Multiphase Balance of Diversity and Convergence in Multiobjective Optimization

Haitham Seada, Mohamed Abouhawwash, and Kalyanmoy Deb, *Fellow, IEEE*

***Abstract*—In multiobjective optimization, defining a good solution is a multifactored process. Most existing evolutionary multi- or many-objective optimization (EMO) algorithms have utilized two factors: 1) domination and 2) crowding levels of each solution. Although these two coarse-grained factors are found to be adequate in many EMO algorithms, their relative importance in an algorithm has been a matter of great concern to many current studies. We argue that beside these issues, other more fine-grained factors are of importance. For example, since extreme objective-wise solutions are important in establishing a noise-free and stable normalization process, reaching extreme solutions is more crucial than finding other solutions. In this paper, we propose an integrated algorithm, B-NSGA-III, that produces much better convergence and diversity preservation. For this purpose, in addition to emphasizing extreme objective-wise solutions, B-NSGA-III tries to find solutions near intermediate undiscovered regions of the front. B-NSGA-III addresses critical algorithmic issues of convergence and diversity-preservation directly through recent progresses in literature and integrates all these critical fine-grained factors seamlessly in an alternating phases scheme. The proposed algorithm is shown to perform better than a number of commonly used existing methods.**

***Index Terms*—Evolutionary, Karush–Kuhn–Tucker proximity metric (KKTPM), local search (LS), multiobjective, optimization.**

I. Introduction

BALANCING convergence and diversity has been drawing researchers' interest since the first ever multiobjective optimization algorithm [1]. Most early studies gave higher priority to convergence over diversity [2]. Gradually, researchers started to realize that strictly following this specific order might be too restrictive [3]. In this section, we will discuss a selected set of notable efforts showing the progression of research in this topic.

Manuscript received August 26, 2017; revised May 13, 2018; accepted August 2, 2018. Date of publication September 19, 2018; date of current version May 29, 2019. *(Corresponding author: Haitham Seada.)*

H. Seada and K. Deb are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: seadahai@msu.edu; kdeb@msu.edu).

M. Abouhawwash is with the Department of Mathematics, Faculty of Science, Mansoura University, Mansoura 35516, Egypt (e-mail: saleh1284@mans.edu.eg).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author. This includes extended analysis and additional tables and figures supporting the contribution presented in the original manuscript. This file is 1.28 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2018.2871362

A. Toward Better Balance

A number of studies on evolutionary multiobjective optimization [4], [5] expressed the need for emphasizing diversity preservation early on during a run. From a different perspective, the desired balance can be attained through variation operators. The study conducted by Tan *et al.* [6] explored this idea in the context of binary chromosome representation. They proposed an adaptive variation operator (AVO) that tunes crossover and mutation to emphasize diversity (a consequence of exploration) in the beginning, then move gradually toward emphasizing convergence (a consequence of exploitation).

Due to the recent success of many-objective optimization algorithms (>3 objectives) [3], [7]–[9], balancing convergence and diversity becomes even more challenging than before. As the number of objectives grows, the percentage of nondominated solutions increases significantly. Consequently, one of the most widely used converging forces—nondomination—becomes ineffective. Although this effect can be generally considered a disadvantage, it can be beneficial in the realm of many objectives. As the dimensionality of the objective space increases, evolutionary multi- or many-objective optimization (EMO) algorithms become more prone to loosing parts of the Pareto front due to premature convergence. In such cases, maintaining mild convergence pressure allows for more exploration, which in turn can result in better diversity. This is the reason behind the reduced selection pressure adopted in NSGA-III [7].

In this context, Li *et al.* [10] proposed an interesting extension to MOEA/D. Their approach treats subproblems and solutions as two different types of agents, representing both convergence and diversity. After each agent—from both sides—creates its own preference list, a stable matching algorithm [11] dictates the final subproblem-solution associations. Those selected solutions move to the next generation.

A line of MOEA/D modification was initiated by the work of Zhang *et al.* [12] in 2009 (MOEA/D-DRA), which allows for distributing resources dynamically across subproblems. It uses a utility function for each subproblem that changes based on its improvement. Although the range of utility values is narrow, it is able using standard operators to achieve better results than its predecessor. Wang *et al.* [13] proposed a modification of MOEA/D replacement strategy. Unlike MOEA/D they allow a solution to be replaced by another coming from a different subproblem, if the replacing solution is a better fit, hence the name "global replacement." Another recent study by Yuan *et al.* [14] proposed a modified version of MOEA/D that achieves better balance. In their study they use a slightly

relaxed mating restriction (controlled by a probability parameter δ). It is worth noting that MOEA/D-DU is a steady state algorithm. The study also includes a similar extension of EFR [15].

Most of these studies do not propose truly *automatic* balancing approaches. We can generally classify them into two categories. The first category follows a predefined preference scheme to achieve the desired balance, either by focusing on convergence then diversity or doing the opposite! Although following either way can show some merit on a selected set of problems, none of the two approaches can be considered *globally* useful. In addition, many problems would not fit in any of these two extremes. A parallel emphasis would be more robust and predictable over the whole spectrum of optimization problems. This parallel approach is adopted by the second category, however, this category uses an additional user-defined parameter to indicate the relative effort put to either convergence or diversity. Now it is the user's responsibility to find the right value for this parameter, which is a very challenging task given a new—possibly black-box—optimization problem. Thus, none of these approaches can be considered truly *automatic*.

One of the contributions of this paper is the infinite seamless alternation of phases it follows, in order to reach the desired balance without adding explicit preferential parameters.

For the sake of completeness, a different yet related line of research tries to solve the same problem by optimizing performance metrics. Several performance metrics have been utilized in this regard. These metrics have the ability to assess both convergence and diversity simultaneously. Emmerich *et al.* [16] and Naujoks *et al.* [17] used the widely adopted hypervolume metric in their metamodel-assisted SMS-EMOA and NDS-HV, respectively. Other notable efforts [18]–[20] used the averaged Hausdorff distance (Δ_p) metric proposed by Schutze *et al.* [21]. Δ_p avoids the high computational complexity of calculating hypervolume in higher dimensions. These methods are promising, yet they are optimizing a model (the adopted metric) of the actual convergence-diversity problem, which makes them highly dependent on the metric itself. In this paper, the balance is achieved on an algorithmic level. Hence, it can be tailored to specific situations and certain parts of the Pareto front according to the dynamics of the optimization process itself.

B. Local Search

In a multiobjective optimization context, local search (LS) refers to optimizing an aggregate (combined) form of all the original objective functions. Several studies used LS in EMO. Ishibuchi and Murata [22] started this interesting combination in IM-MOGLS. Their approach uses a simple weighted-sum aggregate (scalarization) function to combine all objectives. IM-MOGLS then starts after the proposal of NSGA-II, Ishibuchi and Narukawa [24] proposed an NSGA-II extension that uses LS to solve multiobjective 0/1 knapsack problems. Other researchers followed the same path by adding LS to already existing EMO algorithms. Knowles and Corne [25] proposed a similar extension to PAES.

Harada *et al.* [26] proposed the notion of a Pareto descent direction. Their algorithm, Pareto descent method (PDM), solves a linear programming problem to get each of these directions. PDM is an evolutionary strategy-like algorithm where mutation is replaced by an LS in the Pareto descent direction of each solution. Later, Bosman proposed and tested several multiobjective gradient-based algorithms in [27]. Starting at some solution, he also provided an analytical representation of the directions in which objective values can not deteriorate (they either improve or remain constant). And although Bosman's work considers EMO algorithms for hybridization, it need not be limited to the evolutionary domain. Other LS strategies have also been proposed including directed search and continuation methods proposed by Schütze *et al.* [28].

