

# Finding Multiple Alternate Solutions Using Evolutionary Multi-objective Optimization

Kalyanmoy Deb and Ritam Guha

Michigan State University, East Lansing, USA, {kdeb,guharita}@msu.edu

COIN Report 2025005

**Abstract**—In many practical problem-solving tasks, instead of a single desired solution, often the goal is to find multiple alternate solutions (optimal or otherwise). In such tasks, numerical methods are employed to find one solution at a time by making it the focus of the ensuing computational task. On the other hand, evolutionary multi-objective optimization (EMO) algorithms – population-based computational procedures – have demonstrated their ability to find multiple Pareto-optimal solutions resulting from optimizing two or more conflicting objectives simultaneously. In this paper, we highlight the principles of an EMO procedure and how it can be extended to find multiple alternate solutions for a number of different problem-solving tasks encountered in practice. This ‘multiobjectivization’ task requires researchers to choose at least two conflicting goals arising from the specific problem-solving task and apply a suitably modified version of an existing EMO algorithm to find multiple alternate solutions. These extensions broaden EMO research and its applications, and propose a unified approach for solving various practical problem-solving tasks.

**Index Terms**—Multiple alternate solutions, multi-objective optimization, Multi-modal solutions, Pareto-optimal solutions

## I. INTRODUCTION

Many problem-solving tasks naturally result in multiple alternate solutions. However, for implementation purposes, only one of them must be chosen at the end. But the availability of multiple alternate solutions help a user in various ways. First, the availability of multiple solutions for a problem provides an added flexibility to the user to analyze different solutions and pick a single solution for implementation. Second, the knowledge of the gamut of all possible alternate solutions provides valuable information about the extent of possible solutions to the problem. The solutions together may provide a circumstantial picture of what solution to fall back to for an adverse situation for the currently adopted solution.

One simple-minded strategy for finding multiple solutions is to discover them one at a time. The original problem can be parameterized so that certain parameter values will result in one of the alternate solutions; a change in the parameter values will result in another alternate solution, and so on. A computing strategy would then be to solve each parameterized version of the problem at a time to obtain the respective alternate solution. This *generative* multi-solution computing strategy may not be the most computationally effective method, since there is no parallel search advantage

can be obtained by independently finding one solution at a time.

In the *simultaneous* computing approach, multiple alternate solutions can be targeted to be found in a single simulation run. In principle, such an approach can establish an inherent parallel search effort for finding multiple alternate solutions simultaneously.

Recent advances in solving multi-objective optimization problems using evolutionary algorithms (EAs) have amply demonstrated that a well-distributed multiple alternate Pareto-optimal solutions can be discovered using evolutionary multi-objective optimization (EMO) algorithms. In this paper, we extend the principles of EMO algorithms in the sense of multiobjectivization [1] and demonstrate that many practical problem-solving tasks can be addressed with an extension of the EMO approach to find multiple alternate solutions.

We would like to highlight that iteration-wise intermediate solutions created by an optimization algorithm before arriving at the final optimized solutions can be stored and manipulated to select a few as alternate solutions. Since such intermediate solutions are highly dependent on the chosen algorithm and are not the expected outcome of the optimization process, we do not recommend such a procedure to find alternate solutions.

In the remainder of the paper, we briefly contrast the working principles of generative and simultaneous computing approaches for finding multiple alternate solutions in Section II. Then, in Section III, we list a number of traditional and recent problem-solving tasks and demonstrate how a bi-objective reformulation of the problem can be solved by a modified EMO algorithm to find multiple alternate solutions of the original problem. Finally, conclusions are drawn in Section IV.

## II. GENERATIVE AND SIMULTANEOUS COMPUTING APPROACHES

A generative computing algorithm can be used to find one of the desired alternate solutions at a time. Usually the original problem is reformulated (or parameterized) so that a single solution becomes the target. Once the target solution is found by a suitable computing algorithm, it is saved in an archive and the problem is reformulated (with a different parameter setting) so that a different alternate solution becomes the new target. This process is continued until enough (or no

further new) alternate solutions are discovered. Instead of a parameterization, in some occasions, the original problem is modified with a squashing model so that already-found alternate solutions are no more important to the modified problem.

As clear from the above descriptions, there can be a number of hurdles with a generative approach:

- First, since each solution is found by an independent run, not much computational gain can be obtained by the knowledge of finding previous solutions to find the next solution. This lack of inherent parallelism in finding multiple solutions causes most criticism for generative computing approaches.
- Second, The parameterization, or otherwise, involves a number of parameters, which must be set right for the respective solution to be sufficiently and deservedly different from other alternate solutions. This inverse effect of how parameters must be changed to have a desired difference between two alternate solutions is difficult to know beforehand.
- Since multiple solutions are found by independent runs, the obtained solutions largely depend on the effectiveness of each run. This process, at the end, may end up suggesting inferiority of certain obtained solutions due to lack of global information of other solutions when a specific run is executed.
- If alternate solutions must have sufficient differences among them, before they can be considered a set of alternate solutions, a small error in locating a previous solution can magnify the errors in subsequent solutions. Since each application to find an alternate solution is independent, once it is found inaccurately, there is no global measure that can correct an incorrectly found previous solution.

On the other hand, a simultaneous computing algorithm is geared to find multiple alternate solutions in a single application. Although a generative computing approach can be used by actively utilizing the archive of already-obtained solutions in finding a new solution with the flexibility to re-optimize a previous sub-problem, the overall algorithm can be complex and computationally expensive to execute for re-optimizations. A population-based computing approach which can create and evaluate a set of alternate solutions in every step of the procedure makes more sense for this purpose. Evolutionary algorithms (EAs) provide such a computing paradigm, in which a population of solutions are evolved in comparison to each other and by creating new offspring solutions from already-found good parent population members [2].

