

# An Evolutionary Many-Objective Optimization Algorithm Using Reference-point Based Non-dominated Sorting Approach, Part I: Solving Problems with Box Constraints

Kalyanmoy Deb, *Fellow, IEEE* and Himanshu Jain

**Abstract**—Having developed multi-objective optimization algorithms using evolutionary optimization methods and demonstrated their niche on various practical problems involving mostly two and three objectives, there is now a growing need for developing evolutionary multi-objective optimization (EMO) algorithms for handling many-objective (having four or more objectives) optimization problems. In this paper, we recognize a few recent efforts and discuss a number of viable directions for developing a potential EMO algorithm for solving many-objective optimization problems. Thereafter, we suggest a reference-point based many-objective NSGA-II (we call it NSGA-III) that emphasizes population members which are non-dominated yet *close* to a set of supplied reference points. The proposed NSGA-III is applied to a number of many-objective test problems having two to 15 objectives and compared with two versions of a recently suggested EMO algorithm (MOEA/D). While each of the two MOEA/D methods works well on different classes of problems, the proposed NSGA-III is found to produce satisfactory results on all problems considered in this study. This paper presents results on unconstrained problems and the sequel paper considers constrained and other specialties in handling many-objective optimization problems.

**Index Terms**—Many-objective optimization, evolutionary computation, large dimension, NSGA-III, non-dominated sorting, multi-criterion optimization.

## I. INTRODUCTION

Evolutionary multi-objective optimization (EMO) methodologies have amply shown their niche in finding a set of well-converged and well-diversified non-dominated solutions in different two and three-objective optimization problems since the beginning of nineties. However, in most real-world problems involving multiple stake-holders and functionalities, there often exists many optimization problems that involve four or more objectives, sometimes demanding to have 10 to 15 objectives [1], [2]. Thus, it is not surprising that handling a large number of objectives had been one of the main research activities in EMO for the past few years. Many-objective

problems pose a number of challenges to any optimization algorithm, including an EMO. First and foremost, the proportion of non-dominated solutions in a randomly chosen set of objective vectors becomes exponentially large with an increase of number of objectives. Since the non-dominated solutions occupy most of the population slots, any elite-preserving EMO faces a difficulty in accommodating adequate number of new solutions in the population. This slows down the search process considerably [3], [4]. Second, implementation of a diversity-preservation operator (such as the crowding distance operator [5] or clustering operator [6]) becomes a computationally expensive operation. Third, visualization of a large-dimensional front becomes a difficult task, thereby causing difficulties in subsequent decision-making tasks and in evaluating the performance of an algorithm. For this purpose, the performance metrics (such as hyper-volume measure [7] or other metrics [3], [8]) are either computationally too expensive or may not be meaningful.

An important question to ask is then ‘Are EMOs useful for many-objective optimization problems?’. Although the third difficulty related to visualization and performance measures mentioned above cannot be avoided, some algorithmic changes to the existing EMO algorithms may be possible to address the first two concerns. In this paper, we review some of the past efforts [9], [10], [11], [12], [13], [14] in devising many-objective EMOs and outline some viable directions for designing efficient many-objective EMO methodologies. Thereafter, we propose a new method that uses the framework of NSGA-II procedure [5] but works with a set of supplied or predefined reference points and demonstrate its efficacy in solving two to 15-objective optimization problems. In this paper, we introduce the framework and restrict to solving unconstrained problems of various kinds, such as having normalized, scaled, convex, concave, disjointed, and focusing on a part of the Pareto-optimal front. Practice may offer a number of such properties to exist in a problem. Therefore, an adequate test of an algorithm for these eventualities remains as an important task. We compare the performance of the proposed NSGA-III with two versions of an existing many-objective EMO (MOEA/D [10]), as the method is somewhat similar to the proposed method. Interesting insights about working of both versions of MOEA/D and NSGA-III are revealed. The proposed NSGA-III is also evaluated for its use in a few other interesting multi-objective optimization and

K. Deb is with Department of Electrical and Computer Engineering, Michigan State University, 428 S. Shaw Lane, 2120 EB, East Lansing, MI 48824, USA, e-mail: kdeb@egr.msu.edu (see <http://www.egr.msu.edu/~kdeb>). Prof. Deb is also a visiting professor at Aalto University School of Business, Finland and University of Skövde, Sweden.

H. Jain is with Eaton Corporation, Pune 411014, India, email: himanshu.j689@gmail.com.

Copyright (c) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to [pubs-permissions@ieee.org](mailto:pubs-permissions@ieee.org).

decision-making tasks. In the sequel of this paper, we suggest an extension of the proposed NSGA-III for handling many-objective constrained optimization problems and a few other special and challenging many-objective problems.

In the remainder of this paper, we first discuss the difficulties in solving many-objective optimization problems and then attempt to answer the question posed above about the usefulness of EMO algorithms in handling many objectives. Thereafter, in Section III, we present a review of some of the past studies on many-objective optimization including a recently proposed method MOEA/D [10]. Then, in Section IV, we outline our proposed NSGA-III procedure in detail. Results on normalized DTLZ test problems up to 15 objectives using NSGA-III and two versions of MOEA/D are presented next in Section V-A. Results on scaled version of DTLZ problems suggested here are shown next. Thereafter in subsequent sections, NSGA-III procedure is tested on different types of many-objective optimization problems. Finally, NSGA-III is applied to two practical problems involving three and nine objective problems in Section VII. Conclusions of this extensive study are drawn in Section VIII.

## II. MANY-OBJECTIVE PROBLEMS

Loosely, many-objective problems are defined as problems having four or more objectives. Two and three-objective problems fall into a different class as the resulting Pareto-optimal front, in most cases, in its totality can be comprehensively visualized using graphical means. Although a strict upper bound on the number of objectives for a many-objective optimization problem is not so clear, except a few occasions [15], most practitioners are interested in a maximum of 10 to 15 objectives. In this section, first, we discuss difficulties that an existing EMO algorithm may face in handling many-objective problems and investigate if EMO algorithms are useful at all in handling a large number of objectives.

### A. Difficulties in Handling Many Objectives

It has been discussed elsewhere [4], [16] that the current state-of-the-art EMO algorithms that work under the principle of domination [17] may face the following difficulties:

- 1) *A large fraction of population is non-dominated:* It is well-known [3], [16] that with an increase in number of objectives, an increasingly larger fraction of a randomly generated population becomes non-dominated. Since most EMO algorithms emphasize non-dominated solutions in a population, in handling many-objective problems, there is not much room for creating new solutions in a generation. This slows down the search process and therefore the overall EMO algorithm becomes inefficient.
- 2) *Evaluation of diversity measure becomes computationally expensive:* To determine the extent of crowding of solutions in a population, the identification of neighbors becomes computationally expensive in a large-dimensional space. Any compromise or approximation in diversity estimate to make the computations faster

may cause an unacceptable distribution of solutions at the end.

- 3) *Recombination operation may be inefficient:* In a many-objective problem, if only a handful of solutions are to be found in a large-dimensional space, solutions are likely to be widely distant from each other. In such a population, the effect of recombination operator (which is considered as a key search operator in an EMO) becomes questionable. Two distant parent solutions are likely to produce offspring solutions that are also distant from parents. Thus, special recombination operators (mating restriction or other schemes) may be necessary for handling many-objective problems efficiently.
- 4) *Representation of trade-off surface is difficult:* It is intuitive to realize that to represent a higher dimensional trade-off surface, exponentially more points are needed. Thus, a large population size is needed to represent the resulting Pareto-optimal front. This causes difficulty for a decision-maker to comprehend and make an adequate decision to choose a preferred solution.
- 5) *Performance metrics are computationally expensive to compute:* Since higher-dimensional sets of points are to be compared against each other to establish the performance of one algorithm against another, a larger computational effort is needed. For example, computing hyper-volume metric requires exponentially more computations with the number of objectives [18], [19].
- 6) *Visualization is difficult:* Finally, although it is not a matter related to optimization directly, eventually visualization of a higher-dimensional trade-off front may be difficult for many-objective problems.

The first three difficulties can only be alleviated by certain modifications to existing EMO methodologies. The fourth, fifth, and sixth difficulties are common to all many-objective optimization problems and we do not address them adequately here.

### B. EMO Methodologies for Handling Many-Objective Problems

Before we discuss the possible remedies for three difficulties mentioned above, here we highlight two different many-objective problem classes for which existing EMO methodologies can still be used.

First, existing EMO algorithms may still be useful in finding a preferred subset of solutions (a partial set) from the complete Pareto-optimal set. Although the preferred subset will still be many-dimensional, since the targeted solutions are focused in a small region on the Pareto-optimal front, most of the above difficulties will be alleviated by this principle. A number of MCDM-based EMO methodologies are already devised for this purpose and results on as large as 10-objective problems have shown to perform well [20], [21], [22], [23].

Second, many problems in practice, albeit having many objectives, often degenerate to result in a low-dimensional Pareto-optimal front [4], [11], [24], [25]. In such problems, identification of redundant objectives can be integrated with an EMO to find the Pareto-optimal front that is low-dimensional.

Since the ultimate front is as low as two or three-dimensional, existing EMO methodologies should work well in handling such problems. A previous study of NSGA-II with a principal component analysis (PCA) based procedure [4] was able to solve as large as 50-objective problems having a two-objective Pareto-optimal front.

### C. Two Ideas for a Many-Objective EMO

Keeping in mind the first three difficulties associated with a domination-based EMO procedure, two different strategies can be considered to alleviate the difficulties:

- 1) *Use of a special domination principle:* The first difficulty mentioned above can be alleviated by using a special domination principle that will adaptively discretize the Pareto-optimal front and find a well-distributed set of points. For example, the use of  $\epsilon$ -domination principle [26], [27] will make all points within  $\epsilon$  distance from a set of Pareto-optimal points  $\epsilon$ -dominated and hence the process will generate a finite number of Pareto-optimal points as target. Such a consideration will also alleviate the second difficulty of diversity preservation. The third difficulty can be taken care of by the use of a mating restriction scheme or a special recombination scheme in which near-parent solutions are emphasized (such as SBX with a large distribution index [28]). Other special domination principles [29], [30] can also be used for this purpose. Aguirre and Tanaka [31] and Sato et al. [32] suggested the use of a subset of objectives for dominance check and using a different combination of objectives in every generation. The use of fixed cone-domination [33], [34] or variable cone-domination [35] principles can also be tried. These studies were made in the context of low-dimensional problems and their success in solving many-objective optimization is yet to be established.
- 2) *Use of a predefined multiple targeted search:* It has been increasingly clear that it is too much to expect from a single population-based optimization algorithm to have convergence of its population near the Pareto-optimal front and simultaneously its distributed uniformly around the entire front in a large-dimensional problem. One way to handle such many-objective optimization problems would be to aid the diversity maintenance issue by some external means. This principle can directly address the second difficulty mentioned above. Instead of searching the entire search space for Pareto-optimal solutions, multiple predefined targeted searches can be set by an algorithm. Since optimal points are found corresponding to each of the targeted search task, the first difficulty of dealing with a large non-dominated set is also alleviated. The recombination issue can be addressed by using a mating restriction scheme in which two solutions from neighboring targets are participated in the recombination operation. Our proposed algorithm (NSGA-III) is based on this principle and thus we discuss this aspect in somewhat more detail. We suggest two different ways to implement the predefined multiple targeted search principle:

- a) A set of predefined *search directions* spanning the entire Pareto-optimal front can be specified beforehand and multiple searches can be performed along each direction. Since the search directions are widely distributed, the obtained optimal points are also likely to be widely distributed on the Pareto-optimal front in most problems. Recently proposed MOEA/D procedure [10] uses this concept.
- b) Instead of multiple search directions, multiple predefined *reference points* can be specified for this purpose. Thereafter, points corresponding to each reference point can be emphasized to find set of widely distributed set of Pareto-optimal points. A few such implementations were proposed recently [36], [37], [38], [14], and this paper suggests another approach extending and perfecting the algorithm proposed in the first reference [36].

## III. EXISTING MANY-OBJECTIVE OPTIMIZATION ALGORITHMS

Garza-Fabre, Pulido and Coello [16] suggested three single-objective measures by using differences in individual objective values between two competing parents and showed that in 5 to 50-objective DTLZ1, DTLZ3 and DTLZ6 problems the convergence property can get enhanced compared to a number of existing methods including the usual Pareto-dominance based EMO approaches. Purshouse and Fleming [39] clearly showed that diversity preservation and achieving convergence near the Pareto-optimal front are two contradictory goals and usual genetic operators are not adequate to attain both goals simultaneously, particularly for many-objective problems. Another study [40] extends NSGA-II by adding diversity-controlling operators to solve six to 20-objective DTLZ2 problems. Köppen and Yoshida [41] claimed that NSGA-II procedure in its originality is not suitable for many-objective optimization problems and suggested a number of metrics that can potentially replace NSGA-II's crowding distance operator for better performance. Based on simulation studies on two to 15-objective DTLZ2, DTLZ3 and DTLZ6 problems, they suggested to use a substitute assignment distance measure as the best strategy. Hadka and Reed [42] suggested an ensemble-based EMO procedure which uses a suitable recombination operator adaptively chosen from a set of eight to 10 different pre-defined operators based on their generation-wise success rate in a problem. It also uses  $\epsilon$ -dominance concept and an adaptive population sizing approach that is reported to solve up to eight-objective test problems successfully. Bader and Zitzler [43] have suggested a fast procedure for computing sample-based hyper-volume and devised an algorithm to find a set of trade-off solutions for maximizing the hyper-volume. A growing literature on approximate hyper-volume computation [44], [18], [45] may make such an approach practical for solving many-objective problems.

The above studies analyze and extend previously-suggested evolutionary multi-objective optimization algorithms for their suitability to solving many-objective problems. In most cases,



the results are promising and the suggested algorithms must be tested on other more challenging problems than the usual normalized test problems such as DTLZ problems. They must also be tried on real-world problems. In the following paragraphs, we describe a recently proposed algorithm which fits well with our description of a many-objective optimization algorithm given in Section II-C and closely matches with our proposed algorithm.

MOEA/D [10] uses a predefined set of weight vectors to maintain a diverse set of trade-off solutions. For each weight vector, the resulting problem is called a sub-problem. To start with, every population member (with size same as the number of weight vectors) is associated with a weight vector randomly. Thereafter, two solutions from neighboring weight vectors (defined through a niching parameter ( $T$ )) are mated and an offspring solution is created. The offspring is then associated with one or more weight vectors based on a performance metric. Two metrics are suggested in the study. A penalized distance measure of a point from the ideal point is formed by weighted sum (weight  $\theta$  is another algorithmic parameter) of perpendicular distance ( $d_2$ ) from the reference direction and distance ( $d_1$ ) along the reference direction:

$$\text{PBI}(\mathbf{x}, \mathbf{w}) = d_1 + \theta d_2. \quad (1)$$

We call this procedure here as MOEA/D-PBI. The second approach suggested is the use of Tchebycheff metric using a utopian point  $\mathbf{z}^*$  and the weight vector  $\mathbf{w}$ :

$$\text{TCH}(\mathbf{x}, \mathbf{w}, \mathbf{z}^*) = \max_{i=1}^M w_i |f_i(\mathbf{x}) - z_i^*|. \quad (2)$$

In reported simulations [10], the ideal point was used as  $\mathbf{z}^*$  and zero-weight scenario is handled by using a small number. We call this procedure as MOEA/D-TCH here. An external population maintains the non-dominated solutions. The first two difficulties mentioned earlier are negotiated by using an explicit set of weight vectors to find points and the third difficulty is alleviated by using a mating restriction scheme. Simulation results were shown for two and three-objective test problems only and it was concluded that MOEA/D-PBI is better for three-objective problems than MOEA/D-TCH and the performance of MOEA/D-TCH improved with an objective normalization process using population minimum and maximum objective values. Both versions of MOEA/D require to set a niching parameter ( $T$ ). Based on some simulation results on two and three-objective problems, authors suggested the use of a large fraction of population size as  $T$ . Additionally, MOEA/D-PBI requires an appropriate setting of an additional parameter – penalty parameter  $\theta$ , for which authors have suggested a value of 5.

A later study by the developers of MOEA/D suggested the use of differential evolution (DE) to replace genetic recombination and mutation operators. Also, further modifications were done in defining neighborhood of a particular solution and in replacing parents in a given neighborhood by the corresponding offspring solutions [46]. We call this method as MOEA/D-DE here. Results on a set of mostly two and a three-objective linked problems [47] showed better performance with MOEA/D-DE compared to other algorithms.

As mentioned, MOEA/D is a promising approach for many-objective optimization as it addresses some of the difficulties mentioned above well, but the above-mentioned MOEA/D studies did not quite explore their suitability to a large number of objectives. In this paper, we apply them to problems having up to 15 objectives and evaluate their applicability to truly many-objective optimization problems and reveal interesting properties of these algorithms.

Another recent study [14] follows our description of a many-objective optimization procedure. The study extends the NSGA-II procedure to suggest a hybrid NSGA-II (HN algorithm) for handling three and four-objective problems. Combined population members are projected on a hyper-plane and a clustering operation is performed on the hyper-plane to select a desired number of clusters which is user-defined. Thereafter, based on the diversity of the population, either a local search operation on a random cluster member is used to move the solution closer to the Pareto-optimal front or a diversity enhancement operator is used to choose population members from all clusters. Since no targeted and distributed search is used, the approach is more generic than MOEA/D or the procedure suggested in this paper. However, the efficiency of HN algorithm for problems having more than four objectives is yet to be investigated to suggest its use for many-objective problems, in general. We now describe our proposed algorithm.

#### IV. PROPOSED ALGORITHM: NSGA-III

The basic framework of the proposed many-objective NSGA-II (or NSGA-III) remains similar to the original NSGA-II algorithm [5] with significant changes in its selection mechanism. But unlike in NSGA-II, the maintenance of diversity among population members in NSGA-III is aided by supplying and adaptively updating a number of well-spread reference points. For completeness, we first present a brief description of the original NSGA-II algorithm.

Let us consider  $t$ -th generation of NSGA-II algorithm. Suppose the parent population at this generation is  $P_t$  and its size is  $N$ , while the offspring population created from  $P_t$  is  $Q_t$  having  $N$  members. The first step is to choose the best  $N$  members from the combined parent and offspring population  $R_t = P_t \cup Q_t$  (of size  $2N$ ), thus allowing to preserve elite members of the parent population. To achieve this, first, the combined population  $R_t$  is sorted according to different non-domination levels ( $F_1$ ,  $F_2$  and so on). Then, each non-domination level is selected one at a time to construct a new population  $S_t$ , starting from  $F_1$ , until the size of  $S_t$  is equal to  $N$  or for the first time exceeds  $N$ . Let us say the last level included is the  $l$ -th level. Thus, all solutions from level  $(l+1)$  onwards are rejected from the combined population  $R_t$ . In most situations, the last accepted level ( $l$ -th level) is only accepted partially. In such a case, only those solutions that will maximize the diversity of the  $l$ -th front are chosen. In NSGA-II, this is achieved through a computationally efficient yet approximate niche-preservation operator which computes the *crowding distance* for every last level member as the summation of objective-wise normalized distance between two

neighboring solutions. Thereafter, the solutions having larger crowding distance values are chosen. Here, we replace the crowding distance operator with the following approaches (subsections IV-A to IV-E).

