Hindawi
Advances in Operations Research
Volume 2021, Article ID 6699277, 16 pages
https://doi.org/10.1155/2021/6699277

Research Article

Understanding Population Dynamics in Multi- and Many-Objective Evolutionary Algorithms for High-Resolution Approximations

Hugo Monzón Maldonado[1,2,3], **Hernán Aguirre**[1,2,3], **Sébastien Verel**[2,3,4], **Arnaud Liefooghe**[2,3,5], **Bilel Derbel**[2,3,5] and **Kiyoshi Tanaka**[1,2,3]

[1]*Faculty of Engineering, Shinshu University, Nagano, Japan*
[2]*MODŌ-International Associated Laboratory, Nagano, France*
[3]*MODŌ-International Associated Laboratory, Nagano, Japan*
[4]*Univ. Littoral Côte d'Opale, UR 4491, LISIC, F-62100 Calais, France*
[5]*University of Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France*

Correspondence should be addressed to Hernán Aguirre; ahernan@shinshu-u.ac.jp

Received 12 November 2020; Revised 30 September 2021; Accepted 16 October 2021; Published 30 December 2021

Academic Editor: Mhand Hifi

Achieving a high-resolution approximation and hitting the Pareto optimal set with some if not all members of the population is the goal for multi- and many-objective optimization problems, and more so in real-world applications where there is also the desire to extract knowledge about the problem from this set. The task requires not only to reach the Pareto optimal set but also to be able to continue discovering new solutions, even if the population is filled with them. Particularly in many-objective problems where the population may not be able to accommodate the full Pareto optimal set. In this work, our goal is to investigate some tools to understand the behavior of algorithms once they converge and how their population size and particularities of their selection mechanism aid or hinder their ability to keep finding optimal solutions. Through the use of features that look into the population composition during the search process, we will look into the algorithm's behavior and dynamics and extract some insights. Features are defined in terms of dominance status, membership to the Pareto optimal set, recentness of discovery, and replacement of optimal solutions. Complementing the study with features, we also look at the approximation through the accumulated number of Pareto optimal solutions found and its relationship to a common metric, the hypervolume. To generate the data for analysis, the chosen problem is MNK-landscapes with settings that make it easy to converge, enumerable for instances with 3 to 6 objectives. Studied algorithms were selected from representative multi- and many-objective optimization approaches such as Pareto dominance, relaxation of Pareto dominance, indicator-based, and decomposition.

1. Introduction

Multiobjective evolutionary algorithms (MOEAs) have been the choice to solve complex multiobjective optimization problems from a variety of domains [1, 2] and in cases where mathematical methods computational costs are high or the problem representation is not compatible (simulators). Their application mainly had focused on bi- and triobjective cases [3], with an increasing demand on real-world problems of four or more objectives, the so-called many-objective problems [4]. These had been attracting the attention of the research community [5–9] to develop algorithms aimed at them.

However, in high-dimensional spaces, difficulties arise by the curse of dimensionality, as shown by early studies [10–14]; algorithms' efficiency and effectiveness are impacted by the size of the objective space, demanding research in new algorithms, design components, and tuning procedures [15, 16], for which a deeper understanding is needed.

Better algorithms will not only fulfill the main goal of solving these many-objective problems but also ease an important secondary goal in real-world applications, which

is to extract knowledge about the problem from the obtained approximation of the Pareto optimal set. Such a task requires a high-resolution approximation, that is, it contains as many nondominated solutions as possible, with good convergence and diversity properties.

Most studies evaluate these properties on the final population, implicitly bounding the approximation obtained by the algorithm to its size; when for many-objective problems, a Pareto optimal set is expected to be much larger than the chosen population size. Even if some algorithms include a bounded archive to collect the approximation, their size can still be insufficient. Thus, to get the highest possible resolution approximation, it is common to compute it from all solutions visited by the algorithm. However, MOEAs are not designed to maximize such approximation, which again demands a better understanding of them to clarify the mechanism that can facilitate such a task.

To better illustrate this, let us assume a scenario where an algorithm is able to converge with its population to a subset of the Pareto optimal set. Improving the resolution of the approximation from this point implies that the algorithm somehow must continue finding other optimal solutions even if the population is already filled with them. For this, the algorithm has at its disposition (i) operators of variation to continuously explore solutions not seen before and (ii) its selection mechanism to preserve some optimal solutions for exploitation by the operators while also allowing their replacement to give room to the newly discovered ones. In addition, population size also plays a role since it determines the upper limit for the number of optimal solutions that an algorithm can keep simultaneously at a given time.

