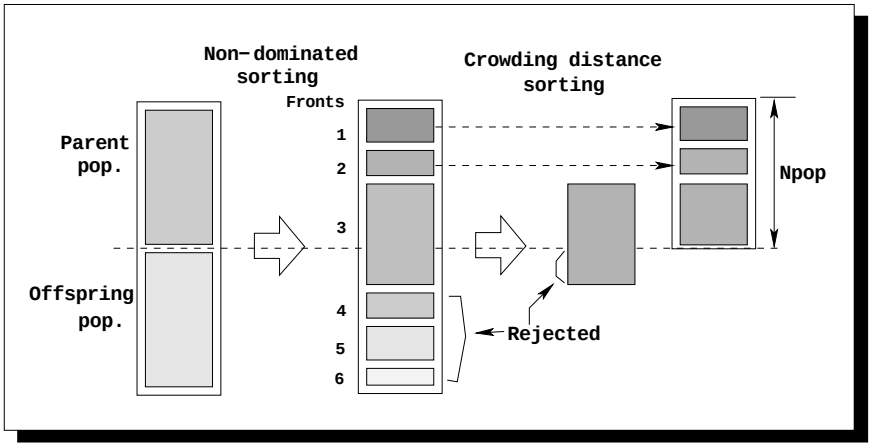

Figure 3: A schematic of the NSGA-II procedure.

a *crowded-sorting* of the solutions of front i (the last front which could not be accommodated fully) is performed by using a *crowding distance metric* and the adequate number of solutions are picked from the top of the list. The crowding distance of a solution in a non-dominated front is a measure of crowding by other members of the front. In the NSGA-II implementation, a metric totalling the objective-wise distances between neighboring solutions is used (described in the following section). The reproduction operator uses a *crowded tournament* selection method in which two solutions are compared and the winner is declared as the one residing on a better non-dominated front, or the one having a larger crowding distance value. For details, readers are encouraged to refer to the original NSGA-II study [4].

5.1.1 NSGA-II Crowding Distance Calculation

One primary objective of a multi-objective optimization problem is to preserve diversity along the front. To keep diversity one needs to measure the density of solutions in a non-dominated front. The very simple way of doing it is by clustering using Euclidean distances between solutions, but the amount of computation required is very large. In NSGA-II this computation is done in a fast manner. Crowding distance is measured by adding objective-wise normalized difference between two neighboring solutions of a solution i, as shown in Figure 4. The extreme solutions are assigned an arbitrarily large crowding distance. Thereafter, solutions are sorted from largest crowding distance to lowest and the required number of solutions are picked from the top of the sorted list.

6 Proposed Modifications to NSGA-II

Though NSGA-II provides importance to the extreme solutions of the non-dominated front, the main emphasis is placed in maintaining a good diversity of solutions on the entire front. However, as

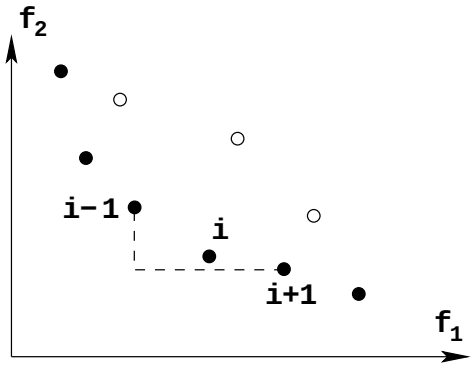

Figure 4: NSGA-II crowding distance calculation.

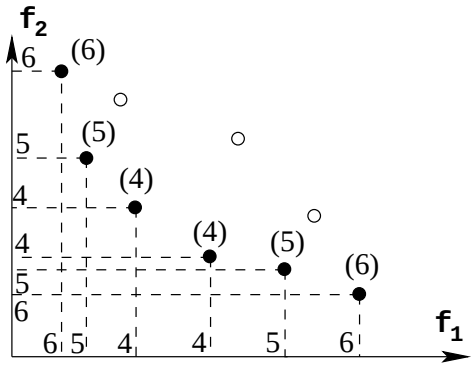

Figure 5: Extremized crowding distance calculation.

discussed above, to find the nadir point we would need to distribute NSGA-II population members around the extreme solutions of the non-dominated front. Thus, for finding the nadir point in a computationally effective manner, we need to emphasize solutions near extreme non-dominated solutions more than that is currently provided by the NSGA-II procedure. To implement such a concept, the crowding distance calculation is modified here.

6.1 Worst Crowded NSGA-II

Since the worst objective values for the Pareto-optimal solutions constitute the nadir point, we employ a crowded distance scheme which emphasizes the worst objective values front-wise. Solutions on a particular front are first sorted from minimum to maximum based on each objective (for minimization problems) and a rank equal to the position of the solution in the sorted list is assigned. Thereafter, the maximum rank assigned to a solution due to all objectives is declared as the crowding distance measure. This way, the solution with the maximum objective value of any objective gets the maximum crowded distance. Since the NSGA-II procedure emphasizes the non-dominated solutions and solutions having larger crowding distance values, all worst objective values in every front get emphasized, thereby constituting a process which may help find the nadir point.

6.2 Extremized Crowded NSGA-II

However, such an emphasis to only the worst objective solutions front-wise may have at least two difficulties: (i) it may not maintain enough diversity in the subpopulation near each worst objective vector, which may cause a premature convergence to a suboptimal solution and (ii) solutions corresponding to the individual worst objective vectors alone may not dominate a number of non-Pareto-optimal solutions, thereby finding a non-dominated front with unwanted solutions. Such a front will then make a wrong estimation of the nadir point. To avoid both these possibilities, we suggest a different crowding distance measure. Solutions on a particular front are first sorted from maximum to minimum based on each objective. A solution closer to either extreme objective vectors (minimum or maximum objective values) gets a higher rank compared to that of an intermediate solution. Thus, the extreme two solutions for every objective get a rank equal to N' (number of solutions in the front), the solutions next to these extreme solutions get a rank $(N'-1)$ and so on. Figure 5 shows this rank-assignment procedure. After a rank is assigned to a solution by each objective, the maximum value of the assigned ranks is declared as the crowding distance. The crowding distance values are shown within brackets in the figure. This procedure emphasizes the extreme solutions (minimum and maximum) of each objective and produces a hierarchy by

emphasizing the solutions closer to the extreme solutions.

