Evolutionary Path Control Strategy for Solving Many-Objective Optimization Problem

Proteek Chandan Roy, Md. Monirul Islam, *Member, IEEE*, Kazuyuki Murase, and Xin Yao, *Fellow, IEEE*

***Abstract*—The number of objectives in many-objective optimization problems (MaOPs) is typically high and evolutionary algorithms face severe difficulties in solving such problems. In this paper, we propose a new scalable evolutionary algorithm, called evolutionary path control strategy (EPCS), for solving MaOPs. The central component of our algorithm is the use of a reference vector that helps simultaneously minimizing all the objectives of an MaOP. In doing so, EPCS employs a new fitness assignment strategy for survival selection. This strategy consists of two procedures and our algorithm applies them sequentially. It encourages a population of solutions to follow a certain path reaching toward the Pareto optimal front. The essence of our strategy is that it reduces the number of nondominated solutions to increase selection pressure in evolution. Furthermore, unlike previous work, EPCS is able to apply the classical Pareto-dominance relation with the new fitness assignment strategy. Our algorithm has been tested extensively on several scalable test problems, namely five DTLZ problems with 5 to 40 objectives and six WFG problems with 2 to 13 objectives. Furthermore, the algorithm has been tested on six CEC09 problems having 2 or 3 objectives. The experimental results show that EPCS is capable of finding better solutions compared to other existing algorithms for problems with an increasing number of objectives.**

***Index Terms*—Evolutionary algorithm, multiobjective optimization, path control strategy, reference vector.**

I. Introduction

Recently, a growing interest has been seen for solving multiobjective optimization problems (MOPs) with a number of objectives considerably larger than two or three [31], [39]. It is because real-world problems intrinsically have several objectives and from a practical point of view it is often desirable for most applications to include as many objectives as possible. While formulating an optimization problem, designers and decision-makers usually prefer to put every performance index related to the problem as an objective, thereby totaling a large number of objectives. Common appearance of such problems can be found in design optimization [40]. Farina and Amato [20] have suggested to use the term many-objective for such problems, an expression introduced in the operation research community to denote optimization problems with more than two or three objectives. Devising algorithms for solving many-objective optimization problems (MaOPs) are challenging and highly desirable.

A number of evolutionary multiobjective (EMO) algorithms have been proposed after the first pioneering work by Schaffer [43], [44]. Many such algorithms (e.g., nondominated sorting genetic algorithm-II (NSGA-II) [12] and strength Pareto evolutionary algorithm 2 (SPEA2) [57]) use the Pareto dominance relation as a primary selection criterion and a diversity measure based on the Euclidean distance in the objective space as a secondary selection criterion. While these algorithms have been successfully employed in various optimization scenarios with two or three objectives, they face severe difficulties with the increase of objectives [30], [31]. A large number of objectives, in general, increase the dimensionality of the objective space as well as Pareto optimal frontier. This in turn increases the probability of having any two arbitrary solutions to be nondominated to each other. As reason for this behavior, it is sometimes argued that the number of nondominated solutions grows rapidly if further objectives are added to an MOP (empirically studied in [30], [33], [34], and [40], and proven in [52]). This situation spoils proper selection pressure required for evolutionary progress and has been identified a key problem for solving MaOPs [30].

The aforementioned problem can be tackled by the way of inducing a preference ordering over the points in the nondominated solution set or by the reduction of objectives. Some work (see [42]) uses a relaxed form of Pareto-dominance or a modified rank definition [19] to obtain preference ordering. The aim is to make an algorithm more scalable, but relaxation and modification have their limitation, i.e., increases nondominated solutions exponentially with the increase of objectives [44]. An alternative is to use a method that does not rely on the Pareto ranking to sort a population. The simplest non-Pareto based methods use either an aggregation or indicator function. We shall describe algorithms based on these functions in the next section.

A direct way to reduce the number of nondominated solutions is to use few objectives instead of all objectives [16]. When objective reduction is possible, a lower dimensional problem becomes more amenable to EMO algorithms and to decision makers as well. In the process of objective reduction,

Manuscript received September 13, 2013; revised February 11, 2014, March 22, 2014, and May 28, 2014; accepted June 25, 2014. Date of publication July 18, 2014; date of current version March 13, 2015. This paper was recommended by Associate Editor J. Zhang.

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This PDF includes 15 tables, which show detail results of various comparisons. This material is 125 KB in size.

P. C. Roy is with Mississippi State University, Starkville, MS 39762 USA (e-mail: proteek_buet@yahoo.com).

M. M. Islam is with the Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh (e-mail: mdmonirulislam@cse.buet.ac.bd).

K. Murase is with the University of Fukui, Fukui 910-8507, Japan (e-mail: murase@u-fukui.ac.jp).

X. Yao is with the University of Birmingham, Birmingham, West Midlands B15 2TT, U.K. (e-mail: x.yao@cs.bham.ac.uk).

Digital Object Identifier 10.1109/TCYB.2014.2334632

the challenge consists in how to determine a minimum set of objectives preserving important characteristics of the original problem. Alternatively, we can optimize a lower number of objectives separately by partitioning the objective space into a number of subspaces and can apply one or several generations of the evolutionary search in each subspace [3]. Space partition, however, may lead to a potential bias. This is due to the fact that the union of the Pareto sets of all subproblems does typically not form the Pareto set of the full MOP [45].

Although a number of algorithms have been proposed for solving MOPs, the development of EMO algorithms for MaOPs is a relatively young field and not studied thoroughly enough [45]. We here propose an algorithm, called evolutionary path control strategy (EPCS), for efficiently solving MaOPs. Our algorithm uses a new fitness assignment strategy for survival selection. Here the aim is to increase the selection pressure during evolution. The strategy assigns fitness to the individuals of an evolving population based on their distances with respect to a particular path, termed reference vector and given reference points. These distances serve as the preference objectives by which the survival decision for nondominated solutions is made through a direct application of the standard multiobjective domination concept.

The rest of this paper is organized as follows. We first describe some previous work, specifically algorithms related to MaOP, in Section II. Afterwards, we give a detailed description of the proposed algorithm in Section III. The performance of the proposed approach and its comparison with other algorithms are presented in Section IV. We close with a summary and an outlook to implied future works in Section V.

II. Previous Work

The individuals (solutions) of an evolving population represent samples in the decision space. A fitness assignment strategy tries to rank the solutions according to their usefulness. Several fitness assignment strategies have been proposed in the context of EMO algorithms. These strategies can be categorized into: 1) classical method; 2) aggregation method; and 3) indicator method.

A. Classical Method

The classical method assigns M fitness values to a solution based on each of M objectives of a given problem. This method neither modifies the objective functions nor utilizes any additional information for computing the fitness values. Let $\mathbf{x}^i$ is the ith solution vector of the population. In the classical method, the fitness of $\mathbf{x}^i$, $\mathbf{f}(\mathbf{x}^i)$, is defined as a vector of M components i.e., $\mathbf{f}(\mathbf{x}^i) = (f_1(\mathbf{x}^i), f_2(\mathbf{x}^i), \ldots f_M(\mathbf{x}^i))$ with $\mathbf{x} \in S$. Here M is the number of objectives to be optimized and S represents the domain of $\mathbf{x}$. Many previous works (see [1], [12], [15], [31], [57]) used the classical method for fitness assignment.

B. Aggregation Method

As the name suggest, the aggregation method combines all objectives in some way and assigns one fitness value to a solution [37]. It is believed that under mild conditions Pareto optimal solution to an MOP could be an optimal solution of a scalar optimization problem [54]. Given an M-objective optimization problem of minimizing $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots f_M(\mathbf{x})$. Aggregation based EMO algorithms solve a single-objective problem instead of an M-objective optimization problem. That is

$$\text{Minimize} \sum_{j=1}^{M} w_j \times f_j(\mathbf{x}) \tag{1}$$

where w_j is the jth component of a chosen weight vector used for scalarizing the objectives. It is now clear that the aggregation method is easy to describe and, on a first sight, scale well because all objectives are somehow combined. The main advantage of such combination is that it eliminates the problem of exponential increase of nondominated solutions with the increase of objectives. However, for high dimensions, the method reaches its limit. This is because it is hard (or even impossible) to determine a good set of weights that grow with the number of objectives to be optimized.

A good number of EMO algorithms (see [27]–[29], [54]) adopts the above idea for fitness assignment to some extent. In [29], one scalar function but with a different weight vector is used for evaluating every solution of the population. In contrast, multiple single objective Pareto sampling (MSOPS) [27] and its variant MSOPS-II [28] utilize several scalar aggregation functions. Multiobjective evolutionary algorithm based on decomposition (MOEA/D) decomposes an MOP into a number of scalar optimization subproblems and optimizes them simultaneously in a single run [54]. Recently, the performance of this algorithm has been enhanced by generating the reference points via systematic sampling and by employing an adaptive epsilon scheme for balancing between convergence and diversity [4].

By aggregating objectives, several EMO algorithms (see [17], [23], [47]) incorporate the decision maker's preference into optimization. The decision maker, however, can give preferences before searching or during searching in an interactive manner. Ideally, by providing such clues, the decision-maker is interested to choose a region of her/his interest. When reference points $\mathbf{z}$ along with the relative preferences (weights) are given, the goal is to find solutions close to those points. One common form of this approach is

$$\text{Minimize} \sum_{j=1}^{M} w_j \times f_j(\mathbf{x} - \mathbf{z}). \tag{2}$$

The main problem of objective aggregation is that not all Pareto optimal solutions can be investigated when the true Pareto front is nonconvex. Therefore, EMO algorithms utilizing objective aggregation have difficulty in finding solutions uniformly distributed over a nonconvex trade off surface [11].

Instead of using a single set of preferences, it is possible to use multiple sets of preferences. This leads to the concept of co-evolving a family of decision-maker preferences together with a population of candidate solutions, called preference-inspired co-evolutionary algorithm (PICEA) [49]. An advantage of this approach is that different sets of preferences may lead to different subsets of the Pareto front. As a result, no separate mechanism is needed for producing diverse solutions to cover the entire Pareto front. PICEA considers preferences as family of goals, which are randomly generated in the objective space within the bounds defined by the vectors of ideal and anti-ideal performances. These bounds can be easily estimated using the nondominated solutions found so far.