The reader can notice that different studies used different ways to combine objectives. Some used achievement scalarization functions (ASFs) [29], while others tried to reach a weighted combination of the objectives based on finding a promising direction that may improve their values. Using both approaches is dependent on the underlying search directions/weights. However, these directions may harm diversity if they are calculated based on potentially decreasing objective values only. In addition, just using a weighted sum approach yields the algorithm unable to attain nonconvex sections of the Pareto front. We recommend using the former approach, ASF, as a general way of combining objectives in any algorithm, as it theoretically enables the algorithm to reach any point, regardless of the convexity of the region at which this point lies on the front. In addition, given a reference directions-based algorithm like B-NSGA-III , ASF can use those readily available directions.

On the other hand, we also noticed that using ASF can significantly distort the contour lines of the aggregate function if the direction is too flat or steep. Consequently, ASF shows poor performance when used to reach objective-wise extreme points. This means that ASF is not a universal formulation that can be used unconditionally.

In this paper, we are not concerned with the specific single objective optimization algorithm used by the single objective optimizers used in these studies. We are more concerned with the formulation of the aggregate function itself, and how LS can be employed in the course of a bigger algorithm, in order to serve the ultimate purpose of automatically balancing convergence and diversity. One of the contributions of this paper is using different formulations according to the current state/phase of the optimization process.

C. Karush–Kuhn–Tucker Proximity Measures

Since its introduction, Karush–Kuhn–Tucker (KKT) conditions have been used extensively by researchers [30]–[33]. KKT conditions are necessity conditions that each optimal point must satisfy. The opposite is not true, however. A KKT point is not necessarily optimal. Despite this fact many researchers and even popular softwares use some form of KKT

conditions to check the optimality of their solutions. Thus, it would be more accurate if we said that these studies/softwares search for KKT points, instead of actual optimal points. To ensure optimality another set of more involved conditions (sufficiency conditions) should be used. But, sufficiency conditions are more difficult to apply in real-world problems. For example, KKT conditions are sufficient (ensure global optimality) if the problem satisfies moderate convexity assumptions, a rare case in practice.

Another problem researchers face is that KKT conditions are "singular," i.e., they hold only at KKT points. They do not provide any clue regarding how far an arbitrary solution can be from being a true KKT point. Thus, in their original form, KKT conditions cannot be used as a convergence metric. Dutta *et al.* [34] tackled this problem and proposed their KKT proximity metric (KKTPM) for single objective optimization problems. The key to overcome the singularity problem was to relax the complementary slackness condition of the original KKT formulation. Deb and Abouhawwash [35] extended this paper to multiobjective optimization. The computational complexity of these approaches remained an obstacle toward using KKTPM efficiently in optimization, as each single KKTPM calculation required solving a quadratic programming problem to get the corresponding Lagrange multipliers. In order to overcome this obstacle, the authors proposed *approximate* KKTPM which is still reliable yet much faster [36]. In this paper, we use *approximate* KKTPM to identify weakly converged solutions, then try to help them improve.

In a precursor study [37], we proposed a preliminary version of our multiphased algorithm. Here, we extensively modify and extend the algorithm as follows. In [37], too much resources were wasted trying to fill *uncoverable* gaps in problems having discontinuous Pareto optimal fronts. Our modified algorithm, B-NSGA-III, never *retries* to cover a gap unless a new promising starting point is found during evolution. B-NSGA-III uses a normalized version of the biased weighted sum (BWS) LS objective function proposed in [37]. This modification allows for using a fixed value for ϵ regardless of the dimensionality (number of objectives) of the problem in hand. We also observed that the way [37] picks a *possibly dominated* starting point for LS in *Phase-2* wastes solution evaluations (SEs). Here, such a starting point is restricted to be nondominated. B-NSGA-III also has the option of using numerical partial derivatives to calculate KKTPM in problems having nondifferential objectives and/or constraints. In addition, our simulations and results have been extended significantly to adequately test the proposed idea. Finally, B-NSGA-III is now available as an Open Source Software for researchers willing do further investigate this path.[1]

II. Proposed B-NSGA-III

B-NSGA-III retains the general outline of U-NSGA-III [9]. Starting with a randomly generated initial population, B-NSGA-III generates an equal number of offspring individuals (solutions/points) using niche-based tournament selection, simulated binary crossover, and polynomial mutation [38]. The two populations are then combined and the ideal point is updated. The combined population goes through nondominated sorting [39] and the next population is formed by collecting individuals front by front starting at the first front. Since population size is fixed, the algorithm will typically reach a situation where the number of individuals needed to complete the next population is less than the number of individuals available in the front currently being considered. B-NSGA-III collects only as many individuals as it needs using a *niching* procedure. This niching procedure normalizes the objectives of all fronts considered. Then, using a fixed set of evenly distributed reference directions (in the normalized objective space) preference is given to those solutions representing the least represented reference directions in the objective space so far. Interested readers are encouraged to consult [9] for more details.

U-NSGA-III maintains a constant preference of convergence over diversity. A solution in front $n + 1$ will never be considered for inclusion in the next population unless all solutions in front n are already included. This convergence-always-first scheme has been recently criticized by several researchers [3], [40]. B-NSGA-III breaks this constant emphasis on convergence. Every α generations, the proposed algorithm changes its survival selection strategy, by favoring solutions solely representing some reference directions over other—possibly dominating—redundant solutions. A redundant solution is a solution that is not the best representative of its niche, i.e., there exists another solution—in the same front—that is closer to the reference direction representing their niche. Consequently, B-NSGA-III can lose a better ranked solution in favor of a solution in a later front, but this will only happen if the latter is the best in its own niche, while the former (better ranked solution) is outperformed in its own niche by an even better solution.

Another difference between U-NSGA-III and B-NSGA-III is that U-NSGA-III treats all individuals/regions of the search space equally at all generations. And although it might seem better from a generic point of view, we claim the opposite. One of the most important resources in optimization is the number of function evaluations (FEs) consumed to reach a solution. In a multiobjective optimization scenario, we use the term SE instead, as evaluating a single solution involves evaluating more than one function. Being fair the way U-NSGA-III is, can lead to wasting SEs on easy sections of the Pareto front. Those wasted SEs could have been put into better use, if they were directed toward reaching more difficult sections of the front. Obviously, in order to achieve the maximum possible utilization of SEs, a truly dynamic algorithm that gives more attention to more difficult sections/points of the front is needed. However, designing such an algorithm needs an oracle that knows deterministically the difficulty of attaining each point on the front relative to others. Unfortunately, for an arbitrary optimization problem, perfecting such an oracle is a far-fetched dream so far, despite some studies [41]. However, recent studies show some clues that can drive creating a nondeterministic version of the targeted oracle. We summarize these clues in the following points.