For two-objective problems and for higher-objective problems having a one-dimensional Pareto-optimal front (that is, an M-objective problem having $(M-2)$ redundant objectives), this crowding distance assignment is similar to the worst crowding distance assignment scheme (as the minimum-rank solution of one objective is the maximum-rank solution of at least one other objective). However, for problems having a large-dimensional Pareto-optimal hyper-surface, the effect of extremized crowding is different from that in the worst crowding scheme. In the three-objective problem shown in Figure 2, the extremized crowding scheme will not only emphasize the extreme points A, B, and C, but also solutions on lines AB, BC, and CA and solutions near them. This has two advantages: (i) a good diversity of solutions will be maintained, thereby allowing genetic operators (crossover and mutation) to find better solutions and not causing a premature convergence and (ii) the presence of these solutions will reduce the chance of having spurious non-Pareto-optimal solutions to remain in the best non-dominated front, thereby causing a reliable (and stable) computation of the nadir point. Moreover, since the intermediate portion of the Pareto-optimal region is not emphasized, finding the extreme solutions should be quicker than the original NSGA-II, especially for problems having a large number of objectives.

6.3 VEGA Based Approach

If thought carefully, the worst crowding distance calculation method proposed above has a similarity with another multi-objective GA, known as VEGA (vector evaluated GA) [15]. In this approach, a GA population is subdivided into smaller groups equal to the number of objective functions in each generation. Individuals in the first group are reproduced based on first objective, second group based on second objective, and so on. Although the selection operator of VEGA is restricted to each group, crossover and mutation operators are applied on the entire population as usual. Because of the selective search made for each objective independently to a subpopulation, individuals having better values of each objective are emphasized. Although VEGA has a tendency to converge to the extreme solutions of the Pareto-optimal front, no domination check or no explicit diversity-preserving mechanism (as they are used in NSGA-II) is made in VEGA. In this study, we also evaluate the VEGA approach for estimating the nadir point and compare with the original NSGA-II and with the two proposed modifications of NSGA-II.

7 Test Problems

The algorithms discussed in Section 6 are tested on a number of test problems starting from two objectives and up to 20 objectives. These test problems are designed in such a way that the complexity level of convergence and shape of the Pareto-optimal front can be tuned by using different parameters and functionals. In these problems, the Pareto-optimal front can be determined analytically, thereby allowing to compute the true nadir objective vector. Since the true nadir point is known in these problems, we define the performance measure of an algorithm by simply computing the normalized Euclidean difference (D) between the true nadir point and the obtained nadir point in the objective space.

$$D = \sqrt{\sum_{i=1}^{M} \left(\frac{z_i^{\text{nad}} - z_i^{\text{est}}}{z_i^{\text{nad}} - z_i^{*}} \right)^2}, \tag{1}$$

where z_i^{nad} is the true nadir objective value for i-th objective function, z_i^* is the ideal objective value for i-th objective vector, and z_i^{est} is the estimated nadir objective value for i-th objective vector. The estimated nadir objective vector at any generation is constructed from the solutions of the best non-dominated front of the population at that generation. The above performance metric D is computed at each generation and when a value smaller than a threshold ($\eta = 0.01$ used here)

is found the simulation is terminated and the algorithm is said to be successful in finding a point close to the true nadir point.

For an application of the proposed approach to any arbitrary problem, first the ideal point ($\mathbf{z}^*$) and the worst objective vector ($\mathbf{z}^w$) can be computed by minimizing and maximizing, respectively, each objective function independently. Thereafter, the proposed EMO approach can be applied and the normalized Euclidean distance between the ideal point and the estimated nadir point can be recorded with the generation counter:

$$\text{Normalized Distance} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(\frac{z_i^{\text{est}} - z_i^*}{z_i^w - z_i^*}\right)^2}. \tag{2}$$

If in a problem, the worst objective vector $\mathbf{z}^w$ (refer to Figure 1) is the same as the nadir vector, the above normalized distance value will be one. For other scenarios, the normalized distance value will be smaller than one. When the change in this distance value is not significant for a continual number of generations, the algorithm can be terminated and the obtained nadir point can be declared. We shall demonstrate the use of this procedure in Section 7.4.

To understand the efficiency of each of the proposed algorithms, input parameters are kept fixed for a particular problem. For all problems, we use the SBX crossover with a probability of 0.9 and polynomial mutation operator with a probability of $1/n$ (n is the number of variables) and distribution index of 20 [3]. The population size and distribution index for crossover are set according to the problem and are mentioned below. Each algorithm is run 11 times, each time starting from a different random initial population, however all algorithms are started with identical initial populations. The number of generations required to satisfy the termination criterion ($D \leq \eta$) is noted for each simulation run and the best, median and worst number of generations are presented.

7.1 Two-Objective Problems

Actually, the payoff table method can be reliably used to find the nadir point for a two-objective optimization problem and there is no need to use an evolutionary approach. It is the higher-objective problems in which the payoff table and other suggested methods have difficulties and there is a strong need for other better methods. However, here we still apply the proposed EMO procedures to a few two-objective problems for completeness.

Five two-objective ZDT test problems [16] were designed using three functionals: f_1, g and h. Function f_1 is responsible for causing difficulties along Pareto-optimal front; g tests the ability to converge to the Pareto-optimal front; and h controls the shape of the Pareto-optimal front. We have chosen five test problems (ZDT1 to ZDT4 and ZDT6) here. ZDT1 is a 30-variable problem with a convex Pareto-optimal front with the nadir objective vector $(1, 1)^T$. ZDT2 is also a 30-variable problem but has a concave Pareto-optimal front. The corresponding nadir objective vector is also $(1, 1)^T$. ZDT3 problem (a 30-variable problem) has a discontinuous Pareto-optimal front and its nadir objective vector $(0.85, 1.0)^T$. ZDT4 is the most difficult among these test problems due to the presence of 99 local non-dominated fronts. A multi-objective optimization algorithm needs to overcome these local non-dominated fronts before converging to the true Pareto-optimal front. This is a 10-variable problem with the nadir point at $(1, 1)^T$. The test problem ZDT6 is also a 10-variable problem with a non-convex Pareto-optimal front. This problem causes a non-uniformity in the distribution of solutions along the Pareto-optimal front. The nadir objective vector of this problem is $(1, 0.92)^T$. The mathematical structure of these test problems is given below:

$$\begin{array}{ll} \text{Minimize} & f_1(\mathbf{x}), \\ \text{Minimize} & f_2(\mathbf{x}) = g(\mathbf{x})h(f_1(\mathbf{x}), g(\mathbf{x})). \end{array} \tag{3}$$