C. Indicator Method

A key question when tackling any optimization problem is defining its goal used for evaluating the quality of solutions. Among the various quality measures hypervolume [58] or S-metric [56] is the only quality measure known to be Pareto-compliant. Knowles and Corne [33] were the first to propose the integration of the hypervolume measure into an optimization process. In [21], Fleischer suggested recasting the MOP to a single objective one by maximizing *S*-metric. Huband *et al*. [25] presented an EMO algorithm using a modified SPEA2, where the hypervolume related measure replaces the original density estimation technique. In [6], the *S*-metric was utilized for selection with the hope of finding a set of solutions well distributed on the Pareto front. Indicator-based evolutionary algorithm (IBEA) used the binary hypervolume to compare solutions and to assign corresponding fitness values [59].

The main drawback of hypervolume based algorithm is their extreme computational overhead. To overcome this problem, Bader and Zitzler [5] proposed a hypervolume estimation algorithm for multiobjective optimization (HypE). Recently, While *et al*. [50] described a faster technique for calculating the exact hypervolume. However, this measure is sensitive to the presence or absence of extreme points in the obtained solution set. The runtime complexity of the exact hypervolume calculation is still exponential to this date.

III. EPCS

Analyzing the pros and cons of classical, aggregation, and indicator-based EMO approaches discussed in the previous section, we propose a new algorithm, EPCS, suitable for solving MaOPs. The algorithm has the following new features.

1) A reference vector is introduced to assist an evolutionary process by reducing the number of nondominated solutions.
2) A new fitness assignment procedure is proposed for evaluating the individuals of a population. The procedure uses the classical method, the aggregation method or a combination of these two methods. However, unlike the aggregation method, our assignment procedure does not create any problem for applying the Pareto-dominance relation in survival selection.
3) Our algorithm simultaneously minimizes all objectives of an MaOP, but gives proper importance to different objectives. To achieve this, a reference vector is defined at each generation and evolution is encouraged to follow it.
4) A fast and scalable algorithm for solving MaOPs whose computational complexity is the same as the state-of-the-art algorithms e.g., NSGA-II [12].

Like other EMO algorithms, our algorithm starts with a population consisting of N individuals generated uniformly at random and applies crossover and mutation on the individuals to produce N offspring. It then defines a reference vector using the reference points and the maximum objective values of the parents and offspring. Finally, the algorithm uses the reference vector for assigning fitness values to all the $2N$ individuals. It is now time to describe the algorithm in detail. We first present the algorithm in the pseudo code form and then describe its different components elaborately.

Algorithm 1: *Main Loop*

Input: Let M is the number of objectives, N is the population size, g is total number of generations, t is the number of generations used for applying the Fitness Assignment Procedure 1, k_1 and k_2 are path control parameters, c_1 is a global diversification parameter and c_2 is a local diversification parameter.

Output: A set of nondominated solutions.

```
1.  $P_{\text{old}} \leftarrow$Randomly initialize all the $N$ individuals of the population
2.  for $i \leftarrow 1$ to $g$
3.      do
4.          Calculate fitness values for all individuals of $P_{\text{old}}$
5.          $P_{\text{mate}} \leftarrow$ Construct $N/2$ mating pairs from $P_{\text{old}}$
6.          $P_{\text{new}} \leftarrow$ Apply crossover on the pairs of $P_{\text{mate}}$ and create $N$ new offspring
7.          $P_{\text{new}} \leftarrow$ Apply mutation on the individuals of $P_{\text{new}}$
8.          $P \leftarrow P_{\text{old}} \cup P_{\text{new}}$
9.          Find $f_j^{\max}$ for all $j = 1$ to $M$, where $f_j^{\max}$ is the maximum value of the $j$-th objective in $P_{\text{old}}$
10.         Define a reference vector going through the origin (reference points) and the points $\mathbf{r}(f_1^{\max}, f_2^{\max}, \ldots, f_M^{\max})$
11.         $d_{\text{avg}} \leftarrow$ Find the average of the Euclidean distances between $\mathbf{r}$ and all individuals of $P_{\text{old}}$
12.         if $(i < t)$
13.            then $d_1 \leftarrow k_1 \times d_{\text{avg}}$
14.                 $d_2 \leftarrow k_2 \times d_{\text{avg}}$
15.                 $P_{\text{temp}} \leftarrow$ Fitness Assignment Procedure 1 $(P, \mathbf{r}, d_1, d_2)$
16.            else $d_v \leftarrow (1 + c_1) \times d_{\text{avg}}$
17.                 $P_{\text{temp}} \leftarrow$ Fitness Assignment Procedure 2 $(P, \mathbf{r}, d_v)$
18.         $P_{\text{old}} \leftarrow$ Select the $N$ best individuals from $P_{\text{temp}}$ using the Pareto-dominance relation and crowding distance as the primary and secondary selection criteria, respectively
19. return A set of nondominated solutions from $P_{\text{old}}$
```

```
Algorithm 2: Fitness Assignment Procedure 1 $(P, \mathbf{r}, d_1, d_2)$
Input: The combined Population $P$ of size $2N$, the reference
       vector $\mathbf{r}$ and two distances $d_1$ and $d_2$.
Output: $P$
1.  for $i = 1$ to $2N$
2.      do $d_o(\mathbf{x}^i) \leftarrow$ Find the Euclidean distance from the
           origin to $\mathbf{x}^i$, where $\mathbf{x}^i$ is the $i$-th individual of $P$
3.         $d_r(\mathbf{x}^i) \leftarrow$ Find the Euclidean distance from $\mathbf{x}^i$ to $\mathbf{r}$
4.         if $(d_r(\mathbf{x}^i) < d_1)$
5.            then $h_1(\mathbf{x}^i) \leftarrow d_r(\mathbf{x}^i) + d_o(\mathbf{x}^i)$
6.                 Assign $h_1(\mathbf{x}^i)$ to each objective of $\mathbf{x}^i$
7.            else if $(d_r(\mathbf{x}^i) < d_2)$
8.                    then keep the original value for each
                           objective of $\mathbf{x}^i$
9.                    else $h_2(\mathbf{x}^i) \leftarrow (d_2 - d_r(\mathbf{x}^i)) + d_o(\mathbf{x}^i)$
10.                        Assign $h_2(\mathbf{x}^i)$ to each objective
                           of $\mathbf{x}^i$
11. return $P$
```

```
Algorithm 3: Fitness Assignment Procedure 2 $(P, \mathbf{r}, d_v)$
Input: The combined Population $P$ of size $2N$, the reference
       vector $\mathbf{r}$ and the Euclidean distance $d_v$.
Output: $P$
1.  for $i = 1$ to $2N$
2.      do $d_o(\mathbf{x}^i) \leftarrow$ Find the Euclidean distance from the
           origin to $\mathbf{x}^i$
3.         if $(d_o(\mathbf{x}^i) < d_v)$
4.            then $n_c \leftarrow 0$, where $n_c$ is the number of objec-
                   tives which have greater values than the
                   bounded values of the objectives defined
                   in the step 6
5.                 for $j = 1$ to $M$
6.                     if $f_j(\mathbf{x}^i) > (1 + c_2) \times f_j^{\max}$
7.                        then $n_c \leftarrow n_c + 1$
8.                 if $n_c > 0$
9.                    then for $j = 1$ to $M$
10.                           $f_j(\mathbf{x}^i) \leftarrow n_c \times f_j(\mathbf{x}^i)$
11.                   else keep the original value for each
                           objective of $\mathbf{x}^i$
12.           else Assign $d_o(\mathbf{x}^i)$ to each objective of $\mathbf{x}^i$
13. return $P$
```

A. Reference Vector

A decision maker usually chooses only one solution from a set of obtained solutions. The considerations for choosing the solution may not be known *a priori*. Not only that, the decision maker may change her/his choice after analyzing the obtained solutions. It is, therefore, reasonable not to assign any preference level (usually given by the weight) to an objective while solving an MOP. This suggests that an EMO algorithm has to simultaneously optimize all the objectives of MaOPs according to their given impotences. Considering this fact, our algorithm does not multiply each objective or its distance from the reference point by a weight. It rather defines a reference vector using the reference points at every generation. This is advantageous in the sense that if the decision maker wants, then she/he can refine her/his previous choice(s) after looking at the solutions. This may be an interactive version of our proposed algorithm.

As the direction of any evolutionary process is dependent on the basis on which selection takes place, EPCS encourages evolution toward the region we define best. Such a region for a minimization problem, for example, may be the region surrounded by (or closest to) the origin. The origin in this case can be thought as reference points i.e., ideal points. Note that EPCS can use any other points as reference points, because it does not put any constraint in selecting such points. It was shown that minimal solutions of reference points based methods were weakly Pareto optimal independently of how the points were chosen (see [51]). Furthermore, if the solutions are unique, they are Pareto optimal [45]. A benefit of using reference points is that any Pareto optimal solution can be found by altering these points only [47], [51]. Some previous studies (see [17], [23], [47]) showed benefit of using reference points in solving different MOPs.

Unlike most previous work, EPCS uses the reference points in defining a reference vector that goes through the multidimensional objective space and directed toward the reference points. If evolution follows this line for reaching toward the global optima, it is natural that all the objectives will be minimized simultaneously. That is why EPCS utilizes the distance from the reference vector for assigning the fitness values to the individuals of the population. Another essence of the reference vector is that our algorithm uses it for reducing nondominated solutions. We discuss this point at length in the next section.

Without loss of generality, we assume that the objectives values of all individuals are nonnegative real numbers. Location of absolute minima in this case is the origin i.e., $O(0_1, 0_2, \ldots, 0_M)$ for both the feasible and nonfeasible regions of the objective space. To define the reference vector at each generation, our algorithm first finds the maximum value of each component of the objective vector of that generation. Let $f_j^{\max}$ is the maximum value of the jth component of the objective vector and $R^{\max}(f_1^{\max}, f_2^{\max}, \ldots, f_M^{\max})$ is a point in the objective space. We then define the reference vector that goes through the points O and $R^{\max}$. The maximum value roughly indicates how far one solution is from its optimal value along that dimension at any particular generation. Hence, if we consider the reference vector as an evolutionary path, it approximately gives necessary importance to the objective while the optimization process goes on. The reference vector drawn for two objectives with different importance is shown in Fig. 1. As EPCS computes $f_j^{\max}$ at every generation and uses it for defining the reference vector, this vector may change as evolution goes on.