#### A. Classification of Population into Non-dominated Levels

The above procedure of identifying non-dominated fronts using the usual domination principle [17] is also used in NSGA-III. All population members from non-dominated front level 1 to level  $l$  are first included in  $S_t$ . If  $|S_t| = N$ , no further operations are needed and the next generation is started with  $P_{t+1} = S_t$ . For  $|S_t| > N$ , members from one to  $(l-1)$  fronts are already selected, that is,  $P_{t+1} = \cup_{i=1}^{l-1} F_i$ , and the remaining ( $K = N - |P_{t+1}|$ ) population members are chosen from the last front  $F_l$ . We describe the remaining selection process in the following subsections.

#### B. Determination of Reference Points on a Hyper-Plane

As indicated before, NSGA-III uses a predefined set of reference points to ensure diversity in obtained solutions. The chosen reference points can either be predefined in a structured manner or supplied preferentially by the user. We shall present results of both methods in the results section later. In the absence of any preference information, any predefined structured placement of reference points can be adopted, but in this paper we use Das and Dennis's [48] systematic approach<sup>1</sup> that places points on a *normalized hyper-plane* – a  $(M-1)$ -dimensional unit simplex – which is equally inclined to all objective axes and has an intercept of one on each axis. If  $p$  divisions are considered along each objective, the total number of reference points ( $H$ ) in an  $M$ -objective problem is given by:

$$H = \binom{M+p-1}{p}. \quad (3)$$

For example, in a three-objective problem ( $M = 3$ ), the reference points are created on a triangle with apex at  $(1, 0, 0)$ ,  $(0, 1, 0)$  and  $(0, 0, 1)$ . If four divisions ( $p = 4$ ) are chosen for each objective axis,  $H = \binom{3+4-1}{4} = 15$  reference points will be created. For clarity, these reference points are shown in Figure 1. In the proposed NSGA-III, in addition to emphasizing non-dominated solutions, we also emphasize population members which are in some sense *associated* with each of these reference points. Since the above-created reference points are widely distributed on the entire normalized hyper-plane, the obtained solutions are also likely to be widely distributed on or near the Pareto-optimal front. In the case of a user-supplied set of preferred reference points, ideally the user can mark  $H$  points on the normalized hyper-plane or indicate any  $H$ ,  $M$ -dimensional vectors for the purpose. The proposed algorithm is likely to find near Pareto-optimal solutions corresponding to the supplied reference points, thereby allowing this method to be used more from the point of view of a combined application of decision-making and many-objective optimization. The procedure is presented in Algorithm 1.

<sup>1</sup>Any other structured distribution with or without a biasing on some part of the Pareto-optimal front can be used as well.

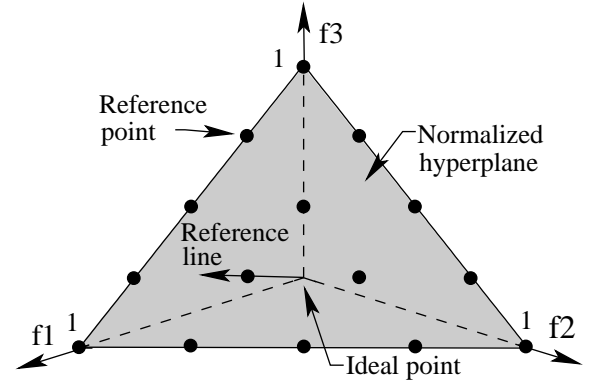


Fig. 1. 15 reference points are shown on a normalized reference plane for a three-objective problem with  $p = 4$ .

---

#### Algorithm 1 Generation $t$ of NSGA-III procedure

---

**Input:**  $H$  structured reference points  $Z^s$  or supplied aspiration points  $Z^a$ , parent population  $P_t$

**Output:**  $P_{t+1}$

- 1:  $S_t = \emptyset$ ,  $i = 1$
  - 2:  $Q_t = \text{Recombination+Mutation}(P_t)$
  - 3:  $R_t = P_t \cup Q_t$
  - 4:  $(F_1, F_2, \dots) = \text{Non-dominated-sort}(R_t)$
  - 5: **repeat**
  - 6:    $S_t = S_t \cup F_i$  and  $i = i + 1$
  - 7: **until**  $|S_t| \geq N$
  - 8: Last front to be included:  $F_l = F_i$
  - 9: **if**  $|S_t| = N$  **then**
  - 10:    $P_{t+1} = S_t$ , **break**
  - 11: **else**
  - 12:    $P_{t+1} = \cup_{j=1}^{l-1} F_j$
  - 13:   Points to be chosen from  $F_l$ :  $K = N - |P_{t+1}|$
  - 14:   Normalize objectives and create reference set  $Z^r$ :  
        $\text{Normalize}(f^n, S_t, Z^r, Z^s, Z^a)$
  - 15:   Associate each member  $s$  of  $S_t$  with a reference point:  
        $[\pi(s), d(s)] = \text{Associate}(S_t, Z^r)$    %  $\pi(s)$ : closest reference point,  $d$ : distance between  $s$  and  $\pi(s)$
  - 16:   Compute niche count of reference point  $j \in Z^r$ :  $\rho_j = \sum_{s \in S_t/F_l} ((\pi(s) = j) ? 1 : 0)$
  - 17:   Choose  $K$  members one at a time from  $F_l$  to construct  $P_{t+1}$ :  
        $\text{Niching}(K, \rho_j, \pi, d, Z^r, F_l, P_{t+1})$
  - 18: **end if**
- 

#### C. Adaptive Normalization of Population Members

First, the ideal point of the population  $S_t$  is determined by identifying the minimum value ( $z_i^{\min}$ ), for each objective function  $i = 1, 2, \dots, M$  in  $\cup_{\tau=0}^t S_\tau$  and by constructing the ideal point  $\bar{z} = (z_1^{\min}, z_2^{\min}, \dots, z_M^{\min})$ . Each objective value of  $S_t$  is then translated by subtracting objective  $f_i$  by  $z_i^{\min}$ , so that the ideal point of translated  $S_t$  becomes a zero vector. We denote this *translated* objective as  $f'_i(\mathbf{x}) = f_i(\mathbf{x}) - z_i^{\min}$ . Thereafter, the extreme point in each objective axis is identified by finding the solution ( $\mathbf{x} \in S_t$ ) that makes the following achievement scalarizing function minimum with weight vector

$\mathbf{w}$  being the axis direction:

$$ASF(\mathbf{x}, \mathbf{w}) = \max_{i=1}^M f'_i(\mathbf{x})/w_i, \quad \text{for } \mathbf{x} \in S_t. \quad (4)$$

For  $w_i = 0$ , we replace it with a small number  $10^{-6}$ . For  $i$ -th translated objective direction  $f'_i$ , this will result in an extreme objective vector,  $\mathbf{z}^{i,\max}$ . These  $M$  extreme vectors are then used to constitute a  $M$ -dimensional linear hyper-plane. The intercept  $a_i$  of the  $i$ -th objective axis and the linear hyper-plane can then be computed (see Figure 2) and the objective functions can be normalized as follows:

$$f_i^n(\mathbf{x}) = \frac{f'_i(\mathbf{x})}{a_i - z_i^{\min}} = \frac{f_i(\mathbf{x}) - z_i^{\min}}{a_i - z_i^{\min}}, \quad \text{for } i = 1, 2, \dots, M. \quad (5)$$

Note that the intercept on each normalized objective axis is now at  $f_i^n = 1$  and a hyper-plane constructed with these intercept points will make  $\sum_{i=1}^M f_i^n = 1$ .

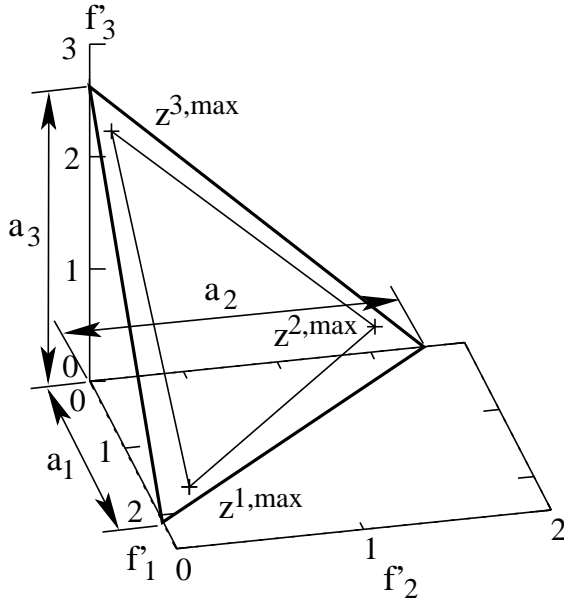


Fig. 2. Procedure of computing intercepts and then forming the hyper-plane from extreme points are shown for a three-objective problem.

In the case of structured reference points ( $H$  of them), the original reference points calculated using Das and Dennis's [48] approach already lie on this normalized hyper-plane. In the case of preferred reference points by the user, the reference points are simply mapped on to the above-constructed normalized hyper-plane using equation 5. Since the normalization procedure and the creation of the hyper-plane is done at each generation using extreme points ever found from the start of the simulation, the proposed NSGA-III procedure adaptively maintains a diversity in the space spanned by the members of  $S_t$  at every generation. This enables NSGA-III to solve problems having a Pareto-optimal front whose objective values may be differently scaled. The procedure is also described in Algorithm 2.

---

#### Algorithm 2 Normalize ( $\mathbf{f}^n, S_t, Z^r, Z^s/Z^a$ ) procedure

---

**Input:**  $S_t, Z^s$  (structured points) or  $Z^a$  (supplied points)

**Output:**  $\mathbf{f}^n, Z^r$  (reference points on normalized hyper-plane)

---

```

1: for  $j = 1$  to  $M$  do
2:   Compute ideal point:  $z_j^{\min} = \min_{\mathbf{s} \in S_t} f_j(\mathbf{s})$ 
3:   Translate objectives:  $f'_j(\mathbf{s}) = f_j(\mathbf{s}) - z_j^{\min} \quad \forall \mathbf{s} \in S_t$ 
4:   Compute extreme points:  $\mathbf{z}^{j,\max} = \mathbf{s} : \arg\min_{\mathbf{s} \in S_t} ASF(\mathbf{s}, \mathbf{w}^j)$ , where  $\mathbf{w}^j = (\epsilon, \dots, \epsilon)^T$ ,  $\epsilon = 10^{-6}$ , and  $w_j^j = 1$ 
5: end for
6: Compute intercepts  $a_j$  for  $j = 1, \dots, M$ 
7: Normalize objectives ( $\mathbf{f}^n$ ) using Equation 5
8: if  $Z^a$  is given then
9:   Map each (aspiration) point on normalized hyper-plane using Equation 5 and save the points in the set  $Z^r$ 
10: else
11:    $Z^r = Z^s$ 
12: end if
```

---

#### D. Association Operation

After normalizing each objective adaptively based on the extent of members of  $S_t$  in the objective space, next we need to *associate* each population member with a reference point. For this purpose, we define a *reference line* corresponding to each reference point on the hyper-plane by joining the reference point with the origin. Then, we calculate the perpendicular distance of each population member of  $S_t$  from each of the reference lines. The reference point whose reference line is closest to a population member in the normalized objective space is considered associated with the population member. This is illustrated in Figure 3. The procedure is presented in

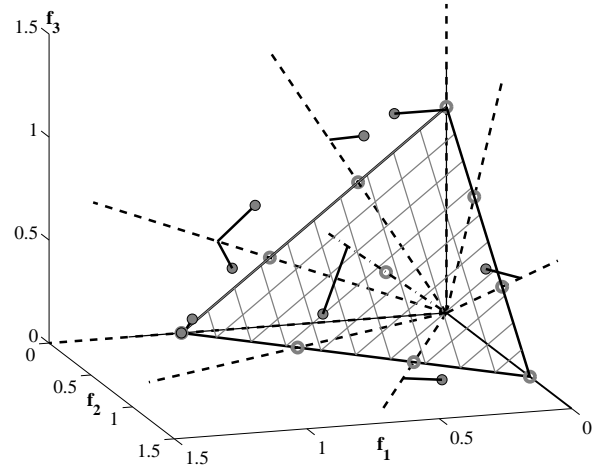


Fig. 3. Association of population members with reference points is illustrated.

#### Algorithm 3.

#### E. Niche-Preservation Operation

It is worth noting that a reference point may have one or more population members associated with it or need not have any population member associated with it. We count the

**Algorithm 3** Associate ( $S_t, Z^r$ ) procedure

---

**Input:**  $Z^r, S_t$   
**Output:**  $\pi(s \in S_t), d(s \in S_t)$

- 1: **for** each reference point  $z \in Z^r$  **do**
- 2:   Compute reference line  $w = z$
- 3: **end for**
- 4: **for** each  $s \in S_t$  **do**
- 5:   **for** each  $w \in Z^r$  **do**
- 6:     Compute  $d^\perp(s, w) = s - w^T s / \|w\|$
- 7:   **end for**
- 8:   Assign  $\pi(s) = w : \operatorname{argmin}_{w \in Z^r} d^\perp(s, w)$
- 9:   Assign  $d(s) = d^\perp(s, \pi(s))$
- 10: **end for**

---

number of population members from  $P_{t+1} = S_t / F_l$  that are associated with each reference point. Let us denote this niche count as  $\rho_j$  for the  $j$ -th reference point. We now devise a new niche-preserving operation as follows. First, we identify the reference point set  $J_{\min} = \{j : \operatorname{argmin}_j \rho_j\}$  having minimum  $\rho_j$ . In case of multiple such reference points, one ( $\bar{j} \in J_{\min}$ ) is chosen at random.

If  $\rho_{\bar{j}} = 0$  (meaning that there is no associated  $P_{t+1}$  member to the reference point  $\bar{j}$ ), there can be two scenarios with  $\bar{j}$  in set  $F_l$ . First, there exists one or more members in front  $F_l$  that are already associated with the reference point  $\bar{j}$ . In this case, the one having the shortest perpendicular distance from the reference line is added to  $P_{t+1}$ . The count  $\rho_{\bar{j}}$  is then incremented by one. Second, the front  $F_l$  does not have any member associated with the reference point  $\bar{j}$ . In this case, the reference point is excluded from further consideration for the current generation.

In the event of  $\rho_{\bar{j}} \geq 1$  (meaning that already one member associated with the reference point exists in  $S_t / F_l$ ), a randomly<sup>2</sup> chosen member, if exists, from front  $F_l$  that is associated with the reference point  $\bar{j}$  is added to  $P_{t+1}$ . The count  $\rho_{\bar{j}}$  is then incremented by one. After niche counts are updated, the procedure is repeated for a total of  $K$  times to fill all vacant population slots of  $P_{t+1}$ . The procedure is presented in Algorithm 4.

### F. Genetic Operations to Create Offspring Population

After  $P_{t+1}$  is formed, it is then used to create a new offspring population  $Q_{t+1}$  by applying usual genetic operators. In NSGA-III, we have already performed a careful elitist selection of solutions and attempted to maintain diversity among solutions by emphasizing solutions closest to the reference line of each reference point. Also, as we shall describe in Section V, for a computationally fast procedure, we have set  $N$  almost equal to  $H$ , thereby giving equal importance to each population member. For all these reasons, we do not employ any explicit selection operation with NSGA-III. The population  $Q_{t+1}$  is constructed by applying the usual crossover and mutation operators by randomly picking parents from  $P_{t+1}$ . However, to create offspring solutions closer to

<sup>2</sup>The point closest to the reference point or using any other diversity preserving criterion can also be used.

**Algorithm 4** Niching ( $K, \rho_j, \pi, d, Z^r, F_l, P_{t+1}$ ) procedure

---

**Input:**  $K, \rho_j, \pi(s \in S_t), d(s \in S_t), Z^r, F_l$   
**Output:**  $P_{t+1}$

- 1:  $k = 1$
- 2: **while**  $k \leq K$  **do**
- 3:    $J_{\min} = \{j : \operatorname{argmin}_{j \in Z^r} \rho_j\}$
- 4:    $\bar{j} = \operatorname{random}(J_{\min})$
- 5:    $I_{\bar{j}} = \{s : \pi(s) = \bar{j}, s \in F_l\}$
- 6:   **if**  $I_{\bar{j}} \neq \emptyset$  **then**
- 7:     **if**  $\rho_{\bar{j}} = 0$  **then**
- 8:        $P_{t+1} = P_{t+1} \cup \{s : \operatorname{argmin}_{s \in I_{\bar{j}}} d(s)\}$
- 9:     **else**
- 10:        $P_{t+1} = P_{t+1} \cup \operatorname{random}(I_{\bar{j}})$
- 11:     **end if**
- 12:      $\rho_{\bar{j}} = \rho_{\bar{j}} + 1, F_l = F_l \setminus s$
- 13:      $k = k + 1$
- 14:   **else**
- 15:      $Z^r = Z^r \setminus \{\bar{j}\}$
- 16:   **end if**
- 17: **end while**

---

parent solutions (to take care of third difficulty mentioned in Section II-A), we suggest using a relatively larger value of distribution index in the SBX operator.

### G. Computational Complexity of One Generation of NSGA-III

The non-dominated sorting (line 4 in Algorithm 1) of a population of size  $2N$  having  $M$ -dimensional objective vectors require  $O(N \log^{M-2} N)$  computations [49]. Identification of ideal point in line 2 of Algorithm 2 requires a total of  $O(MN)$  computations. Translation of objectives (line 3) requires  $O(MN)$  computations. However, identification of extreme points (line 4) require  $O(M^2 N)$  computations. Determination of intercepts (line 6) requires one matrix inversion of size  $M \times M$ , requiring  $O(M^3)$  operations. Thereafter, normalization of a maximum of  $2N$  population members (line 7) require  $O(N)$  computations. Line 8 of Algorithm 2 requires  $O(MH)$  computations. All operations in Algorithm 3 in associating a maximum of  $2N$  population members to  $H$  reference points would require  $O(MNH)$  computations. Thereafter, in the niching procedure in Algorithm 4, line 3 will require  $O(H)$  comparisons. Assuming that  $L = |F_l|$ , line 5 requires  $O(L)$  checks. Line 8 in the worst case requires  $O(L)$  computations. Other operations have smaller complexity. However, the above computations in the niching algorithm need to be performed a maximum of  $L$  times, thereby requiring larger of  $O(L^2)$  or  $O(LH)$  computations. In the worst case scenario ( $S_t = F_l$ , that is, the first non-dominated front exceeds the population size),  $L \leq 2N$ . In all our simulations, we have used  $N \approx H$  and  $N > M$ . Considering all the above considerations and computations, the overall worst-case complexity of one generation of NSGA-III is  $O(N^2 \log^{M-2} N)$  or  $O(N^2 M)$ , whichever is larger.