Three functionals f_1, g and h used to construct the ZDT problems are given below:

ZDT1: $n = 30$, $f_1(\mathbf{x}) = x_1$, $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$, $h(\mathbf{x}) = 1 - \sqrt{f_1/g}$
ZDT2: $n = 30$, $f_1(\mathbf{x}) = x_1$, $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$, $h(\mathbf{x}) = 1 - (f_1/g)^2$
ZDT3: $n = 30$, $f_1(\mathbf{x}) = x_1$, $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$, $h(\mathbf{x}) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1)$
ZDT4: $n = 10$, $f_1(\mathbf{x}) = x_1$, $g(\mathbf{x}) = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i))$,
$h(\mathbf{x}) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(10\pi f_1)$
ZDT6: $n = 10$, $f_1(\mathbf{x}) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$, $g(\mathbf{x}) = 1 + 9[(\sum_{i=2}^{n} x_i)/9]^{0.25}$,
$h(\mathbf{x}) = 1 - (f_1/g)^2$

Table 1 shows the number of generations needed to find a near nadir point (within $\eta = 0.01$) by different algorithms. For the easier problems (ZDT1, ZDT2 and ZDT3), we use a crossover index of 2 and for ZDT4 and ZDT6, we use a value of 10. It is clear from the results that the performances of the worst crowded and extremized crowded NSGA-II are more or less the same and slightly better than the original NSGA-II algorithm for more complex problems, such as ZDT4 and ZDT6.

Table 1: Comparative results for two-objective problems.

Test Problem	Pop. size	Number of generations								
		NSGA-II			Worst crowd. NSGA-II			Extr. crowd. NSGA-II		
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
ZDT1	100	32	44	80	31	41	45	31	42	48
ZDT2	100	45	56	83	44	72	97	34	67	93
ZDT3	100	33	40	55	33	40	113	28	36	45
ZDT4	100	176	197	257	148	201	224	165	191	219
ZDT6	100	137	151	161	125	130	143	126	132	135

To investigate the rate of convergence of these procedures, the best run of each algorithm is plotted in Figures 6 and 7 for problems ZDT4 and ZDT6, respectively. It is observed that

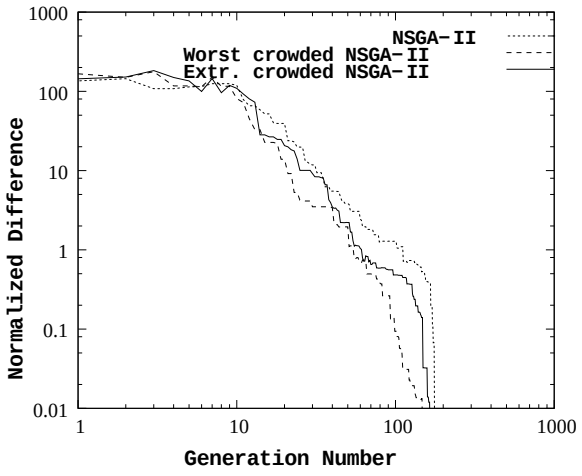

Figure 6: Normalized difference measure for ZDT4.

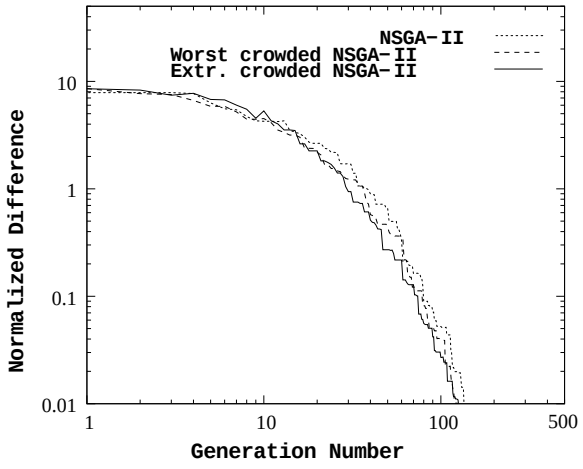

Figure 7: Normalized difference measure for ZDT6.

the convergence patterns are also almost the same for all algorithms. It is concluded that for two-objective optimization problems the use of extremized crowding or the worst crowded approach may not be necessary. The procedure of finding the complete Pareto-optimal front and then deriving

the nadir point from the front is a computationally viable approach for two-objective problems, but as discussed earlier, may be replaced by the conventional pay-off table method.

Before we present the results for problems having many objectives, let us investigate how the VEGA approach would fair in estimating the nadir point in the ZDT test problems. We use VEGA with 500 population members and run it for 1,000 generations. Even with such a large number of evaluations, the VEGA approach could not find the nadir objective vector, as shown in Table 2. The table shows that there is not much of a reduction in the normalized Euclidean difference (equation 1) of the estimated nadir point from the actual one from the initial to the final generation. Because of it's inability to solve these two-objective problems (in which the NSGA-II approach and its modifications performed well), the VEGA approach is not applied for higher-objective problems.

Table 2: Performance of the VEGA approach in estimating the nadir point on ZDT problems.

Test problem	Pop. size	Difference from true nadir point	
		Initial pop.	Final pop.
ZDT1	500	12.36	7.77
ZDT2	500	6.55	3.48
ZDT3	500	3.17	3.28
ZDT4	500	145.82	149.45
ZDT6	500	9.06	9.34

7.2 Problems with More Objectives

To test the algorithms for three and more objectives, we choose three DTLZ test problems [6]. These problems are designed in a manner so that they can be extended to any number of objectives. The first problem, DTLZ1, is constructed to have a linear Pareto-optimal front:

$$\left.\begin{array}{rl} \text{Minimize} & f_1(\mathbf{x}) = \frac{1}{2}x_1x_2\cdots x_{M-1}(1+g(\mathbf{x}_M)), \\ \text{Minimize} & f_2(\mathbf{x}) = \frac{1}{2}x_1x_2\cdots(1-x_{M-1})(1+g(\mathbf{x}_M)), \\ \vdots & \vdots \\ \text{Minimize} & f_{M-1}(\mathbf{x}) = \frac{1}{2}x_1(1-x_2)(1+g(\mathbf{x}_M)), \\ \text{Minimize} & f_M(\mathbf{x}) = \frac{1}{2}(1-x_1)(1+g(\mathbf{x}_M)), \\ \text{subject to} & 0 \le x_i \le 1, \quad \text{for } i = 1,2,\ldots,n. \end{array}\right\} \tag{4}$$