It is now clear that the reference vector can be an evolutionary path. As mentioned earlier, when the number of objectives increases, most of the solutions generated by an EMO algorithm becomes nondominated by the Pareto-dominance relation. This, in turn, slows down convergence in reaching toward the Pareto optimal front. Our algorithm comes up with a new way to tackle this issue. Roughly speaking, EPCS selects those solutions that are close to the origin and close to the reference vector as well. It assigns fitness values in such a way that the solutions do not get distract increasingly from the

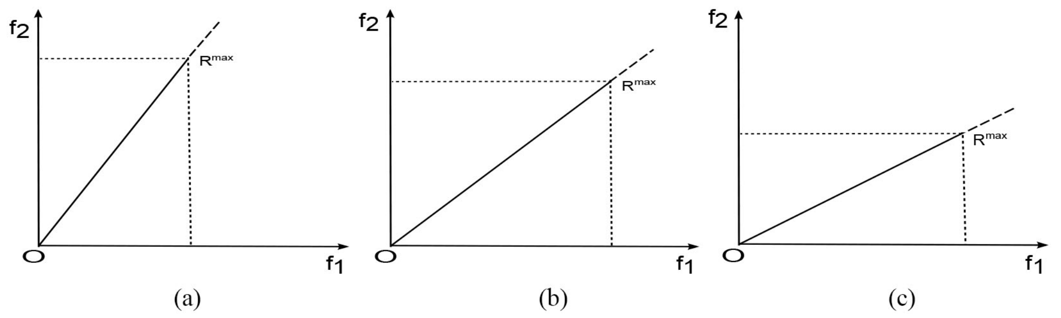


Fig. 1. Relation between the reference vector ($OR^{\max}$) and the importance of an objective. Here f_1 and f_2 are two objectives that are going to be optimized. $R^{\max}$ is a point in the 2-D objective space whose coordinates have the maximum objective values at any particular generation. (a) f_2 is given more importance than f_1. (b) f_1 and f_2 have equal importance. (c) f_1 is given more importance than f_2.

reference vector, rather come closer as evolution progresses. After defining the reference vector, our algorithm applies the genetic operators for creating offspring.

B. Mating Selection, Crossover, and Mutation

EPCS uses the crowded tournament selection for matting and simulated binary crossover (SBX) and polynomial mutation for perturbing the current solutions [11]. Like some other previous studies (see [1] and [15]), the proposed algorithm adopts nondominated sorting by the Pareto-dominance relation as the primary selection criterion and the crowding distance measure as the secondary selection criterion. This is, however, not an inherent constraint for our algorithm; in fact, any suitable operator can be used. It is because EPCS is a general evolutionary framework for MaOPs. The use of popular operators gives us an opportunity to make a fair comparison with other work.

C. Fitness Assignment Strategy

The proposed EPCS uses a fitness assignment strategy consisting of two procedures: fitness assignment Procedure 1 (Algorithm 2) and fitness assignment Procedure 2 (Algorithm 3). The Procedure 1 is mainly used to encourage the population reaching near to the Pareto optimal front, while the Procedure 2 is used to push the population more toward the optimal front as well as to diversify the solutions in the entire front.

In its current implementation, EPCS applies the Procedure 1 for the first t generations and the Procedure 2 for the remaining $(g - t)$ generations of an evolutionary process. Here g is the total number of generations and t is a user specified parameter. The value of t can be roughly, for example, from 25% to 75% of g. This value may vary due to the creation of the initial population and/or the characteristics of a problem. For example, if random initial solutions are far from the Pareto optimal front, then an algorithm needs many generations to drive the population closer to the optimal front. The value of t in such a case would be large; otherwise, it would be small.

1) Fitness Assignment Procedure 1: The aim of this procedure is assigning fitness to the individuals in such a way so that it is possible to reduce nondominated solutionis, which in turn will increase selection pressure in evolution. To achieves this, the Procedure 1 divides the objective space into three regions based on the Euclidean distance from the reference vector (Fig. 2). We termed these regions as Region 1, Region 2, and Region 3. The Procedure 1 assigns modified fitness to the individuals resided in the Region 1 and Region 3, while it uses the original objectives value in assigning fitness to the individuals resided in the Region 2. The modified fitness is assigned in such a way so that it does not create any problem for applying the Pareto dominance relation for comparing obtained solutions.

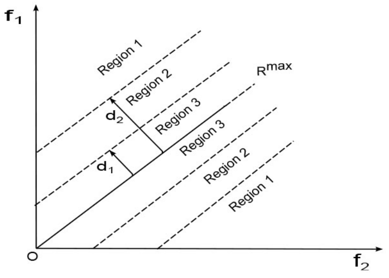


Fig. 2. Division of the objective space with respect to the reference vector $OR^{\max}$.

Two distances, d_1 and d_2, are chosen adaptively for dividing the objective space where $d_1 < d_2$ (Fig. 2). As generation progresses, these distances may change according to d_{avg}, the average Euclidean distance of the population from the reference vector. It is because EPCS computes d_{avg} in every generation (Step 11 of Algorithm 1). As $d_1 < d_2$, the user defined parameter k_1 should be set less than the user defined parameter k_2 (Steps 13 and 14 of Algorithm 1) . The individuals of the population may reside in different regions based on d_1 and d_2. Let the distance from the ith individual $\mathbf{x}^i$ to the reference vector $\mathbf{r}$ is $d_r(\mathbf{x}^i)$. If this distance is greater than d_2, the individual is assumed to be in the Region 1. On the other hand, if $d_r(\mathbf{x}^i)$ is equal to or smaller than d_2 but greater than d_1, the individual resides in the Region 2. Otherwise, the individual is in the Region 3. We use the classical method (Section II-A) for assigning fitness to the individuals resided in the Region 2. However, we use the weighted sum approach for assigning fitness to the individuals in the Regions 1 and 3 (Steps 5 and 9 of Algorithm 2).

Suppose $Q(f_1, f_2, \ldots, f_M)$, a point in the M-dimensional objective space, represents an individual $\mathbf{x}^i$, $\mathbf{q}$ is a position vector of Q and $\mathbf{r}$ is the position vector of $R^{\max}$. The unit vector into the direction of $\mathbf{r}$ is $\mathbf{r}/\,||\mathbf{r}||$. The projection of $\mathbf{q}$ into

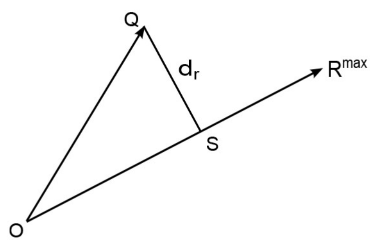


Fig. 3. Measurement of distance d_r from any point Q to the reference vector $OR^{\max}$ in the objective space.

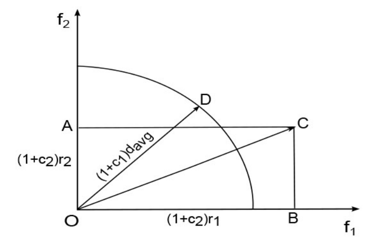


Fig. 4. Division of the objective space with respect to the hypersphere with radius d_v.

the direction of that unit vector is $\mathbf{q} \cdot \mathbf{r}/\,||\mathbf{r}||$ (Fig. 3). Thus, the shortest distance from $\mathbf{x}^i$ to $\mathbf{r}$ is

$$d_r(\mathbf{x}^i) = \sqrt{\mathbf{q} \cdot \mathbf{q} - \frac{\mathbf{q}.\mathbf{r}}{||\mathbf{r}||} \cdot \frac{\mathbf{q}.\mathbf{r}}{||\mathbf{r}||}}. \tag{3}$$

Similarly, the Euclidean distance from the origin to $\mathbf{x}^i$ is given by

$$d_o(\mathbf{x}^i) = \sqrt{f_1^2 + f_2^2 + \ldots + f_M^2}. \tag{4}$$

As mentioned earlier, EPCS tries to reach the Pareto optimal front by following the reference vector. To incorporate this goal, it uses $d_r(\mathbf{x}^i)$ and $d_o(\mathbf{x}^i)$ in h_1^i and h_2^i, which our algorithm employs for assigning fitness to $\mathbf{x}^i$ (Steps 5 and 9 of Algorithm 2). Although $d_r(\mathbf{x}^i)$ and $d_o(\mathbf{x}^i)$ are used in [4] for assigning fitness to individuals, there are three major differences. Firstly, we do not use $d_r(\mathbf{x}^i)$ and $d_o(\mathbf{x}^i)$ for assigning fitness to all individuals of the population, as our algorithm uses the original objectives value for assigning fitness when $d_r(\mathbf{x}^i) < d_2$ (Step 8 of Algorithm 2). Secondly, EPCS does not assign only one value as fitness; rather it assigns fitness as a vector of M values, where M is the number of objectives to be optimized. Our algorithm assigns the same value i.e., h_1 or h_2 to each of the M values when $d_r(\mathbf{x}^i) < d_1$ or $d_r(\mathbf{x}^i) > d_2$ (Steps 6 and 9 of Algorithm 2). This makes possible to apply the Pareto-dominance comparison to all individuals of the population. Thirdly, we do not multiply each objective by a weight. In contrast, existing EMO algorithms based on the weighted sum approach require M weights for an MOP with M objectives. The number of weights for these algorithms, therefore, increases with the increase of objectives.

One may wonder why we divide the objective space into three regions, although it is possible to divide the space more or less number of regions. Some problems may occur naturally if we do either. If we divide the space into two, then any one of the two regions would be the region of preference as our strategy suggests. Assume the region further from the reference vector is the preferred region where the original objective value is kept. An evolving population in this condition will tend to be diversified and the convergence rate will slow down. On the other hand, if the preferred region is the region closer to the reference vector, then the whole population would tend to come close to this region. It is often be the case that the entire population may gather very near to the reference vector before reaching to the Pareto optimal front. Thus, the diversity of the population will be lost, resulting slow convergence. On the other hand, if we divide the space more than three regions, then it will be difficult to decide the preference among the regions. Convergence rate will also slow down due to more than one preferred regions for which there will be more than one direction for converging the population. All these points signify the reasons for using three well-defined regions.

2) Fitness Assignment Procedure 2: The aim of the Procedure 2 is same as the Procedure 1 i.e., assigning fitness to the individuals in such a way so that it is possible to reduce nondominated solutionis. Like the Procedure 1, it also assigns fitness to all individuals of a population as a vector of M values. However, unlike the former one (Fig. 2), the Procedure 2 divides the objective space into two regions and uses the Euclidian distances from the reference points not the reference vector for such a division (Fig. 4). Furthermore, the division is done here perpendicular to the one done by the Procedure 1. This kind of division facilitates in distributing solutions over the whole Pareto front.