[1] https://www.coin-laboratory.com/evolib

1) Researchers have repeatedly shown the important role normalization plays especially in achieving better coverage of the Pareto front. Usually, the extreme points of the current population dictate normalization parameters. During evolution, as new extreme points appear all previously normalized objective values become outdated, and normalization is repeated. Hence, the importance of extreme points in optimization. And as pointed in [42], the earlier we reach extreme points, the better normalization we have and the better coverage we attain.
2) Reaching some parts of the Pareto front may require more effort than others. Several test as well as real-world problems exhibit such behavior [43], [44]. Usually such difficult regions appear as gaps in the first front. In reference directions-based optimization algorithms (like MOEA/D , NSGA-III, and U-NSGA-III), gaps can be identified by looking for reference directions having no associations so far.
3) In multiobjective optimization, all nondominated solutions are considered equally good, thus deserving equal attention. This is not ideal though. Being nondominated with respect to each other does not mean that two solutions are equally converged. The recently published approximate KKTPM now enables us to efficiently differentiate nondominated solutions based on their proximity from local optima.

These clues are realized in B-NSGA-III through several phases. In α generations, the algorithm switches back and forth among three different phases. Each phase uses one some LS operator to fulfill its goal. The following section discusses these phases in greater detail.

A. Alternating Phases

One naive approach is to use sequential phases. Given their relative importance the first phase may seek extreme points. Once found, the algorithm moves to subsequent phases and never looks back. But, as shown in [42], reaching true extreme points is not a trivial task. Even using LS, several optimizations might be needed to attain one extreme point. And since we can never safely assume that we have reached the true extreme points, this sequential design is not recommended. The same argument is valid for covering gaps. In an earlier generation, although your solutions may not provide the desired spread, it might be the case that there are no gaps within the small region they cover. As generations proceed and solutions expand, gaps may appear. This is a frequent pattern, that is, likely to repeat through an optimization run. Again, the sequential pattern is prone to failure, as we can never know if more gaps will appear in the future.

Another more involved yet simple approach is to move from one phase to the next after a fixed number of generations (or SEs). Once, the algorithm reaches the final phase it goes back to the first cyclically. This cyclic approach is more appealing, but how many generations (or SEs) to wait before switching from one phase to the next? Obviously, it is never easy to tell. In addition, using this rigid design obligates the algorithm to spend resources (SEs) in *possibly unnecessary* phases, just because it is their turn in the alternation cycle.

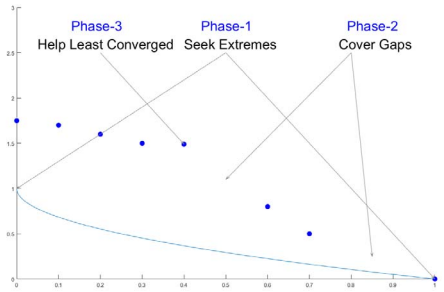

Fig. 1. *Phase-1*, *Phase-2*, and *Phase-3* in action.

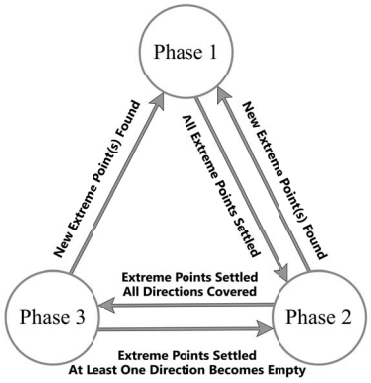

Fig. 2. Alternation of phases.

B-NSGA-III alternates among three phases *dynamically* and *adaptively*. Fig. 1 shows the three phases in action. During *Phase-1*, the algorithm seeks extreme points. *Phase-2* is where an attempt is made to cover gaps found in the nondominated front. Finally, during *Phase-3* the focus is shifted toward helping poorly converged nondominated solutions. In order to avoid the shortcomings of the two aforementioned approaches, B-NSGA-III watches for specific incidents that trigger transitions from one phase to another. Those transitions are completely unrestricted, i.e., B-NSGA-III can move from any phase to the other if the appropriate trigger is observed. Fig. 2 shows all possible transitions along with their triggers.

In the first α generation, the algorithm puts itself in *Phase-1*. Algorithm 1 shows the details of this phase. For an M objectives problem, *Phase-1* uses an Extreme-LS operator (discussed later) to search for its M extreme points. If all extreme points remain unchanged from one α generation to the next, B-NSGA-III assumes *temporarily* that these settled points are the true extreme points, and moves to *Phase-2*. While being in phases 2 or 3, finding a better extreme point through evolution indicates that those extreme points previously settled are not the true ones. Consequently, B-NSGA-III returns to *Phase-1* in search for better extreme points again.

As mentioned earlier, B-NSGA-III gives a chance to possibly dominated solutions that solely represent their niche, every α generations. This is shown in Algorithm 2, line 3 and expanded in Algorithm 3. The points surrounding each

Algorithm 1 Phase 1

Input: parent population (P), offspring (O) population size (N), reference directions (D), ideal point (I), intercepts (T), maximum number of function evaluations ($FeMax$), maximum number of local search operations per iteration β, augmentation factor ϵ

Output: None

1: $All \leftarrow P \vee O$
2: $E \leftarrow getExtremePoints(All)$
3: **for** i = 1 to M **do**
4: $E_i \leftarrow localSearch_{BWS}(E_i, I, T, FeMax, \epsilon)$, $i = 1, \ldots, M$
5: $O(randomIndex) \leftarrow E_i$, $1 \leq randomIndex \leq |O|$
6: $i \leftarrow i + 1$
7: **end for**

Algorithm 2 Phases 2 and 3 in a B-NSGA-III Generation

Input: parents (P), offspring (O), number of objectives (M), population size (N), reference directions (D), ideal point (I), intercepts (T), maximum number of function evaluations ($FeMax$), maximum number of local search operations per iteration β, last point used to cover direction d ($prev_d$) for all d in D

Output: New Population ($\hat{P}$)

1: $F \leftarrow nonDominatedSorting(All)$
2: $All \leftarrow P \vee O$
3: $\hat{P} \leftarrow getBestWithinNiche(d, O)\ \forall d \in D$
4: **if** $stagnant(E)$ **then**
5: $D_{empty} \leftarrow \{d \in D \mid (\nexists x)[x \in F_1 \wedge x \notin d_{surroundings}]\}$
6: $s_{kktpm} \leftarrow calculateKKTPM(s) \quad \forall s \in \hat{P}$
7: **for** i = 1 to β **do**
8: **if** $D_{empty} \neq \phi$ **then** {Phase-2}
9: $d \leftarrow randomPick(D_{empty})$
10: $s \leftarrow \{x \in F_1 \mid (\nexists y)[y \in F_1 \wedge \perp_d (y) \geq \perp_d (x)]\}$
11: **if** $prev_d = null$ or $\perp_d (s) < \perp_d (prev_d)$ **then**
12: $prev_d \leftarrow s$
13: **else**
14: $s \leftarrow null$
15: $D_{empty} \leftarrow D_{empty} \setminus \{d\}$
16: **end if**
17: **else**{Phase-3}
18: $s \leftarrow \{x \in \hat{P} \mid (\nexists y)[y \in \hat{P} \wedge y_{kktpm} \geq x_{kktpm}]\}$
19: **end if**
20: **if** $s \neq null$ **then**
21: $\hat{s} \leftarrow localSearch_{ASF}(s, I, T, FeMax)$
22: $\hat{P} \leftarrow \hat{P} \vee \{\hat{s}\}$
23: $\beta \leftarrow \beta + 1$
24: **end if**
25: **end for**
26: **end if**

Algorithm 3 getBestWithinNiche(...)