However, the standard population approach of EAs, when applied to a single objective function, cannot automatically sustain multiple, diverse, and good solutions over a large number of generations. Even in problems having equally good alternate solutions in the search space (in a multi-modal sense), a genetic drift like effect sets in to the population and eventually the whole population converges to a single

solution. Although multiple solutions can be co-survived with a large enough population size to delay the drift process [2], it will not be most computationally effective approach. However, by modifying the selection operator, multiple stable *niches* can be established near each alternate solution and multiple solutions can be found in a single application of the so-called evolutionary multi-modal optimization (EMmO) algorithm [3].

However, if the original problem involves multiple conflicting objectives, the problem naturally results in multiple Pareto-optimal (PO) solutions. With the help of a niche-preserving selection operator and an emphasis for non-dominated solutions within a population, several evolutionary multi-objective optimization (EMO) algorithms have demonstrated their ability to discover multiple PO solutions in a single application [4]–[7].

In principle, either EMmO or EMO algorithm is suitable for finding multiple alternate solutions. In the former case, the single objective function being optimized must be multi-modal in nature, having multiple optimal solutions in the search space. In the latter case, at least two conflicting objectives are required and no multi-modality is needed for any objective function. Therefore, we choose the latter strategy and reformulate the original problem with two conflicting goals (of which, one usually comes with the task description and other is suitably chosen from the task description to provide a conflict with the first goal in arriving at multiple alternate solutions). Since the conflict between two goals will naturally produce a set of PO solutions of the resulting bi-objective problem, each PO solution will correspond to an alternate solution of the original problem. In some problems, more than two conflicting goals can also be chosen to find more functionally meaningful alternate solutions.

### III. CHOSEN PROBLEM-SOLVING TASKS

In this section, we present a number of problem-solving tasks which result in multiple alternate solutions and demonstrate how a suitable multi-objective reformulation can be constructed and solved to find them.

#### A. Finding Multiple Optimal Solutions for a Multi-Modal Single-Objective Function

In this task, a multi-modal single-objective function, with or without the presence of constraints, is optimized and the goal is to find more than one optimal solutions of the function. Multiple solutions, if present in a problem, are of importance to users, as they provide flexibility and knowledge to the users by providing more than one options for solving the problem.

The standard EAs, when applied to such multi-modal problems, eventually converge to a single optimum, hopefully to one of the global optima. EAs always thrive to find better and better solutions of the search space. Thus, even if an EA run tends to get stuck to a local optimum, as soon as a better (objective value) solution is discovered in a population, EA's focus will move to the new and better solution. This eventually allows the EA to converge to one of the global optimal solutions. Despite using a population, EAs, in their standard

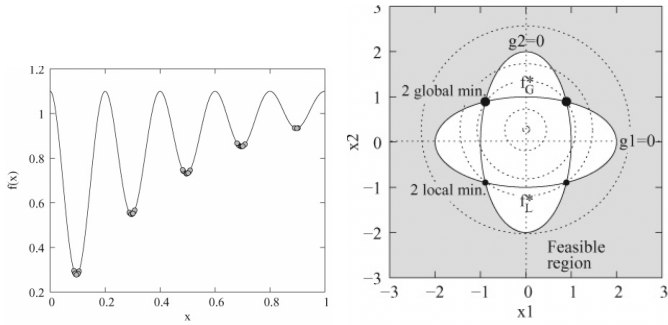
forms, cannot sustain population members on multiple global regions parallelly. Depending on the population size, a drift eventually occurs favoring one of the global optimum, causing the whole population to converge to a single optimum.

Researchers have modified the selection operator of an EA, by restricting two *neighboring* solutions to be compared against each other. Since this prevents two population members representing two distant optima to be compared, co-existence of multiple optimal solutions within a population is possible, like multiple niches (animals, amphibians, humans, etc.) co-exist in nature. Several different modifications to the selection operator have been suggested [8].

A recent study [9] has proposed a bi-objective approach for finding multiple optimal solutions:

$$\begin{aligned} & \text{Minimize} && (f(\mathbf{x}), N(\mathbf{x})), \\ & \text{subject to} && g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J, \\ & && x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n, \end{aligned} \quad (1)$$

where  $N(\mathbf{x})$  can be any local optimality metric which indicates the potential for  $\mathbf{x}$  to be a local optimum. Figures 1a and 1b show results from the bi-objective approach solved using a modified NSGA-II with  $N(\mathbf{x})$  defined as the proportion of  $H$  neighboring solutions better than  $\mathbf{x}$ . It is able to find multiple optimal solutions for an unconstrained and a constrained problem. More details can be found in [9]. Since the minimum of  $N(\mathbf{x})$  occurs at every local optimum, irrespective of local or global optima with different  $f(\mathbf{x})$  values, all optima of the problem becomes the weak Pareto-optimal set of the above bi-objective problem. By modifying the domination definition, standard EMO algorithms can be used to find multiple alternate optima simultaneously in a single run.



(a) Unconstrained problem with one global and four local minima (b) Constrained problem with two global and two local minima

Fig. 1: Bi-objective formulation of multi-modal problems finds multiple global and local optima simultaneously. (From [9])

### B. Finding Multiple Alternate Points with Common Features

In this task, no optimal solutions are sought, rather a set of alternate solutions  $\mathbf{X} = \{\mathbf{x}^{(i)}, i = 1, 2, \dots, K\}$  having certain common properties is the target. For example, the following task requires finding a set of feasible solutions:

$$\begin{aligned} & \text{Find} && \mathbf{X}, \\ & \text{subject to} && g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J, \\ & && h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K, \\ & && x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n. \end{aligned} \quad (2)$$

This task is also known as the *constraint satisfaction* problem.