Our Procedure 2 first calculates d_{avg}, the average Euclidian distance between all the individuals in the population and the reference points. It then defines a bounded region by an M-dimensional hyper-sphere, whose radius $d_v = (1 + c_1) \times d_{\text{avg}}$. Here c_1 is a global diversification parameter and defines the rate by which the global boundary should vary in the subsequent generations. However, putting a boundary does not introduce any constraint, as d_v varies with the change of d_{avg} at every generation (Steps 10, 11, and 16 of Algorithm 1). The Procedure 2 divides the objective space into two regions: one is inside d_v and another is outside d_v (Fig. 4).

Based on their distances from the origin, some individuals may resided inside d_v and some may outside. If the Euclidian distance, $d_o(\mathbf{x}^i)$, between the individual $\mathbf{x}^i$ and the reference points is greater than or equal to d_v, then EPCS assigns $d_o(\mathbf{x}^i)$ to each objective of $\mathbf{x}^i$ (Step 12 of Algorithm 2). This indicates that all the individuals beyond d_v will be compared only using their Euclidean distance, irrespective of the values of different objectives. The benefit is the reduction of nondominated solutions and the preservation of the best solutions that are nearer to the reference points. Due to the stochastic nature of evolution, an evolutionary process will discover newer search points in the bounded region, pushing an evolving population more toward the Parteo-optimal front and diversifying the population as well.

If $d_o(\mathbf{x}^i)$ is smaller than d_v, then EPCS counts the number of objectives (n_c), which have a value greater than a bounded value of those objectives. The bounded value is computed as $(1+c_2) \times f_j^{\max}$. Here c_2 is a local diversification parameter and defines a rate by which each objective can vary in the subsequent generations. When n_c of $\mathbf{x}^i$ is greater than zero, all of its objective is multiplied by n_c. Otherwise, EPCS keeps the original values of all the objectives. The diversification boundaries for a 2-D case are shown in Fig. 4 The line OD defines the global boundary, which equals to d_v. As Fig. 4 is drawn for a 2-D case, $\mathbf{r}$ has two components: r_1 and r_2. Consequently, there are two local boundaries. One is $OA = (1 + c_2) \times r_1$ and the other is $OB = (1 + c_2) \times r_2$. The shape of the Pareto optimal front may be convex, concave, flat, discrete or mixed type. It is, however, hard to know the shape in advance. Our algorithm takes the advantage of randomization and stochastic nature of evolutionary algorithms. By controlling c_1 and c_2, we can control the diversity of an evolving population according to the characteristics of a problem or the interests of a user.

As generation progresses, solutions inside the hyper-sphere may become nondominated. It is, however, not a major problem for EPCS, as after the first t generations the population is expected to have reached close to the Pareto-optimal front. Now, only few generations are necessary to drive the population more close to the front. For solutions inside or outside d_v, EPCS also uses here the Pareto-dominance relation as the primary selection criterion and the crowding distance as the secondary selection criterion. However, solutions will mostly be selected from the less crowded region because most of the solutions are expected to be nondominated with the progress of generations. At the end of evolution, we will thus get a set of diverse solutions

3) Algorithmic Complexity: As mentioned before, the proposed EPCS uses tournament selection, simulated binary crossover, polynomial mutation, and nondominated sorting with the crowding distance measure for survival selection. These components are used in the most popular NSGA-II algorithm, whose complexity is $O(MN^2)$. It is evident from Section III that in addition to the aforementioned components, EPCS has four more major components. The complexity of the additional components are as follows.

1) Finding the reference vector requires $O(MN)$ operations.
2) Application of the fitness assignment Procedure 1 requires $O(MN)$ operations.
3) Calculation of the distance from the reference vector to an individual takes $O(M)$ operations.
4) Application of the fitness assignment Procedure 2 incurs $O(MN)$ operations.

It is now clear that none of the four additional components used by EPCS requires more than $O(MN^2)$ operations. Hence, the overall complexity of EPCS is $O(MN^2)$.

IV. Experiment Studies

In order to know how the efficacy of an algorithm changes with the increase of objectives, it is better to use scalable test functions for simulation. The well known and most widely used scalable test suit is the DTLZ family [13]. There are nine functions in this family. We used five of them (DTLZ1−DTLZ4 and DTLZ7) in our simulation. We have also used six problems from each of the recently introduced WFG [26] and CEC09 [55] test suites. All the problems used in our simulation did not have any constraint.

A. Experiment Setup

The performance of EPCS on the DTLZ test suit is going to be compared with six other algorithms: DM1 [1], HYPE [5], MSOPS [27], GDE3 [35], MOEA/D [54], and IBEA [59]. The algorithms MSOPS and MOEA/D are aggregation based algorithms, while HYPE and IBEA are indicator based algorithms. DM1 is a recently introduced algorithm that incorporates a mechanism for promoting diversity in evolutionary many-objective optimization. In [35], differential evolution has been generalized for global optimization with an arbitrary number of objectives and constraints. EPCS has been implemented by the authors and its source code is available from Roy (pcr56@msstate.edu). We implemented DM1 (NSGA-II with standard SBX parameters and only DM1 mechanism [1]). The source codes of HYPE, MSOPS, and IBEA are obtained from PISA,[1] while those of MOEA/D and GD3 are obtained from Zhang.[2] All algorithms have been implemented on a PC with a Intel(R) Core(TM) i7-2670QM CPU running at 2.2 GHz with 8GB RAM.

For all the competing algorithms, we used the same settings for the common parameters, like N and g. The population size, N, was set to 100 for all problems irrespective of the number of objectives. This was done following the previous work (see [1] and [60]). However, this is not an inherent constraint in our EPCS; the population size could be adapted like in [24] and [32]. Total number of generations, g, was set to 1000 for DTLZ3, while it was set to 300 for other DTLZ problems. The remaining parameters of other algorithms were chosen according to the obtained source codes.

The crossover probability and its distribution index for EPCS were set to 1 and 15, respectively. We used the mutation probability, $1/n$ and the distribution index, 20. Here n denotes the number of variables in a problem. Our EPCS has five parameters: t, k_1, k_2, c_1, and c_2. For all the DTLZ test instances, the value of t was set to 150, k_1 to $0.1/M$ and k_2 to $1/M$. The values of c_1 and c_2 for DTLZ1 were set to 0.02 and 0.01, respectively, while these for DTLZ2 were set to 0.001 and 0.0005, respectively. For DTLZ3, DTLZ4, and DTLZ7, we set the value of c_1 and c_2 to 0.002 and 0.001, respectively. These values were not meant to be optimal, which were obtained after some preliminary runs.

The performance of an EMO algorithm is usually evaluated from two different aspects: convergence and diversity. We used generational distance (GD) to measure convergence, while inverse generational distance (IGD) and hypervolume (HV) for measuring both convergence and diversity [9]. These metrics have been widely used in the EMO literature (see [1], [24], [36], [48]). In addition to observing convergence and diversity, we also used another performance indicator, called spacing (SP) [9], for measuring uniformity of the obtained solutions. The analytical form of the Pareto optimal fronts of DTLZ problems is known. We used it for

[1]http://www.tik.ee.ethz.ch/pisa/

[2]http://cswww.essex.ac.uk/staff/zhang/MOEAcompetition/CEC09final/code/

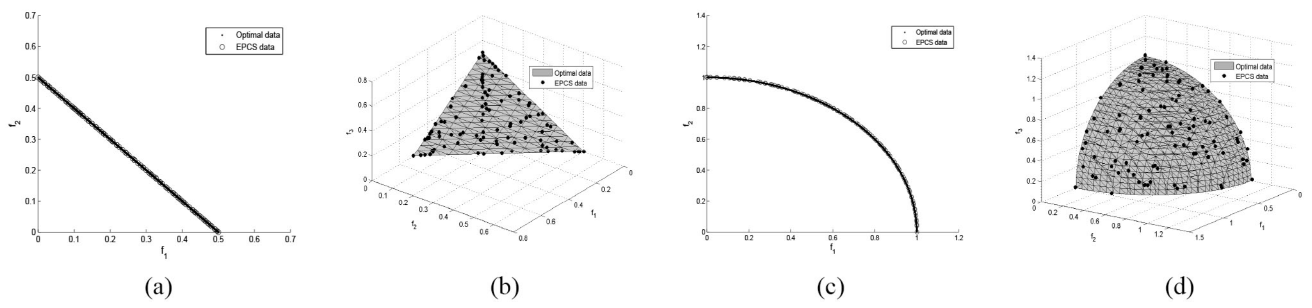


Fig. 5. Nondominated solutions obtained by EPCS for DTLZ1 and DTLZ3 with two and three objectives. (a) DTLZ1 with two objectives. (b) DTLZ1 with three objectives. (c) DTLZ3 with two objectives. (d) DTLZ3 with three objectives.

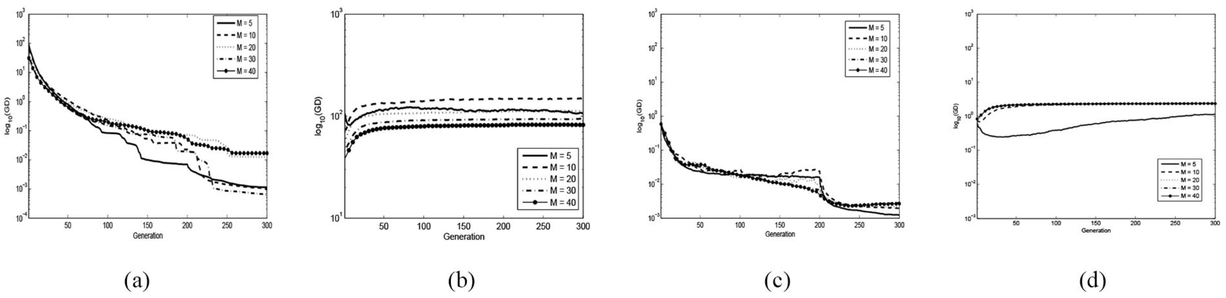


Fig. 6. Convergence process of DTLZ1 and DTLZ4 with *M* = 5, 10, 20, 30 and 40. Here *M* denotes the number of objectives. (a) DTLZ1 with the new fitness assignment strategy. (b) DTLZ1 with the classical fitness assignment strategy. (c) DTLZ4 with the new fitness assignment strategy. (d) DTLZ4 with the classical fitness assignment strategy.

computing GD of these problems. Although a small GD value indicates good convergence ability of an algorithm, it does not guarantee a good distribution of obtained solutions. A reference set is needed for computing IGD. Using the analytical form of the optimal front, we generated 100 000 optimal points uniformly at random and used them as the set of reference points.