Input: merged parents and offspring (All), reference directions (D)

Output: selected Individuals (one from each niche) ($\hat{P}$)

1: $\hat{P} \leftarrow \phi$
2: **for all** $d \in D$ **do**
3: $S \leftarrow getSurroundings(d, All)$
4: $X_d \leftarrow \{x \in S \mid (\nexists y)[y \in S \ \wedge y_{rank} \geq x_{rank}]\}$
5: $x_d \leftarrow \{x \in X_d \mid (\nexists y)[y \in S \wedge \perp_d (y) \geq \perp_d (x)]\}$
6: $\hat{P} \leftarrow \hat{P} \cup \{x_d\}$
7: **end for**

Algorithm 4 fillUpPop(...)

Input: merged parents and offspring (All), population size (N), partially full new population ($\hat{P}$)

Output: completely full new population ($\hat{P}$)

1: $All \leftarrow All \setminus \hat{P}$
2: **while** $|\hat{P}| < N$ **do**
3: $Z \leftarrow \{x \in All \mid (\nexists y)[y \in All \wedge y_{rank} \geq x_{rank}]\}$
4: $z \leftarrow pickRandom(Z)$
5: $All \leftarrow All \setminus \{z\}$
6: $\hat{P} \leftarrow \hat{P} \vee \{z\}$
7: **end while**

nonempty reference direction are collected from the merged population (parents and offspring) (line 3), and the best ranked point is selected to represent this direction/niche (line 4). If more than one point share the same rank, the point closest to the direction is selected (line 5). Obviously, as opposed to U-NSGA-III , points in B-NSGA-III compete only with their niche peers, which means that an inferiorly ranked point from one niche, can be included because it is the best representative of its niche, while a superior point from another niche is left out because a better representative if its niche exists.

Once in *Phase-2*, B-NSGA-III looks for reference directions having no associations in the first front (*empty directions*). These directions represent gaps in the nondominated front (Algorithm 2, line 5). If several such directions exist, one is picked randomly (line 9) and the closest first front point to this direction is saved (line 10) to be used later as a starting point in LS. Notice that lines 11–16 ensures that B-NSGA-III will not retry to cover a gap until a closer starting point than the one previously used is found. If no empty directions exist, B-NSGA-III moves to *Phase-3*, looking for the least converged point among those selected so far. This point should have the highest KKTPM among all (line 18). An ASF LS operator (discussed later) is employed in both cases (line 21), either to cover a gap (*Phase-2*) or to bring a poorly converged point closer to the front (*Phase-3*). In order to keep SEs as low as possible, a maximum of β LS operations are allowed, even if the number of gaps is more than β. Notice the ability of the algorithm to move directly from *Phase-1* to *Phase-3* if no gaps are found.

It is worth noting that phases 2 and 3 may run simultaneously. If the number of empty directions (gaps) is less than β, B-NSGA-III moves to *Phase-3* and uses the remaining budget to help poorly converged solutions. Obviously, *Phase-1* has the highest priority followed by *Phase-2* then *Phase-3*. The

following two sections discuss both Extreme-LS and ASF-LS operators in detail.

Finally, since all the three phases are not guaranteed to completely fill the next population, a final pass is made to fill up the next population using points that B-NSGA-III has discarded so far. Algorithm 4 shows that the best ranked points—out of those not included yet in the next population—are given higher priority.

B. Two Local Search Operators

As mentioned earlier, B-NSGA-III uses two different LS operators. In each, all the objectives are combined into some aggregate function (scalarization). Any single objective optimizer can be used to minimize these aggregate functions. Here, we chose to use MATLAB's `fmincon()` optimization routine, a point-to-point deterministic optimizer. Point-to-point optimizers use less FEs compared to set-based methods (e.g., evolutionary algorithms). But, they are also less guaranteed to reach global optima. Yet, in an alternating multiphased algorithm like B-NSGA-III the embedded single objective optimizer is not expected to reach the global optimum in one shot. That is why `fmincon()` fits our criteria for an embedded single objective optimizer. Earlier we discussed the role of our two LS operators. Next, we discuss their formulations and how they fit into their designated roles.

1) Extreme-LS: Phase-1 uses Extreme-LS to find extreme points. This operator is formulated simply as a BWS aggregate function of all objectives (1). $\tilde{f}_k(x)$ represents the normalized value of objective k. When seeking the ith extreme point, the term *biased* refers to the significantly smaller weight (we call it *augmentation factor*) multiplied by the ith objective, compared to the weights of all other objectives. Although, weighted sum aggregate functions are straightforward and easy to implement, they (including ours) can only reach points lying on convex sections of the Pareto front. While, this makes them less plausible as a generic formulation, they perfectly serve their purpose in B-NSGA-III, since extreme points by definition can never lie in a nonconvex section of the Pareto front. Unlike [37], the operator we are proposing here is normalized based on the number of objectives. This allows using the same *augmentation factor* for different dimensions. It is important to note that adding the ith objective term to the formula helps avoiding weakly dominated points

$$\underset{\mathbf{x}}{\text{Minimize}} \;\; \text{BWS}_{\text{i}}(\mathbf{x}) = \epsilon\tilde{f}_i(x) + \sum_{j=1,\; j\neq i}^{M} \frac{w_j\tilde{f}_j(x)}{M-1} \tag{1}$$

where ϵ is set as one percent of $\min_{j=1, j\neq i}^{M} w_j$.

2) Achievement Scalarization Function LS: As mentioned earlier, a generic LS operator that is required to get an arbitrary Pareto point cannot rely on BWS. That is why we use ASF to formulate our second LS operator, ASF-LS. The formulation in (2) shows that ASF-LS targets the intersection between the provided direction and the Pareto front. Since, B-NSGA-III is a reference direction-based algorithm, ASF-LS can follow these already existing directions. And because of its ability to reach points lying on both convex and nonconvex sections of the Pareto front, this is the operator employed in both *Phase-2* and *Phase-3* of B-NSGA-III. It is worth noting that according to our earlier experiments, ASF-LS does not perform as well if used to find extreme points. This can be attributed to the steep gradient of the aggregate ASF function (at these points) on one side of the global optimum, which usually misleads the single objective optimizer. Hence, we need both operators in B-NSGA-III, each playing its designated role

$$\begin{aligned} &\underset{\mathbf{x}}{\text{Minimize}} \;\; \text{ASF}\big(\mathbf{x}, \mathbf{z}^r, \mathbf{w}\big) = \max_{i=1}^{M}\left(\frac{\tilde{f}_i(x) - u_i}{w_i}\right) \\ &\text{subject to} \;\; g_j(\mathbf{x}) \leq 0, \quad j = 1, 2, \ldots, J. \end{aligned} \tag{2}$$

III. Computational Complexity

As a multiphase algorithm, the complexity of a single generation in B-NSGA-III depends on the phase in effect at the time of this generation. A detailed discussion of the computational complexity of B-NSGA-III is provided in the supplementary material. In summary, assuming that the computational complexity of a single LS operation is $O(X)$, the number of inequality constraints is G and the number of variables is n, the complexity of B-NSGA-III is bounded by either $O(N^2 \log^{M-2} N)$ or $O(MX)$ or $O(\beta X)$ or $O(NG^{1+\psi})$ or $O(Nn^{1+\psi})$, whichever is higher, where ψ can take several specific values in the range [2, 3] depending on the LU decomposition algorithm used in KKTPM calculations.