The functional $g(\mathbf{x}_M)$ requires $|\mathbf{x}_M| = k$ variables:

$$g(\mathbf{x}_M) = 100\left[|\mathbf{x}_M| + \sum_{x_i \in \mathbf{x}_M}(x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))\right]. \tag{5}$$

The total number of variables is $n = M + k - 1$. The value of k is kept 5 in this problem, so that $n = M + 4$. The Pareto-optimal hyper-plane intercepts each objective axis at 0.5. Thus, the nadir objective vector is at $(0.5, 0.5, \ldots, 0.5)^T$ and the ideal objective vector is at $(0, 0, \ldots, 0)^T$.

The Pareto-optimal front of the second test problem, DTLZ2, is a quadrant of a unit sphere centered at the origin of the objective space. The mathematical formulation of the problem is given

below:

$$\left.\begin{array}{ll}
\text{Min} & f_1(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos(x_1\pi/2)\cos(x_2\pi/2)\cdots\cos(x_{M-2}\pi/2)\cos(x_{M-1}\pi/2),\\
\text{Min} & f_2(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos(x_1\pi/2)\cos(x_2\pi/2)\cdots\cos(x_{M-2}\pi/2)\sin(x_{M-1}\pi/2),\\
\vdots & \vdots\\
\text{Min} & f_{M-1}(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos(x_1\pi/2)\sin(x_2\pi/2),\\
\text{Min} & f_M(\mathbf{x}) = (1+g(\mathbf{x}_M))\sin(x_1\pi/2),\\
 & 0 \le x_i \le 1, \quad \text{for } i = 1,2,\ldots,n,\\
\text{where} & g(\mathbf{x}_M) = \sum_{x_i\in\mathbf{x}_M}(x_i-0.5)^2.
\end{array}\right\} \quad (6)$$

The value of k is chosen to be 10, so that total number of variables is $n = M + 9$. The nadir objective vector is at $(1,1,\ldots,1)^T$ and the ideal objective vector is at $(0,0,\ldots,0)^T$.

The third test problem, DTLZ5, is somewhat modified from the original DTLZ5 and has a one-dimensional Pareto-optimal curve in the M-dimensional space:

$$\left.\begin{array}{ll}
\text{Minimize} & f_1(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos(\theta_1)\cos(\theta_2)\cdots\cos(\theta_{M-2})\cos(\theta_{M-1}),\\
\text{Minimize} & f_2(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos(\theta_1)\cos(\theta_2)\cdots\cos(\theta_{M-2})\sin(\theta_{M-1}),\\
\vdots & \vdots\\
\text{Minimize} & f_{M-1}(\mathbf{x}) = (1+g(\mathbf{x}_M))\cos(\theta_1)\sin(\theta_2),\\
\text{Minimize} & f_M(\mathbf{x}) = (1+g(\mathbf{x}_M))\sin(\theta_1),\\
\text{where} & \theta_i = \frac{\pi}{4(1+g')}\left(1+2g'x_i\right), \quad \text{for } i\in I, \quad \theta_1 = \frac{\pi}{2}x_1,\\
 & g(\mathbf{x}_M) = \sum_{x_i\in\mathbf{x}_M} 100(x_i-0.5)^2, \quad g' = \min(0.3, g(\mathbf{x}_M)),\\
 & L \le x_i \le U, \text{ for } i\in I, \quad 0\le x_i\le 1, \quad \text{for } i\notin I.
\end{array}\right\} \quad (7)$$

Here the variable index $I = \{2,3,\ldots,(M-1)\}$. The value of k is taken as 10, so that total number of variables is $n = M + 9$. The ideal objective vector is at $(0,0,\ldots,0)^T$ and the nadir objective vector is at $\left((\frac{1}{\sqrt{2}})^{M-2},(\frac{1}{\sqrt{2}})^{M-2},(\frac{1}{\sqrt{2}})^{M-3},\ (\frac{1}{\sqrt{2}})^{M-4},\ldots,(\frac{1}{\sqrt{2}})^0\right)^T$. For all DTLZ problems, we use a crossover index of 10.

7.2.1 Three-Objective DTLZ Problems

All three algorithms are run with 100 population members for three-objective DTLZ1, DTLZ2 and DTLZ5. For DTLZ5, we use bounds $L = 0.1$ and $U = 0.9$. Table 3 shows the number of generations needed to find a solution close to (within a normalized difference of 0.01) the actual nadir point. It

Table 3: Comparative results for three-objective DTLZ problems.

Test problem	Pop. size	Number of generations								
		NSGA-II			Worst crowd. NSGA-II			Extr. crowd. NSGA-II		
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
DTLZ1	100	223	366	610	171	282	345	188	265	457
DTLZ2	100	75	111	151	38	47	54	41	49	55
DTLZ5	100	63	80	104	59	74	86	62	73	88

is observed that the worst crowded NSGA-II and the extremized crowded NSGA-II perform more or less similar to each other and are somewhat better than the original NSGA-II.

7.2.2 Five-Objective DTLZ Problems

Next, we study the performance of all three NSGA-II procedures on five-objective DTLZ problems. It is now quite evident from Table 4 that the proposed modifications to the NSGA-II procedure

Table 4: Comparative results for five-objective DTLZ problems.

Test problem	Pop. size	Number of generations								
		NSGA-II			Worst crowd. NSGA-II			Extr. crowded NSGA-II		
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
DTLZ1	100	2,342	3,136	3,714	611	790	1,027	353	584	1,071
DTLZ2	100	650	2,142	5,937	139	166	185	94	114	142
DTLZ5	100	52	66	77	51	66	76	49	61	73

perform much better than the original NSGA-II. The best simulation of the original NSGA-II takes 2,342 generations to estimate the nadir point, whereas the extremized crowded NSGA-II requires only 353 generations and the worst-crowded NSGA-II requires only 611 generations for the DTLZ1 problem. In the case of DTLZ2 problem, the minimum number of generations required for NSGA-II, extremized crowded NSGA-II and the worst crowded NSGA-II are 650, 94 and 139, respectively. The median generation counts of the modified NSGA-II methods for 11 independent runs are also much better than that of the original NSGA-II. Since the original NSGA-II procedure attempts to find the entire Pareto-optimal set properly before a true estimate of the nadir point can be made, it requires more generations for problems with a large number of objectives.