In this work, we aim to understand selection and population size effects on the algorithms' ability to improve the resolution of a well-converged approximation so that a high-resolution can be achieved, with the hope that this is a step towards a better understanding of them. Looking at some representative algorithms with different selection mechanisms from multi- and many-objective optimization, such as Pareto dominance, relaxation of Pareto dominance, indicator-based, and decomposition, under various population sizes and varying number of problem objectives. Only the operation of variation remains the same for all algorithms.

As the study focuses on the postconvergence state to the Pareto optimal set, which in many-objective problems can be an issue, we have chosen a test problem that can be tuned to present a relatively small search space with mild difficulty so the algorithms can hit the Pareto optimal set. MNK-landscape [12] allows us to tune not only the number of objectives but also variables and introduce random nonlinear correlations between them. In our experiments, we created problem instances with $M = 3 - 6$ objectives, $N = 20$ binary variables, and $K = 1$ epistasis that defines the correlation between them. Taking advantage of the small number of variables, we find by enumeration the true Pareto optimal set of these instances to compare against it the approximations obtained by the algorithms. This knowledge is used to investigate their behavior and performance on cases when our population size is large enough to contain the whole Pareto optimal set and when it can only contain a fraction of it.

Instead of only using quality indicators [17] to measure the algorithms' ability to reach better approximation sets, we will look into the dynamics of the search process through features on the population, to gain insights into how and why they work or fail. MOEAs are tracked over the generations, and by analyzing their change, we can look at the algorithms' dynamics. The features are the ones presented by Aguirre et al. [18], defined as generational assessment indices, they are independent of the selection mechanism used by the algorithm and can study their effects by looking at solutions in terms of dominance status, membership to the Pareto optimal set, recentness of discovery, and how their numbers change generation by generation. Since features will be measured on the population after truncation, they in fact are tracing the combined effect of variation operators and selection mechanisms, giving the dynamics of the population after truncation, that is, survival selection.

Complementing these features, we will also look at the accumulated gain of the Pareto optimal solutions, a basic operator for resolution, as well as a commonly used quality indicator, the hypervolume [19] of the approximation.

While only one benchmark problem may seem not enough, this work's main focus is the algorithms and what can the features tell us on their dynamics. To keep the study manageable, we have chosen a problem capable of generating situations that allow the algorithms to achieve convergence quickly and see how each of them deals with the discovery of new Pareto optimal solutions. In particular, we hope to make the following contributions:

(i) Look at the effect of population size on the achieved approximation through the resolution index

(ii) Show how the accumulated number of PO solutions is related to performance

(iii) Use features on the composition of the population to study algorithm dynamics

(iv) Explain some behaviors in terms of the population composition change

The organization of this work is as follows. Section 2 presents the search-assessment indices that are used as features to look at the dynamics. Section 3 describes the experimental setup, test problem, and the algorithms analyzed. Section 4 focuses on the performance and analysis of the dynamics of the population based on the proposed features. Section 5 summarizes this work and proposes some future directions.

2. Understanding Population Dynamics

Looking into the effects of algorithm choices for operators and selection mechanisms naturally leads to analyzing the population composition, so let's first define some sets around it.

Let $P(t)$ be the population at generation t, $\mathscr{F}_1(t) \subseteq \mathscr{P}(t)$ the first front, that is, the nondominated solutions, POS is the Pareto optimal set of the problem, and $\cup_{k=0}^{t-1} \mathscr{F}_1(k)$ is the union of all nondominated solutions from generation 0 to $t-1$. Do note that this set is used to check if a solution has

appeared in any generation from the beginning to just before the current one, so Pareto dominance is not used to construct this set.

Using the previous sets, we can compute the generational search-assessment indices [18] used to track the progress of the search. In this work, for simplicity, we will refer to them as population features or just features. They look at the composition of the population in a given generation t, by counting how many solutions are part of some set (the current generation nondominated set, POS), counting their dominance status, or looking into their recentness (has been part of the population in any previous generation). The features range lies in the interval $[0, |P|]$, where $|P|$ denotes the population size, and their full mathematical definition is presented in Table 1.