IV. Results

The set of problems used here are carefully selected and/or modified to exhibit different types of difficulties. Both unconstrained and constrained problems are taken into account. Some of them have disconnected Pareto fronts. Others, challenge the convergence ability of the multiobjective algorithm, while others test its ability to cover the entire front evenly (diversity). Some of these problems are differentiable while others are not. We even modified some problems to exhibit certain types of behavior, different from what they were originally designed for. Our problems cover a wide range of dimensionality as well, namely 2, 3, 5, and 10 objectives. B-NSGA-III is compared to two other state-of-the-art reference direction-based algorithms, U-NSGA-III and MOEA/D. Since the original study [3] did not mention a specific way to handle constraints, we use MOEA/D with unconstrained problems only. U-NSGA-III on the other hand is applied to all problems. Population size is kept at a small value to make a strict test of the algorithms. We use inverted generational distance (IGD) and generational distance (GD) to test both overall performance and convergence, respectively. hypervolume (HV) results are also provided in the supplementary material along with their statistical significance probabilities. Sometimes, we also show how KKTPM progresses during optimization. All results presented here are the medians of 31 independent runs of each algorithm on each problem. Table I shows the parameters used by B-NSGA-III in each problem. The other two algorithms use the same values for common parameters. More in depth discussion about our experimental design can be found in the supplementary material. Before

TABLE I
PARAMETERS USED BY B-NSGA-III. M IS THE NUMBER OF OBJECTIVES. N IS POPULATION SIZE. SEs STANDS FOR SOLUTION EVALUATIONS. α IS THE FREQUENCY EXPLAINED IN SECTION II WHILE β IS THE MAXIMUM NUMBER OF LS OPERATIONS PER GENERATION. FE_{MAX} IS THE MAXIMUM LIMIT OF FEs THE SINGLE OBJECTIVE OPTIMIZER (FMINCON()) CAN USE. THE FINAL COLUMN SHOWS THE VALUES OF ϵ IN (1)

Problem	M	N	SEs	α	β	FE_{max}	ϵ
ZDT3	2	72	8000	10	2	200	0.1
ZDT4	2	48	30000	10	2	200	0.1
ZDT6	2	48	5000	10	2	200	0.1
OSY	2	48	15000	10	2	200	0.1
TNK	2	24	5000	10	2	200	0.1
DTLZ4	3	36	7000	10	2	200	0.1
DTLZ4	5	120	25000	10	2	200	0.1
DTLZ4	10	276	50000	10	2	200	0.1
DTLZ7	3	92	20000	10	2	200	0.1
UF1	2	200	50000	10	2	200	0.1
UF2	2	200	50000	10	2	200	0.1
WFG1	3	92	20000	10	2	200	0.1

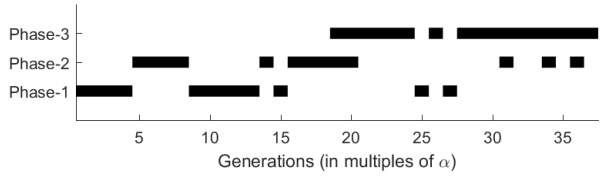

Fig. 3. Typical alternation of phases of B-NSGA-III while solving ZDT4.

going through the details of our simulations results, we emphasize the alternating nature of B-NSGA-III by showing it in action. Fig. 3 shows a typical alternation of phases performed by B-NSGA-III while solving ZDT4 [44].

A. Multiobjective Problems

We start our experiments by solving three unconstrained bi-objective problems, ZDT3, ZDT4, and ZDT6 [43] to test the ability of our algorithm to deal with disconnected Pareto fronts, local Pareto fronts, and variable density objective spaces, respectively. As shown in Supplementary Figs. S1-a, S1-b, and S1-c, for ZDT3, B-NSGA-III, and U-NSGA-III achieve better convergence compared to MOEA/D. In both ZDT4 and ZDT6, B-NSGA-III is a clear winner in terms of both convergence and diversity as shown in Fig. 4 and Supplementary Figs. S2-a, S2-b, and S2-c.

The same outcome can be observed with constrained problems, see Fig. 5 and Supplementary Figs. S3-a, S3-b, and S3-c. For these two problems, TNK and OSY [2], we exclude MOEA/D since the original study did not provide a specific way to handle constraints. Notice how B-NSGA-III was able to cover the entire Pareto front providing a nice distribution in Fig. 4(c), Supplementary Fig. S2-c, and Fig. 5(c) using a limited number of FEs (see Table I). Both, U-NSGA-III and MOEA/D need more SEs to catch up with B-NSGA-III, if they ever do.

The only exceptions to the superiority of B-NSGA-III are with UF1 and UF2, where MOEA/D significantly outperforms both B-NSGA-III and U-NSGA-III. See Supplementary Figs. S4-a, S4-b, S4-c, S5-a, S5-b, and S5-c. Although the three algorithms struggle and none of them is able to fully cover the Pareto front, MOEA/D achieves a better spread on average. This can be attributed to the restricted (neighborhood) mating nature of MOEA/D which gives the algorithm a better chance to follow the very unusual paths in the variable space leading to the true Pareto front in these two problems. This also motivates us to explore a multiphase version of MOEA/D for similar problems in the future.

Looking at KKTPM yields another perspective that can confirm the convergence edge B-NSGA-III has in most problems. Supplementary Figs. S6-a, S6-b, and S6-c show how the median KKTPM of the nondominated solutions progresses throughout optimization in ZDT4, ZDT6, and OSY, respectively. Obviously, B-NSGA-III can reach Pareto optimal points (either local or global) much faster than U-NSGA-III and MOEA/D. Combined with the previous GD plots, the reader can easily conclude that B-NSGA-III hits global Pareto optimal points much earlier than U-NSGA-III and MOEA/D. Fluctuations can be seen in initial generations (see Supplementary Fig. S6-b) because of the rapidly changing number of nondominated solutions at early generations.

For three objectives, we use DTLZ4 and DTLZ7 problems [44]. DTLZ4 tests the ability of an optimization algorithm to diversify solutions in a variable density objective space. A randomly generated set of Pareto optimal solutions will be dense near the f_M-f_1 plane, and as you move away from it, solutions get sparser. Degree of density/sparseness can be controlled via the parameter γ (called α in the original study and changed here not to be confused with our α parameter shown in Table I). However, DTLZ4 does not test the ability of an algorithm to converge. Actually, all randomly generated solutions are relatively close to the Pareto front (compared to other problems from the same family, like DTLZ1). And since B-NSGA-III is designed to tackle both convergence and diversity simultaneously, we need a problem that is able to test both capabilities at the same time. We can get such a problem by multiplying the $g(x)$ function of DTLZ4 by a constant factor D. The larger D is, the more distant randomly generated solutions will be from the Pareto front. In this paper, we use $\gamma = 20$ and $D = 100$. On the other hand, the Pareto front of DTLZ7 has four disconnected regions. One region is relatively easier than the others. An optimization algorithm can easily get attracted to the easily attainable region and ignore some/all others.

Fig. 6(a)–(c) shows the overall performance of B-NSGA-III compared to U-NSGA-III and MOEA/D on DTLZ4. B-NSGA-III is better in terms of both overall performance and convergence. Median Pareto fronts (those having median IGD values) shown in Fig. 6(d) and Supplementary Figs. S7-a and S7-b confirm the overall superiority of B-NSGA-III and U-NSGA-III over MOEA/D, especially in terms of diversity. The superiority of B-NSGA-III is even more evident on DTLZ7 as show in Fig. 7 and Supplementary Figs. S8-a and S8-b.