The difference between the worst crowded and extremized crowded NSGA-II methods is also clear from the table. For a problem having a large number of objectives, the extremized crowded NSGA-II emphasizes both best and worst extreme solutions for each objective maintaining an adequate diversity among the population members. The NSGA-II operators are able to exploit such a diverse population and make a faster progress towards the extreme Pareto-optimal solutions needed to estimate the nadir point correctly.

These results imply that for a problem having a large number of objectives, an emphasis for the individual-best solutions (instead of all non-dominated solutions) is a faster approach for locating the nadir point. If the problem degenerates to a have a lower-dimensional Pareto-optimal front or the problem has a fewer objectives, still the proposed modifications (of emphasizing the individual-best solutions) is somewhat quicker compared to the procedure of finding the complete front using the original NSGA-II procedure. Figures 8 and 9 shows the convergence rate for the best runs of three algorithms on DTLZ1 and DTLZ2, respectively.

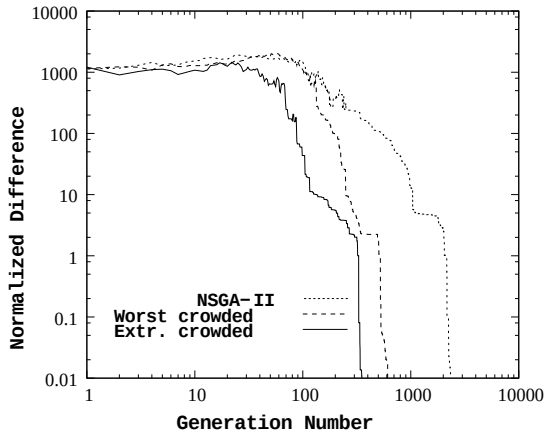

Figure 8: Performance of three methods on five-objective DTLZ1.

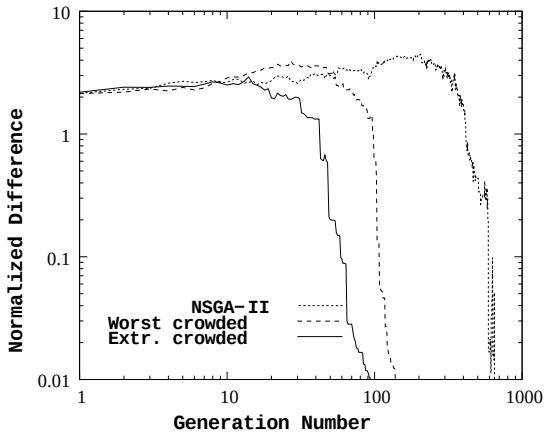

Figure 9: Performance of three methods on five-objective DTLZ2.

For the DTLZ5 problem, we change the bounds to $L = 0.4$ and $U = 0.6$. Simulation results show that there is no significant advantage of the proposed modifications, because the five-objective problem degenerates to have a one-dimensional curve as the Pareto-optimal front and all methods find it easier to locate the true nadir point quickly.

7.2.3 Ten-Objective DTLZ Problems

Table 5 presents the number of generations required to find a point close (within $\eta = 0.01$) to the nadir point by the three NSGA-II procedures for 10-objective DTLZ problems. It is clear that the

Table 5: Comparative results for 10-objective DTLZ problems.

Test problem	Pop size	Number of generations								
		NSGA-II			Worst crowd. NSGA-II			Extr. crowd. NSGA-II		
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
DTLZ1	200	17,581	21,484	33,977	1,403	1,760	2,540	1,199	1,371	1,790
DTLZ2	200	–	–	–	520	823	1,456	388	464	640
DTLZ5	200	45	53	60	43	53	57	45	51	64

extremized crowded NSGA-II performs an order of magnitude better than the original NSGA-II and is also better than the worst crowded NSGA-II. Both DTLZ1 and DTLZ2 problems have 10-dimensional Pareto-optimal fronts and the extremized crowded NSGA-II makes a good balance of maintaining diversity and emphasizing worst objective vectors so that the nadir point estimation is quick. In the case of 10-objective DTLZ2 problem, the original NSGA-II could not even find the nadir objective vector after 50,000 generations (and achieved a normalized difference measure (equation 1) of 5.936). Figure 10 shows a typical convergence pattern of the proposed and original NSGA-II algorithms on 10-objective DTLZ1. The plots demonstrate that for a large number of

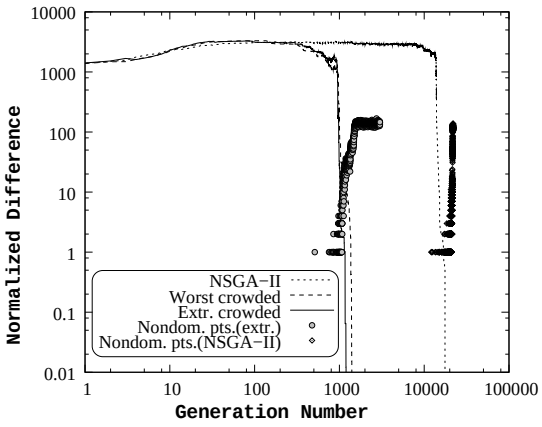

Figure 10: Performance of three methods on 10-objective DTLZ1.