### H. Parameter-Less Property of NSGA-III

Like in NSGA-II, NSGA-III algorithm does not require to set any new parameter other than the usual genetic parameters such as the population size, termination parameter, crossover and mutation probabilities and their associated parameters. The number of reference points  $H$  is not an algorithmic parameter, as this is entirely at the disposal of the user. The population size  $N$  is dependent on  $H$ , as  $N \approx H$ . The location of the reference points is similarly dependent on the preference information that the user is interested to achieve in the obtained solutions.

We shall now present simulation results of NSGA-III for various many-objective scenarios and compare its performance with MOEA/D and classical methods.

## V. RESULTS

In this section, we provide the simulation results of NSGA-III on three to 15-objective optimization problems. Since our method has a framework similar to MOEA/D in that both types of algorithms require a set of user-supplied reference points or weight vectors, we compare our proposed method with different versions of MOEA/D (codes from MOEA/D website [50] are used). The original MOEA/D study proposed two procedures (MOEA/D-PBI and MOEA/D-TCH), but did not solve four or more objective problems. Here, along with our algorithm, we investigate the performance of these MOEA/D algorithms on three to 15-objective problems.

As a performance metric, we have chosen the inverse generational distance (IGD) metric [51], [47] which as a single metric which can provide a combined information about the convergence and diversity of the obtained solutions. Since reference points or reference directions are supplied in NSGA-III and MOEA/D algorithms, respectively, and since in this section we show the working of these methods on test problems for which the exact Pareto-optimal surface is known, we can exactly locate the targeted Pareto-optimal points in the normalized objective space. We compute these targeted points and call them a set  $\mathbf{Z}$ . For any algorithm, we obtain the final non-dominated points in the objective space and call them the set  $\mathbf{A}$ . Now, we compute the IGD metric as the average Euclidean distance of points in set  $\mathbf{Z}$  with their nearest members of all points in set  $\mathbf{A}$ :

$$\text{IGD}(\mathbf{A}, \mathbf{Z}) = \frac{1}{|\mathbf{Z}|} \sum_{i=1}^{|\mathbf{Z}|} \min_{j=1}^{|\mathbf{A}|} d(\mathbf{z}_i, \mathbf{a}_j), \quad (6)$$

where  $d(\mathbf{z}_i, \mathbf{a}_j) = \|\mathbf{z}_i - \mathbf{a}_j\|_2$ . It is needless to write that a set with a smaller IGD value is better. If no solution associated with a reference point is found, the IGD metric value for the set will be large. For each case, 20 different runs from different initial populations are performed and best, median and worst IGD performance values are reported. For all algorithms, the population members from the final generation are presented and used for computing the performance measure.

Table I shows the number of chosen reference points ( $H$ ) for different sizes of a problem. The population size  $N$  for NSGA-

III is set as the smallest multiple of four<sup>3</sup> higher than  $H$ . For MOEA/D procedures, we use a population size,  $N' = H$ , as suggested by their developers. For three-objective problems,

TABLE I  
NUMBER OF REFERENCE POINTS/DIRECTIONS AND CORRESPONDING POPULATION SIZES USED IN NSGA-III AND MOEA/D ALGORITHMS.

| No. of objectives ( $M$ ) | Ref. pts./ Ref. dim. ( $H$ ) | NSGA-III popsize ( $N$ ) | MOEA/D popsize ( $N'$ ) |
|---------------------------|------------------------------|--------------------------|-------------------------|
| 3                         | 91                           | 92                       | 91                      |
| 5                         | 210                          | 212                      | 210                     |
| 8                         | 156                          | 156                      | 156                     |
| 10                        | 275                          | 276                      | 275                     |
| 15                        | 135                          | 136                      | 135                     |

we have used  $p = 12$  in order to obtain  $H = \binom{3-1+12}{12}$  or 91 reference points (refer to equation 3). For five-objective problems, we have used  $p = 6$ , so that  $H = 210$  reference points are obtained. Note that as long as  $p \geq M$  is not chosen, no intermediate point will be created by Das and Dennis's systematic approach. For eight-objective problems, even if we use  $p = 8$  (to have at least one intermediate reference point), it requires 5,040 reference points. To avoid such a situation, we use two layers of reference points with small values of  $p$ . On the boundary layer, we use  $p = 3$ , so that 120 points are created. On the inside layer, we use  $p = 2$ , so that 36 points are created. All  $H = 156$  points are then used as reference points for eight-objective problems. We illustrate this scenario in Figure 4 for a three-objective problem using  $p = 2$  for boundary layer and  $p = 1$  for the inside layer. For 10-objective

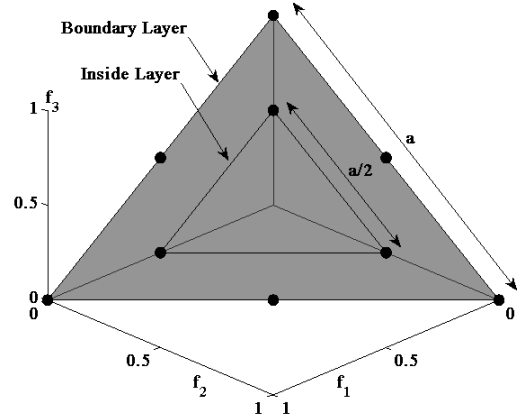


Fig. 4. The concept for two-layered reference points (with six points on the boundary layer ( $p = 2$ ) and three points on the inside layer ( $p = 1$ )) is shown for a three-objective problem, but are implemented for eight or more objectives in the simulations here.

problems as well, we use  $p = 3$  and  $p = 2$  for boundary and inside layers, respectively, thereby requiring a total of  $H = 220 + 55$  or 275 reference points. Similarly, for 15-objective problems, we use  $p = 2$  and  $p = 1$  for boundary and inside layers, respectively, thereby requiring  $H = 120 + 15$  or 135 reference points.

<sup>3</sup>Since no tournament selection is used in NSGA-III, a factor of two would be adequate as well, but as we re-introduce tournament selection in the constrained NSGA-III in the sequel paper [52], we keep population size as a multiple of four here as well to have a unified algorithm.



Table II presents other NSGA-III and MOEA/D parameters used in this study. MOEA/D requires additional parameters

TABLE II  
PARAMETER VALUES USED IN NSGA-III AND TWO VERSIONS OF  
MOEA/D.  $n$  IS THE NUMBER OF VARIABLES.

| Parameters                           | NSGA-III | MOEA/D |
|--------------------------------------|----------|--------|
| SBX probability [28], $p_c$          | 1        | 1      |
| Polynomial mutation prob. [3], $p_m$ | $1/n$    | $1/n$  |
| $\eta_c$ [28]                        | 30       | 20     |
| $\eta_m$ [28]                        | 20       | 20     |

which we have set according to the suggestions given by their developers. The neighborhood size  $T$  is set as 20 for both approaches and additionally the penalty parameter  $\theta$  for the MOEA/D-PBI approach is set as 5.

#### A. Normalized Test Problems

To start with, we use three to 15-objective DTLZ1, DTLZ2, DTLZ3 and DTLZ4 problems [53]. The number of variables are  $(M + k - 1)$ , where  $M$  is number of objectives and  $k = 5$  for DTLZ1, while  $k = 10$  for DTLZ2, DTLZ3 and DTLZ4. In all these problems, the corresponding Pareto-optimal fronts lie in  $f_i \in [0, 0.5]$  for DTLZ1 problem, or  $[0, 1]$  for other DTLZ problems. Since they have an identical range of values for each objective, we call these problems ‘normalized test problems’ in this study. Table III indicates the maximum number of generations used for each test problem.

Figure 5 shows NSGA-III obtained front for the three-objective DTLZ1 problem. This particular run is associated with the median value of IGD performance metric. All 91 points are well distributed on the entire Pareto-optimal set. Results with the MOEA/D-PBI are shown in Figure 6. It is clear that MOEA/D-PBI is also able to find a good distribution of points similar to that of NSGA-III. However, Figure 7 shows that MOEA/D-TCH is unable to find a uniform distribution of points. Such a distribution was also reported in the original MOEA/D study [10].

Table III shows that for the DTLZ1 problem NSGA-III performs slightly better in terms of the IGD metric, followed by MOEA/D-PBI. For the five-objective DTLZ1 problem, MOEA/D performs better than NSGA-III, but in 8, 10, and 15-objective problems, NSGA-III performs better. MOEA/D-TCH consistently does not perform well in all higher dimensions of the problem. This observation is similar to that concluded in the original MOEA/D study [10] based on two and three-objective problems.

For DTLZ2 problems, the performance of MOEA/D-PBI is consistently better than NSGA-III, however NSGA-III performs better than MOEA/D-TCH approach. Figures 8, 9 and 10 show the distribution of obtained points for NSGA-III, MOEA/D-PBI, and MOEA/D-TCH algorithms on three-objective DTLZ2 problem, respectively. Figures show that the performances of NSGA-III and MOEA/D-PBI are comparable to each other, whereas the performance of MOEA/D-TCH is poor.

Figure 11 shows the variation of the IGD metric value with function evaluations for NSGA-III and MOEA/D-PBI approaches for the eight-objective DTLZ2 problem. Average

IGD metric value for 20 runs are plotted. It is clear that both approaches are able to reduce the IGD value with elapse of function evaluations.

Similar observation is made for the DTLZ3 problem. This problem introduces a number of local Pareto-optimal fronts that provide a stiff challenge for algorithms to come close to the global Pareto-optimal front. While the performance of NSGA-III and MOEA/D-PBI are similar, with a slight edge for MOEA/D-PBI, the performance of MOEA/D-TCH is poor. However, in some runs, MOEA/D-PBI is unable to get close to the Pareto-optimal front, as evident from a large value of IGD metric value. Figure 12 shows variation of the average IGD metric of 20 runs with function evaluations for NSGA-III and MOEA/D-PBI approaches. NSGA-III manages to find a better IGD value than MOEA/D-PBI approach after about 80,000 function evaluations.

Problem DTLZ4 has a biased density of points away from  $f_M = 0$ , however the Pareto-optimal front is identical to that in DTLZ2. The difference in the performances between NSGA-III and MOEA/D-PBI is clear from this problem. Both MOEA/D algorithms are not able to find an adequate distribution of points, whereas NSGA-III algorithm performs as it did in other problems. Figures 13, 14 and 15 show the obtained distribution of points on the three-objective DTLZ4 problem. The algorithms are unable to find near  $f_3 = 0$  Pareto-optimal points, whereas the NSGA-III is able to find a set of well-distributed points on the entire Pareto-optimal front. These plots are made using the median performed run in each case. Table III clearly show that the IGD metric values for NSGA-III algorithm are better than those of MOEA/D algorithms. Figure 16 shows the value path plot of all obtained solutions for the 10-objective DTLZ4 problem by NSGA-III. A spread of solutions over  $f_i \in [0, 1]$  for all 10 objectives and a trade-off among them are clear from the plot. In contrast, Figure 17 shows the value path plot for the same problem obtained using MOEA/D-PBI. The figure clearly shows that MOEA/D-PBI is not able to find a widely distributed set of points for the 10-objective DTLZ4 problem. Points having larger values of objectives  $f_1$  to  $f_7$  are not found by MOEA/D-PBI approach.

The right-most column of Table III presents the performance of the recently proposed MOEA/D-DE approach [46]. The approach uses differential evolution (DE) instead of SBX and polynomial mutation operators. Two additional parameters are introduced in the MOEA/D-DE procedure. First, a maximum bound ( $n_r$ ) on the number of weight vectors that a child can be associated to is introduced and is set as  $n_r = 2$ . Second, for choosing a mating partner of a parent, a probability ( $\delta$ ) is set for choosing a neighboring partner and the probability  $(1 - \delta)$  for choosing any other population member. Authors suggested to use  $\delta = 0.9$ . We use the same values of these parameters in our study. Table III indicates that this version of MOEA/D does not perform well on the normalized DTLZ problems, however it performs better than MOEA/D-PBI on DTLZ4 problem. Due to its poor performance in general in these problems, we do not apply it any further.

In the above runs with MOEA/D, we have used SBX recombination parameter index  $\eta_c = 20$  (as indicated in

TABLE III  
BEST, MEDIAN AND WORST IGD VALUES OBTAINED FOR NSGA-III AND TWO VERSIONS OF MOEA/D ON  $M$ -OBJECTIVE DTLZ1, DTLZ2, DTLZ3  
AND DTLZ4 PROBLEMS. BEST PERFORMANCE IS SHOWN IN BOLD.

| Problem | $M$ | MaxGen | NSGA-III                                 | MOEA/D-PBI                               | MOEA/D-TCH             | MOEA/D-DE                                |
|---------|-----|--------|--|--|------------------------|--|
| DTLZ1   | 3   | 400    | $4.880 \times 10^{-4}$                   | <b><math>4.095 \times 10^{-4}</math></b> | $3.296 \times 10^{-2}$ | $5.470 \times 10^{-3}$                   |
|         |     |        | <b><math>1.308 \times 10^{-3}</math></b> | $1.495 \times 10^{-3}$                   | $3.321 \times 10^{-2}$ | $1.778 \times 10^{-2}$                   |
|         |     |        | $4.880 \times 10^{-3}$                   | <b><math>4.743 \times 10^{-3}</math></b> | $3.359 \times 10^{-2}$ | $3.394 \times 10^{-1}$                   |
|         | 5   | 600    | $5.116 \times 10^{-4}$                   | <b><math>3.179 \times 10^{-4}</math></b> | $1.124 \times 10^{-1}$ | $2.149 \times 10^{-2}$                   |
|         |     |        | $9.799 \times 10^{-4}$                   | <b><math>6.372 \times 10^{-4}</math></b> | $1.129 \times 10^{-1}$ | $2.489 \times 10^{-2}$                   |
|         |     |        | $1.979 \times 10^{-3}$                   | <b><math>1.635 \times 10^{-3}</math></b> | $1.137 \times 10^{-1}$ | $3.432 \times 10^{-2}$                   |
|         | 8   | 750    | <b><math>2.044 \times 10^{-3}</math></b> | $3.914 \times 10^{-3}$                   | $1.729 \times 10^{-1}$ | $3.849 \times 10^{-2}$                   |
|         |     |        | <b><math>3.979 \times 10^{-3}</math></b> | $6.106 \times 10^{-3}$                   | $1.953 \times 10^{-1}$ | $4.145 \times 10^{-2}$                   |
|         |     |        | $8.721 \times 10^{-3}$                   | <b><math>8.537 \times 10^{-3}</math></b> | $2.094 \times 10^{-1}$ | $4.815 \times 10^{-2}$                   |
|         | 10  | 1000   | <b><math>2.215 \times 10^{-3}</math></b> | $3.872 \times 10^{-3}$                   | $2.072 \times 10^{-1}$ | $4.253 \times 10^{-2}$                   |
|         |     |        | <b><math>3.462 \times 10^{-3}</math></b> | $5.073 \times 10^{-3}$                   | $2.147 \times 10^{-1}$ | $4.648 \times 10^{-2}$                   |
|         |     |        | $6.869 \times 10^{-3}$                   | <b><math>6.130 \times 10^{-3}</math></b> | $2.400 \times 10^{-1}$ | $4.908 \times 10^{-2}$                   |
|         | 15  | 1500   | <b><math>2.649 \times 10^{-3}</math></b> | $1.236 \times 10^{-2}$                   | $3.237 \times 10^{-1}$ | $8.048 \times 10^{-2}$                   |
|         |     |        | <b><math>5.063 \times 10^{-3}</math></b> | $1.431 \times 10^{-2}$                   | $3.438 \times 10^{-1}$ | $8.745 \times 10^{-2}$                   |
|         |     |        | <b><math>1.123 \times 10^{-2}</math></b> | $1.692 \times 10^{-2}$                   | $3.634 \times 10^{-1}$ | $1.008 \times 10^{-1}$                   |
| DTLZ2   | 3   | 250    | $1.262 \times 10^{-3}$                   | <b><math>5.432 \times 10^{-4}</math></b> | $7.499 \times 10^{-2}$ | $3.849 \times 10^{-2}$                   |
|         |     |        | $1.357 \times 10^{-3}$                   | <b><math>6.406 \times 10^{-4}</math></b> | $7.574 \times 10^{-2}$ | $4.562 \times 10^{-2}$                   |
|         |     |        | $2.114 \times 10^{-3}$                   | <b><math>8.006 \times 10^{-4}</math></b> | $7.657 \times 10^{-2}$ | $6.069 \times 10^{-2}$                   |
|         | 5   | 350    | $4.254 \times 10^{-3}$                   | <b><math>1.219 \times 10^{-3}</math></b> | $2.935 \times 10^{-1}$ | $1.595 \times 10^{-1}$                   |
|         |     |        | $4.982 \times 10^{-3}$                   | <b><math>1.437 \times 10^{-3}</math></b> | $2.945 \times 10^{-1}$ | $1.820 \times 10^{-1}$                   |
|         |     |        | $5.862 \times 10^{-3}$                   | <b><math>1.727 \times 10^{-3}</math></b> | $2.953 \times 10^{-1}$ | $1.935 \times 10^{-1}$                   |
|         | 8   | 500    | $1.371 \times 10^{-2}$                   | <b><math>3.097 \times 10^{-3}</math></b> | $5.989 \times 10^{-1}$ | $3.003 \times 10^{-1}$                   |
|         |     |        | $1.571 \times 10^{-2}$                   | <b><math>3.763 \times 10^{-3}</math></b> | $6.301 \times 10^{-1}$ | $3.194 \times 10^{-1}$                   |
|         |     |        | $1.811 \times 10^{-2}$                   | <b><math>5.198 \times 10^{-3}</math></b> | $6.606 \times 10^{-1}$ | $3.481 \times 10^{-1}$                   |
|         | 10  | 750    | $1.350 \times 10^{-2}$                   | <b><math>2.474 \times 10^{-3}</math></b> | $7.002 \times 10^{-1}$ | $2.629 \times 10^{-1}$                   |
|         |     |        | $1.528 \times 10^{-2}$                   | <b><math>2.778 \times 10^{-3}</math></b> | $7.266 \times 10^{-1}$ | $2.873 \times 10^{-1}$                   |
|         |     |        | $1.697 \times 10^{-2}$                   | <b><math>3.235 \times 10^{-3}</math></b> | $7.704 \times 10^{-1}$ | $3.337 \times 10^{-1}$                   |
|         | 15  | 1000   | $1.360 \times 10^{-2}$                   | <b><math>5.254 \times 10^{-3}</math></b> | 1.000                  | $3.131 \times 10^{-1}$                   |
|         |     |        | $1.726 \times 10^{-2}$                   | <b><math>6.005 \times 10^{-3}</math></b> | 1.084                  | $3.770 \times 10^{-1}$                   |
|         |     |        | $2.114 \times 10^{-2}$                   | <b><math>9.409 \times 10^{-3}</math></b> | 1.120                  | $4.908 \times 10^{-1}$                   |
| DTLZ3   | 3   | 1000   | <b><math>9.751 \times 10^{-4}</math></b> | $9.773 \times 10^{-4}$                   | $7.602 \times 10^{-2}$ | $5.610 \times 10^{-2}$                   |
|         |     |        | $4.007 \times 10^{-3}$                   | <b><math>3.426 \times 10^{-3}</math></b> | $7.658 \times 10^{-2}$ | $1.439 \times 10^{-1}$                   |
|         |     |        | <b><math>6.665 \times 10^{-3}</math></b> | $9.113 \times 10^{-3}$                   | $7.764 \times 10^{-2}$ | $8.887 \times 10^1$                      |
|         | 5   | 1000   | $3.086 \times 10^{-3}$                   | <b><math>1.129 \times 10^{-3}</math></b> | $2.938 \times 10^{-1}$ | $1.544 \times 10^{-1}$                   |
|         |     |        | $5.960 \times 10^{-3}$                   | <b><math>2.213 \times 10^{-3}</math></b> | $2.948 \times 10^{-1}$ | $2.115 \times 10^{-1}$                   |
|         |     |        | $1.196 \times 10^{-2}$                   | <b><math>6.147 \times 10^{-3}</math></b> | $2.956 \times 10^{-1}$ | $8.152 \times 10^1$                      |
|         | 8   | 1000   | $1.244 \times 10^{-2}$                   | <b><math>6.459 \times 10^{-3}</math></b> | $6.062 \times 10^{-1}$ | $2.607 \times 10^{-1}$                   |
|         |     |        | $2.375 \times 10^{-2}$                   | <b><math>1.948 \times 10^{-2}</math></b> | $6.399 \times 10^{-1}$ | $3.321 \times 10^{-1}$                   |
|         |     |        | <b><math>9.649 \times 10^{-2}</math></b> | 1.123                                    | $6.808 \times 10^{-1}$ | 3.923                                    |
|         | 10  | 1500   | $8.849 \times 10^{-3}$                   | <b><math>2.791 \times 10^{-3}</math></b> | $7.174 \times 10^{-1}$ | $2.549 \times 10^{-1}$                   |
|         |     |        | $1.188 \times 10^{-2}$                   | <b><math>4.319 \times 10^{-3}</math></b> | $7.398 \times 10^{-1}$ | $2.789 \times 10^{-1}$                   |
|         |     |        | <b><math>2.083 \times 10^{-2}</math></b> | 1.010                                    | $8.047 \times 10^{-1}$ | $2.998 \times 10^{-1}$                   |
|         | 15  | 2000   | $1.401 \times 10^{-2}$                   | <b><math>4.360 \times 10^{-3}</math></b> | 1.029                  | $2.202 \times 10^{-1}$                   |
|         |     |        | $2.145 \times 10^{-2}$                   | <b><math>1.664 \times 10^{-2}</math></b> | 1.073                  | $3.219 \times 10^{-1}$                   |
|         |     |        | <b><math>4.195 \times 10^{-2}</math></b> | 1.260                                    | 1.148                  | $4.681 \times 10^{-1}$                   |
| DTLZ4   | 3   | 600    | <b><math>2.915 \times 10^{-4}</math></b> | $2.929 \times 10^{-1}$                   | $2.168 \times 10^{-1}$ | $3.276 \times 10^{-2}$                   |
|         |     |        | <b><math>5.970 \times 10^{-4}</math></b> | $4.280 \times 10^{-1}$                   | $3.724 \times 10^{-1}$ | $6.049 \times 10^{-2}$                   |
|         |     |        | $4.286 \times 10^{-1}$                   | $5.234 \times 10^{-1}$                   | $4.421 \times 10^{-1}$ | <b><math>3.468 \times 10^{-1}</math></b> |
|         | 5   | 1000   | <b><math>9.849 \times 10^{-4}</math></b> | $1.080 \times 10^{-1}$                   | $3.846 \times 10^{-1}$ | $1.090 \times 10^{-1}$                   |
|         |     |        | <b><math>1.255 \times 10^{-3}</math></b> | $5.787 \times 10^{-1}$                   | $5.527 \times 10^{-1}$ | $1.479 \times 10^{-1}$                   |
|         |     |        | <b><math>1.721 \times 10^{-3}</math></b> | $7.348 \times 10^{-1}$                   | $7.491 \times 10^{-1}$ | $4.116 \times 10^{-1}$                   |
|         | 8   | 1250   | <b><math>5.079 \times 10^{-3}</math></b> | $5.298 \times 10^{-1}$                   | $6.676 \times 10^{-1}$ | $2.333 \times 10^{-1}$                   |
|         |     |        | <b><math>7.054 \times 10^{-3}</math></b> | $8.816 \times 10^{-1}$                   | $9.036 \times 10^{-1}$ | $3.333 \times 10^{-1}$                   |
|         |     |        | <b><math>6.051 \times 10^{-1}</math></b> | $9.723 \times 10^{-1}$                   | 1.035                  | $7.443 \times 10^{-1}$                   |
|         | 10  | 2000   | <b><math>5.694 \times 10^{-3}</math></b> | $3.966 \times 10^{-1}$                   | $7.734 \times 10^{-1}$ | $2.102 \times 10^{-1}$                   |
|         |     |        | <b><math>6.337 \times 10^{-3}</math></b> | $9.203 \times 10^{-1}$                   | $9.310 \times 10^{-1}$ | $2.885 \times 10^{-1}$                   |
|         |     |        | <b><math>1.076 \times 10^{-1}</math></b> | 1.077                                    | 1.039                  | $6.422 \times 10^{-1}$                   |
|         | 15  | 3000   | <b><math>7.110 \times 10^{-3}</math></b> | $5.890 \times 10^{-1}$                   | 1.056                  | $4.500 \times 10^{-1}$                   |
|         |     |        | <b><math>3.431 \times 10^{-1}</math></b> | 1.133                                    | 1.162                  | $6.282 \times 10^{-1}$                   |
|         |     |        | 1.073                                    | 1.249                                    | 1.220                  | <b><math>8.477 \times 10^{-1}</math></b> |