B. Many-Objective Optimization

A many-objective optimization problem, is a problem having more than three objectives. Both MOEA/D and

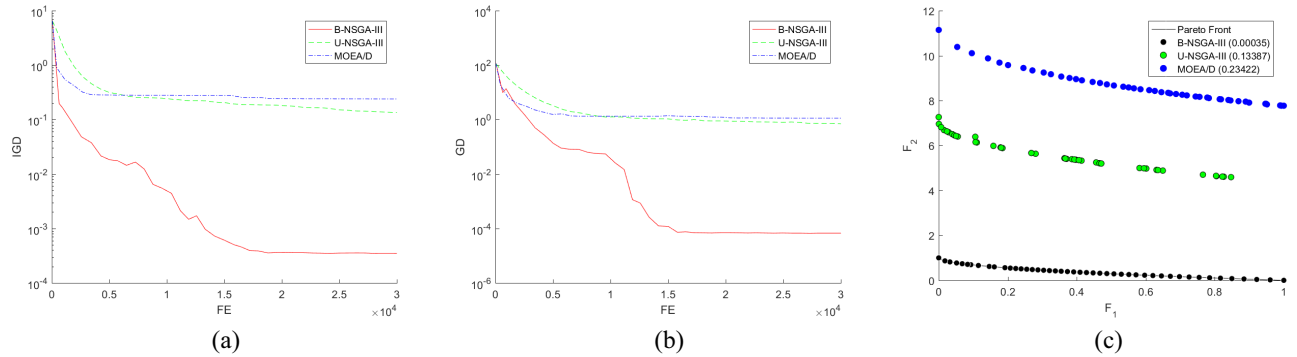

Fig. 4. Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective ZDT4. (a) IGD comparison on ZDT4. (b) GD comparison on ZDT4. (c) Fronts comparison on ZDT4.

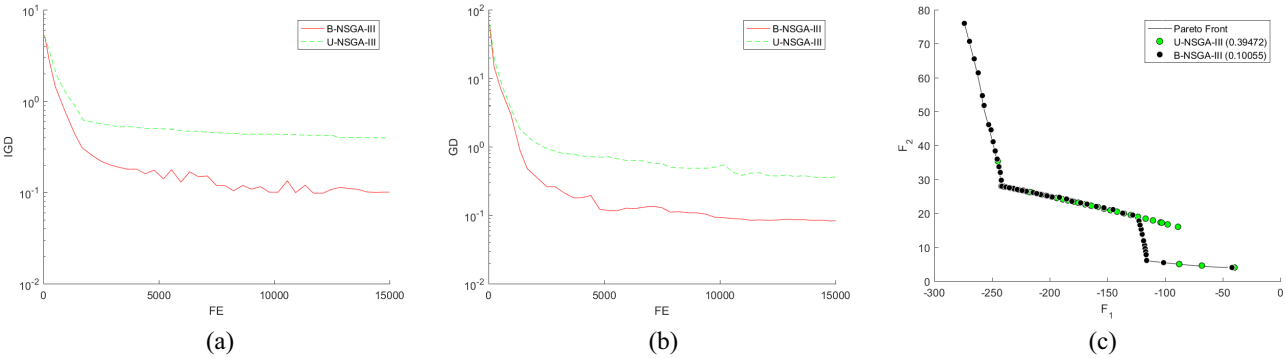

Fig. 5. Performance of B-NSGA-III, U-NSGA-III, and MOEA/D on bi-objective OSY. (a) IGD comparison on OSY. (b) GD comparison on OSY. (c) Fronts comparison on OSY.

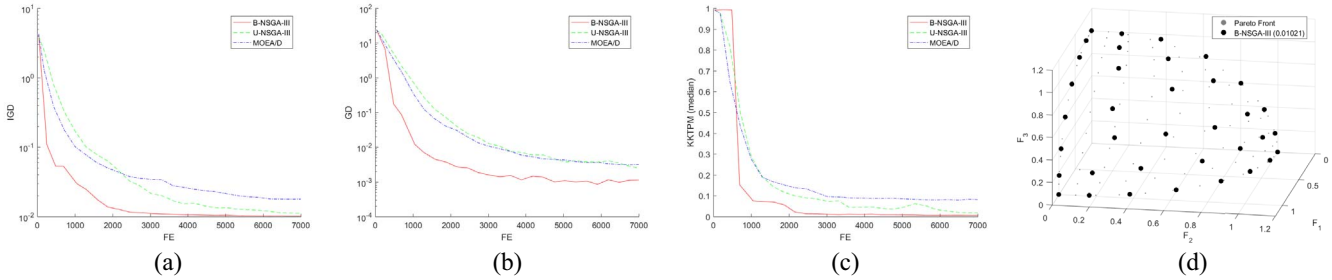

Fig. 6. Performance of B-NSGA-III, U-NSGA-III, and MOEA/D and median final fronts of U-NSGA-III on DTLZ4 (3 objectives). (a) IGD comparison on DTLZ4 (3 obj.). (b) GD comparison on DTLZ4 (3 obj.). (c) KKTPM comparison on DTLZ4 (3 obj.). (d) B-NSGA-III Median front.

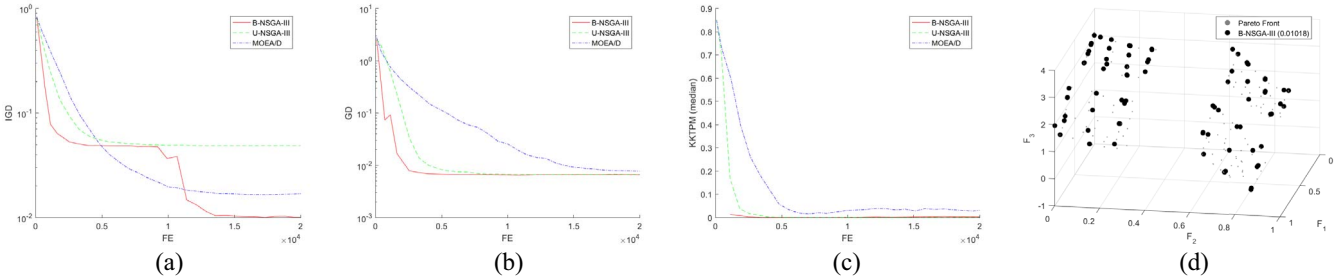

Fig. 7. Performance of B-NSGA-III, U-NSGA-III, and MOEA/D and median final fronts of U-NSGA-III on DTLZ7 (3 objectives). (a) IGD comparison on DTLZ7 (3 obj.). (b) GD comparison on DTLZ7 (3 obj.). (c) KKTPM comparison on DTLZ7 (3 obj.). (d) B-NSGA-III Median front.