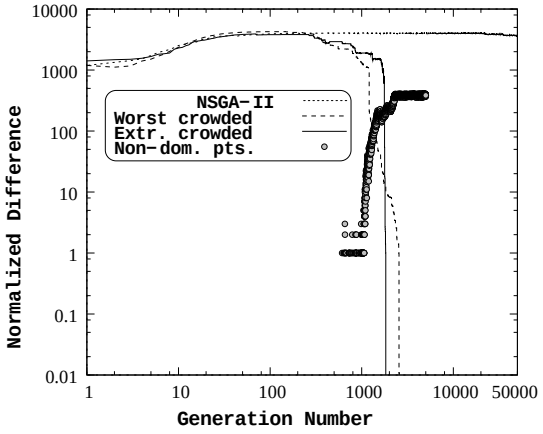

Figure 11: Performance of three methods on 15-objective DTLZ1.

generations the estimated nadir point is away from the true nadir point, but after some generations (around 1,000 in this problem) the estimated nadir point comes quickly near the true point. To understand the dynamics of the movement of the extremized crowded NSGA-II population with generation counter, we have counted the number of solutions in the population which dominate

the true nadir point and plotted in Figures 10 and 11. In DTLZ1, it is seen that the first point dominating the actual nadir point appears in the population at around 750 generations and since then when an adequate number of such solutions appear in the population, the population very quickly converges to the correct Pareto-optimal regions for estimating the nadir point. A similar phenomenon occurs for the worst crowded NSGA-II, but is not shown here for clarity. The figure also plots the number of non-dominated points for the original NSGA-II and and a similar dynamics is observed.

In case of DTLZ5 problem, we change the bounds to $L = 0.45$ and $U = 0.55$. The performance of all three algorithms are almost the same, due to the reasons outlined in previous subsection.

7.2.4 15-Objective DTLZ Problems

Next, we apply all three NSGA-II procedures on 15-objective DTLZ1 and DTLZ2 problems. Table 6 shows number of generations needed to find a point within a normalized Euclidean difference of 0.01 from the true nadir point. The original NSGA-II simulations were run for a maximum of 50,000 generations and the best normalized Euclidean difference measure (equation 1) comes at 3554.587 and 8.632 (which is much larger than zero) for DTLZ1 and DTLZ2, respectively, whereas both proposed modifications of NSGA-II are able to find a good estimate of the nadir point in a reasonable number of evaluations for both problems. Figure 11 shows the normalized difference

Table 6: Comparative results for 15-objective DTLZ problems.

Test problem	Pop. size	Number of generations								
		NSGA-II			Worst crowd. NSGA-II			Extr. crowd. NSGA-II		
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst
DTLZ1	500	–	–	–	2,567	3,144	8,645	1,829	2,372	2,841
DTLZ2	500	–	–	–	988	1,166	1,729	693	858	1,620

measure obtained using all three methods on 15-objective DTLZ1. The original NSGA-II procedure is unable to come closer to the nadir point. A similar convergence pattern as that observed in the 10-objective DTLZ1 is also observed here.

7.3 Scale-up Performance

We now compare the overall function evaluations needed to estimate the nadir point within a normalized difference measure (equation 1) of 0.01 from the true nadir point for DTLZ1 and DTLZ2 test problems by the extremized crowded NSGA-II. Although we presented the results for problems up to 15 objectives in earlier subsections, here we show scale-up performance of the extremized crowded NSGA-II up to 20 objectives. Figure 12 plots the best, median, and worst of 11 runs of extremized crowded NSGA-II and original NSGA-II on DTLZ1. First of all, the figure clearly shows that the original NSGA-II is unable to scale up to 10 or 15 objectives. In the case of 15-objective DTLZ1, the original NSGA-II is more than two orders of magnitude worse than that of the extremized crowded NSGA-II. For the 15-objective DTLZ1 problem, the original NSGA-II with more than 200 million function evaluations obtained a frontier having normalized difference of 12.871 from the true nadir point.

Figure 13 shows the performance on DTLZ2. After 670 million function evaluations, the original NSGA-II is still not able to come close (with a normalized difference of 0.01) to the true nadir point. However, the extremized crowded NSGA-II takes an average 429,000 evaluations. Because of the computational inefficiencies associated with the original NSGA-II approach, we do not perform any simulation for 15 or more objective DTLZ2 problem.

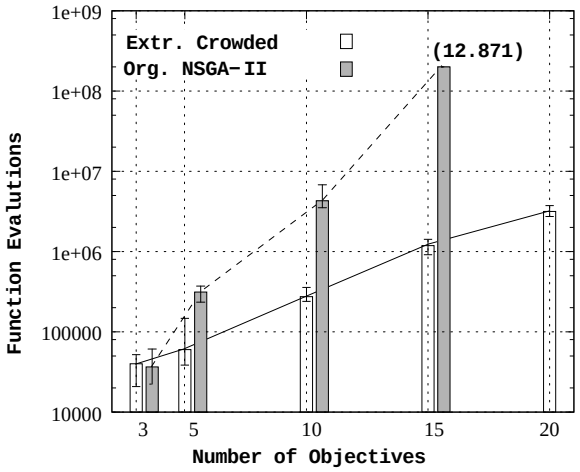

Figure 12: Function evaluations versus number of objectives for DTLZ1.

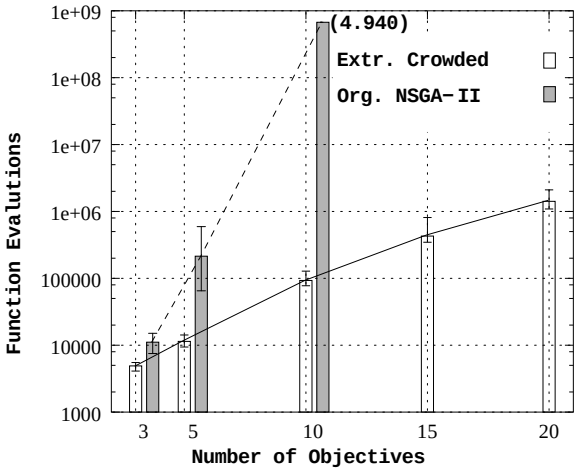

Figure 13: Function evaluations versus number of objectives for DTLZ2.

The nature of the plots for the extremized crowded NSGA-II is found to be sub-linear on logarithmic axes. This indicates that a lesser than exponential scaling property of the proposed extremized crowded NSGA-II. It is important to highlight here that estimating the nadir point requires identification of the worst objective vector corresponding to the Pareto-optimal solutions. Since this requires that an evolutionary procedure first puts its population members on the Pareto-optimal front, an adequate computational effort must be spent to achieve this task. However, the earlier tables have indicated that the computational effort needed with the extremized crowded NSGA-II is much smaller compared to the original NSGA-II, which attempts to find the entire Pareto-optimal front and then constructs the nadir point.

7.4 Two Linear Problems

Next, we consider two, three-objective linear programming problems described elsewhere [11]. The study explained why it was difficult to use the payoff table method to find the nadir point for these two problems. We consider each problem in turn.