One of the ways to find multiple alternate (feasible) solutions of the above problem is to solve the following two objective problem:

$$\begin{aligned} & \text{Minimize} && (\|\mathbf{x} - \mathbf{x}^{(L)}\|, -\|\mathbf{x} - \mathbf{x}^{(L)}\|), \\ & \text{subject to} && \mathbf{x} \in \mathcal{X}, \\ & && x_i = x_i^{(L)} + l_i(x_i^{(U)} - x_i^{(L)})/H_i, \\ & && l_i \in \{0, 1, \dots, H_i\}, \quad i = 1, \dots, n, \end{aligned} \quad (3)$$

where  $\mathcal{X}$  is the set of feasible solutions in the search space. For any two solutions  $\mathbf{x}$  and  $\mathbf{y}$  in the feasible search space satisfying  $f(\mathbf{x}) < f(\mathbf{y})$  (in which  $f(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^{(L)}\|$ ) has objective vectors  $(f(\mathbf{x}), -f(\mathbf{x}))$  and  $(f(\mathbf{y}), -f(\mathbf{y}))$ . It is clear that these two objective vectors are non-dominated each other. Since this is true for any two feasible solutions, all feasible solutions constitute the Pareto-optimal set of the above two-objective optimization problem.

In order to find a finite set of alternate solutions, the variable space can be discretized as indicated by the final line in the above formulation. Thus, instead of using real-parameter  $\mathbf{x}$  as a variable vector, a discrete variable vector  $\mathbf{l} \in \mathbb{Z}_{\geq 0} = \{0, 1, 2, \dots, H\}$  can be used. Figure 2 shows the result of the proposed bi-objective algorithm in finding a well-distributed set of feasible solutions for the following problem:

$$\begin{aligned} & \text{Minimize} && f(\mathbf{x}) = (x_1 - 1.5)^2 + (x_2 - 5.5)^2, \\ & \text{subject to} && 4x_1 + x_2^2 - 25 \leq 0, \\ & && -2x_1 + x_2 + 1 \leq 0, \\ & && -2 \leq x_1 \leq 8, \quad -6 \leq x_2 \leq 6. \end{aligned} \quad (4)$$

The solutions are obtained from a single run of NSGA-II, using a population size of 300 and 100 generations. The algorithm employs simulated binary crossover (SBX) with a crossover parameter  $\eta_c = 5$  and polynomial mutation (PM) with a mutation parameter  $\eta_m = 10$ .

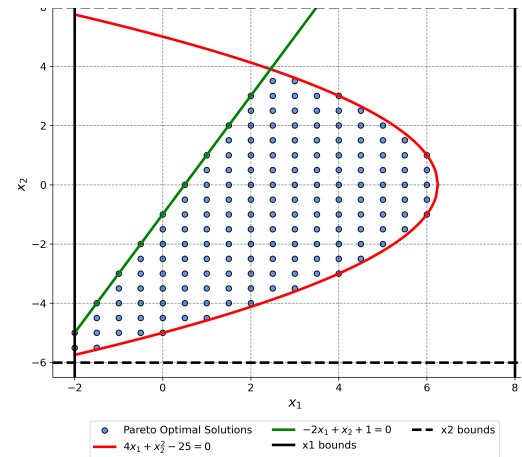


Fig. 2: A well-distributed set of feasible points for Problem (4).

1) *Finding Multiple Roots*: A special case of the above task is to find multiple alternate roots of a function, or for a number of functions:

$$\begin{aligned} & \text{Find} && \mathbf{X}, \\ & \text{subject to} && h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K. \end{aligned} \quad (5)$$

The proposed bi-objective approach to find multiple roots is as follows:

$$\begin{aligned} & \text{Minimize} \quad (\|\mathbf{x} - \mathbf{x}^{(0)}\|, -\|\mathbf{x} - \mathbf{x}^{(0)}\|), \\ & \text{subject to} \quad h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K, \end{aligned} \quad (6)$$

where  $\mathbf{x}^{(0)}$  is any solution. Once the first root ( $\mathbf{x}^{(1)}$ ) is found, the above formulation can be changed by replacing  $\mathbf{x}^{(0)}$  with  $\mathbf{x}^{(1)}$ .

Figure 3 shows how the above bi-objective approach finds four roots of  $h(x) = \sin(\pi x)$  in  $-0.5 \leq x \leq 3.5$  starting from  $x^{(0)} = -0.5$ . To obtain the results, a single run of NSGA-II is performed with a population size of 500 and 200 generations. The algorithm uses SBX with a  $\eta_c = 1$  along with PM with  $\eta_m = 1$ .

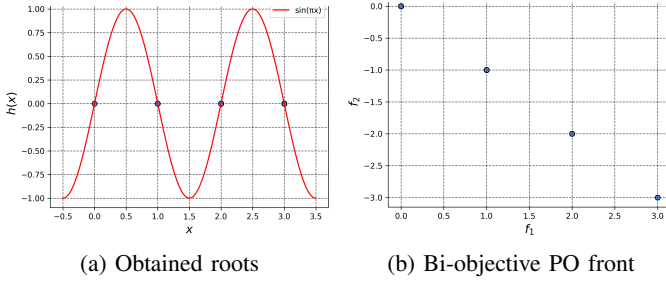


Fig. 3: Four roots are found for  $\sin(\pi x)$  in  $-0.5 \leq x \leq 3.5$  using the proposed bi-objective approach.