U-NSGA-III are known to be very efficient with this category of problems. In this section, we consider a difficult problem (our modified version of DTLZ4) and compare the performance of B-NSGA-III to these two powerful algorithms. Here, we use two instances of the problem, one with 5 objectives and the other with 10. For the 5 objectives

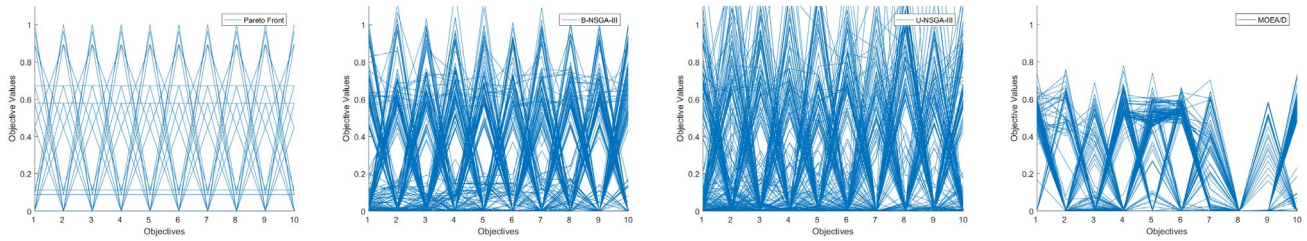

Fig. 8. PCPs of B-NSGA-III, U-NSGA-III, and MOEA/D on DTLZ4 (10 objectives). (a) Reference PCP (True PF). (b) B-NSGA-III. (c) U-NSGA-III. (d) MOEA/D.

instance, Supplementary Fig. S9-a is a parallel coordinate plot (PCP) showing the 25%, median and 75% quantile values of each objective among the three algorithms. The values plotted for each algorithm are taken from the final population of the run having the median IGD value of its 31 independent runs. Unmarked lines represent the benchmark quantile values of the selected Pareto optimal set. Lines marked with circles, squares, and triangles represent B-NSGA-III, U-NSGA-III, and MOEA/D respectively. The closer the lines representing one algorithm to the benchmark lines, the better this algorithm performs. All three algorithms successfully attain the 25% quantile values. However, median and 75% quantile results vary. Obviously, B-NSGA-III achieves the best approximation of the two benchmark lines among the three algorithms. U-NSGA-III is still acceptable while MOEA/D misses the target lines completely. In addition, IGD and GD median comparisons are shown in Supplementary Figs. S9-b and S9-c, respectively. Obviously, B-NSGA-III outperforms the other two algorithms in terms of overall performance and convergence.

The same conclusions can be drawn for the ten objectives problem instance. Since most of the Pareto points are on the edges of the hyper simplex, the median of each objective is the same as the minimum (zero). Supplementary Fig. S10-a shows that unlike both U-NSGA-III and MOEA/D, B-NSGA-III achieves a close approximation of the benchmark lines. Those results are supported by IGD and GD results in Supplementary Figs. S10-b and S10-c, respectively. In order to further confirm our results, we have included the PCPs of all nondominated solutions of the median run of each algorithm in Fig. 8. Obviously, the PCP of B-NSGA-III in Fig. 8(b) is closer to the benchmark PCP [Fig. 8(a)] than both the PCPs of U-NSGA-III and MOEA/D.

C. Using Numerical Gradients

Up to this point, our approach has been using exact analytical gradients—provided by the user—to calculate KKTPM. However, for many problems—especially real-world problems—these gradients may not be available in their analytical form. In such cases we resort to using numerical gradients. In most cases numerical gradients yield acceptable gradient approximations, but this comes on the expense of consuming more SEs. In order to investigate the effect of using numerical gradients on our approach, first we use numerical gradients with a problem for which we know the exact analytical gradients, namely OSY. Then we apply the same approach to WFG1 [45], a problem with nondifferentiable objective functions. Supplementary Fig. S11 clearly shows how using numerical gradients has a minor negative effect on the performance of B-NSGA-III. As the reader can observe, B-NSGA-III using numerical gradients still outperforms U-NSGA-III. In addition, although using numerical gradients consumes more SEs, it is clear from both figures that the magnitude of sacrifice is minimal. B-NSGA-III with numerical gradients even catches up with the exact gradients version in some cases.

The reader can easily reach a similar conclusion by looking at Supplementary Figs. S12 and S13. Here, we use a slightly modified version of WFG1. We noticed that in order to reach about 100 Pareto front points in the original version of WFG1, we can only use a code whose precision goes beyond $\approx 10^{-50}$! This is true because of the very small power used at the third transformation of WFG1 (0.02). To the best of our knowledge, most of the codes available are not even close to this level of precision. In order to solve this problem we changed the power of the third transformation from 0.02 to 0.2.

Since, all the objectives of WFG1 are nondifferentiable, the only possible option for B-NSGA-III is to use numerical gradients. Even with the burden of additional SEs, B-NSGA-III (with numerical gradients) still outperforms both MOEA/D and U-NSGA-III. It is worth noting that, although Supplementary Figs. S12-a and S13-a look similar, the points obtained by B-NSGA-III are more converged (closer to the front) than those obtained by U-NSGA-III. This difference in convergence can be visually observed through the profile views of the same two plots (not included due to space limitations). The good performance of numerical gradient based B-NSGA-III can be attributed to the very conservative approach B-NSGA-III follows with KKTPM calculations, which is summarized in the following points.

1) Since our modified niching approach is applied every α generations, KKTPM calculations are only possible in $1/\alpha$ of the total number of generations. Typically, in our case—since $\alpha = 10$—KKTPM values may be calculated in only 10% of all generations at max.
2) Even when it comes to these 10% of generations, *Phases-I* and *II* do not require any KKTPM

calculations. And as we showed earlier, in most problems B-NSGA-III spends the majority of its time in these two phases.

3) *Phase-III* is where KKTPM calculations are required. In this phase, KKTPM values are calculated only for those individuals selected so far, which is usually less than N (population size).
4) B-NSGA-III uses forward difference in all numerical gradient calculations.

Median SE budget consumed by numerical gradient calculations in OSY and WFG1 are <43% and <28%, respectively, which shows how this conservative approach prevents numerical gradient calculations from dominating the whole process.

V. Conclusion

In this paper, we have proposed B-NSGA-III , a multiphased many-objective evolutionary optimization algorithm capable of automatically balancing convergence and diversity of population members. B-NSGA-III does not use explicit preferential parameters or weights to distribute its effort among the two aspects. It rather waits for signals based on which it changes from one phase to another. This alternation of phases is completely unrestricted and the exact way it happens adapts automatically to the problem in hand. Two types of LS operators are used during optimization. The first is designed to find extreme points while the second is more suited to move arbitrary points toward the Pareto optimal front. Approximate KKTPM is used to identify weakly dominated points that need to be locally enhanced. On a wide range of problems with different attributes and difficulties, B-NSGA-III shows superior performance to a number of state-of-the-art algorithms for an equal number of SEs. B-NSGA-III should be viewed as one among many possible cooperative structures of different entities. Different extensions can be proposed through either substituting one/some of these entities and/or changing their cooperative structure. One possible extension is to substitute our ASF-based LS with a quasi-Newton optimizer, motivated by the potential such optimizers exhibits elsewhere [46].We argue that our open source implementation of B-NSGA-III can serve as a starting point for researchers willing to further exploit these possibilities toward balanced many-objective optimization.

References

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

[2] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, vol. 16. Chichester, U.K.: Wiley, 2001.

[3] 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.

[4] K. Deb and T. Goel, "Controlled elitist non-dominated sorting genetic algorithms for better convergence," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, 2001, pp. 67–81.

[5] P. A. N. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 174–188, Apr. 2003.

[6] K. C. Tan, S. C. Chiam, A. A. Mamun, and C. K. Goh, "Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization," *Eur. J. Oper. Res.*, vol. 197, no. 2, pp. 701–713, 2009.

[7] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[8] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 602–622, Aug. 2014.

[9] H. Seada and K. Deb, "A unified evolutionary optimization procedure for single, multiple, and many objectives," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 358–369, Jun. 2016.

[10] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 6, pp. 909–923, Dec. 2014.

[11] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *Amer. Math. Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[12] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2009, pp. 203–208.

[13] Z. Wang, Q. Zhang, M. Gong, and A. Zhou, "A replacement strategy for balancing convergence and diversity in MOEA/D," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2014, pp. 2132–2139.