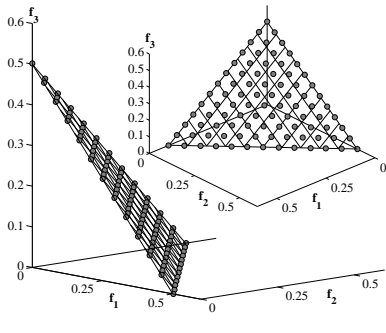


Fig. 5. Obtained solutions by NSGA-III for DTLZ1.

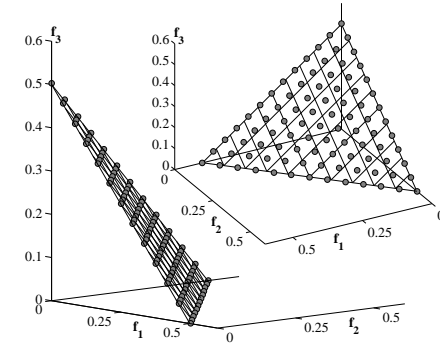


Fig. 6. Obtained solutions by MOEA/D-PBI for DTLZ1.

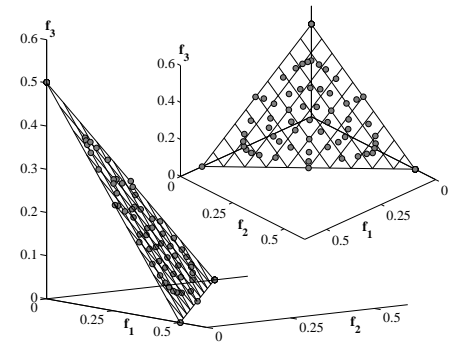


Fig. 7. Obtained solutions by MOEA/D-TCH for DTLZ1.

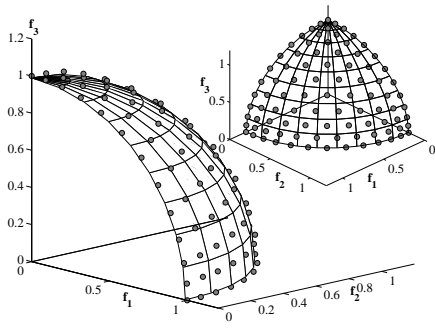


Fig. 8. Obtained solutions by NSGA-III for DTLZ2.

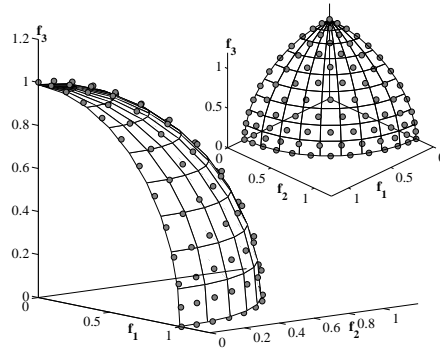


Fig. 9. Obtained solutions by MOEA/D-PBI for DTLZ2.

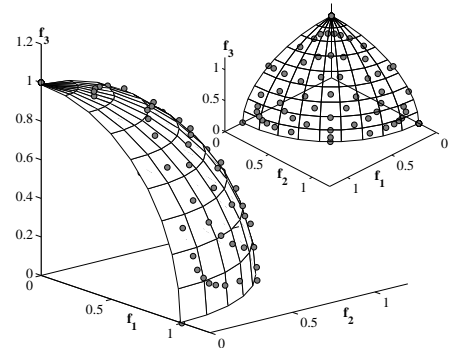


Fig. 10. Obtained solutions by MOEA/D-TCH for DTLZ2.

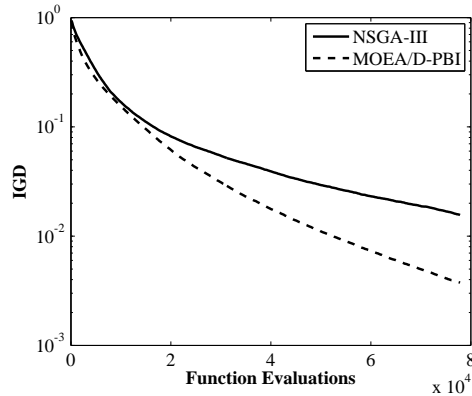


Fig. 11. Variation of IGD metric value with NSGA-III and MOEA/D-PBI for DTLZ2.

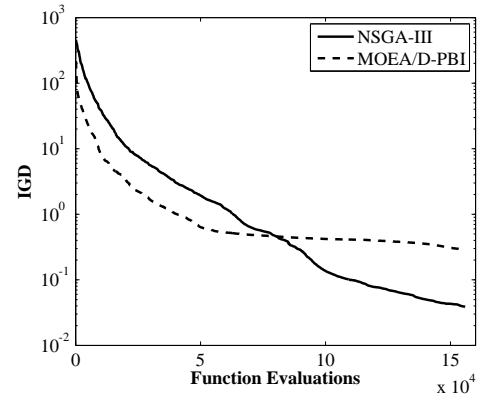


Fig. 12. Variation of IGD metric value with NSGA-III and MOEA/D-PBI for DTLZ3.

Table II) mainly because this value was chosen in the original MOEA/D study [10]. Next, we investigate MOEA/D-PBI's performance with  $\eta_c = 30$  (which was used with NSGA-III). Table IV shows that the performance of MOEA/D does not change much with the above change in  $\eta_c$  value.

Next, we apply NSGA-III and MOEA/D-PBI approaches to two WFG test problems. The parameter settings are the same as before. Figures 18 and 19 show the obtained points on WFG6 problem. The convergence is slightly better with NSGA-III approach. Similarly, Figures 20 and 21 show a similar performance comparison of the above two approaches for WFG7 problem. Table V shows the performances of

the two approaches up to 15-objective WFG6 and WFG7 problems.

Based on the results on three to 15-objective normalized DTLZ and WFG test problems, it can be concluded that (i) MOEA/D-PBI performs consistently better than MOEA/D-TCH approach, (ii) MOEA/D-PBI performs best in some problems, whereas the proposed NSGA-III approach performs best in some other problems, and (iii) in a non-uniformly distributed Pareto-optimal front (like in the DTLZ4 problem), both MOEA/D approaches fail to maintain a good distribution of points, while NSGA-III performs well.

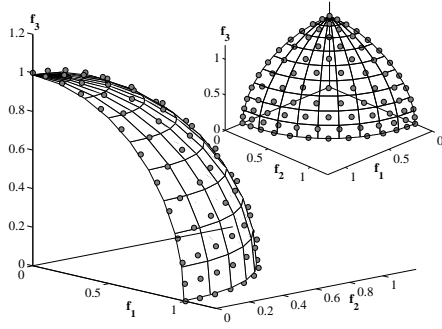


Fig. 13. Obtained solutions by NSGA-III for DTLZ4.

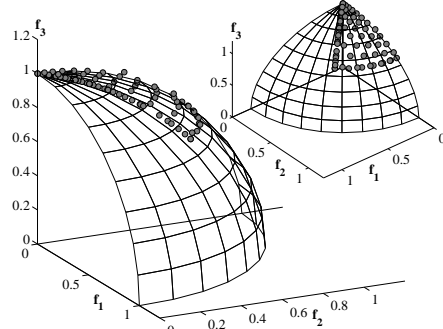


Fig. 14. Obtained solutions by MOEA/D-PBI for DTLZ4.

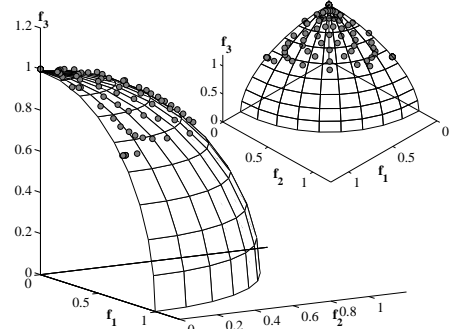


Fig. 15. Obtained solutions by MOEA/D-TCH for DTLZ4.

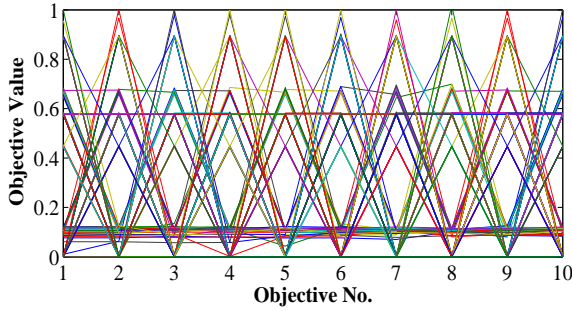


Fig. 16. NSGA-III Solutions are shown using 10-objective value path format for DTLZ4.

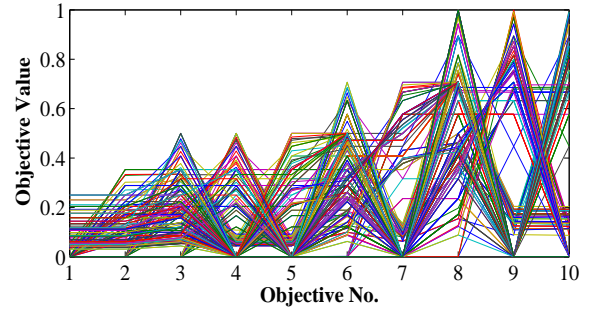


Fig. 17. MOEA/D-PBI Solutions are shown using 10-objective value path format for DTLZ4.

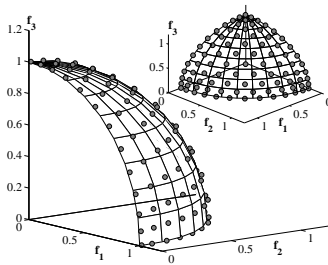


Fig. 18. NSGA-III solutions are shown on three-objective WFG6 problem.

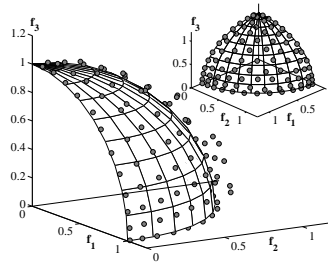


Fig. 19. MOEA/D-PBI solutions are shown on three-objective WFG6 problem.

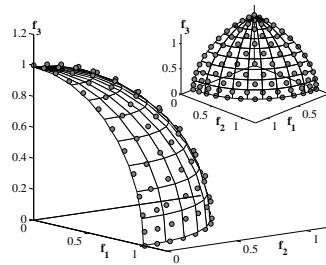


Fig. 20. NSGA-III solutions are shown on three-objective WFG7 problem.

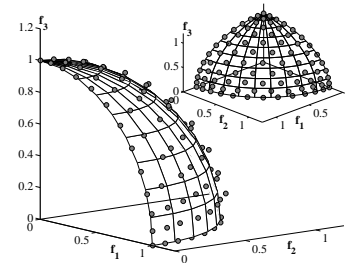


Fig. 21. MOEA/D-PBI solutions are shown on three-objective WFG7 problem.