For computing HV, a reference point is needed. The anti-optimal point or worst possible point is usually chosen as the reference point [50]. If such a point is not known or does not exist, one suggestion is to take, in each objective, the worst value from any of the sets being compared [50]. In this paper, the reference point was chosen by adding a small constant, δ, to the worst point. Without the small increment, the extremal points might register no volume and not contribute to HV [50]. We set δ equal to 1.5 and 3 for DTLZ1-4 and DTLZ7 problems, respectively. We used 100 000 points for computing HV. We normalized HV by the Pareto optimal points. That is, if HV is 1 then the obtained solutions are on the Pareto optimal front. A similar technique is used for computing HV in previous studies (see [53]). To compute SP, we do not require any reference point or set. The solutions in the approximation set are used for computing this metric [9].

B. Experiment Results

Fig. 5 shows 2-D and 3-D plots using the nondominated solutions obtained by EPCS in a single run. We chose the DTLZ2 and DTLZ3 problems with two and three objectives for such plots. It is evident from this figure that the solutions are very close to the Pareto optimal front and they are distributed in the whole region of the front. Before comparing the results of our EPCS with other algorithms, we first present experimental evidences for different features of our algorithm.

1) Evolutionary Progress: Evolutionary progress with respect to generation is presented in Fig. 6. We chose only DTLZ1 and DTLZ4 for brevity. The following observations can be made from this figure.

1) Convergence rate of EPCS is almost similar for a particular problem irrespective of the number of objectives. Specifically, it does not slow down much due to increment of objectives. A very different scenario is observed when we use the classical fitness assignment strategy in which no convergence actually takes place.
2) Sudden drop of the GD value is observed in some cases, for example, after $t = 150$ generations. This is due to the fact that EPCS switches from the fitness assignment Procedure 1 to 2 after the said generations. The Procedure 1 tries to hold a limited but sufficient diversity during evolution, while the Procedure 2 increases diversity. Thus, the solutions in the population get diversified, helping to reduce GD by a good amount within the first several generations of the Procedure 2.
3) At the end of an evolutionary process, the solution quality of the obtained set is very much similar for the same problem with a different number of objectives, although the ratio of the highest to the lowest number of objectives is 8:1. This signifies the scalability of our EPCS.

2) Dealing With Nondominated Solutions: Our new fitness assignment strategy reduces the number of nondominated

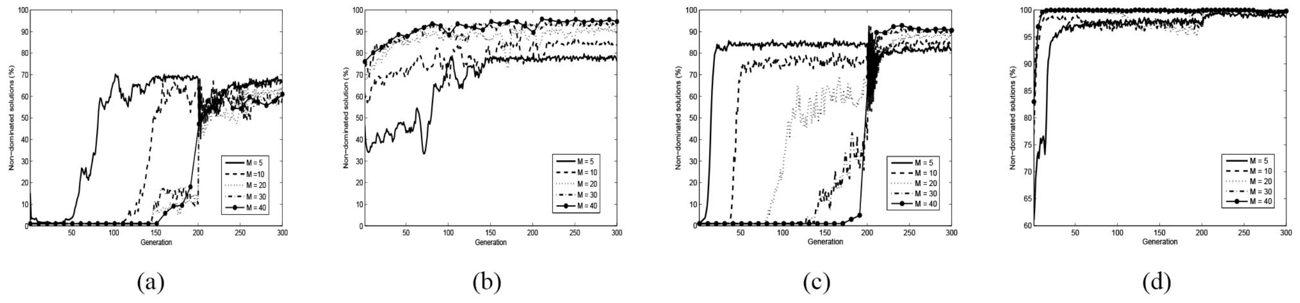


Fig. 7. Nondominated solutions produced by EPCS with modified fitness values and original fitness values for DTLZ1 and DTLZ2. (a) DTLZ1 (with modified fitness values). (b) DTLZ1 (with original fitness values). (c) DTLZ2 (with modified fitness values). (d) DTLZ2 (with original fitness values).

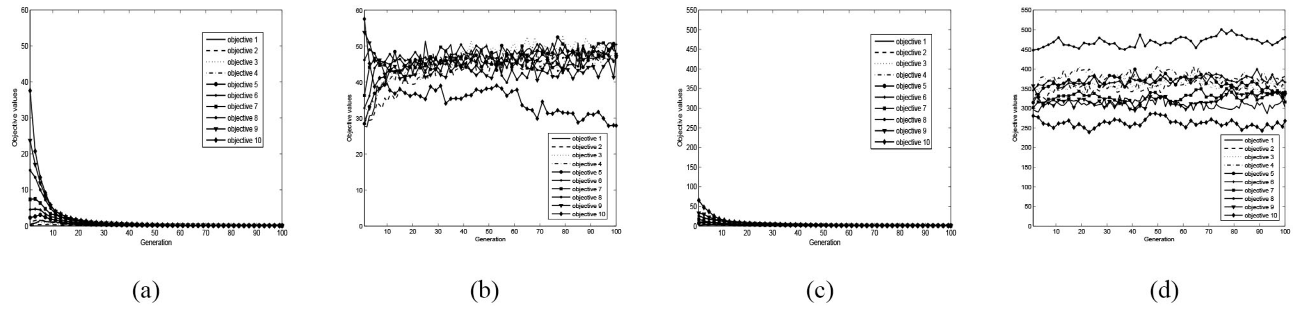


Fig. 8. Simultaneous minimization of all objectives for DTLZ1 and DTLZ3 with 10 objectives. (a) DTLZ1 with the new fitness assignment strategy. (b) DTLZ1 with the classical fitness assignment strategy. (c) DTLZ3 with the new fitness assignment strategy. (d) DTLZ3 with the classical fitness assignment strategy.

solutions for solving MaOPs with an aim to increase selection pressure. We show this for DTLZ1 and DTLZ2 problems. The following characteristics tend to emerge if we scrutinize these figures.

1) The new fitness assignment strategy with modified fitness values reduces nondominated solutions compared to those obtained for the classical fitness assignment strategy with the original fitness values. This reduction is more prominent at the beginning of the evolutionary process, especially for the first t generations of evolution when EPCS uses the fitness assignment Procedure 1.
2) Increasing objectives does not increase the number of nondominated solutions much when we compare solutions based on the modified fitness values. This is, however, not the case for the original fitness values.

3) Simultaneous Minimization of All Objectives: An important feature of EPCS is that our algorithm minimizes all the objectives properly in order to obtain a set of good quality solutions. For the sake of clarity, we chose here DTLZ1 and DTLZ3 with only 10 objectives to visualize how the value of each object minimized as evolution progresses. It can be seen from Fig. 8 that, for DTLZ1 and DTLZ3, the average initial values of different objectives are different. The proposed algorithm minimizes the high-valued objectives more quickly than the low-valued ones. And the final average values of each objectives come closer to one another. This scenario is quite different when we apply the classical fitness assignment strategy, which fails to minimizes the objectives.

4) Comparison: This section presents comparison of EPCS with DM1 [1], HYPE [5], MSOPS [27], IBEA [59], MOEA/D [54], and GDE3 [35]. We considered five different number of objectives for each of the five different DTLZ problems. This makes the total number of problem instances to 25. All the algorithms were independently run 10 times for each problem instance.

Tables I and II summarizes the results of all seven algorithms for DTLZ1 and DTLZ3 problems. However, the detail results for DTLZ2, DTLZ4, and DTLZ7 problems are given in the supplementary materials. In terms of average GD, EPCS outperformed the competing algorithms on the majority of the DTLZ problem instances. Except DTLZ7, EPCS reached very close to the Pareto optimal front in most of the cases, even problems with a large number of objectives. A very recent study has revealed that EMO algorithms using the crowding distance as a selection criterion are not suitable for problems with a disconnected optimal front, like DTLZ7 [7]. Note that our EPCS also outperformed the other algorithms on DTLZ7.

As mentioned earlier, IGD and HV measure both convergence and diversity of the obtained solutions. EPCS achieved the best IGD and HV values for the same number of 12 instances. In terms of IGD, MOEA/D is a close competitor of EPCS, which achieved the best IGD value for the eight instances. This algorithm, however, achieved the best HV value for only four instances. From the results of GD, IGD, and HV, we can conclude that EPCS does not compromise with diversity in reaching very close to the Pareto optimal front.

A very different scenario is observed when we compare EPCS with respect to SP. Although IBEA was worse compared to EPCS in terms of GD, IGD, and HV, it outperformed EPCS in terms of SP. A good SP, however, does not indicate that

TABLE I
COMPARISON BETWEEN EPCS, DM1 [1], HYPE [5], MSOPS [27], GDE3 [35], MOEA/D [54], AND IBEA [59] FOR DTLZ1. EACH RESULT REPRESENTS AVERAGE OF TEN INDEPENDENT RUNS. THE BEST RESULTS ARE IN BOLD

Obj	Algorithm	GD	IGD	HV	SP
	EPCS	0.032826	0.098338	0.999189	0.053796
	DM1	0.413339	0.988651	0.463088	0.144059
	HYPE	3.896014	2.902610	0.140670	6.561140
5	MSOPS	0.812166	1.067975	0.576547	0.275905
	IBEA	0.092113	0.522730	0.848372	**0.028184**
	MOEA/D	**0.013024**	**0.064881**	**0.999879**	0.082779
	GDE3	0.106991	0.166421	0.961754	0.075737
	EPCS	**0.036352**	**0.125227**	**0.999590**	0.104243
	DM1	5.522970	9.782591	0.000000	1.111152
	HYPE	3.401907	1.991946	0.295143	12.86229
10	MSOPS	0.845442	1.184065	0.484052	0.906178
	IBEA	0.115727	0.707869	0.625747	**0.032895**
	MOEA/D	0.167341	0.194750	0.981445	0.581111
	GDE3	11.390740	4.401762	0.000000	5.754356
	EPCS	**0.072964**	**0.183860**	**0.997668**	0.185176
	DM1	2.362644	7.060653	0.000000	0.938650
	HYPE	3.456631	4.084097	0.013821	6.220383
20	MSOPS	2.260267	2.994712	0.234888	3.765049
	IBEA	0.138650	0.937664	0.518194	**0.019446**
	MOEA/D	0.132204	0.198913	0.976267	0.262688
	GDE3	74.19705	11.02369	0.000000	78.61775
	EPCS	**0.074703**	**0.203927**	**0.996306**	0.226602
	DM1	1.097720	1.984909	0.202276	1.499319
	HYPE	2.732765	2.737450	0.139155	12.34055
30	MSOPS	0.537217	1.147579	0.624463	0.978879
	IBEA	0.092979	0.832119	0.550843	**0.021172**
	MOEA/D	0.159463	0.270440	0.945890	0.481095
	GDE3	72.80482	12.94036	0.000000	112.0841
	EPCS	0.157778	**0.299386**	**0.941218**	0.361017
	DM1	2.092723	8.397977	0.040756	0.783590
	HYPE	1.238360	2.247890	0.068007	8.231189
40	MSOPS	0.595066	1.330462	0.467174	1.313336
	IBEA	**0.034790**	0.755009	0.564243	**0.021744**
	MOEA/D	0.172482	0.329656	0.871368	0.535051
	GDE3	67.80594	12.80697	0.000000	117.1270