The first two indices are obtained from the Pareto dominance relationship. The first index is the number of *nondominated* solutions in the population at generation t, which may include solutions that are part of the POS. The second index is the number of *dominated* solutions, and it considers all the dominated solutions in the population at time t. Other indices are computed by comparing the set of nondominated solutions in the current population $\mathscr{F}_1(t)$ with the POS or with the POS and nondominated solutions at previous generations. These indices are as follows: *Pareto optimal solutions count* all the solutions in the current generation that are part of the POS. Pareto optimal possibly *new* considers solutions that are part of the POS and appeared in this generation but not in the previous one. *Pareto optimal absolutely new* considers solutions that are part of the POS and only appeared in the current generation, while *Pareto optimal not absolutely new* counts optimal solutions that appeared in any previous generation and also in the current one. *Pareto optimal dropped* counts how many Pareto optimal solutions that were present in the previous generation are not present in the current one. *Nondominated non-Pareto optimal* counts the solutions that are nondominated but also not part of the POS in the current generation.

We will also analyze the algorithms' ability to achieve a high-resolution approximation of the POS, that is, being able to not only evolve the population towards the POS and reach some Pareto optimal solutions but keep discovering them, even if the population has converged to good solutions. This becomes challenging in many-objective problems, as common choices for population sizes may not be enough to contain the complete POS. For this, we have chosen to first compute the approximation set of our algorithm $\mathscr{A}(T)$, which contains all nondominated solutions discovered by it. That is, looking at the nondominated sets $\mathscr{F}_1(t)$ from generation T to 0, collect all nondominated solutions, eliminating duplicates and applying Pareto dominance to eliminate solutions that have become dominated by the presence of better solutions found in later generations. More formally:

$$\mathscr{A}(t) = \{\mathbf{x} | \mathbf{x} \in \mathscr{X}(t) = \mathscr{A}(t-1) \cup \mathscr{F}_1(t) \backslash \mathscr{A}(t-1) \cap \mathscr{F}_1(t) \nexists \mathbf{y} \in \mathscr{X}(t) \mathbf{y} \succcurlyeq \mathbf{x}\}, \tag{1}$$

where generation 0 approximation set is simply its nondominated set

$$\mathscr{A}(0) = \mathscr{F}_1(0), \tag{2}$$

and $\mathbf{y} \succcurlyeq \mathbf{x}$ denotes that solution $\mathbf{y}$ Pareto dominates $\mathbf{x}$, defined as follows for maximization problems:

$$\mathbf{y} \succcurlyeq \mathbf{x} \quad \text{if and only if} \quad \begin{array}{ll} \forall i \in 1, \ldots, M, & f_i(\mathbf{y}) \geq f_i(\mathbf{x}) \wedge, \\ \exists i \in 1, \ldots, M & f_i(\mathbf{y}) > f_i(\mathbf{x}). \end{array} \tag{3}$$

Using these definitions, we can define the resolution index or $\alpha \in [0, 1]$ of the approximation at generation t as follows:

$$\alpha(t) = \frac{|\{\mathbf{x} | \mathbf{x} \in \mathscr{A}(t) \wedge \mathbf{x} \in \text{POS}\}|}{|\text{POS}|}, \tag{4}$$

which gives the fraction of the accumulated number of Pareto optimal solutions found until generation t to the size of the POS. Here, a value of 1 indicates that all Pareto optimal solutions were found by the algorithm.

3. Experimental Setup

To study the dynamics of several MOEAs and measure the ability of the algorithms to find a high-resolution approximation of the POS, we use MNK-landscapes [12, 20], a multiobjective extension of Kauffman's NK-landscapes [21], where M indicates the number of objectives, N the number of decision variables, and K the number of epistatic interactions between variables. The epistatic model is random. Thus, for the same values of N, K, and M, we can generate as many instances as necessary. Here, we use problem instances generated with $M = 3, 4, 5, 6$ objectives, $N = 20$ bits, and $K = 1$ epistatic bit, which determine that the contribution to the fitness of each variable is affected by one other variable (Appendix A includes a more detailed explanation of epistasis as well as its effect on problem difficulty). In this work, our interest is to observe how the population dynamics of the algorithms change when the number of objectives of the problem increases, relating it to the effectiveness of selection to find good approximations of the Pareto optimal set. Thus, we vary M but keep the interactions and number of variables unchanged. We use $N = 20$ and $K = 1$ because it gives us problems with minimum nonlinearity and a rather small search space (2^{20}), where the algorithms are expected to converge to the POS and the challenge is to find most if not all Pareto optimal solutions. These small problems also allow us to enumerate all Pareto optimal solutions to verify the selection effectiveness of the algorithms. Although at this time we use MNK-landscapes, the features used in this work are independent of the benchmark problems used.