TABLE IV  
IGD VALUES OF MOEA/D-PBI APPROACH WITH  $\eta_c = 30$ .

| Function | $M$ | MaxGen | NSGA-III               | MOEA/D-PBI             |
|----------|-----|--------|------------------------|------------------------|
| DTLZ1    | 3   | 400    | $4.880 \times 10^{-4}$ | $4.047 \times 10^{-4}$ |
|          |     |        | $1.308 \times 10^{-3}$ | $1.797 \times 10^{-3}$ |
|          |     |        | $4.880 \times 10^{-3}$ | $5.699 \times 10^{-3}$ |
|          | 5   | 600    | $5.116 \times 10^{-4}$ | $2.849 \times 10^{-4}$ |
|          |     |        | $9.799 \times 10^{-4}$ | $5.717 \times 10^{-4}$ |
|          |     |        | $1.979 \times 10^{-3}$ | $1.647 \times 10^{-3}$ |
|          | 8   | 750    | $2.044 \times 10^{-3}$ | $4.053 \times 10^{-3}$ |
|          |     |        | $3.979 \times 10^{-3}$ | $6.839 \times 10^{-3}$ |
|          |     |        | $8.721 \times 10^{-3}$ | $1.127 \times 10^{-2}$ |
|          | 10  | 1000   | $2.215 \times 10^{-3}$ | $4.057 \times 10^{-3}$ |
|          |     |        | $3.462 \times 10^{-3}$ | $5.251 \times 10^{-3}$ |
|          |     |        | $6.869 \times 10^{-3}$ | $6.300 \times 10^{-3}$ |
|          | 15  | 1500   | $2.649 \times 10^{-3}$ | $1.301 \times 10^{-2}$ |
|          |     |        | $5.063 \times 10^{-3}$ | $1.653 \times 10^{-2}$ |
|          |     |        | $1.123 \times 10^{-2}$ | $2.695 \times 10^{-2}$ |
| DTLZ2    | 3   | 250    | $1.262 \times 10^{-3}$ | $4.535 \times 10^{-4}$ |
|          |     |        | $1.357 \times 10^{-3}$ | $5.778 \times 10^{-4}$ |
|          |     |        | $2.114 \times 10^{-3}$ | $8.049 \times 10^{-4}$ |
|          | 5   | 350    | $4.254 \times 10^{-3}$ | $1.096 \times 10^{-3}$ |
|          |     |        | $4.982 \times 10^{-3}$ | $1.208 \times 10^{-3}$ |
|          |     |        | $5.862 \times 10^{-3}$ | $1.370 \times 10^{-3}$ |
|          | 8   | 500    | $1.371 \times 10^{-2}$ | $2.793 \times 10^{-3}$ |
|          |     |        | $1.571 \times 10^{-2}$ | $3.615 \times 10^{-3}$ |
|          |     |        | $1.811 \times 10^{-2}$ | $5.231 \times 10^{-3}$ |
|          | 10  | 750    | $1.350 \times 10^{-2}$ | $2.286 \times 10^{-3}$ |
|          |     |        | $1.528 \times 10^{-2}$ | $2.433 \times 10^{-3}$ |
|          |     |        | $1.697 \times 10^{-2}$ | $2.910 \times 10^{-3}$ |
|          | 15  | 1000   | $1.360 \times 10^{-2}$ | $5.292 \times 10^{-3}$ |
|          |     |        | $1.726 \times 10^{-2}$ | $5.952 \times 10^{-3}$ |
|          |     |        | $2.114 \times 10^{-2}$ | $8.413 \times 10^{-3}$ |
| DTLZ3    | 3   | 1000   | $9.751 \times 10^{-4}$ | $1.001 \times 10^{-3}$ |
|          |     |        | $4.007 \times 10^{-3}$ | $5.080 \times 10^{-3}$ |
|          |     |        | $6.665 \times 10^{-3}$ | $1.154 \times 10^{-2}$ |
|          | 5   | 1000   | $3.086 \times 10^{-3}$ | $7.191 \times 10^{-4}$ |
|          |     |        | $5.960 \times 10^{-3}$ | $2.344 \times 10^{-3}$ |
|          |     |        | $1.196 \times 10^{-2}$ | $6.118 \times 10^{-3}$ |
|          | 8   | 1000   | $1.244 \times 10^{-2}$ | $6.285 \times 10^{-3}$ |
|          |     |        | $2.375 \times 10^{-2}$ | $2.032 \times 10^{-2}$ |
|          |     |        | $9.649 \times 10^{-2}$ | 1.133                  |
|          | 10  | 1500   | $8.849 \times 10^{-3}$ | $2.848 \times 10^{-3}$ |
|          |     |        | $1.188 \times 10^{-2}$ | $6.110 \times 10^{-3}$ |
|          |     |        | $2.083 \times 10^{-2}$ | $4.834 \times 10^{-1}$ |
|          | 15  | 2000   | $1.401 \times 10^{-2}$ | $4.647 \times 10^{-3}$ |
|          |     |        | $2.145 \times 10^{-2}$ | $1.110 \times 10^{-2}$ |
|          |     |        | $4.195 \times 10^{-2}$ | 1.271                  |
| DTLZ4    | 3   | 600    | $2.915 \times 10^{-4}$ | $1.092 \times 10^{-1}$ |
|          |     |        | $5.970 \times 10^{-4}$ | $4.277 \times 10^{-1}$ |
|          |     |        | $4.286 \times 10^{-1}$ | $5.235 \times 10^{-1}$ |
|          | 5   | 1000   | $9.849 \times 10^{-4}$ | $2.342 \times 10^{-1}$ |
|          |     |        | $1.255 \times 10^{-3}$ | $4.384 \times 10^{-1}$ |
|          |     |        | $1.721 \times 10^{-3}$ | $7.347 \times 10^{-1}$ |
|          | 8   | 1250   | $5.079 \times 10^{-3}$ | $3.821 \times 10^{-1}$ |
|          |     |        | $7.054 \times 10^{-3}$ | $8.015 \times 10^{-1}$ |
|          |     |        | $6.051 \times 10^{-1}$ | $9.686 \times 10^{-1}$ |
|          | 10  | 2000   | $5.694 \times 10^{-3}$ | $5.083 \times 10^{-1}$ |
|          |     |        | $6.337 \times 10^{-3}$ | $8.321 \times 10^{-1}$ |
|          |     |        | $1.076 \times 10^{-1}$ | 1.024                  |
|          | 15  | 3000   | $7.110 \times 10^{-3}$ | $9.406 \times 10^{-1}$ |
|          |     |        | $3.431 \times 10^{-1}$ | 1.157                  |
|          |     |        | <b>1.073</b>           | 1.219                  |

TABLE V  
IGD VALUES FOR NSGA-III AND MOEA/D-PBI APPROACHES ON THREE TO 15-OBJECTIVE WFG PROBLEMS.

| Problem | $M$ | MaxGen | NSGA-III               | MOEA/D-PBI             |
|---------|-----|--------|------------------------|------------------------|
| WFG6    | 3   | 400    | $4.828 \times 10^{-3}$ | $1.015 \times 10^{-2}$ |
|         |     |        | $1.224 \times 10^{-2}$ | $3.522 \times 10^{-2}$ |
|         |     |        | $5.486 \times 10^{-2}$ | $1.066 \times 10^{-1}$ |
|         | 5   | 750    | $5.065 \times 10^{-3}$ | $8.335 \times 10^{-3}$ |
|         |     |        | $1.965 \times 10^{-2}$ | $4.230 \times 10^{-2}$ |
|         |     |        | $4.474 \times 10^{-2}$ | $1.058 \times 10^{-1}$ |
|         | 8   | 1500   | $1.009 \times 10^{-2}$ | $1.757 \times 10^{-2}$ |
|         |     |        | $2.922 \times 10^{-2}$ | $5.551 \times 10^{-2}$ |
|         |     |        | $7.098 \times 10^{-2}$ | $1.156 \times 10^{-1}$ |
|         | 10  | 2000   | $1.060 \times 10^{-2}$ | $9.924 \times 10^{-3}$ |
|         |     |        | $2.491 \times 10^{-2}$ | $4.179 \times 10^{-2}$ |
|         |     |        | $6.129 \times 10^{-2}$ | $1.195 \times 10^{-1}$ |
|         | 15  | 3000   | $1.368 \times 10^{-2}$ | $1.513 \times 10^{-2}$ |
|         |     |        | $2.877 \times 10^{-2}$ | $6.782 \times 10^{-2}$ |
|         |     |        | $6.970 \times 10^{-2}$ | $1.637 \times 10^{-1}$ |
| WFG7    | 3   | 400    | $2.789 \times 10^{-3}$ | $1.033 \times 10^{-2}$ |
|         |     |        | $3.692 \times 10^{-3}$ | $1.358 \times 10^{-2}$ |
|         |     |        | $4.787 \times 10^{-3}$ | $1.926 \times 10^{-2}$ |
|         | 5   | 750    | $8.249 \times 10^{-3}$ | $8.780 \times 10^{-3}$ |
|         |     |        | $9.111 \times 10^{-3}$ | $1.101 \times 10^{-2}$ |
|         |     |        | $1.050 \times 10^{-2}$ | $1.313 \times 10^{-2}$ |
|         | 8   | 1500   | $2.452 \times 10^{-2}$ | $1.355 \times 10^{-2}$ |
|         |     |        | $2.911 \times 10^{-2}$ | $1.573 \times 10^{-2}$ |
|         |     |        | $6.198 \times 10^{-2}$ | $2.626 \times 10^{-2}$ |
|         | 10  | 2000   | $3.228 \times 10^{-2}$ | $1.041 \times 10^{-2}$ |
|         |     |        | $4.292 \times 10^{-2}$ | $1.218 \times 10^{-2}$ |
|         |     |        | $9.071 \times 10^{-2}$ | $1.490 \times 10^{-2}$ |
|         | 15  | 3000   | $3.457 \times 10^{-2}$ | $7.552 \times 10^{-3}$ |
|         |     |        | $5.450 \times 10^{-2}$ | $1.063 \times 10^{-2}$ |
|         |     |        | $8.826 \times 10^{-2}$ | $2.065 \times 10^{-2}$ |

TABLE VI  
BEST, MEDIAN AND WORST IGD AND CONVERGENCE METRIC VALUES OBTAINED FOR NSGA-III AND CLASSICAL GENERATIVE METHODS FOR THREE-OBJECTIVE DTLZ1 AND DTLZ2 PROBLEMS.

| Prob. | FE     | NSGA-III               |                        | Generative Method      |                        |
|-------|--------|------------------------|------------------------|------------------------|------------------------|
|       |        | IGD                    | GD                     | IGD                    | GD                     |
| DTLZ1 | 36,400 | $4.880 \times 10^{-4}$ | $4.880 \times 10^{-4}$ | $6.400 \times 10^{-2}$ | $1.702 \times 10^1$    |
|       |        | $1.308 \times 10^{-3}$ | $6.526 \times 10^{-4}$ | $8.080 \times 10^{-2}$ | $1.808 \times 10^1$    |
|       |        | $4.880 \times 10^{-3}$ | $7.450 \times 10^{-4}$ | $1.083 \times 10^{-1}$ | $1.848 \times 10^1$    |
| DTLZ2 | 22,750 | $1.262 \times 10^{-3}$ | $1.264 \times 10^{-3}$ | $1.113 \times 10^{-3}$ | $9.678 \times 10^{-5}$ |
|       |        | $1.357 \times 10^{-3}$ | $1.270 \times 10^{-3}$ | $6.597 \times 10^{-3}$ | $1.019 \times 10^{-4}$ |
|       |        | $2.114 \times 10^{-3}$ | $1.274 \times 10^{-3}$ | $9.551 \times 10^{-3}$ | $1.082 \times 10^{-4}$ |

### B. Classical Generative Method

For solving many-objective optimization problems, we require a set of reference points – either supplied by the user or systematically constructed as discussed earlier. One may then ponder how the proposed NSGA-III method will compare with a classical generative method in which multiple scalarized single-objective optimization problems can be formulated for each of the preferred or structured reference points and solved independently. For this purpose, we minimize the PBI-metric for each case. Along the reference direction obtained by joining the ideal point ( $\mathbf{z}^*$ ) to the supplied reference point ( $\bar{\mathbf{z}}$ ) dictated by a  $\mathbf{w}$ -vector (a unit vector  $\mathbf{w} = (\bar{\mathbf{z}} - \mathbf{z}^*)/|\bar{\mathbf{z}} - \mathbf{z}^*|$ ), the distance ( $d_1$ ) along the  $\mathbf{w}$ -direction and the distance ( $d_2$ ) perpendicular to the  $\mathbf{w}$ -direction are computed for any point  $\mathbf{x}$ . A weighted-sum of these two directions is then minimized, as follows:

$$\text{Minimize}_{\mathbf{x}} \quad d_1 + \theta d_2 = \mathbf{w}^T \mathbf{f}(\mathbf{x}) + \theta \|\mathbf{f}(\mathbf{x}) - \mathbf{w}^T \mathbf{f}(\mathbf{x}) \mathbf{w}\|. \quad (7)$$

The parameter  $\theta$  is set to 5 for all reference points [10]. In other words, the above minimization process is likely to find the intersection point between the reference direction  $\mathbf{w}$  and the Pareto-optimal surface, if there exists an intersection. Since the ideal point ( $\mathbf{z}^*$ ) is required to be known for this method, we assume origin to be the ideal vector for this problem.

To make a fair comparison, for  $H$  reference points, we allocate a maximum of  $T = FE_{\text{NSGA-III}}/H$  (where  $FE_{\text{NSGA-III}}$  is the total number of solution evaluations needed by NSGA-III) function evaluations to each optimization process. For three-objective DTLZ1 and DTLZ2 problems,  $FE_{\text{NSGA-III}} = 92 \times 400$  or 36,800 was required to find 91 solutions. Therefore, we allocate 36,800/91 or 405 function evaluations to each optimization run by `fmincon` routine of Matlab. A random solution is used for initialization. Figures 22 and 23 show the final solutions obtained by this generative method.

The DTLZ1 problem has multiple local fronts and Matlab's `fmincon` routine could not find a Pareto-optimal solution every time with the allotted number of function evaluations. Table VI shows the IGD and GD metric (average distance of obtained points from the closest reference points, a metric opposite in sense to IGD metric) values. For three-objective DTLZ2 problem, the generative method is able to find a well-distributed set of points, as shown in the figure. However, the table indicates that the distribution of points is not as good as that obtained by NSGA-III with the number of function evaluations restricted in this study. The inherent parallel search of NSGA-III constitutes a more efficient optimization.

### C. Scaled Test Problems

To investigate the algorithm's performance on problems having differently scaled objective values, we consider DTLZ1 and DTLZ2 problems again, but now we modify them as follows. Objective  $f_i$  is multiplied by a factor  $10^i$ . To illustrate, objectives  $f_1$ ,  $f_2$  and  $f_3$  for the three-objective scaled DTLZ1 problem are multiplied by  $10^0$ ,  $10^1$  and  $10^2$ , respectively.

To handle different scaling of objectives and make the distances ( $d_1$  and  $d_2$ ) along and perpendicular to reference

directions in MOEA/D-PBI approach, we normalize the objective values using the procedure suggested for MOEA/D-TCH approach in the original MOEA/D study [10]. We also use the code from MOEA/D web-site [50] to obtain the points. Figures 24, 25, and 26 show the obtained distribution of points using NSGA-III, MOEA/D-PBI and MOEA/D-TCH approaches, respectively. It is clear that both MOEA/D approaches with objective normalization are not able to handle the scaling involved in the objectives adequately, whereas NSGA-III's operators, normalization of objectives, and adaptive hyperplane construction process are able to negotiate the scaling of the objectives quite well. In the scaled problems, IGD metric is computed by first normalizing the objective values using the ideal and nadir points of the exact Pareto-optimal front and then computing the IGD value using the reference points as before. Resulting IGD values are shown in Table VII. Similar

TABLE VII  
BEST, MEDIAN AND WORST IGD VALUES FOR THREE-OBJECTIVE SCALED DTLZ1 AND DTLZ2 PROBLEMS USING NSGA-III AND MOEA/D ALGORITHMS. A SCALING FACTOR OF  $10^i$ ,  $i = 1, 2, \dots, M$ , IS USED.

| Prob. | Max Gen | NSGA-III               | MOEA/D<br>-PBI         | MOEA/D<br>-TCH         |
|-------|---------|------------------------|------------------------|------------------------|
| DTLZ1 | 400     | $3.853 \times 10^{-4}$ | $2.416 \times 10^{-2}$ | $8.047 \times 10^{-2}$ |
|       |         | $1.214 \times 10^{-3}$ | $2.054 \times 10^{-1}$ | $2.051 \times 10^{-1}$ |
|       |         | $1.103 \times 10^{-2}$ | $4.113 \times 10^{-1}$ | $2.704 \times 10^{-1}$ |
| DTLZ2 | 250     | $1.347 \times 10^{-3}$ | $4.749 \times 10^{-2}$ | $2.350 \times 10^{-1}$ |
|       |         | $2.069 \times 10^{-3}$ | $3.801 \times 10^{-1}$ | $5.308 \times 10^{-1}$ |
|       |         | $5.284 \times 10^{-3}$ | $5.317 \times 10^{-1}$ | $5.321 \times 10^{-1}$ |

performance is also observed for the scaled DTLZ2 problem, as shown in Figures 27, 28, and 29 and results are tabulated in Table VII.

It is interesting to note that the MOEA/D-PBI algorithm that worked so well in the normalized test problems in the previous subsection did not perform well for the scaled version of the same problems. Practical problems are far from being normalized and the objectives are usually scaled differently. An efficient optimization algorithm must handle different scaling of objectives as effectively as shown by NSGA-III.

Next, Table VIII shows the IGD values of the obtained solutions for 5, 8, 10, and 15-objective scaled DTLZ1 and DTLZ2 problems. Although difficult to visualize, a small IGD value in each case refers to a well-distributed set of points. Due to the poor performance of MOEA/D algorithms in scaled problems, we did not apply them further to the above higher-dimensional scaled problems.

### D. Convex DTLZ2 Problem

DTLZ1 problem has a linear Pareto-optimal front and DTLZ2 to DTLZ4 problems have concave Pareto-optimal fronts. To investigate the performance of NSGA-III and both versions of MOEA/D on scalable problems having a convex Pareto-optimal front, we suggest a new problem based on DTLZ2 problem. After the objective values ( $f_i$ ,  $i = 1, 2, \dots, M$ ) are computed using the original DTLZ2 problem

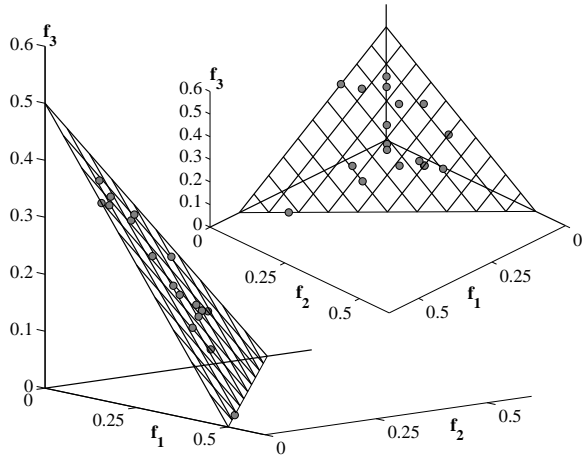


Fig. 22. Solutions obtained by classical generative method for DTLZ1. Only the points obtained close to the true Pareto-optimal front are shown.

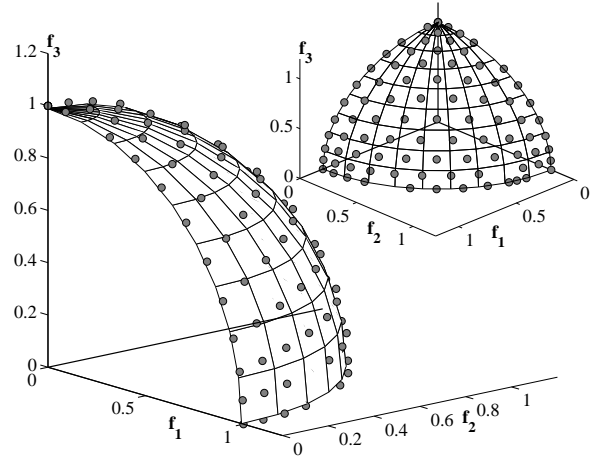


Fig. 23. Solutions obtained by classical generative method for DTLZ2.

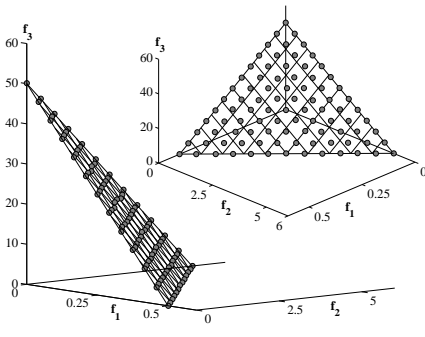


Fig. 24. Obtained solutions by NSGA-III for scaled DTLZ1.