#### 2) Finding Multiple Solutions on Constraint Boundary:

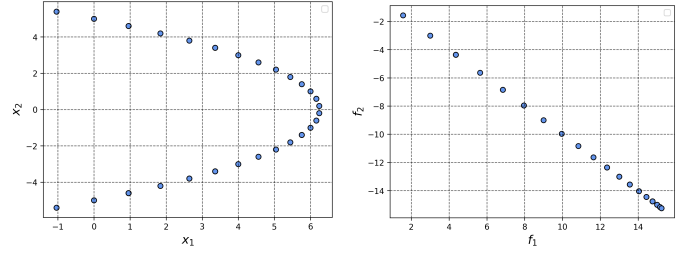
The task of finding a set of solutions on the intersection of a number of constraint boundaries is equivalent to the previous task mentioned in Subsection III-B1:

$$\begin{aligned} & \text{Find} \quad \mathbf{X}, \\ & \text{subject to} \quad g_j^S(\mathbf{x}) = 0, \quad j = 1, 2, \dots, J, \\ & \quad \quad \quad x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n. \end{aligned} \quad (7)$$

In this task, lower and upper boundaries of desired solutions may or may not be supplied. The resulting bi-objective problem can be formulated with a supplied  $\mathbf{x}^{(L)}$ , if any, or any guess solution  $\mathbf{x}^{(0)}$ . Figure 4 shows points obtained on the constraint  $g(\mathbf{x}) \equiv 4x_1 + x_2^2 - 25 = 0$  within  $-2 \leq x_1 \leq 8$  and  $-6 \leq x_2 \leq 6$ . The respective PO front of the bi-objective problem is also shown. NSGA-II is run with 1,000 individuals for 200 generations. The algorithm utilizes SBX with  $\eta_c = 1$  and PM with  $\eta_m = 1$ . To ensure an effective diversity of the solution set, a repair operator is incorporated, enforcing a repair multiple of 0.04 on  $x_1$ , where  $x_1$  is constrained to be a multiple of 0.04. Due to the handling of equality constraints, this problem may require a large number of evaluations, but a variable-wise niching approach may produce a better distribution of points on the constraint boundary. However, the principle of the bi-objective approach is clear from the figure.

#### C. Finding Iso-Objective Points

A simplified extension of the root finding task is to find multiple  $\mathbf{x}$ -vectors for computing contours of a function  $f(\mathbf{x})$ .



(a) Points on constraint boundary (b) Bi-objective PO front

Fig. 4: Bi-objective approach to find points on a constraint boundary.

For a contour level of  $f(x) = \bar{f}$ , following task needs to be solved to multiple iso-contour points on the contour line:

$$\begin{aligned} & \text{Find} \quad \mathbf{X}, \\ & \text{subject to} \quad f(\mathbf{x}) = \bar{f}. \end{aligned} \quad (8)$$

This problem is similar to the root-finding problem with  $K = 1$  and  $h(\mathbf{x}) = f(\mathbf{x}) - \bar{f}$ .

$$\begin{aligned} & \text{Minimize} \quad (\|\mathbf{x} - \mathbf{x}^{(0)}\|, -\|\mathbf{x} - \mathbf{x}^{(0)}\|), \\ & \text{subject to} \quad h(\mathbf{x}) = 0. \end{aligned} \quad (9)$$

After the points are found, they can be arranged in a suitable order for making a continuous contour line. For plotting a contour with a different level, the above task can be repeated with different  $\bar{f}$  value. For space restrictions and similarity with the root finding problem, we do not show any results for this task here.

#### D. Finding Relaxed Constrained Solutions

In most constrained optimization problems, the task is to find the optimal *feasible* solution having the minimum objective function value. This often results in a single solution which lies on the intersection of active constraints. The task is also referred to as a *hard-constraint* problem, as all active constraint functions are expected to be satisfied without fail. However, in many practical scenarios, certain constraints can be relaxed to some extent, say with a limit  $e_j$  (a small positive value) if a much better objective value can be obtained. Mathematically, soft-constraints are set as  $g_j^S(\mathbf{x}) \leq e_j$ , instead of the original constraint  $g_j(\mathbf{x}) \leq 0$ . Hard-constraints ( $g_j^H(\mathbf{x}) \leq 0$ ) are treated as they are. In such scenarios, it is difficult for users to predefine  $e_j$  for every constraint before any optimization is performed. They will rather expect the optimization task to find a few relaxed solutions having different but small constraint violations and then choose a single preferred one based on the trade-off between the extent of constraint violation and the improvement of the objective function.

If thought carefully, this relaxed constrained optimization task requires that multiple alternate near-constraint solutions be found for a subsequent choice of a single preferred solution.

A single-objective constrained minimization problem (only inequality constraints are considered for brevity):

$$\begin{aligned} & \text{Minimize} && f(\mathbf{x}), \\ & \text{subject to} && g_j^S(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J^S, \\ & && g_j^H(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J^H, \end{aligned} \quad (10)$$

can be converted into a bi-objective optimization problem as follows:

$$\begin{aligned} & \text{Minimize} && (\text{CV}^S(\mathbf{x}), f(\mathbf{x})), \\ & \text{subject to} && g_j^H(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J^H, \end{aligned} \quad (11)$$

where the constraint violation ( $\text{CV}^S$ ) for soft constraints are defined as follows:

$$\text{CV}^S(\mathbf{x}) = \sum_{j=1}^{J^S} \langle g_j^S(\mathbf{x}) \rangle, \quad (12)$$

where  $\langle \cdot \rangle$  is the operand, if the operand is positive; otherwise, zero.

A careful thought will reveal that one of the extreme PO solutions for the above bi-objective problem is the minimum  $\text{CV}^S$  solution having a value zero, meaning that all soft-constraints are satisfied. Hence, this extreme PO solution corresponds to the constrained minimum solution ( $\mathbf{x}^*$ ) of the original problem. The other extreme PO solution corresponds to the constrained minimum solution of the following problem:

$$\begin{aligned} & \text{Minimize} && f(\mathbf{x}), \\ & \text{subject to} && g_j^H(\mathbf{x}) \leq 0, \quad j = 1, 2, \dots, J^H, \end{aligned} \quad (13)$$

which may not be of much interest to the user. But the solutions closer to the constrained minimum solution  $\mathbf{x}^*$  on the PO set are of interest to the user, as they provide compromise solutions, each of which violates soft-constraints to an extent, but also makes a better objective value than  $f(\mathbf{x}^*)$ . The trade-off between soft-constraint violation  $\text{CV}^S$  and improvement in  $f$  can be analyzed and a single preferred solution can be chosen in a most information manner. Notice that each of these compromise solutions satisfy all hard-constraints.