7.4.1 Problem Pekka1

The first problem is a seven-variable problem with one equality constraint.

$$\begin{aligned} \text{Maximize} \quad & 11x_2 + 11x_3 + 12x_4 + 9x_5 + 9x_6 - 9x_7, \\ \text{Maximize} \quad & 11x_1 + 11x_3 + 9x_4 + 12x_5 + 9x_6 - 9x_7, \\ \text{Maximize} \quad & 11x_1 + 11x_2 + 9x_4 + 9x_5 + 12x_6 + 12x_7, \\ \text{Subject to} \quad & \textstyle\sum_{i=1}^{7} x_i = 1, \\ & x_i \geq 0, \quad i = 1, 2, \ldots, 7. \end{aligned} \tag{8}$$

To use an evolutionary optimization procedure, we bound $x_i \in [0, 1]$ and perform the following repair mechanism: $x_i \leftarrow x_i / \sum_{i=1}^{7} x_i$ to make every solution a feasible one. All three NSGA-II approaches are applied with a standard parameter setting: (population size = 100, crossover probability = 0.9, crossover index = 10, mutation probability = 1/7, and mutation index = 20). Figure 14 shows the populations obtained with NSGA-II and the extremized crowded NSGA-II approaches. The nadir point obtained using both approaches is $(0, 0, 0)^T$. It is interesting to note how the extremized crowded NSGA-II is able to find both minimum and maximum objective values

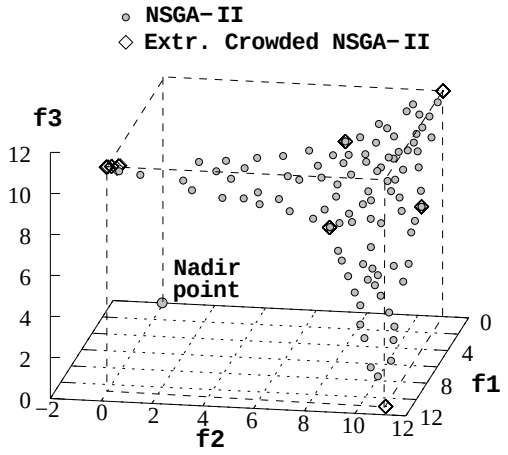

Figure 14: Populations obtained for problem Pekka1 using extremized crowded and original NSGA-II approaches.

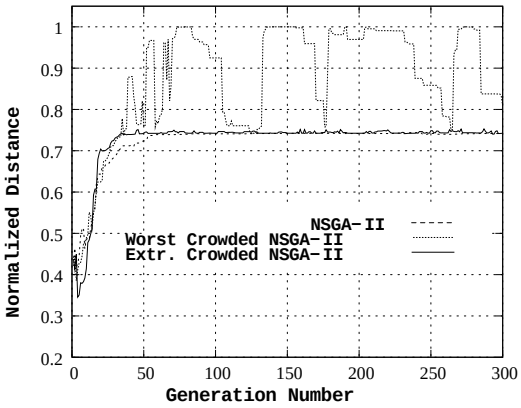

Figure 15: Normalized distance value stabilizes somewhat quicker with the extremized Crowded NSGA-II in problem Pekka1.

corresponding to the Pareto-optimal solutions, the presence of which allows a correct estimation of the nadir point.

In Figure 15, we plot the normalized distance metric (equation 2) versus generation number. In problem Pekka1, we first find the ideal vector ($\mathbf{z}^* = (12, 12, 12)^T$ and worst objective vector $\mathbf{z}^w = (-9, -9, 0)^T$ using a single-objective evolutionary algorithm and then calculate the distance measure at every generation. For the nadir point, the normalized distance measure is 0.7423. The figure shows that the extremized crowded NSGA-II reaches somewhat quicker near this desired value compared to the original NSGA-II. Due to continuous changes in the extreme solutions in their vicinities the true nadir point changes marginally. However, the performance of the worst crowded NSGA-II is poor. The method is not able to make a stable estimate of the nadir point. Since only the worst solutions are emphasized in this method, the frontier with only these three solutions causes a number of non-Pareto-optimal solutions to become non-dominated and to remain in the population. The presence of these spurious solutions makes a wrong estimate of the nadir point. Although a point close to the nadir point was found repeatedly, this point is difficult to retain. On the other hand, the presence of individual best objective solutions (shown in Figure 14) and often their neighboring solutions in the population, ensured in the extremized crowded NSGA-II, reduces the presence of such spurious solutions, thereby causing a stable and correct estimate of the nadir point.

7.4.2 Problem Pekka2

The second problem is formulated as follows:

$$\begin{aligned} \text{Maximize} \quad & (x_1, x_2, x_3), \\ \text{Subject to} \quad & x_1 + 2x_2 + 2x_3 \leq 8, \\ & 2x_1 + 2x_2 + x_3 \leq 8, \\ & 3x_1 - 2x_2 + 4x_3 \leq 12, \\ & x_i \geq 0, \quad i = 1, 2, 3. \end{aligned} \tag{9}$$

Although no upper bound on decision variables is given in the above formulation, we have used an upper bound $x_i \leq 10$. The constraints are handled using the constraint-tournament selection procedure [4, 3], in which the domination principle given in definition 1 is modified to emphasize feasible solutions over infeasible solutions. The original study [11] identified the nadir point as

$(0, 0, 0)^T$. Figure 16 shows the Pareto-optimal front (shown dark-shaded) and two populations obtained using NSGA-II and extremized crowded NSGA-II. Although NSGA-II makes a good

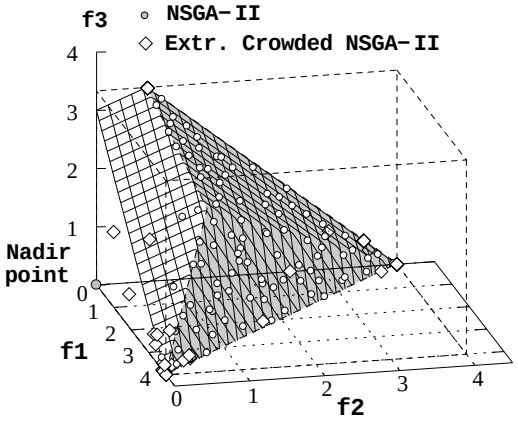

Figure 16: Populations obtained for problem Pekka2 using extremized crowded and the original NSGA-II approaches.