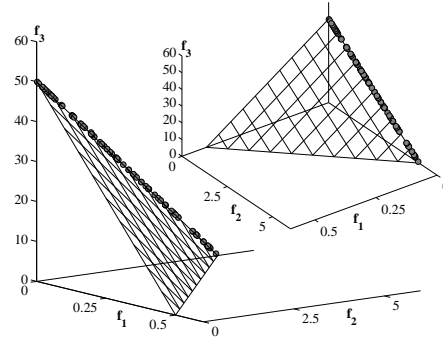


Fig. 25. Obtained solutions by MOEA/D-PBI for scaled DTLZ1.

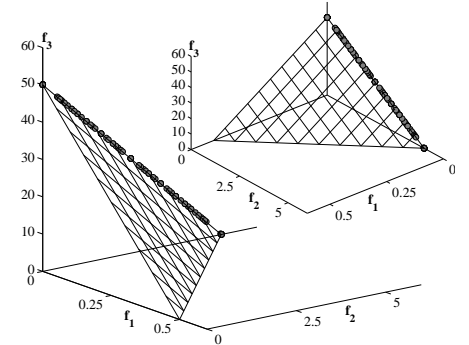


Fig. 26. Obtained solutions by MOEA/D-TCH for scaled DTLZ1.

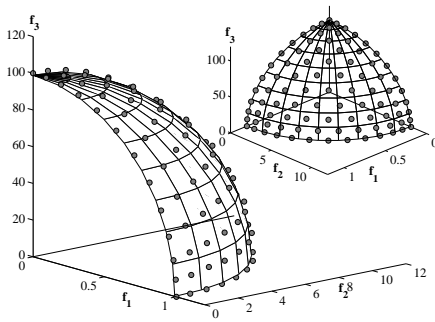


Fig. 27. Obtained solutions by NSGA-III for scaled DTLZ2.

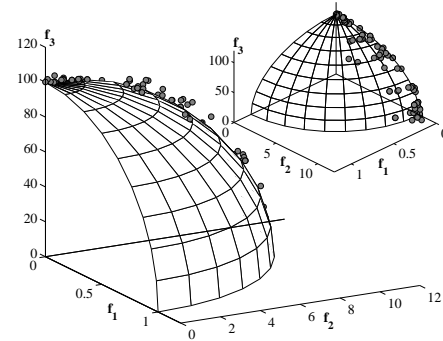


Fig. 28. Obtained solutions by MOEA/D-PBI for scaled DTLZ2.

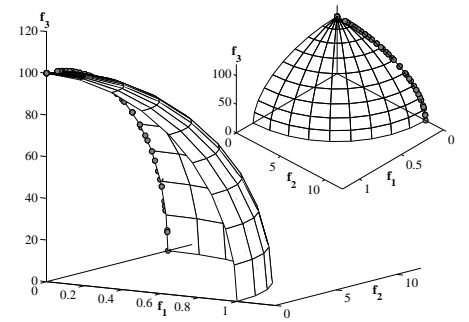


Fig. 29. Obtained solutions by MOEA/D-TCH for scaled DTLZ2.

description, we map them as follows:

$$\begin{aligned} f_i &\leftarrow f_i^4, \quad i = 1, 2, \dots, M-1, \\ f_M &\leftarrow f_M^2. \end{aligned}$$

Thus, the Pareto-optimal surface is given as

$$f_M + \sum_{i=1}^{M-1} \sqrt{f_i} = 1. \quad (8)$$

Figure 30 shows the obtained points on a three-objective convex Pareto-optimal front problem with  $H = 91$  reference

points. As the figure shows, the Pareto-optimal surface is almost flat at the edges, but changes sharply in the intermediate region. Although the reference points are uniformly placed on a normalized hyper-plane making an equal angle to all objective axes, the distribution of points on the Pareto-optimal front is not uniform. Despite such a nature of the front, NSGA-III is able to find a widely distributed set of points on the edges and also in the intermediate region of the surface. Note that the range of each objective for the Pareto-optimal set is identical (within  $[0,1]$ ), hence MOEA/D-PBI is expected to

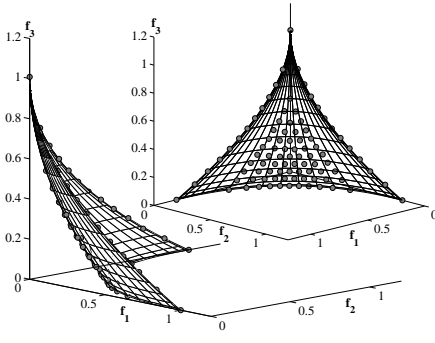


Fig. 30. Obtained solutions by NSGA-III for the convex Pareto-optimal front problem.

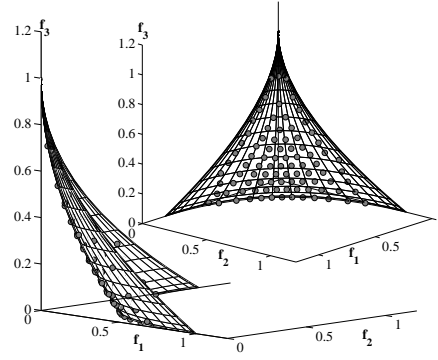


Fig. 31. Obtained solutions by MOEA/D-PBI for the convex Pareto-optimal front problem.

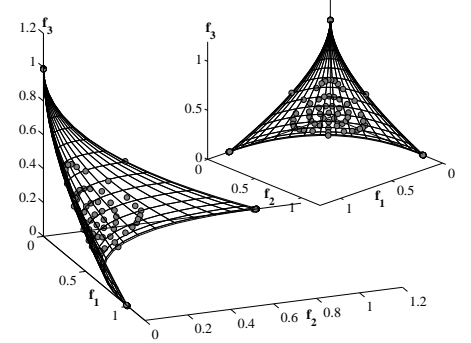


Fig. 32. Obtained solutions by MOEA/D-TCH for the convex Pareto-optimal front problem.

TABLE VIII  
BEST, MEDIAN AND WORST IGD VALUES FOR SCALED  $M$ -OBJECTIVE DTLZ1 AND DTLZ2 PROBLEMS.

| Problem | $M$ | Scaling factor | MaxGen | NSGA-III               |
|---------|-----|----------------|--------|------------------------|
| DTLZ1   | 3   | $10^i$         | 400    | $3.853 \times 10^{-4}$ |
|         |     |                |        | $1.214 \times 10^{-3}$ |
|         |     |                |        | $1.103 \times 10^{-2}$ |
|         | 5   | $10^i$         | 600    | $1.099 \times 10^{-3}$ |
|         |     |                |        | $2.500 \times 10^{-3}$ |
|         |     |                |        | $3.921 \times 10^{-2}$ |
|         | 8   | $3^i$          | 750    | $4.659 \times 10^{-3}$ |
|         |     |                |        | $1.051 \times 10^{-2}$ |
|         |     |                |        | $1.167 \times 10^{-1}$ |
|         | 10  | $2^i$          | 1000   | $3.403 \times 10^{-3}$ |
|         |     |                |        | $5.577 \times 10^{-3}$ |
|         |     |                |        | $3.617 \times 10^{-2}$ |
|         | 15  | $1.2^i$        | 1500   | $3.450 \times 10^{-3}$ |
|         |     |                |        | $6.183 \times 10^{-3}$ |
|         |     |                |        | $1.367 \times 10^{-2}$ |
| DTLZ2   | 3   | $10^i$         | 250    | $1.347 \times 10^{-3}$ |
|         |     |                |        | $2.069 \times 10^{-3}$ |
|         |     |                |        | $5.284 \times 10^{-3}$ |
|         | 5   | $10^i$         | 350    | $1.005 \times 10^{-2}$ |
|         |     |                |        | $2.564 \times 10^{-2}$ |
|         |     |                |        | $8.430 \times 10^{-2}$ |
|         | 8   | $3^i$          | 500    | $1.582 \times 10^{-2}$ |
|         |     |                |        | $1.788 \times 10^{-2}$ |
|         |     |                |        | $2.089 \times 10^{-2}$ |
|         | 10  | $3^i$          | 750    | $2.113 \times 10^{-2}$ |
|         |     |                |        | $3.334 \times 10^{-2}$ |
|         |     |                |        | $2.095 \times 10^{-1}$ |
|         | 15  | $2^i$          | 1000   | $2.165 \times 10^{-2}$ |
|         |     |                |        | $2.531 \times 10^{-2}$ |
|         |     |                |        | $4.450 \times 10^{-2}$ |

perform better. As shown in Figure 31, a nicely distributed set of points are found in the middle part of the front, but the algorithm fails to find points on the boundary of the Pareto-optimal front. This happens due to the very small slope of the surface at the boundary region. The use of penalty parameter value  $\theta = 5$  finds a non-boundary point to have a better PBI metric value for the boundary reference directions. A larger value of  $\theta$  may allow MOEA/D-PBI to find the exact boundary points, thereby requiring to do a parametric study with  $\theta$ . However, the advantage of no additional parameter in NSGA-III becomes clear from this problem. MOEA/D-TCH performs poorly again, as shown in Figure 32.

Table IX shows the best, median and worst IGD values of 20 independent runs (obtained using equation 6) for all three methods. Clearly, NSGA-III performs the best by finding a set of points having an order of magnitude smaller IGD values in three, five, eight, 10, and 15-objective versions of the convex DTLZ2 problem.

TABLE IX  
BEST, MEDIAN AND WORST IGD VALUES FOR THE CONVEX DTLZ2 PROBLEM.

| $M$ | MaxGen | NSGA-III               | MOEA/D-PBI             | MOEA/D-TCH             |
|-----|--------|------------------------|------------------------|------------------------|
| 3   | 250    | $2.603 \times 10^{-3}$ | $3.050 \times 10^{-2}$ | $6.835 \times 10^{-2}$ |
|     |        | $4.404 \times 10^{-3}$ | $3.274 \times 10^{-2}$ | $6.871 \times 10^{-2}$ |
|     |        | $8.055 \times 10^{-3}$ | $3.477 \times 10^{-2}$ | $6.912 \times 10^{-2}$ |
| 5   | 750    | $7.950 \times 10^{-3}$ | $1.082 \times 10^{-1}$ | $1.211 \times 10^{-1}$ |
|     |        | $1.341 \times 10^{-2}$ | $1.103 \times 10^{-1}$ | $1.223 \times 10^{-1}$ |
|     |        | $1.917 \times 10^{-2}$ | $1.117 \times 10^{-1}$ | $1.235 \times 10^{-1}$ |
| 8   | 2000   | $2.225 \times 10^{-2}$ | $2.079 \times 10^{-1}$ | $2.235 \times 10^{-1}$ |
|     |        | $2.986 \times 10^{-2}$ | $2.086 \times 10^{-1}$ | $2.396 \times 10^{-1}$ |
|     |        | $4.234 \times 10^{-2}$ | $2.094 \times 10^{-1}$ | $2.490 \times 10^{-1}$ |
| 10  | 4000   | $7.389 \times 10^{-2}$ | $2.117 \times 10^{-1}$ | $2.373 \times 10^{-1}$ |
|     |        | $9.126 \times 10^{-2}$ | $2.119 \times 10^{-1}$ | $2.472 \times 10^{-1}$ |
|     |        | $1.051 \times 10^{-1}$ | $2.120 \times 10^{-1}$ | $2.514 \times 10^{-1}$ |
| 15  | 4500   | $2.169 \times 10^{-2}$ | $3.681 \times 10^{-1}$ | $3.851 \times 10^{-1}$ |
|     |        | $2.769 \times 10^{-2}$ | $3.682 \times 10^{-1}$ | $3.911 \times 10^{-1}$ |
|     |        | $4.985 \times 10^{-2}$ | $3.683 \times 10^{-1}$ | $3.972 \times 10^{-1}$ |

To demonstrate the distribution of obtained points on a higher-objective problem, we show the obtained points of the median performed run of the 15-objective convex DTLZ2 problem on a value path using NSGA-III and MOEA/D-PBI approaches in Figures 33 and 34, respectively. It is clear from the plots that NSGA-III is able to find a good spread of solutions in the entire range of Pareto-optimal front ( $f_i \in [0, 1]$  for all  $i$ ), but MOEA/D-PBI is unable to find solutions having larger objective values.

Before we evaluate the proposed NSGA-III approach further, we highlight a couple of properties of MOEA/D-TCH and MOEA/D-PBI approaches that become clear from the above simulation results:

- 1) As observed in three-objective results with MOEA/D-TCH above (Figure 7, for example), it ends up generating a non-uniform distribution of points. This is due to the way the Tchebysheff metric is constructed. Moreover, multiple weight vectors produce an identical



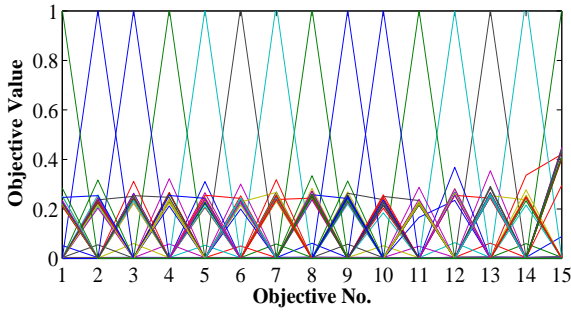


Fig. 33. NSGA-III Solutions are shown using 15-objective value path format for the convex problem.

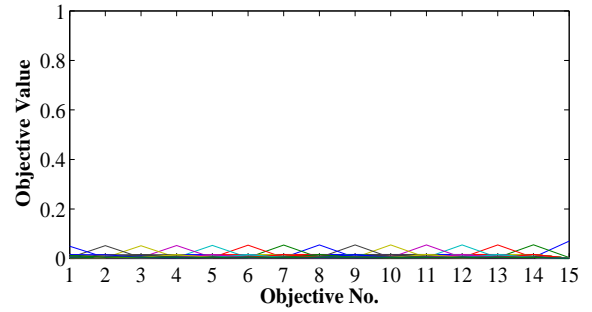


Fig. 34. MOEA/D-PBI Solutions are shown using 10-objective value path format for the convex problem.

extreme solution, thereby wasting in computational efforts.

- 2) Interestingly, depending on the parameter  $\theta$ , MOEA/D-PBI is potentially capable of generating dominated solutions. This certainly results in a waste of computational effort.

Although special care can be taken to reduce the chance of above, they are important for one to be aware while working with MOEA/D-PBI or MOEA/D-TCH approaches.

## VI. FURTHER INVESTIGATIONS OF NSGA-III

Having performed well on all test problems in above sections, we now investigate NSGA-III's performance in certain special types of many-objective problem solving tasks.

### A. Finding a Preferred Part of Pareto-front

In many-objective optimization problems, the user may not always be interested in finding the entire Pareto-optimal front. Practically speaking, an user is often interested in a preferred part of the Pareto-optimal front. In such a scenario, the user may represent his/her preferred region using a few representative reference points (or aspiration points). The aim in such a many-objective optimization task is to find the Pareto-optimal points that are in some sense *closest* to the supplied reference points.

Recall from Section IV-B that the proposed NSGA-III algorithm can also be applied with a set of  $H$  user-supplied reference points, instead of Das and Dennis's structured reference points used so far. Here, we demonstrate the working of NSGA-III for scaled DTLZ1 and DTLZ2 problems for a few user-supplied preference information. In such a case, in order to determine the right normalized hyper-plane, by default, we supply  $M$  additional reference points, one at each objective axis at an intercept of unity. The presence of these extreme points will attempt to keep extreme Pareto-optimal points, thereby forming a proper normalized hyper-plane needed to have appropriate scaling of the objectives.

For both scaled DTLZ1 and DTLZ2 problems, we supply 10 reference points in the middle of the normalized hyper-plane (scenario 1, as shown in Figure 35). As mentioned, three additional reference points  $(1, 0, 0)^T$ ,  $(0, 1, 0)^T$  and  $(0, 0, 1)^T$  are also included for the execution of the algorithm, thereby making a total of 13 supplied reference points. Figure 37

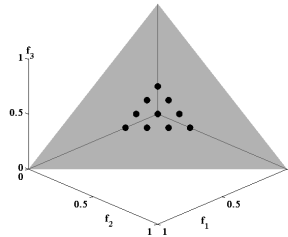


Fig. 35. Reference points used for Scenario 1 on normalized hyper-plane.

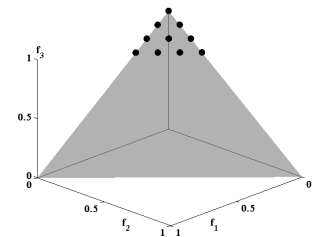


Fig. 36. Reference points used for Scenario 2 on normalized hyper-plane.

shows the points obtained by NSGA-III on the scaled Pareto-optimal front of DTLZ1 problem. Figure 38 shows 10 obtained points for the scaled DTLZ2 problem for scenario 1. Next, we supply a set of 10 reference points on a different part of the Pareto-optimal, as shown as scenario 2 in Figure 36. Figure 38 shows the obtained points on the scaled DTLZ2 problem. In both cases, a set of 10 near-Pareto-optimal points are obtained close to where the reference points were supplied.

Best, median and worst IGD values for the obtained points for three and 10-objective scaled DTLZ1 and DTLZ2 problems are shown in Table X. The IGD values are computed by normalizing the obtained objective values by theoretical ideal and nadir points and by projecting 10 supplied reference points on the normalized Pareto-optimal front. A small IGD value in each case ensures the convergence and diversity of the obtained solutions. The success of NSGA-III in finding just 10 Pareto-optimal points on 10-objective scaled problems indicates its potential use in finding a handful of preferred solutions in a many-objective practical optimization problem.

### B. Randomly generated preferred reference points

The supplied preferred reference points considered in the previous subsection are structured. In fact, they were created using Das and Dennis's [48] structured reference point creation procedure. Here, we investigate NSGA-III's ability to deal with a set of randomly created reference points. The procedure used here is the same as in previous subsection –  $M$  additional axis points are included for NSGA-III to adaptively create its hyper-plane.

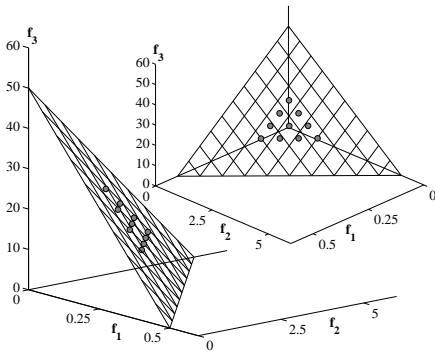


Fig. 37. Preferred solutions (scenario 1) obtained by NSGA-III for three-objective DTLZ1.

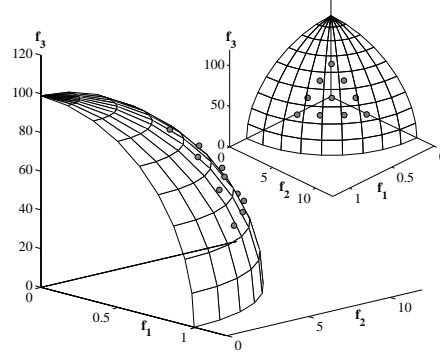


Fig. 38. Preferred solutions (scenario 1) obtained by NSGA-III for three-objective DTLZ2.