Table 1: Generational search-assessment indices.

Name	Abbreviation	Formula
Nondominated	ND	$\{\mathbf{x}\|\mathbf{x} \in \mathscr{F}_1(t)\}$
Dominated	DOM	$\{\mathbf{x}\|\mathbf{x} \in P \wedge \mathbf{x} \notin \mathscr{F}_1(t)\}$
Pareto optimal	PO	$\{\mathbf{x}\|\mathbf{x} \in \mathscr{F}_1(t) \wedge \mathbf{x} \in \mathrm{POS}\}$
PO possibly new	POpnew	$\{\mathbf{x}\|\mathbf{x} \in \mathscr{F}_1(t) \wedge \mathbf{x} \notin \mathscr{F}_1(t-1) \wedge \mathbf{x} \in \mathrm{POS}\}$
PO absolutely new	POA	$\{\mathbf{x}\|\mathbf{x} \in \mathscr{F}_1(t) \wedge \mathbf{x} \notin \cup_{k=0}^{t-1} \mathscr{F}_1(k) \wedge \mathbf{x} \in \mathrm{POS}\}$
PO dropped	POdrop	$\{\mathbf{x}\|\mathbf{x} \in \mathscr{F}_1(t-1) \wedge \mathbf{x} \notin \mathscr{F}_1(t) \wedge \mathbf{x} \in \mathrm{POS}\}$
Nondominated non-PO	NDNP	$\{\mathbf{x}\|\mathbf{x} \in \mathscr{F}_1(t) \wedge \mathbf{x} \notin \mathrm{POS}\}$

Measures are taken on the population $P(t)$ and checking membership to the current nondominated set $\mathscr{F}_1(t)$, the previous one $\mathscr{F}_1(t-1)$, all seen nondominated sets $\cup_{k=0}^{t-1}\mathscr{F}_1(k)$, and/or the Pareto optimal set POS.

The exact number of Pareto optimal solutions found by enumeration and the total number of nondominated fronts are shown in Table 2 under columns |POS| and Fronts, respectively. The same table also shows the corresponding fraction (%) of the population sizes $|P|$ to the |POS| for various population sizes investigated here. Note that for all objectives, we chose 50, 100, and 200 as the population size, which are values commonly used in the literature. On $M = 3$ objectives, these population sizes can hold around 30%, 60%, and 130% of the POS of the problem instances used in the study. On problems with more than 3 objectives, these percentages drop significantly. Thus, to investigate the effects of population size when we increase the number of objectives, we also chose population sizes that can contain around 30% and 60% of the POS on $M > 3$ objectives.

We analyze NSGA-II (nondominated sorting genetic algorithm II) [22], AεSεH (Adaptive ϵ-Sampling ϵ-Hood) [23], IBEA (indicator-based evolutionary algorithm) [24], and MOEA/D (multiobjective evolutionary algorithm based on decomposition) [25] which are some representative algorithms from the main approaches to multi- and many-objective optimization, that is, Pareto dominance, extensions of Pareto dominance, indicator-based, and decomposition-based. These algorithms differ in the way ranking and selection of the individuals is performed but share the property of implementing a kind of $(\mu + \lambda)$ elitism to select the next population (Appendix B includes a brief description of the algorithms, focusing particularly on selection). They were chosen given they represent an easy-to-understand implementation of commonly used approaches; some are the common choice in literature; and their inner workings do not add too much noise to the effect of the selection operator. In our study of MOEAs, we focus on the population after truncation. Thus, the features capture the collective effect of the operators, parent selection, and truncation selection.

The reader is referred to the original papers for more details. For each algorithm, an out-of-the-box implementation and recommended parameters are considered. Additional settings are as follows. All algorithms use a two-point crossover with rate pc = 1 and bit-flip mutation with rate pm = $(1/n)$. For AεSεH, the reference neighborhood size is set to 20 individuals and the function used for ϵ-dominance is additive ($f_i' = f_i + \epsilon$). The ϵ+ and hypervolume are used as quality indicators in IBEA, that is, $\mathrm{IBEA}_{\epsilon+}$ and $\mathrm{IBEA}_{\mathrm{HV}}$, setting the scaling factor of the quality indicator to $k = 0.001$. For MOEA/D, the Tchebycheff scalarizing function is used, and the neighborhood size is set to 10. The weight vectors are generated using the methodology presented in [26], where the number of weights and population size can be freely set.