[14] Y. Yuan, H. Xu, B. Wang, B. Zhang, and X. Yao, "Balancing convergence and diversity in decomposition-based many-objective optimizers," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 180–198, Apr. 2016.

[15] Y. Yuan, H. Xu, and B. Wang, "Evolutionary many-objective optimization using ensemble fitness ranking," in *Proc. Annu. Conf. Genet. Evol. Comput.*, 2014, pp. 669–676.

[16] M. Emmerich, N. Beume, and B. Naujoks, "An EMO algorithm using the hypervolume measure as selection criterion," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, 2005, pp. 62–76.

[17] B. Naujoks, N. Beume, and M. Emmerich, "Multi-objective optimisation using S-metric selection: Application to three-dimensional solution spaces," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 2, 2005, pp. 1282–1289.

[18] C. A. R. Villalobos and C. A. Coello Coello, "A new multi-objective evolutionary algorithm based on a performance assessment indicator," in *Proc. Annu. Conf. Genet. Evol. Comput.*, 2012, pp. 505–512.

[19] G. Rudolph, O. Schütze, C. Grimme, C. Domínguez-Medina, and H. Trautmann, "Optimal averaged Hausdorff archives for bi-objective problems: Theoretical and numerical results," *Comput. Optim. Appl.*, vol. 64, no. 2, pp. 589–618, 2016.

[20] O. Schütze *et al.*, "A scalar optimization approach for averaged Hausdorff approximations of the Pareto front," *Eng. Optim.*, vol. 48, no. 9, pp. 1593–1617, 2016.

[21] O. Schutze, X. Esquivel, A. Lara, and C. A. Coello Coello, "Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 4, pp. 504–522, Aug. 2012.

[22] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 28, no. 3, pp. 392–403, Aug. 1998.

[23] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 204–223, Apr. 2003.

[24] H. Ishibuchi and K. Narukawa, "Performance evaluation of simple multiobjective genetic local search algorithms on multiobjective 0/1 knapsack problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 1, 2004, pp. 441–448.

[25] J. D. Knowles and D. W. Corne, "M-PAES: A memetic algorithm for multiobjective optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 1, 2000, pp. 325–332.

[26] K. Harada, J. Sakuma, and S. Kobayashi, "Local search for multiobjective function optimization: Pareto descent method," in *Proc. 8th Annu. Conf. Genet. Evol. Comput.*, 2006, pp. 659–666.

[27] P. A. N. Bosman, "On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 1, pp. 51–69, Feb. 2012.

[28] O. Schütze *et al.*, "The directed search method for multi-objective memetic algorithms," *Comput. Optim. Appl.*, vol. 63, no. 2, pp. 305–332, 2016.

[29] A. P. Wierzbicki, "The use of reference objectives in multiobjective optimization," in *Multiple Criteria Decision Making Theory and Application*. Heidelberg, Germany: Springer, 1980, pp. 468–486.
[30] H.-C. Wu, "The Karush–Kuhn–Tucker optimality conditions in an optimization problem with interval-valued objective function," *Eur. J. Oper. Res.*, vol. 176, no. 1, pp. 46–59, 2007.
[31] T. Antczak, "A new approach to multiobjective programming with a modified objective function," *J. Glob. Optim.*, vol. 27, no. 4, pp. 485–495, 2003.
[32] G. Bigi and M. Castellani, "Second order optimality conditions for differentiable multiobjective problems," *RAIRO Oper. Res.*, vol. 34, no. 4, pp. 411–426, 2000.
[33] K. Miettinen, *Nonlinear Multiobjective Optimization*, vol. 12. New York, NY, USA: Springer, 2012.
[34] J. Dutta, K. Deb, R. Tulshyan, and R. Arora, "Approximate KKT points and a proximity measure for termination," *J. Glob. Optim.*, vol. 56, no. 4, pp. 1463–1499, 2013.
[35] K. Deb and M. Abouhawwash, "An optimality theory-based proximity measure for set-based multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 515–528, Aug. 2016.
[36] K. Deb, M. Abouhawwash, and H. Seada, "A computationally fast convergence measure and implementation for single-, multiple-, and many-objective optimization," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 1, no. 4, pp. 280–293, Aug. 2017.
[37] H. Seada, M. Abouhawwash, and K. Deb, "Towards a better balance of diversity and convergence in NSGA-III: First results," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, 2017, pp. 545–559.
[38] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.
[39] 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.
[40] M. Li, J. Zheng, K. Li, Q. Yuan, and R. Shen, "Enhancing diversity for average ranking method in evolutionary many-objective optimization," *Parallel Problem Solving From Nature, PPSN XI*. Heidelberg, Germany: Springer, 2010, pp. 647–656.
[41] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 450–455, Jun. 2014.
[42] A. K. M. K. A. Talukder, K. Deb, and S. Rahnamayan, "Injection of extreme points in evolutionary multiobjective optimization algorithms," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, 2017, pp. 590–605.
[43] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.
[44] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*. London, U.K.: Springer, 2005, pp. 105–145.
[45] S. Huband, L. Barone, L. While, and P. Hingston, "A scalable multiobjective test problem toolkit," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, 2005, pp. 280–295.
[46] Ž. Povalej, "Quasi-Newton's method for multiobjective optimization," *J. Comput. Appl. Math.*, vol. 255, pp. 765–777, Jan. 2014.

Haitham Seada received the master's degree in bioinformatics (major in computer science) from Zagazig University, Zagazig, Egypt, and the Ph.D. degree in evolutionary multiobjective optimization from Michigan State University, East Lansing, MI, USA.

He is a Data Scientist with Ford Motor Company, Dearborn, MI, USA. He was a Post-Doctoral Research Associate with the Department of Pathology and Laboratory Medicine, Brown University, Providence, RI, USA. He is also the author of several software libraries, both closed and open source, spanning a wide range of technologies. His current research interests include vehicle routing, nonlinear optimization, computational intelligence, analysis of high throughput biological images, and machine learning. More information about him can be found at www.haithamseada.com.

Mohamed Abouhawwash received the master's and Ph.D. degrees in statistics and computer science from Mansoura University, Mansoura, Egypt, in 2011 and 2015, respectively.

He is a Visiting Scholar with the Department of Mathematics and Statistics, Faculty of Science, Thompson Rivers University, Kamloops, BC, Canada. He is an Assistant Professor with the Department of Mathematics, Faculty of Science, Mansoura University. His current research interests include evolutionary algorithms and mathematical optimization.

Dr. Abouhawwash was a recipient of the best master's and Ph.D. thesis awards from Mansoura University in 2012 and 2018, respectively.

Kalyanmoy Deb (SM'02–F'12) received the master's and Ph.D. degrees from the University of Alabama, Tuscaloosa, AL, USA, in 1989 and 1991, respectively.

He is a Koenig Endowed Chair Professor with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI, USA. He has published over 490 research papers with Google Scholar Citations of over 116 000 with an H-index of 110. His current research interests include evolutionary optimization and their application in multicriterion optimization, modeling, and machine learning.

Dr. Deb was a recipient of the IEEE EC Pioneer Award, the Infosys Prize, the TWAS Prize in Engineering Sciences, the CajAstur Mamdani Prize, the Distinguished Alumni Award from IIT Kharagpur, the Edgeworth-Pareto Award, the Bhatnagar Prize in Engineering Sciences, and the Bessel Research Award from Germany. He is on the editorial board of 18 major international journals. He is a fellow of ASME and three Indian science and engineering academies.