Figure 5 shows the PO solutions obtained for the following single-objective constrained problem [10]:

$$\begin{aligned} & \text{Minimize} && f(\mathbf{x}) = (x_1 - 3)^2 + (x_2 - 2)^2, \\ & \text{subject to} && (x_1 - 0.05)^2 + (x_2 - 2.5)^2 - 4.84 \leq 0, \\ & && -x_1^2 - (x_2 - 2.5)^2 + 4.84 \leq 0, \\ & && 0 \leq x_1 \leq 6, \quad 0 \leq x_2 \leq 6. \end{aligned} \quad (14)$$

The constrained minimum solution is marked as ‘Optimum soln.’. The bi-objective approach also finds other trade-off (albeit, infeasible) solutions shown on the ‘Found PO front’. An application of KKT optimality conditions [11] to multiple scalarized problems indicate that the obtained bi-objective front is truly Pareto-optimal.

However, if the constraints are relaxed so that solutions having a total constraint violation  $\text{CV}(\mathbf{x}) \leq \epsilon = 0.1$  are allowed, the obtained PO front contains many such solutions having a smaller objective value than the constrained optimum. This flexibility of our proposed bi-objective approach to find near constraint-minimum solutions with soft-constraints are worth pursuing for real-world problems.

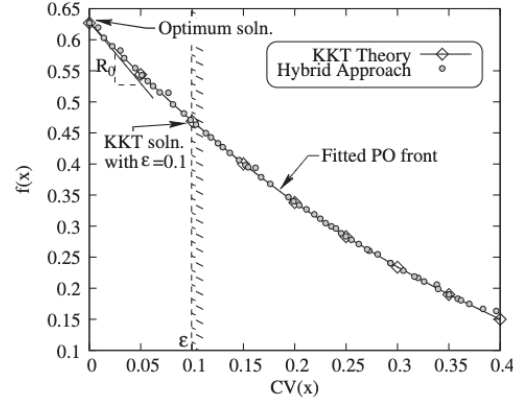


Fig. 5: PO front found by a modified NSGA-II method for a single-objective constrained problem. (Taken from [10])

This concept were also used to introduce one or more secondary objectives and find a biased distribution of multiple alternate solutions close to the minimum of the primary objective function. To alleviate the bloating issue in genetic programming (GP), program size is used as a secondary objective to find a minimum error code with smallest program size, rather than finding a minimum error code with an arbitrarily large program size [12]. In another study, a helper objective of maximizing the difference from the minimum of a primary objective was introduced along with the primary objective of minimizing weight of a structure to obtain solutions better than the minimum-weight structure obtained using the single-objective weight minimization approach. Since the presence of a conflicting objective to the primary objective allows a diverse set of solutions to be present in an evolving population, recombination operator works better, specially in problems where it is difficult to produce distant good solutions [13].

### E. Finding Ideal and Nadir Points

Finding ideal and nadir points in a multi-objective optimization task is important mainly for normalizing each objective function within  $[0, 1]$  so that any Euclidean distance measure in the  $M$ -dimensional objective space can be computed in a meaningful manner. Finding the ideal point

$$\mathbf{z}^{\text{ideal}} = \left( f_1(\mathbf{x}^{*(1)}), f_2(\mathbf{x}^{*(2)}), \dots, f_M(\mathbf{x}^{*(M)}) \right)$$

is relatively easy, as it can be constructed by finding the minimum solution ( $\mathbf{x}^{*(i)}$ ) of every independent constrained problem for each ( $i$ -th) objective function separately.

However, finding the nadir point  $\mathbf{z}^{\text{nadir}}$  is not that easy, as it is defined as the concatenation of worst objective values for the entire PO solution set:

$$\mathbf{z}^{\text{nadir}} = \left( f_1(\mathbf{x}^{\text{worst},(1)}), f_2(\mathbf{x}^{\text{worst},(2)}), \dots, f_M(\mathbf{x}^{\text{worst},(M)}) \right),$$

in which  $\mathbf{x}^{\text{worst},(i)}$  is the maximum objective value of  $f_i$  for all  $\mathbf{x}$  in the PO set. Thus, although the purpose of finding the nadir objective vector is to normalize objectives for a meaningful multi-objective optimization, unless the PO set is

known, the true nadir point cannot be obtained. In the multi-criterion decision-making (MCDM) literature, nadir point is estimated by the pay-off table method [14], which constructs the nadir point from worst objective values of  $M$  optimized solutions executed for computing the ideal point. Clearly, this pay-off table method makes a gross approximation of the nadir point and more accurate methods are certainly called for.

One EMO approach would be to find multiple alternate PO solutions only at the extremes of the PO set, rather than attempting to find a uniformly-distributed set on the entire PO set:

$$\begin{aligned} & \text{Minimize} && (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})), \\ & \text{subject to} && \mathbf{x} \in \mathcal{X}, \\ & && x_i \text{ is an extreme point for all } i, \end{aligned} \quad (15)$$

where  $\mathcal{X}$  is the feasible set of the original problem. Extreme points ( $\mathbf{x}^e$ ) can be defined in many different ways. One approach would be to check if the normalized objective  $f_i(\mathbf{x}^e) = 0$  for at least one  $i = 1, \dots, M$ . Such a task was demonstrated to be computationally tractable using the NSGA-II procedure [15] by redefining crowding distance metric in increasing order of magnitude from intermediate to smallest and largest objective values. Figure 6 shows the extreme points (in solid black) obtained for a three-objective problem using a modified NSGA-II approach [15], which focuses on finding the extreme points only, rather than the entire PO set (shown by open circles). The nadir point can be easily estimated from these extreme points. Due to focusing on a few extreme points, the modified NSGA-II is also more computationally efficient. Restricted reference vectors having above-defined