Each of the selected algorithms was run for a fixed number of T generations, collecting each time in separate files the sets of nondominated solutions $\mathscr{F}_1(t)$ found at generation t, $t = 1, \ldots, T$.

4. Performance and Analysis of Dynamics

4.1. Resolution Index. We first look at the resolution index $\alpha(T)$ of the approximation at the end of the run, that is, the ratio of the accumulated number of PO solutions found to the size of the POS. Figure 1 shows boxplots of the results for all algorithms with population sizes of {50, 100, 200} on 3, 4, 5, and 6 objectives problems. The algorithms are labeled as A, Ie, Ih, M, and N, corresponding to AεSεH, $\mathrm{IBEA}_{\epsilon+}$, $\mathrm{IBEA}_{\mathrm{HV}}$, MOEA/D, and NSGA-II, respectively.

On $M = 3$ objectives, it can be noted that compared to all the other algorithms, AεSεH finds the largest number of PO solutions when population sizes {50, 100, 200} are used. When the population size is 50, MOEA/D finds more PO solutions than NSGA-II. This situation is reversed for population sizes 100 and 200. $\mathrm{IBEA}_{\epsilon+}$ and $\mathrm{IBEA}_{\mathrm{HV}}$ find the lowest number of PO solutions. Note that the differences in resolutions among algorithms reduce as the ratio $|P|/|\mathrm{POS}|$ between population size and POS size increases. For 3 objectives, $|P|/|\mathrm{POS}| \sim \{33, 66, 133\}$ (%) for $|P| = \{50, 100, 200\}$, respectively. Thus, even the smaller population $|P| = 50$ can potentially contain a considerable number of PO solutions, and $|P| = 200$ can contain all of them.

For $M = 4$, the ratios are $(|P|/|\mathrm{POS}|) \sim \{3.2, 6.4, 12.9\}$ (%). For 3.2% and 6.4% ratios, MOEA/D shows an advantage against AεSεH. This advantage disappears when the ratio is 12.9%, that is, population size is 200. For the remaining objectives $M = 5$ and $M = 6$, the ratios are {0.8, 1.6, 3.2} (%) and {0.3, 0.6, 1.2} (%), respectively. For 5 and 6 objectives, MOEA/D finds more PO solutions than any of the other algorithms, with AεSεH as the second-best one. NSGA-II scales up poorly with the increase in the number of objectives, becoming even similar or worse than the IBEAs.

With 3-, 4-, and 5-objectives landscapes on $N = 20$ bits, the algorithms can easily hit the POS after some generations,

Table 2: Number of solutions in the Pareto optimal set |POS| and nondominated fronts in the landscapes with $M = 3, 4, 5$, and 6 objectives. Also, fraction of $|P|/|\text{POS}|$ (in %) for various population sizes $|P|$ investigated in this study. A — indicates that for the corresponding combination of M and $|P|$, no experiments were performed.

M	|POS|	Fronts	$\|P\|/\|\text{POS}\|$ (%)							
			50	100	200	500	1000	2000	4000	5600
3	152	258	32.9	65.8	132.6	—	—	—	—	—
4	1,554	76	3.2	6.4	12.9	32.2	64.4	—	—	—
5	6,265	29	0.8	1.6	3.2	—	—	31.9	63.8	-
6	16,845	22	0.3	0.6	1.2	—	—	—	—	33.2