TABLE II
COMPARISON BETWEEN EPCS, DM1 [1], HYPE [5], MSOPS [27], GDE3 [35], MOEA/D [54], AND IBEA [59] FOR DTLZ3. EACH RESULT REPRESENTS AVERAGE OF TEN INDEPENDENT RUNS. THE BEST RESULTS ARE IN BOLD

Obj	Algorithm	GD	IGD	HV	SP
	EPCS	0.013285	**0.184776**	**0.999769**	0.165717
	DM1	559.8657	328.7860	0.000000	21.15844
	HYPE	35.47658	17.44374	0.093549	9.529538
5	MSOPS	3.508899	4.126575	0.580779	**0.018387**
	IBEA	0.015720	0.619449	0.935837	0.060499
	MOEA/D	**0.012770**	0.188802	0.999698	0.178828
	GDE3	0.199899	0.353321	0.942604	0.158187
	EPCS	**0.006642**	0.316159	**0.999846**	0.391037
	DM1	176.0178	175.5425	0.000000	7.418563
	HYPE	47.17973	26.56151	0.000000	23.49494
10	MSOPS	2.047841	2.433865	0.397683	0.180877
	IBEA	0.620354	1.147053	0.475050	**0.082923**
	MOEA/D	0.231980	**0.314897**	0.991521	0.483641
	GDE3	957.2009	52.71012	0.000000	365.4511
	EPCS	**0.002501**	0.397915	**0.999625**	0.699480
	DM1	241.0584	239.9176	0.000000	8.360024
	HYPE	25.39766	17.37992	0.000000	12.06166
20	MSOPS	20.34418	15.79198	0.000000	6.824772
	IBEA	0.679047	1.386040	0.354108	**0.130909**
	MOEA/D	0.202673	**0.381913**	0.944192	0.601167
	GDE3	1476.918	125.7821	0.000000	939.8038
	EPCS	**0.002548**	**0.441901**	**0.999440**	0.799310
	DM1	262.4708	249.3190	0.000000	7.320301
	HYPE	39.91227	26.51468	0.000000	31.39164
30	MSOPS	11.86537	9.860647	0.000000	6.775918
	IBEA	1.462874	2.252185	0.035533	**0.119444**
	MOEA/D	0.822827	0.672858	0.770306	1.040213
	GDE3	1649.960	234.8678	0.000000	1281.516
	EPCS	**0.003338**	**0.507430**	**0.998819**	0.898371
	DM1	678.0232	358.6638	0.000000	9.884655
	HYPE	37.45071	26.103170	0.015295	26.45185
40	MSOPS	28.40809	14.54832	0.005376	22.41799
	IBEA	1.827912	2.182626	0.259881	**0.277449**
	MOEA/D	0.895152	0.838005	0.792906	0.915537
	GDE3	1771.968	354.7091	0.000000	1595.569

the obtained solutions are close to the Pareto-optimal front and distributed well in the entire front. For example, SP can be 0 when all the solutions are clustered in a small region, but located very far from the Pareto optimal front and not distributed in the entire front.

Following the suggestions from [18] and [22], we used the Wilcoxon signed-rank test [10] on the average values of GD, IGD, HV, and SP. The test was conducted to compare EPCS with any other algorithm in a pairwise manner. We used the average result of each problem, which was obtained from different runs of the same algorithm using the same problem. Suppose μ pairs of average results are obtained by applying EPCS and any other algorithm on μ different problems. At first, we computed the difference in each pair of average results. Let, the difference is δ_i, $i = 1, 2, \ldots, \mu$. According to the absolute values of the differences, we sorted the differences in ascending order and assigned a rank to each difference, starting from the rank 1 for the smallest difference and ending at the rank μ for the largest one. In case, if more than one differences were equal, an average rank was assigned to each of them [10]. The ranks were finally converted to signed-ranks by giving them the sign of the difference. For each problem, we gave a positive rank to EPCS if it was better than the competing algorithm with respect to a particular performance indicator (say, GD). Otherwise, we gave a negative rank. We summed all the positive ranks, R^+ and negative ranks, R^-. Finally, the minimum of R^+ and R^- was taken as the table value, T [10]. We used a significance level of 0.10 for our comparison. Since there are 25 problem instances (five different DTLZ problems and five different number of objectives for each problem), the T value for this significance level should be less than or equal to 100 (the critical value) for rejecting the null hypothesis [10]. We summarize the results of the Wilcoxon test in Table III, where $+$, $-$, and $=$ signs are employed to indicate whether the performance of EPCS is superior, similar or inferior to the competing algorithm. It is evident from this table that EPCS outperformed all the competing algorithms with respect to GD, IGD, and HV. Based on the SP indicator, EPCS outperformed HYPE, MSOPS, MOEA/D, and GDE3, while IBEA outperformed our EPCS. Neither EPCS nor DM1 was found superior when they were compared based on SP.

C. Discussion

This section briefly explains why EPCS achieves better competency over other evolutionary algorithms. The first reason is that EPCS encourages faster convergence, specifically

TABLE III
WILCOXON SIGNED-RANK TEST [10] RESULTS BETWEEN EPCS VERSUS DM1 [1], HYPE [5], MSOPS [27], GDE3 [35], MOEA/D [54], OR IBEA [59] BASED ON AVERAGE GD, IGD, HV, AND SP FOR 25 PROBLEM INSTANCES (FIVE DTLZ PROBLEMS, EACH PROBLEM WITH 5, 10, 20, 30, AND 40 OBJECTIVES). THE SIGNS +, =, AND − INDICATE EPCSS PERFORMANCE WAS STATISTICALLY SUPERIOR, INDIFFERENT FROM OR INFERIOR TO THE COMPETING ALGORITHM, RESPECTIVELY

Algorithm	GD	IGD	HV	SP
DM1	+	+	+	=
HYPE	+	+	+	+
MSOPS	+	+	+	+
IBEA	+	+	+	−
MOEA/D	+	+	+	+
GDE3	+	+	+	+

at earlier generations. This is achieved by utilizing a reference vector and a new fitness assignment strategy. These two components push the population to a narrow region of the objective space and make it possible reducing nondominated solutions (Fig. 7). In contrast, other algorithms we considered here do not use any such scheme. Without any direction, these algorithms may engage in excessive exploration at early generations, visit the same space again and again or focus on a smaller region of its current interest where the good solutions are not located. Although HYPE, MSOPS, IBEA, and MOEA/D eliminate the problem of exponential increase of nondominated solutions, HYPE, and IBEA use a single dimensional search, while MSOPS and MOEA/D use the aggregation function. It has been known that algorithms utilizing the aggregation function do not perform well on nonconvex search space [11]. Single dimensional search may not be suitable in a multiobjective scenario. The algorithms DM1 and GDE3 did not employ any mechanism for reducing nondominated solutions. Hence, they totally failed to converge in many occasions with a increasing number of objectives. For example, the GD values of GDE3 on DTLZ3 with 5 and 40 objectives were 0.199899 and 1771.968, respectively, a serious deterioration of performance with the increase of objectives. In our EPCS, the performance was also degraded with the increase of objectives, but the amount is not too much. This signifies the reduction of the nondominated solutions for solving problems with a increasing number of objectives.

The second reason is the strategy of negotiating two apparently conflicting objectives i.e., convergence and diversity in solving MaOPs. Our EPCS controls the diversity of an evolving population to a certain limit with the help of the reference vector. At the first part of evolution, EPCS encourages the evolving population to progress aggressively near to the Pareto optimal front by staying in a particular location, the Region 2. The population maintains diverse solutions in this location. In the second part of evolution, EPCS encourages more diversity by dividing the objective space perpendicular to the one done in the first part of evolution. Except DM1, other competing algorithms do not employ any explicit mechanism for diversity. However, the working procedure of EPCS that emphasizes convergence in the first part and diversity in the second part is quite different from all other algorithms including DM1. This approach seems to be effective as our experimental evidence suggests.

D. Sensitivity of Parameters

Conceptually, the simplest method to sensitivity analysis is to repeatedly vary one parameter at a time while holding the others fixed [38]. This type of analysis has been referred to as a local sensitivity since it only addresses sensitivity relative to the point estimates chosen and not for the entire parameter distribution. We applied this method to analyze the sensitivities of parameters used in EPCS. As seen from Algorithms 1-3, EPCS involves five parameters: t, k_1, k_2, c_1, and c_2. We give some guidelines how to select t (Section III-C) and k_1 and k_2 (Section III-C1). In our previous experiments, we chose fixed values for all these five parameters, which were obtained after some preliminary runs. We here present the sensitivity of these parameters by varying their values. For brevity, we chose two problems, i.e., DTLZ1 and DTLZ2 and two performance metrics, i.e., GD and IGD, for our sensitivity analysis. However, similar observation can be made for other problems and metrics. The detail results related to parameters' sensitivities are given in the supplementary materials.