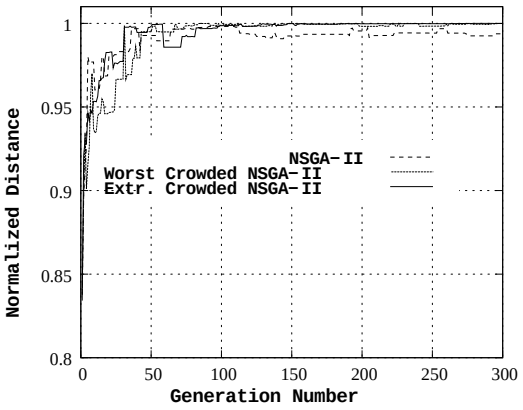

Figure 17: Normalized distance value stabilizes somewhat quicker for the extremized Crowded NSGA-II in problem Pekka2.

approximation of the Pareto-optimal front, the extermized crowded NSGA-II finds solutions near the extreme solutions $(4, 0, 0)^T$, $(0, 4, 0)^T$ and $(0, 0.667, 3.333)^T$. These extreme solutions allow an appropriate computation of the nadir point. Since NSGA-II deals with a finite population, it also finds a few non-Pareto-optimal solutions which unfortunately stays non-dominated with the other NSGA-II solutions. Figure 17 shows the generation-wise plot of the normalized distance measure with $\mathbf{z}^w = (0, 0, 0)^T$, and $\mathbf{z}^* = (4, 4, 3.333)^T$. Since the worst point is identical to the nadir point in this problem, the final normalized distance measure takes a value of one. The figure also indicates that although all three methods are able to make a good estimate of the nadir point almost at the same time, the two modified NSGA-II approaches are better able to maintain the point with generation than the original NSGA-II procedure. Around 100 generations are needed to make a stable estimation of the nadir point in this problem.

8 Conclusions

In this paper, we have proposed a couple of modifications to the NSGA-II procedure for estimating the nadir point in a multi-objective optimization problem. By definition, a nadir point is constructed with the worst objective values corresponding to the Pareto-optimal solutions. Since this requires finding the Pareto-optimal solutions, it is a difficult task. Both the proposed methodologies emphasize the extreme solutions corresponding to each objective front-wise. Since intermediate Pareto-optimal solutions are not important in this task, both procedures have been found to be capable of making a quicker estimate of the nadir point than the original NSGA-II procedure for a number of test problems having two to 20 objectives. The proposed procedures have also been compared with a VEGA-based evolutionary algorithm, which was designed to emphasize the individual best solutions. However, the procedure has not performed well. Based on the study, we conclude the following:

1. Emphasizing both best and worst objective values front-wise has been found to be a better approach than emphasizing only the worst objective values front-wise for solving large-dimensional problems. Since the former approach maintains a diverse set of solutions near worst front-wise objective values, the search is better.

2. The computational effort to estimate the nadir point has been observed to be much better (more than an order of magnitude) for many objectives than the original NSGA-II procedure.

3. For a fewer objectives (up to three-objective) and DTLZ5 problems with a lower-dimensional Pareto-optimal front, both proposed approaches have been observed to perform well. The performance of original NSGA-II has also been found to be similar to the proposed approaches.

Thus, for smaller-objective problems (say up to three-objective or so) and problems having a low-dimensional Pareto-optimal front, there is no motivation to find the nadir point explicitly using the evolutionary approaches. An use of NSGA-II or an equivalent method may be adequate to find a well-represented set of Pareto-optimal solutions, which can then be used for a decision-making. However, if the decision-maker prefers to use an interactive method, the proposed methodologies can also be used to estimate the nadir point for the purpose. On the other hand, for many objectives, the NSGA-II procedure requires a huge number of evaluations to find a representative set of Pareto-optimal solutions. For these problems, the nadir point may be estimated quickly and reliably using the proposed extremized crowded NSGA-II. Thereafter, a classical procedure (using both ideal and nadir points) can be applied interactively with a decision-maker to find a desired Pareto-optimal solution.

Acknowledgements

Authors acknowledge the support provided by STMicroelectronics for performing this study.

References

[1] R. Benayoun, J. de Montgolfier, J. Tergny, and P. Laritchev. Linear programming with multiple objective functions: Step method (STEM). *Mathematical Programming*, 1(3):366–375, 1971.

[2] C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer Academic Publishers, 2002.

[3] K. Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.

[4] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[5] K. Deb and S. Chaudhuri. I-EMO: An interactive evolutionary multi-objective optimization tool. Technical Report KanGAL Report Number 2005005, Department of Mechanical Engineering, IIT Kanpur, India, 2005. Also in Proceedings of Pattern Recognition in Machine Intelligence (PReMI'05), Springer.

[6] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multi-objective optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization*, pages 105–145. London: Springer-Verlag, 2005.

[7] M. I. Dessouky, M. Ghiassi, and W. J. Davis. Estimates of the minimum nondominated criterion values in multiple-criteria decision-making. *Engineering Costs and Production Economics*, 10:95–104, 1986.

[8] M. Ehrgott and D. Tenfelde-Podehl. Computation of ideal and nadir values and implications for their use in mcdm methods. *European Journal of Operational Research*, 151:119–139, 2003.

[9] D. E. Goldberg. *Genetic Algorithms for Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[10] H. Iserman and R. E. Steuer. Computational experience concerning payoff tables and minimum criterion values over the efficient set. *European Journal of Operational Research*, 33(1):91–97, 1988.

[11] P. Korhonen, S. Salo, and R. Steuer. A heuristic for estimating nadir criterion values in multiple objective linear programming. *Operations Research*, 45(5):751–757, 1997.

[12] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer, Boston, 1999.

[13] K. Miettinen. Graphical illustration of pareto optimal solutions. In T. Tanino, T. Tanaka, and M. Inuiguchi, editors, *Multi-Objective Programming and Goal Programming: Theory and Applications*, pages 197–202. Springer-Verlag, Berlin, Heidelberg, 2003.

[14] K. Miettinen, M. M. Mäkelä, and K. Kaario. Experiments with classification-based scalarizing functions in interactive multiobjective optimization.

[15] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100, 1985.

[16] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, 8(2):125–148, 2000.