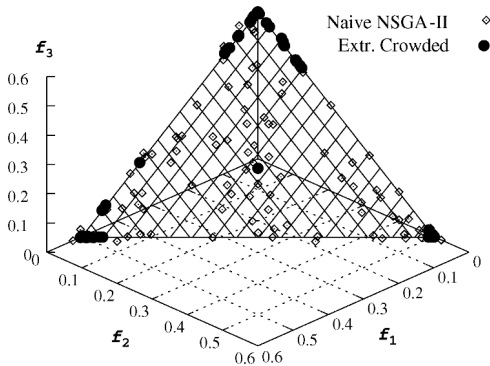


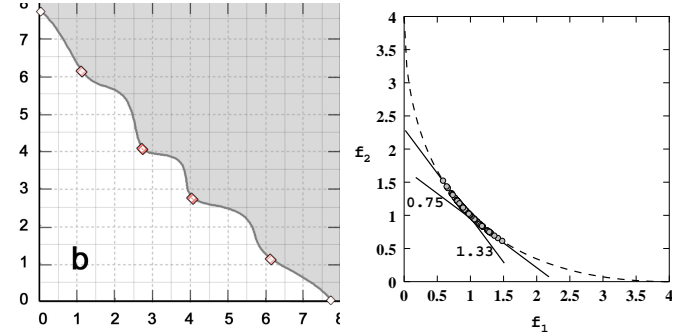
Fig. 6: Modified NSGA-II results on DTLZ1. (Taken from [15])

extreme point property can be used with evolutionary many-objective optimization (EMaO) algorithms, such as, NSGA-III [6] and MOEA/D [7] to estimate nadir points.

Population-based EMO or EMaO algorithms can be extended to find biased and a different distribution of PO points than the standard practice of finding a uniformly distributed set of PO points. We describe here a few such scenarios.

1) *Finding Multiple Knee Points:* Knee points on a PO front, if exists, make a large trade-off, indicating that neighboring PO solutions are not worth considering since they will cause a large loss in some objectives per unit gain in

other objectives. Instead of finding the entire PO set and then discovering knee points, if exists, a modified EMO algorithm can look for the knee points directly. In a previous study with NSGA-II, multiple knee solutions were directly found (Figure 7a) by emphasizing high trade-off solutions in NSGA-II's niching operator [16].



(a) Knee points found directly. (b) Proper PO points found directly. (Taken from [16])

Fig. 7: Special points on PO front are found directly.

2) *Finding Proper PO points:* Proper PO solutions [14] refer to high trade-offs (not to the extent arising for knee points) set by parameters, thereby allowing users to control the extent of proper PO solutions. A modification of the domination principle allows proper (or other more generalized [17]) PO points (Figure 7b) to be found directly.

3) *Finding Convex Part of the PO set:* In another multi-objective optimization task, instead of finding the entire PO set, only the convex part of the PO set may need to be discovered. Convex PO solutions have a direct connotation with weight-based importance of objectives, whereas non-convex part of the PO front cannot be found by any weighted-sum scalarization approach [14]. Hence, it may be worth to find the convex part of a PO front directly by changing domination definition in an EMO algorithm.

4) *Finding Partial PO set:* By using the cone domination principle [14], instead of the standard Pareto dominance principle, a past study has shown how a part of the PO front can be directly discovered quickly. Figure 7b shows the results of NSGA-II having a cone of domination described by  $(-1, 1.33)$  and  $(1, -0.75)$  for a two-objective problem. Note how only the top part of the PO front is found.

## F. Finding an Innovation Path

Next, we discuss completely different task, which is useful and applicable in many practical scenarios.

Consider a scenario, in which users are stuck with an existing solution or a methodology ( $\mathbf{x}^C$ ) which is apparently not the best one to continue with any more. This can be due to the fact that the existing solution was obtained in a much earlier task, which is no more efficient or relevant currently. In such an updated scenario, users find a completely different

target solution ( $\mathbf{x}^T$ ) which is better than ( $\mathbf{x}^C$ ) for a new set of goals:

$$\begin{aligned} & \text{Minimize } (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})), \\ & \text{subject to } \mathbf{x} \in \mathcal{X}. \end{aligned} \quad (16)$$

However, the difference between these two solutions are so drastic in terms of the sheer change in variable values or sub-processes that they involve or in terms of cost, time, or other implications that users are discouraged to make the large change. In the sense of ‘Kaizen’ principle [18], which advocates a number of small, gradual, and practically viable changes to existing solutions, rather than a single large and practically challenging change, our proposed innovation path (IP) approach suggests to find a gradual sequence of intermediate solutions starting from the existing solution  $\mathbf{x}^C$  and reaching near the target solution, thereby advocating to find multiple alternate (ordered) solutions.

The proposal uses a bi-objective formulation to find multiple intermediate solutions from  $\mathbf{x}^C$  toward the target solution  $\mathbf{x}^T$  by minimizing two criteria: (i) difference between  $\mathbf{x}^C$  and any candidate intermediate solution  $\mathbf{x}$ , and (ii) a scalarized version of new objectives for improving the current solution. For example, if  $M$  new goals  $f_i(\mathbf{x})$   $i = 1, \dots, M$  are to be simultaneously minimized for arriving at the target solution, a conflict in the objectives will result in a Pareto-optimal set of solutions. Then, an achievement scalarizing function (ASF) can be minimized for an aspiration point  $\mathbf{z}$  and a weight vector  $\mathbf{w}$ , as the second objective:

$$\text{ASF}(\mathbf{x}) = \max_{i=1}^M \left( \frac{f_i(\mathbf{x}) - z_i}{w_i} \right). \quad (17)$$

Thus, minimizing ASF function, which attempts to improve the objectives, and minimizing the difference from existing solution are two conflicting goal for the proposed bi-objective optimization task [19]:

$$\begin{aligned} & \text{Minimize } (\text{ASF}(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^C\|), \\ & \text{subject to } \mathbf{x} \in \mathcal{X}. \end{aligned} \quad (18)$$

Due to the conflict in two criteria, the solution of the above problem will be a set of trade-off solutions, of which one extreme will be the existing solution  $\mathbf{x}^C$ . This is because this solution will make the second objective minimum (with a value of zero). The minimum-ASF solution will be the other extreme, corresponding to the target solution  $\mathbf{x}^T$ . The intermediate trade-off solutions will make a compromise between the difference from existing solution and improvement of ASF function.