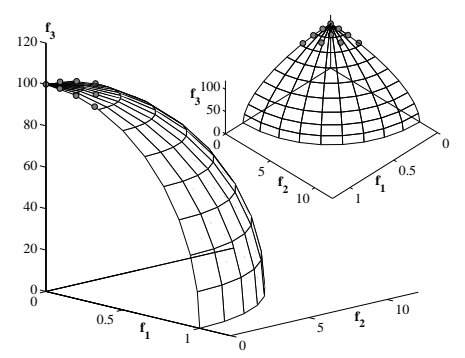


Fig. 39. Preferred solutions (scenario 2) obtained by NSGA-III for three-objective DTLZ2.

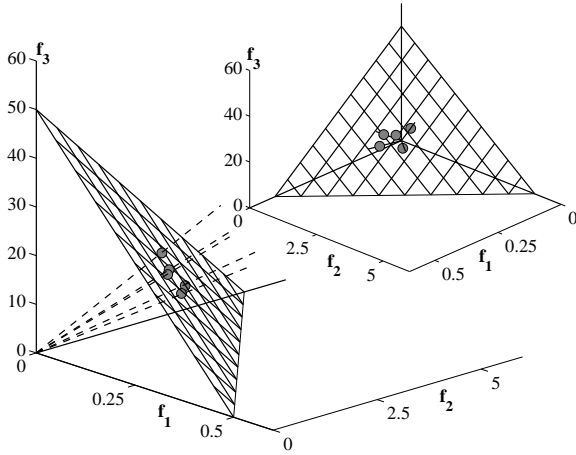


Fig. 40. Preferred solutions obtained by NSGA-III for three-objective DTLZ1.

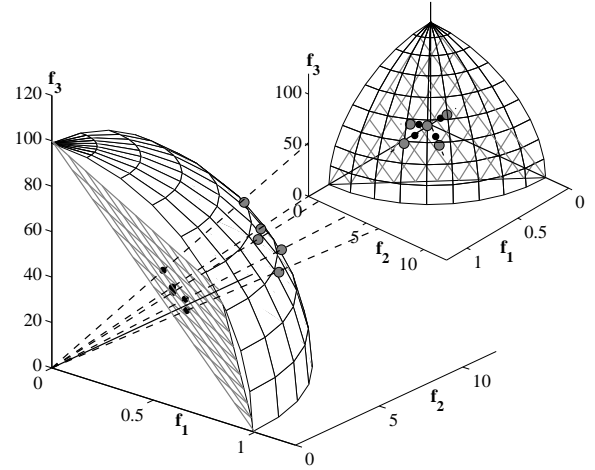


Fig. 41. Preferred solutions obtained by NSGA-III for three-objective DTLZ2.

For decision-making purposes, only a few trade-off points can be analyzed, even for a many-objective problem. Here, we investigate if the proposed NSGA-III is able to find only five Pareto-optimal points in a preferred region for three and 10-objective scaled DTLZ1 and DTLZ2 problems. Five random points are supplied in the intermediate region of the normalized hyper-plane (within  $z_i \in [0.4, 0.6]$  for all  $i$ ). Figures 40 and 41 show the obtained points on scaled three-objective DTLZ1 and DTLZ2 problems, respectively. Table XI shows the IGD values for three and 10-objective problems. In each case, a small IGD value indicates that NSGA-III is adequate to successfully find the desired solutions. This application shows promise in applying NSGA-III to many-objective problems (10-objective or like) with a handful (five or like) of preferred reference points – a matter which is of great importance to practitioners.

### C. Small population size

Previous subsections have shown that NSGA-III can work with a small number of reference points in order to find a few Pareto-optimal points in a preferred region. In previous sections, we have used a population size that is almost equal to the number of reference points. A natural question then arises: ‘Can NSGA-III work well with a small population size?’. We investigate this aspect here.

Table XII tabulates the number of reference points ( $H$ ) and corresponding layer-wise  $p$  and the maximum number of generations considered for different many-objective DTLZ2 problems. We perform this study for three, five, and 10-objective scaled DTLZ2 problems. Identical scaling factors as those shown in Table VIII are used here. In each case, the population size ( $N$ ) is always kept as the smallest number multiple of four and greater than the number of reference points ( $H$ ). A larger number of generations is kept for higher objective problems. The chosen maximum number of generations are also tabulated in the table.

For the three-objective DTLZ2 problem, when we are interested in finding 10 well-distributed Pareto-optimal points, we find that a population of size 12 is adequate to find all 10 Pareto-optimal points after 250 generations (Figure 42). For the 10-objective problem, if we desire 65 reference points (with  $p = 2$ ), we find that a population of size 68 is adequate to find all 65 solutions after 1,000 generations. The best, median and worst IGD values are shown in the table for three, five and 10-objective DTLZ2 problems. These results show that NSGA-III can work with a small population size due to its focus and balanced emphasis for finding near Pareto-optimal points corresponding to a supplied set of reference points.

TABLE X  
BEST, MEDIAN AND WORST IGD VALUES FOR SCALED DTLZ1 AND  
DTLZ2 PROBLEMS WITH 10 REFERENCE POINTS.

| Function                    | $M$ | $N$ | MaxGen | NSGA-III               |
|-----------------------------|-----|-----|--------|------------------------|
| DTLZ1                       | 3   | 28  | 750    | $4.945 \times 10^{-4}$ |
|                             |     |     |        | $4.076 \times 10^{-3}$ |
|                             |     |     |        | $1.898 \times 10^{-2}$ |
| (Scenario 1)<br>(Figure 35) | 10  | 100 | 2000   | $1.671 \times 10^{-3}$ |
|                             |     |     |        | $5.792 \times 10^{-3}$ |
|                             |     |     |        | $1.212 \times 10^{-1}$ |
| DTLZ1                       | 3   | 28  | 750    | $2.742 \times 10^{-4}$ |
|                             |     |     |        | $2.875 \times 10^{-3}$ |
|                             |     |     |        | $1.741 \times 10^{-2}$ |
| (Scenario 2)<br>(Figure 36) | 10  | 100 | 2000   | $4.194 \times 10^{-3}$ |
|                             |     |     |        | $7.393 \times 10^{-3}$ |
|                             |     |     |        | $5.799 \times 10^{-2}$ |
| DTLZ2                       | 3   | 28  | 250    | $6.459 \times 10^{-4}$ |
|                             |     |     |        | $1.473 \times 10^{-3}$ |
|                             |     |     |        | $3.883 \times 10^{-3}$ |
| (Scenario 1)<br>(Figure 35) | 10  | 100 | 1500   | $2.840 \times 10^{-3}$ |
|                             |     |     |        | $4.907 \times 10^{-3}$ |
|                             |     |     |        | $1.074 \times 10^{-2}$ |
| DTLZ2                       | 3   | 28  | 250    | $1.327 \times 10^{-3}$ |
|                             |     |     |        | $2.706 \times 10^{-3}$ |
|                             |     |     |        | $8.099 \times 10^{-3}$ |
| (Scenario 2)<br>(Figure 36) | 10  | 100 | 1500   | $1.576 \times 10^{-2}$ |
|                             |     |     |        | $2.548 \times 10^{-2}$ |
|                             |     |     |        | $5.893 \times 10^{-2}$ |

TABLE XI  
BEST, MEDIAN AND WORST IGD VALUES FOR SCALED DTLZ1 AND  
DTLZ2 PROBLEMS WITH RANDOMLY SUPPLIED REFERENCE POINTS.

| Problem | $M$ | $N$ | MaxGen | NSGA-III               |
|---------|-----|-----|--------|------------------------|
| DTLZ1   | 3   | 28  | 600    | $1.621 \times 10^{-3}$ |
|         |     |     |        | $9.870 \times 10^{-3}$ |
|         |     |     |        | $3.186 \times 10^{-1}$ |
|         | 10  | 100 | 2000   | $3.244 \times 10^{-3}$ |
|         |     |     |        | $8.227 \times 10^{-3}$ |
|         |     |     |        | $1.688 \times 10^{-2}$ |
| DTLZ2   | 3   | 28  | 250    | $5.763 \times 10^{-4}$ |
|         |     |     |        | $1.951 \times 10^{-3}$ |
|         |     |     |        | $6.766 \times 10^{-3}$ |
|         | 10  | 100 | 1500   | $1.317 \times 10^{-2}$ |
|         |     |     |        | $2.463 \times 10^{-2}$ |
|         |     |     |        | $6.439 \times 10^{-2}$ |

#### D. Nadir Point Estimation

In multi-objective optimization problems, the nadir point is important to be found for various reasons. A nadir point is constructed from the worst objective values for the entire Pareto-optimal set. Thus, along with the ideal point, nadir point can be used to normalize the objective functions for a systematic and reliable execution of many classical and non-classical optimization methods. In another study, a nadir point estimation strategy was proposed based on a bilevel evolutionary approach [54]. In that study, three to 20-objective DTLZ1 and DTLZ2 problems were considered for estimating the nadir point. Since studies in the previous two subsections have shown amply that NSGA-III can be used to find a few Pareto-optimal solutions even in 10-objective problems, here we apply NSGA-III to find only the extreme Pareto-optimal points so that the nadir point can be estimated accurately.

For this purpose, we suggest using  $M$  reference points, one on each objective axis, located at unity. We apply NSGA-III to

TABLE XII  
MINIMUM POPULATION SIZE AND BEST, MEDIAN AND WORST IGD  
VALUES OF THE OBTAINED SOLUTIONS FOR THE SCALED DTLZ2  
PROBLEM.

| $M$ | $(p^B, p^I)$ | $H$ | MaxGen | $N$ | NSGA-III               |
|-----|--------------|-----|--------|-----|------------------------|
| 3   | (3,0)        | 10  | 250    | 12  | $1.464 \times 10^{-3}$ |
|     |              |     |        |     | $4.206 \times 10^{-3}$ |
|     |              |     |        |     | $1.524 \times 10^{-1}$ |
| 5   | (2,1)        | 20  | 500    | 20  | $4.952 \times 10^{-3}$ |
|     |              |     |        |     | $8.537 \times 10^{-3}$ |
|     |              |     |        |     | $1.195 \times 10^{-1}$ |
| 10  | (2,1)        | 65  | 1,000  | 68  | $8.641 \times 10^{-3}$ |
|     |              |     |        |     | $1.155 \times 10^{-2}$ |
|     |              |     |        |     | $1.842 \times 10^{-2}$ |

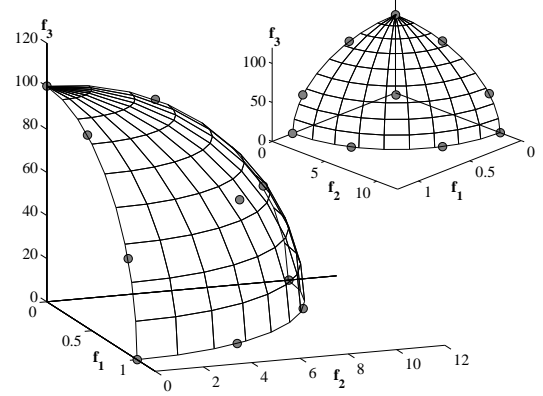


Fig. 42. Obtained points using NSGA-III for the scaled DTLZ2 with a small population size of 12.

three to 20-objective DTLZ1 and DTLZ2 problems and record the overall function evaluations needed to find the true nadir point with the following termination condition. When the error value ( $E$ ) computed as below

$$E = \sqrt{\sum_{i=1}^M \left( \frac{z_i^{\text{nad}} - z_i^{\text{est}}}{z_i^{\text{nad}} - z_i^*} \right)^2}, \quad (9)$$

is less than a threshold (0.01, used here), the run is terminated. Here,  $\mathbf{z}^*$ ,  $\mathbf{z}^{\text{nad}}$  and  $\mathbf{z}^{\text{est}}$  are the ideal point, the true nadir point, and the estimated nadir point by NSGA-III, respectively. Table XIII presents the results and compares them with the evaluations needed to find the nadir point with an identical termination condition in the previous study [54]. In most cases, NSGA-III requires smaller number of function evaluations than the previous procedure to locate the nadir point reliably.

Figure 43 shows the value path plot of obtained points for estimating the nadir point for the 10-objective DTLZ2 problems. Understandably, the points near the extreme of the Pareto-optimal front are found. From this plot, the nadir point of the problem is estimated to be  $(1, 1, 1, 1, 1, 1, 1, 1, 1, 1)^T$ .

## VII. TWO PROBLEMS FROM PRACTICE

After solving a number of test problems, we now apply NSGA-III to a couple of engineering design optimization problems. The first problem has three objectives and the second one has nine objectives.

TABLE XIII  
NADIR POINT ESTIMATION BY NSGA-III IN THREE TO 20-OBJECTIVE  
DTLZ1 AND DTLZ2 PROBLEMS.

| Problem | $M$ | NSGA-III         | NSGA-II [54]   |
|---------|-----|------------------|----------------|
| DTLZ1   | 3   | <b>7,400</b>     | 18,800         |
|         |     | <b>16,660</b>    | 26,500         |
|         |     | 72,180           | <b>45,700</b>  |
|         | 5   | <b>25,000</b>    | 35,300         |
|         |     | 62,900           | <b>58,400</b>  |
|         |     | 152,040          | <b>107,100</b> |
|         | 10  | <b>58,640</b>    | 239,800        |
|         |     | <b>259,760</b>   | 274,200        |
|         |     | 483,880          | <b>358,000</b> |
|         | 15  | <b>356,460</b>   | 750,000        |
|         |     | <b>589,140</b>   | 1,100,000      |
|         |     | <b>1,200,000</b> | 1,800,000      |
|         | 20  | <b>274,800</b>   | 2,600,000      |
|         |     | <b>956,960</b>   | 3,500,000      |
|         |     | <b>1,600,000</b> | 4,000,000      |
| DTLZ2   | 3   | <b>1,368</b>     | 4,100          |
|         |     | <b>1,920</b>     | 4,900          |
|         |     | <b>2,532</b>     | 5,500          |
|         | 5   | <b>4,320</b>     | 9,400          |
|         |     | <b>6,900</b>     | 11,400         |
|         |     | <b>13,000</b>    | 14,200         |
|         | 10  | <b>31,400</b>    | 77,600         |
|         |     | <b>52,840</b>    | 92,800         |
|         |     | <b>76,080</b>    | 128,000        |
|         | 15  | <b>153,540</b>   | 346,500        |
|         |     | <b>213,060</b>   | 429,000        |
|         |     | <b>316,860</b>   | 810,000        |
|         | 20  | <b>309,440</b>   | 1,089,000      |
|         |     | <b>589,520</b>   | 1,415,000      |
|         |     | <b>734,400</b>   | 2,102,000      |

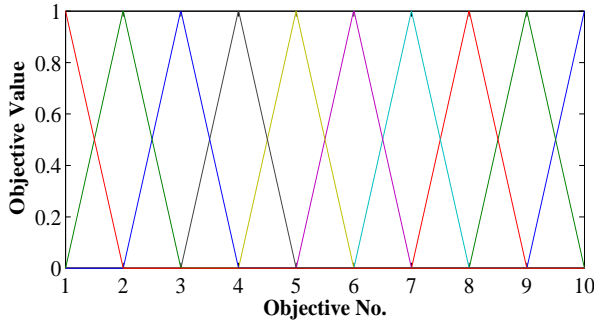


Fig. 43. A value path plot for the 10-objective DTLZ2 problem shows that NSGA-III is able to find the extreme objective values (zero and one) in each objective.

#### A. Crash-worthiness Design of Vehicles for Complete Trade-off Front

This problem aims at structural optimization of the frontal structure of vehicle for crash-worthiness [55]. Thickness of five reinforced members around the frontal structure are chosen as design variables, while mass of vehicle, deceleration during the full frontal crash (which is proportional to biomechanical injuries caused to the occupants) and toe board intrusion in the offset-frontal crash (which accounts for the structural integrity of the vehicle) are taken as objectives. Mathematical formulation for the three objectives can be found in the original study [55].

All objectives are to be minimized. For this problem, we choose  $p = 16$  so that there are  $H = \binom{3-1+16}{16}$  or 153

structured reference points. The reference points are initialized on the entire normalized hyper-plane in the three-objective space. NSGA-III is applied with 156 population size and run for 200 generations. Other parameters are the same as before. Figure 44 shows the obtained points with an open circle. The non-uniform scaling of objectives is clear from

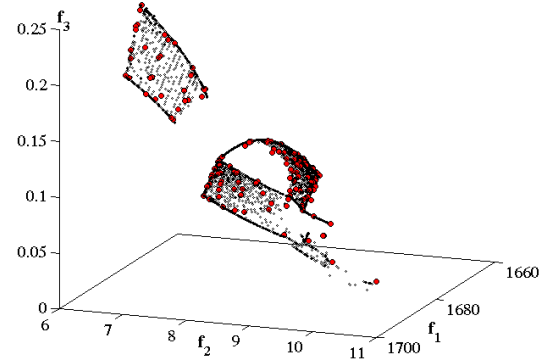


Fig. 44. 60 solutions are found on the entire front on the three-objective crash-worthiness problem.

the figure. Although 153 reference points were used, only 60 of them have representative solutions on the final front. Other 93 reference do not correspond to any Pareto-optimal point, thereby indicating possible 'holes' in the Pareto-optimal front correspond to remaining reference points. The ideal point ( $\mathbf{z}^*$ ) and the nadir point ( $\mathbf{z}^{\text{nad}}$ ) are estimated from the obtained front.

To investigate the nature of the true Pareto-optimal front, next, we create 7,381 reference points  $\mathbf{z}^{\text{ref},i}$  ( $i = 1, \dots, 7381$ ) with  $p = 120$ . Using the ideal and nadir points obtained as above, we normalize the objectives and then solve the achievement scalarization function (ASF) (equation 4) corresponding to each weight vector  $\mathbf{w}^i = \mathbf{z}^{\text{ref},i}$ . Each ASF is minimized using Matlab's fmincon routine. Resulting 7,381 points are collected and dominated points are removed. We noticed that only 4,450 points are non-dominated. These non-dominated points are shown in Figure 44 with a dot. It is clear that the NSGA-III obtained set of 60 points are widely-distributed on the entire front. Interestingly, 4,450 trade-off points reveal the complex nature of Pareto-optimal front (having holes and varying density of points) for this practical problem and the ability of NSGA-III to find a few widely-distributed points on the entire front.

To investigate the closeness of NSGA-III points with the classically optimized solutions, we compute the convergence metric (average distance of NSGA-III points from the closest fmincon-optimized points) and the minimum, median and maximum convergence metric values are found be 0.0019, 0.0022, and 0.0026, respectively. Since these values are small, they indicate that obtained NSGA-III solutions are close to the classically optimized front. This problem demonstrates the use of NSGA-III in finding a representative set of points on the entire Pareto-optimal front on problems having practical difficulties (such as differences in scaling of objectives, potential holes in the Pareto-optimal front, etc.).