As described in Section III, EPCS applies the fitness assignment Procedure 1 for the first t generations and then the fitness assignment Procedure 2 for the remaining $(g - t)$ generations. We used here t = 300 (100% of g), 225 (75% of g), 150 (50% of g), 75 (25% of g) and 0 (0% of g) for our simulation. It is found that the performance of EPCS is dependent on the value of t. A moderate t (e.g., $t = 75$ or 150) was found beneficial for obtaining good GD and IGD values. The smallest t (e.g., $t = 0$) or the largest one (e.g., $t = 300$) actually gave no opportunity for applying the fitness assignment Procedure 1 or 2. Note that our EPCS employs the Procedure 1 for convergence with reasonable diversity, while the Procedure 2 focuses more on diversity. Our experimental evidences suggest that focusing only on convergence or diversity is not beneficial for evolution. Most importantly, focusing only on convergence has a severe negative effect on the performance of EPCS. The effect of t also gives an indication about the essence of using two fitness assignment procedures in solving MaOPs. It would be interesting to see how our algorithm would perform if the two Procedures 1 and 2 are used in an intertwined fashion instead of using fixed parameters to sequentially execute the Procedures. Related to this point, it might be interesting to adaptively decide to execute the Procedure 2. For example, when the variance of the population Euclidean Distances to the reference vector falls below some threshold, EPCS may execute the Procedure 2 to avoid convergence to suboptimal regions of the Pareto front. These two ideas can be explored in future.

We have already discussed that k_2 should be larger than k_1 for proper functioning of EPCS (Section III-C1). We hold this relationship to choose three different sets of values for k_1 and k_2. We termed these sets as Case 1: $k_1 = 0.02$ and $k_2 = 0.10$, Case 2: $k_1 = 0.05$ and $k_2 = 0.10$, and Case 3: $k_1 = 0.01$ and $k_2 = 0.11$. Our simulation suggests that a small ratio between k_2 and k_1 seems to be good for a problem with a certain number of objectives. For example, EPCS with $k_1 = 0.20$ and $k_2 = 1.00$ achieved the best GD values and reasonable IGD values for DTLZ1 with upto 30 objectives. However, the best GD and IGD values for DTLZ1 with 40 objectives were obtained with $k_1 = 0.50$ and $k_2 = 1.00$. Hence, a moderate ratio between k_2 and k_1 can be used for obtaining a reasonable performance of EPCS.

The performance variation of EPCS for different values of c_1 and c_2 was observed. We also used here three different cases. We set $c_1 = 0.01$ and $c_2 = 0.01$ for Case 1, while we set $c_1 = 0.50$ and $c_2 = 0.01$ for Case 2. In Case 3, we set $c_1 = 0.50$ and $c_2 = 0.50$. It can be seen that the performance of EPCS is dependent on the value of c_1 and c_2. Most importantly, when we chose $c_1 = c_2$ and assigned a large value to them (Case 3), EPCS achieved a worse GD for the DTZ1 problem compared to that obtained with a small value (Case 1). This is, however, not true for IGD obtained for the DTLZ1 problem. In fact, we obtained the best IGD and reasonable GD for this problem when we chose $c_1 \gg c_2$ (Case 2). These results suggest that we can get good convergence and diversity by choosing $c_1 \gg c_2$.

Varying parameters one at a time is a typical first step in sensitivity analysis and is a local method. In the Sobol sensitivity analysis method [46], the variances of the key model outputs or metric values are decomposed to estimate global sensitivities to model parameters. This method identifies which parameters are important i.e., which cause the largest variation in the model outputs. Additionally, it identifies interactions between parameters. Global sensitivity indices produced by the Sobel analysis are generally used for estimating the influence of individual parameters or groups of parameters on the model outputs. For example, the first-order and second-order effects describe the sensitivities of each parameter in isolation and the pairwise interaction between parameters, respectively. Total-order effects describe the sum of all sensitivities attributed to each parameter. We again chose here DTLZ1 and DTLZ2 for the Sobel analysis. The bounds of different parameters used in the analysis are $t = 150$ to 300, $k_1 = 0.1$ to 0.5, $k_2 = 1.0$ to 1.1, $c_1 = 0.01$ to 0.50, and $c_2 = 0.001$ to 0.05.

We observed the second order effects obtained by the Sobel analysis for the five different parameters of EPCS. The analysis was carried out for the GD and IGD metrics and the software used for the analysis was obtained from http://www.moeaframework.org/. It is found that all the five parameters have different level of interactions, which vary with the number of objectives and/or problems. For example, for the DTLZ1 problem with five objectives, we see that k_1 interacts more strongly with other parameters. A different scenario is observed when we consider the DTLZ2 problem with five objectives for which c_1 interacts more strongly with other parameters. For DTLZ1 or DTLZ2 with a higher number of objectives, c_1 maintains strong interactions with other parameters. Although we show here the results of the Sobel analysis for the DTLZ1 and DTLZ2 problems, a very similar results can be obtained for other problems.

E. Further Evaluation of EPCS

This section presents the performance evaluation of EPCS on more challenging MOPs from the WFG [26] and CEC09 [55] test suits. For brevity and computational burden, we chose a representative and diverse set of problems from these two suites. Huband *et al.* [26] proposed the WFG test suite having nine scalable MOPs (WFG1-WFG9). We chose six of them including WFG2, WFG3, WFG5, WFG6, WFG8, and WFG9. These problems were nonseparable and representative cases. The detailed characteristics of these problems can be found in [26]. For each of the six WFG problems, we considered six different number of objectives i.e., 2, 3, 5, 7, 10, and 13 objectives. This raises the total number of WFG test instances to 36 (6 × 6). In [55], a set of 13 unconstrained MOP test problems has been proposed. We chose six of them including UF2-UF4 and UF8-UF10. The first three problems have two objectives, while the remaining problems have three objectives.

Similar to our earlier simulations, we used the population size 100 and the total generation 300 for both WFG and CEC09 problems. The value of k_1 was set to 0.1 and k_2 to 1.0 for these problems. We employed $c_1 = 5$ and $c_2 = 2$ for the WFG problems and $c_1 = c_2 = 2$ for the CEC09 problems. The additive epsilon (ϵ_+) [58] and IGD were used for measuring the performance of different algorithms. For the WFG problems, a reference set containing 100 000 points were used for computing both ϵ_+ and IGD. The guidelines provided in [55] were used for computing IGD of CEC09 problems. To compute ϵ_+s of CEC09 problems, we also used here the same points that were used in computing IGD.

We here dropped some algorithms that were used in previous comparison (Tables I and II), as those algorithms did not perform well for the DTLZ test suit. Note that the WFG and CEC09 test suites contain more challenging problems than the DTLZ suit. In our previous comparison, we observed that IBEA [59] performed better than the other hypervolume based algorithm, HYPE [5] and MOEA/D [54] performed better than its predecessor, MSOPS [27]. Hence, we did not choose HYPE and MOSPS for our comparison here. We compare the performances of EPCS with those of DM1 [1], IBEA [59], MOEA/D [54], and GDE3[35] for the WFG problems. The detail comparisons are given in the supplementary materials. It is seen that EPCS performs better than the other four algorithms for most of the problems. For example, out of 36 WFG problems, EPCS achieved the best $\epsilon+$ and IGD values for 18 and 25 WFG problems, respectively. The second best performer was GDE3, which achieved the best $\epsilon+$ and IGD for eights and five problems, respectively. We also applied here the Wilcoxon signed-rank test in the same way as described earlier. The test was performed for a significance level of 0.10 and the summary results are presented in the supplementary materials. It is observed that EPCS exhibits statistically superior performance compared to all the competing algorithms in terms of $\epsilon+$ and IGD.

We present the comparisons between EPCS and other algorithms on the CEC09 problems in the supplementary materials. Unlike the DTLZ and WFG problems, EPCS was found not better for most of the CEC09 problems. In fact, EPCS and GDE3 were performed nearly same on the CEC09 problems. For example, EPCS achieved the best ϵ_+ and IGD for three and two problems, respectively, while GDE3 achieved the best ϵ_+ and IGD for an equal number of three problems. As mentioned earlier, CEC09 problems have only two or three objectives. Hence, the number of nondominated solutions is not an obstacle for solving these problems. This is the main reason why EPCS is not the only best choice compared to the other algorithms for problems having a smaller number of objectives as CEC09 problems.

V. Conclusion

Evolutionary algorithms have been introduced to the MOP community for nearly three decades. However, they are suitable for solving MOPs with only few objectives. Most of these algorithms do not employ any mechanism for controlling the exponential increase of nondominated solutions with the increase of objectives. Thus, the algorithms face difficulty in converging toward the Pareto optimal front of MaOPs. This paper presents EPCS, a new algorithm, for solving MaOPs efficiently. Our algorithm introduces a new fitness assignment strategy for reducing nondominated solutions, specifically at earlier generations of evolution.

Extensive experiments have been carried out in this paper to evaluate how well EPCS performed on different problems in comparison with other EMO algorithms. We applied EPCS on 25 DTLZ, 36 WFG, and six CEC09 problem instances. We compared EPCS with several other algorithms based on a number of performance indicators. Our algorithm was found significantly better than the competing algorithms on the DTLZ and WFG problems in terms of convergence and diversity. Since CEC09 problems had two or three objectives, the reduction of nondominated solution was not essential for these problems and thus EPCS performed equally good compared to the other algorithms.

In its current implementation, EPCS have several user specified parameters although this is not unusual in the field of evolutionary algorithms. No attempts were made in EPCSs experiments to optimize parameters, which were set after some limited preliminary experiments. One of the future improvements to EPCS would be to reduce the number of parameters or make them adaptive. It would be interesting in the future to analyze EPCS further and identify its strengths and weaknesses. Most importantly, a more in-depth analysis to find the exact impact of the distances between the reference points and the Pareto front on the performance of our algorithm. Potential hybridization between EPCS and other algorithms (see [1], [2], [8]) would also be an interesting future research topic.

References

[1] S. F. Adra and P. J. Fleming, "Diversity management in evolutionary many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 15, no. 2, pp. 183–195, Apr. 2011.

[2] S. F. Adra, T. J. Dodd, I. Griffin, and P. J. Fleming, "Convergence acceleration operator for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 825–847, Aug. 2009.

[3] H. Aguirre and K. Tanaka, "Many-objective optimization by space partitioning and adaptive epsilon-ranking on MNK-landscapes," in *Proc. 5th Int. Conf. Evol. Multi-Criterion Optim.*, vol. 5467. 2009, pp. 407–422.

[4] M. Asafuddoula, T. Ray, and R. A. Sarker, "A decomposition based evolutionary algorithm for many objective optimization with systematic sampling and adaptive epsilon control," in *Evolutionary Multi-Criterion Optimization*. Berlin, Germany: Springer, 2013, pp. 413–427.

[5] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, 2009.