However, one of the practicalities of this problem that users may not be interested in a large number of intermediate solutions, but in a handful of (three to five) solutions. Thus, to find a few compromise trade-off solutions, additional step-constraints can be added to the above formulation. For example, the following step-constraint will ensure that two consecutive intermediate IP solutions ( $\mathbf{x}^{(i)}$  and  $\mathbf{x}^{(j)}$ ) have certain minimum difference ( $\Delta$ ) in them:

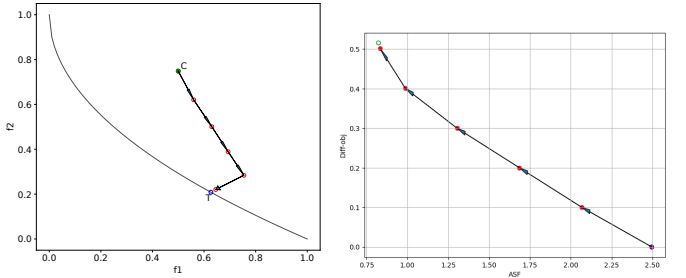
$$G(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \equiv \Delta - \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\| \leq 0. \quad (19)$$

The bi-objective problem in Equation 18 can be modified as follows for  $L$  supplied step-constraints:

$$\begin{aligned} & \text{Minimize } (\text{ASF}(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^C\|), \\ & \text{subject to } G_l(\mathbf{x}, \mathbf{x}^{(k)}) \leq 0, \quad \mathbf{x}^{(k)} \in \mathbf{x}^{*,(k)}, \\ & \quad \forall k = 1, \dots, |\mathbf{x}^*|, \quad l = 1, 2, \dots, L, \\ & \quad \mathbf{x} \in \mathcal{X}. \end{aligned} \quad (20)$$

The set  $\mathbf{x}^{*,(k)}$  contains all intermediate anchor solutions obtained before  $k$ -th one is stabilized. Due to the evolving nature of this set, which in turn defines the constraint set, the problem is difficult to optimize. Also, due to the flexibility of EMO approaches and their ability to find and store multiple alternate solutions, they have a better chance of solving such complex optimization problems.

Figure 8 shows how a modified NSGA-II approach is able to find a few IP solutions for the ZDT1 goal problem using the above bi-objective approach. The existing solution  $\mathbf{x}^C$  turns to be a dominated solution for the new goals  $f_1$  and  $f_2$ , shown in Figure 8a. By setting a target PO solution with ASF set with  $\mathbf{w} = (0.75, 0.25)$  and  $\mathbf{z} = (0, 0)$ , the figure shows how the existing (dominated) solution must be modified, with the step-constraint specified in Equation 19 ( $\Delta = 0.1$ ), gradually to arrive at a solution close to the target. The bi-objective PO front in the ASF-Diff space, where the optimization was run, are shown in Figure 8b, demonstrating the trade-off between improvement of objectives and difference in the solutions.



(a) IP on original objective space.

(b) IP on ASF-Diff space.

Fig. 8: A bi-objective IP-seeking algorithm finds a set of IP solutions. (Taken from [19])

### G. Finding Regularized Trade-off Points

It is observed that multiple PO solutions in a multi-objective optimization task possess certain *common* features among the variables [20]. This happens in most practical problems due to the fact that each PO solution must satisfy certain optimality conditions [4], [14], [21]. The common principles among PO solutions are manifestations of the optimality principles, which dominated and non-Pareto solutions do not possess. For a better understanding of PO solutions and their properties and for a more informed future application of the problem, the common *innovized* principles are worth discovering.

However, in certain scenarios, deciphered ‘‘innovized’’ principles may not be easily interpretable by human decision-makers (DMs) and therefore defeats the purpose of utilizing them for any future use. In a revolutionary manner, DMs may

be interested in finding a new set of trade-off solutions, which are close to the original PO front, but possess certain DM-specified interpretability. For example, the desired relationships among variables can be at most linear and involving a maximum of three variables. The DM does not know a priori and therefore cannot specify which variables must be involved in each relationship, how many such first or zero-th order relationships must exist, and what are the exact mathematical forms of these relationships. A new set of trade-off solutions lying on the so-called *regularized front*, close to the original PO front, will then satisfy DM's interpretability requirement.

Some thoughts will reveal that any parameterized ( $\mathbf{p}$ ) formulation of a specific relationship structure ( $R_{\mathbf{p}}(\mathbf{x}) = 0$ ) of the variables must be optimized first to find the resulting trade-off set ( $T_{\mathbf{p}}(\mathbf{x})$ ). Then, an upper-level optimization in the parameter space can be introduced to optimize a bi-objective problem of minimizing the difference in hypervolume of original PO set and the obtained  $T_{\mathbf{p}}$  set and minimizing the complexity  $\mathcal{C}$  of the relationship structure  $R_{\mathbf{p}}(\mathbf{x})$ :

$$\begin{aligned} & \text{Minimize } (\mathcal{C}(R_{\mathbf{p}}(\mathbf{x})), (\text{HV}(\text{PO}) - \text{HV}(T_{\mathbf{p}}(\mathbf{x})))) , \\ & \text{subject to } \mathbf{x}(\mathbf{p}) = \text{argmin} \{ \mathbf{f}(\mathbf{x}) | R_{\mathbf{p}}(\mathbf{x}) = 0, \mathbf{x} \in \mathcal{X} \} . \end{aligned} \quad (21)$$

The above is a bilevel, bi-objective optimization problem, in which upper-level variables are parameter set  $\mathbf{p}$  and lower-level variables are problem variables  $\mathbf{x}$ .