### B. Car Cab Design with Preference Information

Next, we consider a vehicle performance optimization problem having 11 decision variables involving dimensions of the car body and bounds on natural frequencies. The problem involves nine objectives on ‘roominess’ of the car, fuel economy, acceleration time, and road noise at different speeds. Mathematical formulation can be found elsewhere [56].

In this problem, we demonstrate NSGA-III’s ability to use preference information in finding a few preferred non-dominated solutions on a part of the Pareto-optimal front, instead of on the complete front. We apply NSGA-III with only 10 reference points in the *intermediate* part of the Pareto-optimal front. For this purpose, we first create 10 random points on the entire unit hyper-plane and then ‘shrink’ them within 25% around the centroid of the unit hyper-plane. As suggested in Section VI-A, nine extreme reference points are also included for a better normalization purpose. A population of size 100 is used and NSGA-III is run for 4,000 generations. 20 independent runs are made with different sets of reference points and resulting solutions are compared with the classical achievement scalarizing function approach implemented with Matlab’s *fmincon* routine. To have good representative solutions in the chosen intermediate part, *fmincon* routine is run for 20,000 random reference points initialized in the same intermediate range one at a time. A domination check of final solutions make 11,280 of them non-dominated to each other. Each set of 10 solutions from NSGA-III runs is then compared against these 11,280 *fmincon* solutions and the GD metric value is computed. The solutions from the specific NSGA-III run corresponding to the median GD metric value is shown against the silhouette of 11,280 *fmincon* solutions on a value path plot in Figure 45 to make a visual comparison. The trade-off information and spread of obtained NSGA-III solutions are clear from the plot. While the large number of *fmincon* solutions provide a range of objective values for the chosen range of reference points, only one out of 10 NSGA-III solutions has two objective values ( $f_4$  and  $f_9$ ) outside the range. Other nine solutions quite nicely represent relevant part of the Pareto-optimal front. With NSGA-III’s ability to find just 10 preferred trade-off solutions on a nine-objective non-dominated front should remain as a significant step towards a convenient post-optimal multi-criterion decision-making task.

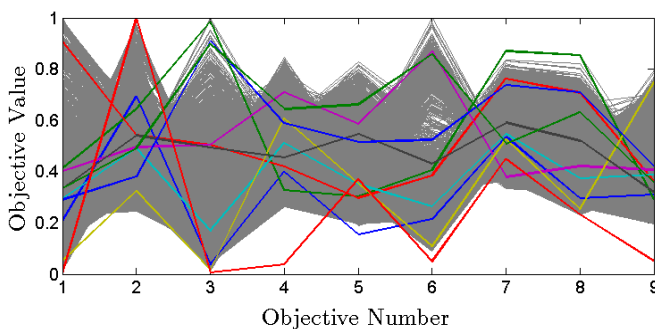


Fig. 45. A value path plot for the nine-objective car cab design problem shows that NSGA-III is able to find as few as 10 preferred solutions within the silhouette of solutions found by classical means.

To quantify the convergence of NSGA-III solutions, the best, median and worst GD metric values against 11,280 *fmincon* solutions on the normalized objective space are computed as follows: 0.0245, 0.0408, and 0.0566. These values are small enough to indicate the closeness of obtained NSGA-III solutions from the *fmincon* solutions. The ability of NSGA-III to find a handful of trade-off points in a preferred part of a nine-objective practical optimization problem having different scaling of objectives shows a promise of its use in practical many-objective optimization problems.

## VIII. CONCLUSIONS

In this paper, we have suggested a reference point based approach to an earlier-proposed NSGA-II framework for solving many-objective optimization problems. The proposed NSGA-III approach has been applied to three to 15-objective existing and new test problems and to three and nine-objective practical problems. The test problems involve fronts that have convex, concave, disjointed, differently scaled, biased density of points across the front, and multimodality involving multiple local fronts where an optimization algorithm can get stuck to. In all such problems, the proposed NSGA-III approach has been able to successfully find a well-converged and well-diversified set of points repeatedly over multiple runs. The performance scaling to 15-objectives is achieved mainly due to the aid in diversity preservation by supplying a set of well-distributed reference points. In higher-dimensional problems, EMO algorithms face with an increasingly difficult task of maintaining diversity as well as in converging to the Pareto-optimal front. The supply of a set of reference points and NSGA-III’s efficient niching methodology in finding a Pareto-optimal solution associated to each reference point has made the diversity preservation of obtained solutions in as large as 15-objectives possible.

The performance of NSGA-III has been compared with several versions of a recently proposed MOEA/D procedure. Although different MOEA/Ds have shown their working on different problems, no single version is able to solve all problems efficiently. Having solved all problems well by the proposed NSGA-III procedure, there is another advantage of it that is worth mentioning here. Unlike MOEA/D versions, NSGA-III procedure does not require any additional parameter to be set.

Furthermore, NSGA-III has been tested for its ability to solve a number of different types of many-objective problem solving tasks. It has been demonstrated that NSGA-III is able to work with a small number of user-supplied structured or randomly assigned reference points, thereby making the method suitable for a many-objective preference-based optimization-cum-decision-making approach. It has also been shown that the NSGA-III procedure can be used to quickly estimate the nadir point in many-objective optimization problems compared to another EMO-based method proposed earlier. For a practical application, only a few handful trade-off points are required for decision-making in multi- and many-objective optimization problems. It has been shown that NSGA-III can be used to find only a few points (20 in five-objective and

65 in a 10-objective problem) with a small population size, thereby reducing the computational efforts.

The proposed NSGA-III approach with a supply of a set of reference points has a similar implicit principle to the hyper-grid based archiving techniques, such as adaptive grid algorithm [57], [58] and  $\epsilon$ -MOEA [59], [60]. Grid based methods require exponentially more grids to be considered with an increase in objectives, whereas  $\epsilon$ -domination concept, although reduce the burden of finding an appropriate diversity in solutions somewhat, still poses enough difficulties in finding well-distributed Pareto-optimal solutions for a large-dimensional problem. However, it remains as an interesting future study to compare NSGA-III's performance with an  $\epsilon$ -domination based algorithm for many-objective optimization problems. NSGA-III's performance has been found to be much better than a classical generating method in many-objective problems. However, a more thorough study using population-based aggregation methods [61], [62] in which multiple search directions are used simultaneously is worth pursuing.

Results on many different problems with box constraints used in this paper have clearly shown promise for NSGA-III's further application in other challenging problems. In the sequel of this paper [52], we have extended NSGA-III to solve constrained many-objective optimization problems having generic inequality and equality constraints and also suggested an adaptive NSGA-III approach for adding and deleting the supplied reference points to maximize the number of obtained Pareto-optimal solutions in a computationally efficient manner. Nevertheless, this paper has addressed the issues and difficulties related to solving many-objective optimization problems, reviewed some past studies in this direction, suggested a viable many-objective evolutionary algorithm, and demonstrated proof-of-principle results on many problems including two engineering design problems.

## REFERENCES

- [1] O. Chikumbo, E. Goodman, and K. Deb, "Approximating a multi-dimensional pareto front for a land use management problem: A modified MOEA with an epigenetic silencing metaphor," in *Proceedings of Congress on Evolutionary Computation (CEC-2012)*, 2012, pp. 1–8.
- [2] C. A. C. Coello and G. B. Lamont, *Applications of Multi-Objective Evolutionary Algorithms*. World Scientific, 2004.
- [3] K. Deb, *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.
- [4] K. Deb and D. Saxena, "Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems," in *Proceedings of the World Congress on Computational Intelligence (WCCI-2006)*, 2006, pp. 3352–3360.
- [5] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [6] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, K. C. Giannakoglou, D. T. Tsahalis, J. Périaux, K. D. Papailiou, and T. Fogarty, Eds. Athens, Greece: International Center for Numerical Methods in Engineering (CIMNE), 2001, pp. 95–100.
- [7] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.
- [8] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.
- [9] E. J. Hughes, "Evolutionary many-objective optimisation: Many once or one many?" in *IEEE Congress on Evolutionary Computation (CEC-2005)*, 2005, pp. 222–227.
- [10] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 6, pp. 712–731, 2007.
- [11] D. K. Saxena, J. A. Duro, A. Tiwari, K. Deb, and Q. Zhang, "Objective reduction in many-objective optimization: Linear and nonlinear algorithms," *IEEE Transactions on Evolutionary Computation*, in press.
- [12] J. A. López and C. A. C. Coello, "Some techniques to deal with many-objective problems," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*. New York: ACM, 2009, pp. 2693–2696.
- [13] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *Proceedings of Congress on Evolutionary Computation (CEC-2008)*, 2008, pp. 2424–2431.
- [14] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, in press.
- [15] H. Nakayama, K. Kaneshige, S. Takemoto, and Y. Watada, "An application of a multiobjective programming technique to construction accuracy control of cable-stayed bridges," *European Journal of Operations Research*, vol. 87, no. 3, pp. 731–738, 1995.
- [16] M. Garza-Fabre, G. T. Pulido, and C. A. C. Coello, "Ranking methods for many-objective optimization," in *Proceedings of Mexican International Conference on Artificial Intelligence (MICA-2009)*. Springer, 2009, pp. 633–645.
- [17] V. Chankong and Y. Y. Haimes, *Multiobjective Decision Making Theory and Methodology*. New York: North-Holland, 1983.
- [18] C. M. Fonseca, L. Paquete, and M. López-Ibáñez, "An improved dimension sweep algorithm for the hypervolume indicator," in *Proceedings of the 2006 Congress on Evolutionary Computation (CEC'06)*. IEEE Press, Piscataway, NJ, 2006, pp. 1157–1163.
- [19] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.
- [20] K. Deb, J. Sundar, N. Uday, and S. Chaudhuri, "Reference point based multi-objective optimization using evolutionary algorithms," *International Journal of Computational Intelligence Research (IJCIR)*, vol. 2, no. 6, pp. 273–286, 2006.
- [21] K. Deb and A. Kumar, "Interactive evolutionary multi-objective optimization and decision-making using reference direction method," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*. New York: The Association of Computing Machinery (ACM), 2007, pp. 781–788.
- [22] —, "Light beam search based multi-objective optimization using evolutionary algorithms," in *Proceedings of the Congress on Evolutionary Computation (CEC-07)*, 2007, pp. 2125–2132.
- [23] U. K. Wickramasinghe and X. Li, "A distance metric for evolutionary many-objective optimization algorithms using user-preferences," in *Proceedings of Artificial Intelligence conference (AI-2009)*, (LNAI 5866). Heidelberg, Germany: Springer-Verlag, 2009, pp. 443–453.
- [24] D. Brockhoff and E. Zitzler, "Dimensionality Reduction in Multiobjective Optimization: The Minimum Objective Subset Problem," in *Operations Research Proceedings 2006*, K. H. Waldmann and U. M. Stocker, Eds. Springer, 2007, pp. 423–429.
- [25] H. K. Singh, A. Isaacs, and T. Ray, "A Pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 4, pp. 539–556, 2011.
- [26] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multi-objective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.
- [27] D. Hadka and P. Reed, "Diagnostic assessment of search controls and failure modes in many-objective evolutionary optimization," *Evolutionary Computation Journal*, in press.
- [28] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.
- [29] M. Farina and P. Amato, "A fuzzy definition of 'optimality' for many-criteria decision-making and optimization problems," *IEEE Trans. on Sys. Man and Cybern.*, vol. 34, no. 3, pp. 315–326, 2004.
- [30] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 38, no. 5, pp. 1402–1412, 2008.



- [31] H. Aguirre and K. Tanaka, "Many-objective optimization by space partitioning and adaptive epsilon-ranking on mnk-landscapes," in *Proceedings of 5th International Conference on Evolutionary Multi-Criterion Optimization (EMO 2009)*, 2009, pp. 407–422, lecture Notes in Computer Science 5467.
- [32] H. Sato, H. E. Aguirre, and K. Tanaka, "Pareto partial dominance moea in many-objective optimization," in *Proceedings of Congress on Computational Intelligence (CEC-2010)*, 2010, pp. 1–8.
- [33] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston: Kluwer, 1999.
- [34] J. Branke, T. Kaußler, and H. Schmeck, "Guidance in evolutionary multi-objective optimization," *Advances in Engineering Software*, vol. 32, pp. 499–507, 2001.
- [35] G. Eichfelder, "Optimal elements in vector optimization with a variable ordering structure," *Journal of Optimization Theory and Applications*, vol. 151, no. 2, pp. 217–240, 2011.
- [36] K. Deb and H. Jain, "Handling many-objective problems using an improved NSGA-II procedure," in *Proceedings of World Congress on Computational Intelligence (WCCI-2012)*, 2012, pp. 1–8.
- [37] Q. Zhang, H. Li, D. Maringer, and E. Tsang, "MOEA/D with NBI-style Tchebycheff approach for portfolio management," in *IEEE Congress on Evolutionary Computation*, 2010.
- [38] A. Zhou, Q. Zhang, and Y. Jin, "Approximating the set of pareto optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm," *IEEE Transaction on Evolutionary Computation*, vol. 13, no. 5, pp. 1167–1189, 2009.
- [39] R. C. Purshouse and P. J. Fleming, "On the evolutionary optimization of many conflicting objectives," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, 2007.
- [40] S. F. Adra and P. J. Fleming, "Diversity management in evolutionary many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 2, pp. 183–195, 2011.
- [41] M. Köppen and K. Yoshida, "Substitute distance assignments in nsga-ii for handling many-objective optimization problems," in *Proceedings of Fourth International Conference on Multi-Objective Optimization, EMO 2007 (LNCS 4403)*. Heidelberg, Germany: Springer-Verlag, 2007, pp. 727–741.
- [42] D. Hadka and P. Reed, "Borg: An auto-adaptive many-objective evolutionary computing framework," *Evolutionary Computation*, p. in press, 2011.
- [43] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," *Evolutionary Computation Journal*, vol. 19, no. 1, pp. 45–76, 2011.
- [44] J. Bader, K. Deb, and E. Zitzler, "Faster hypervolume-based search using Monte Carlo sampling," in *Proceedings of Multiple Criteria Decision Making (MCDM 2008)*. Heidelberg: Springer, 2010, pp. 313–326, INEMS-634.
- [45] L. Bradstreet, L. While, and L. Barone, "A fast incremental hypervolume algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 714–723, 2008.
- [46] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans on Evolutionary Computation*, vol. 12, no. 2, pp. 284–302, 2009.
- [47] Q. Zhang, A. Zhou, S. Z. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the cec-2009 special session and competition," Singapore: Nanyang Technological University, Tech. Rep., 2008.
- [48] I. Das and J. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal of Optimization*, vol. 8, no. 3, pp. 631–657, 1998.
- [49] H. T. Kung, F. Luccio, and F. P. Preparata, "On finding the maxima of a set of vectors," *Journal of the Association for Computing Machinery*, vol. 22, no. 4, pp. 469–476, 1975.
- [50] Q. Zhang, "MOEA/D homepage," <http://dces.essex.ac.uk/staff/zhang/webofmoead.htm>.
- [51] D. V. Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," Dayton, OH: Department of Electrical and Computer Engineering, Air Force Institute of Technology, Tech. Rep. TR-98-03, 1998.
- [52] K. Deb and H. Jain, "An improved NSGA-II procedure for many-objective optimization, Part II: Handling constraints and extending to an adaptive approach," Indian Institute of Technology Kanpur, Tech. Rep. 2012010, 2012.
- [53] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multi-objective optimization," in *Evolutionary Multi-objective Optimization*, A. Abraham, L. Jain, and R. Goldberg, Eds. London: Springer-Verlag, 2005, pp. 105–145.
- [54] K. Deb, K. Miettinen, and S. Chaudhuri, "Towards an estimation of nadir objective vector using a hybrid of evolutionary and local search approaches," *IEEE Transactions on Evolutionary Computation*, in press.
- [55] X. Liao, Q. Li, W. Zhang, and X. Yang, "Multiobjective optimization for crash safety design of vehicle using stepwise regression model," *Structural and Multidisciplinary Optimization*, vol. 35, pp. 561–569, 2008.
- [56] K. Deb, S. Gupta, D. Daum, J. Branke, A. Mall, and D. Padmanabhan, "Reliability-based optimization using evolutionary algorithms," *IEEE Trans. on Evolutionary Computation*, vol. 13, no. 5, pp. 1054–1074, 2009.
- [57] J. D. Knowles and D. W. Corne, "Quantifying the effects of objective space dimension in evolutionary multiobjective optimization," in *Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization*, 2007, pp. 757–771.
- [58] D. W. Corne, N. R. Jerram, J. D. Knowles, and M. J. Oates, "PESA-II: Region-based selection in evolutionary multiobjective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*. San Mateo, CA: Morgan Kaufmann Publishers, 2001, pp. 283–290.
- [59] M. Laumanns, G. Rudolph, and H. P. Schwefel, "A spatial predator-prey approach to multi-objective optimization: A preliminary study," in *Proceedings of the Parallel Problem Solving from Nature, V*, 1998, pp. 241–249.
- [60] K. Deb, M. Mohan, and S. Mishra, "Towards a quick computation of well-spread Pareto-optimal solutions," in *Proceedings of the Second Evolutionary Multi-Criterion Optimization (EMO-03) Conference (LNCS 2632)*, 2003, pp. 222–236.
- [61] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and reviews*, vol. 28, no. 3, pp. 392–403, 1998.
- [62] A. Jaszkiewicz, "Genetic local search for multiple objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, no. 1, pp. 50–71, 2002.



**Kalyanmoy Deb** Kalyanmoy Deb is the Koenig Endowed Chair Professor at the Department of Electrical and Computer Engineering at Michigan State University (MSU), East Lansing, USA. Prior to this position, he was at Indian Institute of Technology Kanpur in India. Prof. Deb's main research interests are in evolutionary optimization algorithms and their application in optimization and machine learning. He is largely known for his seminal research in Evolutionary Multi-Criterion Optimization. He was awarded the prestigious 'Infosys Prize' in 2012, 'TWAS Prize' in Engineering Sciences in 2012, 'CajAstur Mamdani Prize' in 2011, 'JC Bose National Fellowship' in 2011, 'Distinguished Alumni Award' from IIT Kharagpur in 2011, 'Edgeworth-Pareto' award in 2008, Shanti Swarup Bhatnagar Prize in Engineering Sciences in 2005, 'Thomson Citation Laureate Award' from Thomson Reuters. His 2002 IEEE-TEC NSGA-II paper is now judged as the Most Highly Cited paper and a Current Classic by Thomson Reuters having more than 4,000+ citations. He is a fellow of IEEE. He has written two text books on optimization and more than 340 international journal and conference research papers. He is in the editorial board on 18 major international journals, including IEEE TEC.



**Himanshu Jain** Himanshu Jain was born in Nehtaur, India, on December 6, 1989. He received his dual Bachelor's and Master's degree in Mechanical Engineering from Indian Institute of Technology Kanpur in 2012. He was a member of the Kanpur Genetic Algorithms Laboratory (KanGAL) from 2009. He is currently working as an Engineer at Eaton Corporation, Pune. His research interests include evolutionary computation and machine learning.