[6] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *Eur. J. Oper. Res.*, vol. 181, no. 3, pp. 1653–1669, Sep. 2007.

[7] C. K. Chow and S. Y. Yuen, "A multiobjective evolutionary algorithm that diversifies population by its density," *IEEE Trans. Evol. Comput.*, vol. 16, no. 2, pp. 149–172, Apr. 2012.

[8] A. Gaspar-Cunha and A. Vieira, "A multi-objective evolutionary algorithm using neural networks to approximate fitness evaluations," *Int. J. Comput. Syst. Signals*, vol. 6, no. 1, pp. 18–36, 2005.

[9] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. New York, NY, USA: Springer Verlag, 2007.

[10] G. W. Corder and D. I. Foreman, *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*. New York, NY, USA: Wiley, 2009.

[11] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms*. New York, NY, USA: Wiley, 2001.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[13] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multiobjective optimization test problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Honolulu, HI, USA, 2002, pp. 825–830.

[14] K. Deb, M. Mohan, and S. Mishra, "A fast multiobjective evolutionary algorithm for finding well-spread Pareto optimal solutions," Kanpur Genet. Algorithms Lab., Indian Inst. Technol. Kanpur, Kanpur, India, Tech. Rep. 2003002, 2003.

[15] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ϵ-dominated based multiobjective evolutionary algorithm for a quick computation of Pareto-optimal solutions," *Evol. Comput.*, vol. 13, no. 4, pp. 501–525, 2005.

[16] K. Deb and D. K. Saxena, "Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, 2006, pp. 3353–3360.

[17] K. Deb and J. Sundar, "Preference point based multi-objective optimization using evolutionary algorithms," in *Proc. Genet. Evol. Comput. Conf.*, Seattle, WA, USA, 2006, pp. 635–642.

[18] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[19] N. Drechsler, R. Drechsler, and B. Becker, "Multi-objective optimization based on relation favour," in *Proc. Evol. Multi-Criterion Optim. (EMO)*, London, U.K., 2001, pp. 154–166.

[20] M. Farina and P. Amato, "A fuzzy definition of 'optimality' for many criteria optimization problems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 34, no. 3, pp. 315–326, May 2004.

[21] M. Fleischer, "The measure of Pareto optima. applications to multiobjective metaheuristics," in *Proc. 2nd Int. Conf. Evol. Multi-Criterion Optim.*, Faro, Portugal, 2003, pp. 519–533.

[22] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, and F. Herrera, "Genetics-based machine learning for rule induction: State of the art, taxonomy, and comparative study," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 913–941, Dec. 2010.

[23] G. Greenwood, X. Hu, and J. D'Ambrosio, "Fitness functions for multiple objective optimization problems: Combining preferences with Pareto rankings," in *Foundations of Genetic Algorithms 4*. San Francisco, CA, USA: Morgan Kauffman Publishers, 1997, pp. 437–455.

[24] D. Hadka and P. Reed, "Borg: An auto-adaptive many-objective evolutionary computing framework," *Evol. Comput.*, vol. 21, no. 2, pp. 231–259, 2013.

[25] S. Huband, P. Hingston, L. White, and L. Barone, "An evolution strategy with probabilistic mutation for multiobjective optimisation," in *Proc. IEEE Congr. Evol. Comput.*, 2003, pp. 2284–2291.

[26] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, Oct. 2006.

[27] E. J. Hughes, "Multiple single objective Pareto sampling," in *Proc. IEEE Congr. Evol. Comput.*, vol. 4. Dec. 2003, pp. 2678–2684.

[28] E. J. Hughes, "MSOPS-II: A general-purpose many-objective optimiser," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Singapore, Sep. 2007, pp. 3944–3951.

[29] H. Ishibuchi, T. Doi, and Y. Nojima, "Incorporation of scalarizing fitness functions into evolutionary multiobjective optimization algorithms," in *Proc. 9th Int. Conf. Parallel Probl. Solving Nature (PPSN)*, Berlin, Germany, 2006, pp. 493–502.

[30] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary manyobjective optimization: A short review," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 2419–2426.

[31] V. Khare, X. Yao, and K. Deb, "Performance scaling of multiobjective evolutionary algorithms," in *Proc. 2nd Int. Conf. Evol. Multi-Criterion Optim.*, Faro, Portugal, 2003, pp. 376–390.

[32] J. B. Kollat and P. M. Reed, "Comparison of multi-objective evolutionary algorithms for long-term monitoring design," *Adv. Water Resour.*, vol. 29, no. 6, pp. 792–807, 2006.

[33] J. Knowles and D. Corne, "Quantifying the effects of objective spacedimension in evolutionary multiobjective optimization," in *Proc. 4th Int. Conf. Evol. Multi-Criterion Optim.*, Matsushima, Japan, 2007, pp. 757–771.

[34] M. Kööppen and K. Yoshida, "Substitute distance assignment in NSGA-II for handling many-objective optimization problems," in *Proc. 4th Int. Conf. Evol. Multi-Criterion Optim.*, Matsushima, Japan, 2007, pp. 727–741.

[35] S. Kukkonen and J. Lampinen, "GDE3: The third evolution step of generalized differential evolution," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Edinburgh, U.K., 2005, pp. 443–450.

[36] A. Lara, G. Sanchez, C. A. C. Coello, and O. Schütze, "HCS: A new local search strategy for memetic multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 112–132, Feb. 2010.

[37] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston, MA, USA: Kluwer, 1999.

[38] R. V. O'Neill, R. H. Gardner, and J. B. Mankin, "Analysis of parameter error in a nonlinear model," *Ecol. Model.*, vol. 8, pp. 297–311, Jan. 1980.

[39] K. Praditwong and X. Yao, "How well do multi-objective evolutionary algorithms scale to large problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Singapore, 2007, pp. 3959–3966.

[40] R. C. Purshouse and P. J. Fleming, "Evolutionary many-objective optimization: An exploratory analysis," in *Proc. IEEE Congr. Evol. Comput.*, Piscataway, NJ, USA, 2003, pp. 2066–2073.

[41] R. C. Purshouse and P. J. Fleming, "On the evolutionary optimization of many conflicting objectives," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 770–784, Dec. 2007.

[42] H. Sato, H. E. Aguirre, and K. Tanaka, "Controlling dominance area of solutions and its impact on the performance of MOEAs," in *Proc. 4th Int. Conf. Evol. Multi-Criterion Optim.*, Matsushima, Japan, 2007, pp. 5–20.

[43] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," Ph.D. thesis, Dept. Electr. Eng., Vanderbilt Univ., Nashville, TN, USA, 1984.

[44] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic Algorithms," in *Proc. 1st Int. Conf. Genet. Algorithms*, Hillsdale, NJ, USA, 1985, pp. 93–100.

[45] O. Schütze, A. Lara, and C. A. C. Coello, "On the influence of the number of objectives on the hardness of a multiobjective optimization problem," *IEEE Trans. Evol. Comput.*, vol. 15, no. 4, pp. 444–455, Aug. 2011.

[46] I. M. Sobol, "Sensitivity estimates for nonlinear mathematical models," *Math. Model. Comput. Exp.*, vol. 1, no. 4, pp. 407–417, 1993.

[47] L. Thiele, K. Miettinen, P. J. Korhonen, and J. Molina, "A preference-based evolutionary algorithm for multiobjective optimization," *Evol. Comput.*, vol. 17, no. 3, pp. 411–436, 2009.

[48] D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations," Ph.D. dissertation, Dept. Electr. Comput. Eng., Air Force Instit. Technol., Dayton, OH, USA, 1999.

[49] R. Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired coevolutionary algorithms for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 4, pp. 474–494, Aug. 2013.

[50] L. While, L. Bradstreet, and L. Barone, "A fast way of calculating exact hypervolumes," *IEEE Trans. Evol. Comput.*, vol. 16, no. 1, pp. 86–95, Feb. 2012.

[51] A. Wierzbicki, "On the completeness and constructiveness of parametric characterizations to vector optimization problems," *Oper. Res. Spectr.*, vol. 8, no. 2, pp. 73–87, 1986.

[52] P. Winkler, "Random orders," *Order*, vol. 1, no. 4, pp. 317–331, 1985.

[53] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.

[54] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[55] Q. Zhang *et al.*, "Multiobjective optimization test instances for the CEC 2009 special session and competition," School Comput. Sci. Electron. Eng., University of Essex, Colchester, U.K., Tech. Rep. CES-487.

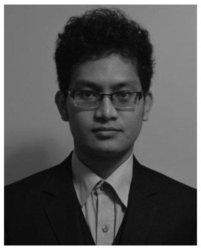
[56] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[57] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," in *Proc. Evol. Methods Design, Optim. Control Appl. Ind. Problems (EUROGEN)*, 2003, pp. 23–35.

[58] E. Zitzler, L. Thiele, M. Laumanns, C. M. Foneseca, and V. F. Grunert, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[59] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proc. 8th Int. Conf. Parallel Probl. Solving Nature (PPSN VIII)*, Birmingham, U.K., 2004, pp. 832–842.

[60] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 5, pp. 1402–1412, Oct. 2008.



Proteek Chandan Roy received the B.Sc. degree in computer science and engineering from the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2011. He is currently pursuing the Ph.D. degree in computer science from Mississippi State University, Starkville, MS, USA.

His current research interests include evolutionary algorithm, multiobjective optimization, planning, and probabilistic model checking.

Md. Monirul Islam (M'10) received the Ph.D. degree from the University of Fukui, Fukui, Japan.

He is a Professor with the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh. His current research interests include evolutionary robotics, evolutionary computation, neural networks, machine learning, and data mining.

Kazuyuki Murase received the M.E. degree in electrical engineering from Nagoya University, Nagoya, Japan, in 1978, and the Ph.D. degree in biomedical engineering from Iowa State University, Ames, IA, USA, in 1983.

He is a Professor with the Department of Human and Artificial Intelligence Systems, University of Fukui, Fukui, Japan. He served as the Founder Chairman of this department in 1999.

Xin Yao (M'91–SM'96–F'03) is a Chair (Professor) of Computer Science at the University of Birmingham, Birmingham, U.K. He is the Director of the Centre of Excellence for Research in Computational Intelligence and Applications, University of Birmingham, and of the Joint USTC Birmingham Research Institute of Intelligent Computation and Its Applications.

Prof. Yao is a Distinguished Lecturer of IEEE Computational Intelligence Society.