A recent study [22] has devised a systematic evolutionary bilevel, bi-objective optimization algorithm to find the regularized front (RF). Figure 9 shows that the RF found for the constrained BNH problem, shown with red points, are close to the original PO set (shown with blue points), but the RF contains following simple and easily interpretable relationship among the two variables of the problem:  $x_2 = 0.5x_1 + 0.5$ .

The RF front is not that different from the original PO front, yet the DM obtains the above simple regularity relationship for RF solutions. It can be argued that such a small sacrifice in optimality for gaining interpretability may be desired in practical problems, making the solution of the above bilevel, bi-objective problem worthy.

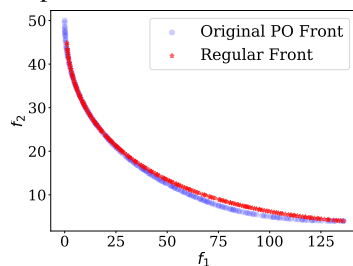


Fig. 9: RF and PF for the BNH problem. (Taken from [22])

#### IV. CONCLUSIONS

Practice often comes with different unorthodox problem-solving tasks requiring not one, but a number of alternate solutions, either to have more fundamental knowledge for solving the problem, or to have a flexibility in choosing a suitable one for practice, or for another practical reasons. It then becomes important to explore if a unified solution strategy can be implemented to find multiple alternate solutions with slight changes, if needed, to suit different kinds of problem-solving tasks. In this paper, we have demonstrated that a bi-objective formulation of a problem with two or

more conflicting goals can provide a computing paradigm resulting in multiple alternate solutions. Such a technique can be generically applied to find multiple alternate solutions.

In this paper, we have presented a number of different problem-solving tasks – multi-modal optimization, multiple root-finding, constraint-satisfaction with multiple points, multiple iso-objective solutions, soft-hard constraint optimization, nadir point estimation, knee point discovery, innovation path discovery, and regularized trade-off set discovery – can all be achieved using a common reformulation of the problem with two or more objectives. The principle of EMO algorithms enables multiple alternate solutions to be discovered for various such problem-solving tasks.

Certainly, many other problem-solving tasks can also be addressed by using a suitable multi-objective approach to find multiple alternate solutions and must be pursued in the future. Additional benefits of using three or more conflicting objectives for such a common approach should be investigated. Computational benefits, if any, compared to the standard practices for solving each task must also be investigated. A similar effort should be made to investigate how convenient the multi-modal computing approach would be to find multiple alternate solutions of each of the problem-solving tasks of this paper. Nevertheless, the benefit of following a single computing approach for solving commonly appearing problem-solving tasks in a unified manner has its own benefits and this paper has exemplified the use of a bi-objective approach to address a plethora of such problem-solving tasks for attracting attention of EMO researchers and practitioners.

#### REFERENCES

- [1] J. Handl, S. Lovell, and J. Knowles, "Multiobjectivization by decomposition of scalar cost functions," *Parallel Problem Solving from Nature-PPSN X*, pp. 31–40, 2008.
- [2] D. E. Goldberg, *Genetic Algorithms for Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
- [3] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*, 1987, pp. 41–49.
- [4] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, UK: Wiley, 2001.
- [5] C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont, *Evol. Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer, 2002.
- [6] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part I: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.
- [7] 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.
- [8] N. Glibovets and N. M. Gulayeva, "A review of niching genetic algorithms for multimodal function optimization," *Cybernetics and Systems Analysis*, vol. 49, pp. 815–820, 2013.
- [9] K. Deb and A. Saha, "Multimodal optimization using a bi-objective evol. algorithms," *Evol. Comput. J.*, vol. 20, no. 1, pp. 27–62, 2012.
- [10] K. Deb and R. Datta, "A bi-objective constrained optimization algorithm using a hybrid evolutionary and penalty function approach," *Engineering Optimization*, vol. 45, no. 5, pp. 503–527, 2012.
- [11] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell, *Engineering Optimization Methods and Applications*. New York : Wiley, 1983.
- [12] S. Bleuler, M. Brack, and E. Zitzler, "Multiobjective genetic programming: Reducing bloat using SPEA2," in *Proceedings of the 2001 Congress on Evolutionary Computation*, 2001, pp. 536–543.

- [13] D. Sharma, K. Deb, and N. N. Kishore, "Towards generating diverse topologies of path tracing compliant mechanisms using a local search based multi-objective genetic algorithm procedure," in *IEEE Congress on Evolutionary Computation '08*, 2008, pp. 2004–2011.
- [14] K. Miettinen, *Nonlinear Multiobjective Opt.* Boston: Kluwer, 1999.
- [15] 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*, vol. 14, no. 6, pp. 821–841, 2010.
- [16] K. Deb and S. Gupta, "Understanding knee points in bicriteria problems and their implications as preferred solution principles," *Engineering Optimization*, vol. 43, no. 11, pp. 1175–1204, 2011.
- [17] K. Deb and M. Ehrgott, "On generalized dominance structures for multi-objective optimization," *Mathematical and Computational Applications*, vol. 28, no. 5, p. 100, 2023.
- [18] R. Maurer and L. A. Hirschman, *Spirit of Kaizen: Creating Lasting Excellence One Small Step at a Time.* McGraw-Hill Professional, 2012.
- [19] A. Khan and K. Deb, "Innovation path: Discovering an ordered set of optimized intermediate solutions from an existing to a desired solution," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2024, pp. 529–537.
- [20] K. Deb and A. Srinivasan, "Innovization: Innovating design principles through optimization." in *Proceedings of the Genetic and Evol. Comput. Conf. (GECCO-2006)*, New York: ACM, 2006, pp. 1629–1636.
- [21] M. Ehrgott, *Multicriteria Optimization.* Berlin: Springer, 2000.
- [22] R. Guha and K. Deb, "Compromising Pareto-optimality with regularity in platform-based multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 6, pp. 1746–1760, 2023.