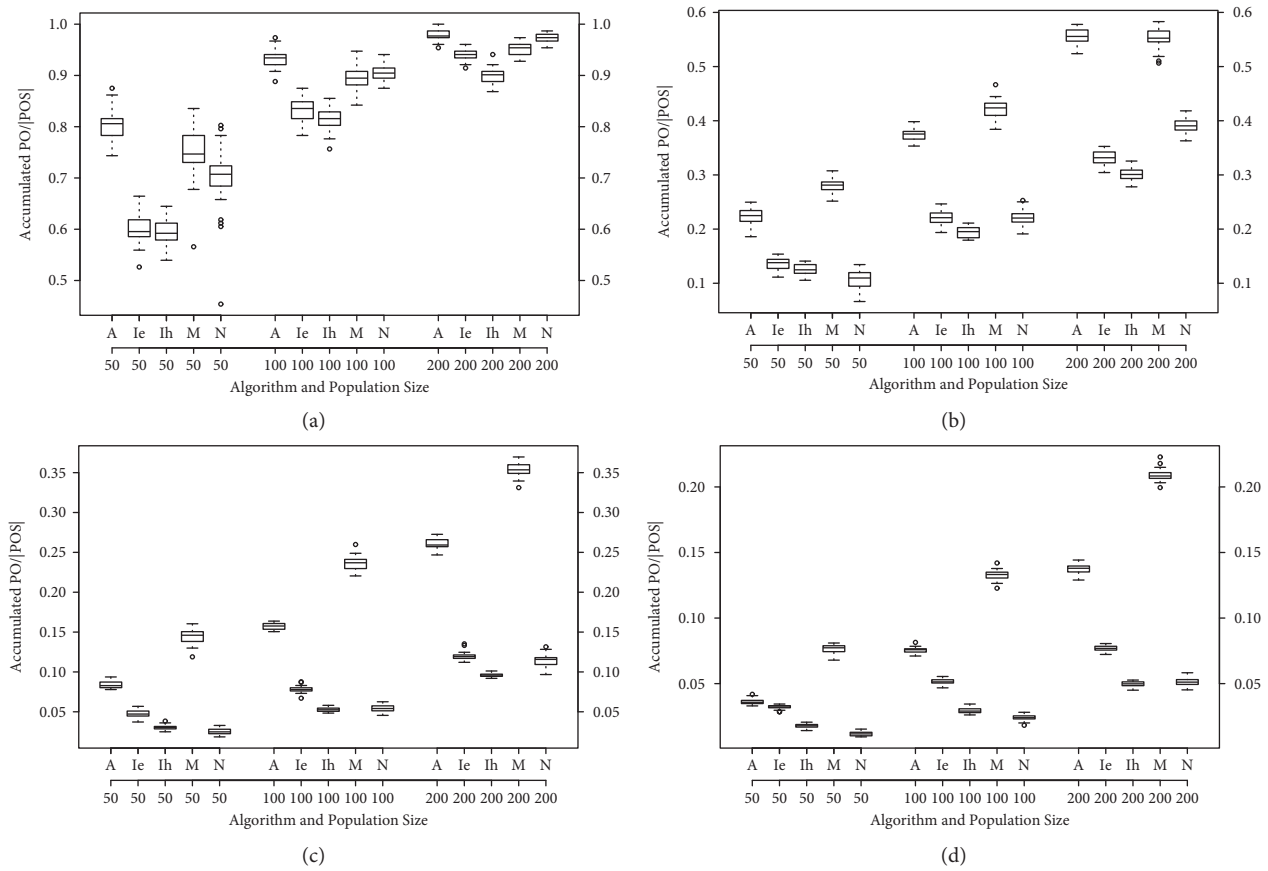


Figure 1: Resolution of the approximation at the end of the run $\alpha(T)$, that is, ratio of the accumulated number of Pareto optimal solutions found to the size of the POS. Population sizes 50, 100, and 200 for 3, 4, 5, and 6 objectives. Algorithms AεSεH (A), $\text{IBEA}_{\varepsilon+}$ (I_ε), IBEA_{HV} (I_h), NSGA-II (N), and MOEA/D (M). (a) $M = 3$ objectives, (b) $M = 4$ objectives, (c) $M = 5$ objectives, and (d) $M = 6$ objectives.

and in particular when $M = 6$, even some random generated solutions are optimal. The results described above show that the algorithms are able to keep discovering PO solutions once they hit the POS, which is particularly relevant for very small values of $|P|/|\text{POS}|$. Note that, for example, on $M = 6$ objectives AεSεH finds around 700 of the 16,845 solutions with a population size of $|P| = 50$ and MOEA/D around 1350. On problems with a large number of objectives, we see that the resolution gap between the algorithms reduces when the population size increases to a range it could hold a large portion of the POS, as shown in Figure 2 where population sizes are roughly 33%–66% of the POS for 4 and 5 objectives. When the ratio is 33%, we see that AεSεH overcomes MOEA/D finding in this case around 1,320 PO solutions against 1,190 in 4 objectives and 5,310 against 5,090 in 5 objectives. Surprisingly, NSGA-II performs better when the ratio is 66% finding more solutions than MOEA/D, while AεSεH still retains its position as the best overall. For 6 objectives, although not depicted here, similar conclusions can be drawn for population size 5,600, or 33% of the POS.

In summary, the trend observed on 3 objectives in Figure 1(a) also repeats for 4, 5, and 6 objectives when similar $|P|/|\text{POS}|$ ratios happen. This suggests that a strong correlation between population size and algorithm performance. What this translates to is that when the population is enough to contain the POS, then the performance gap

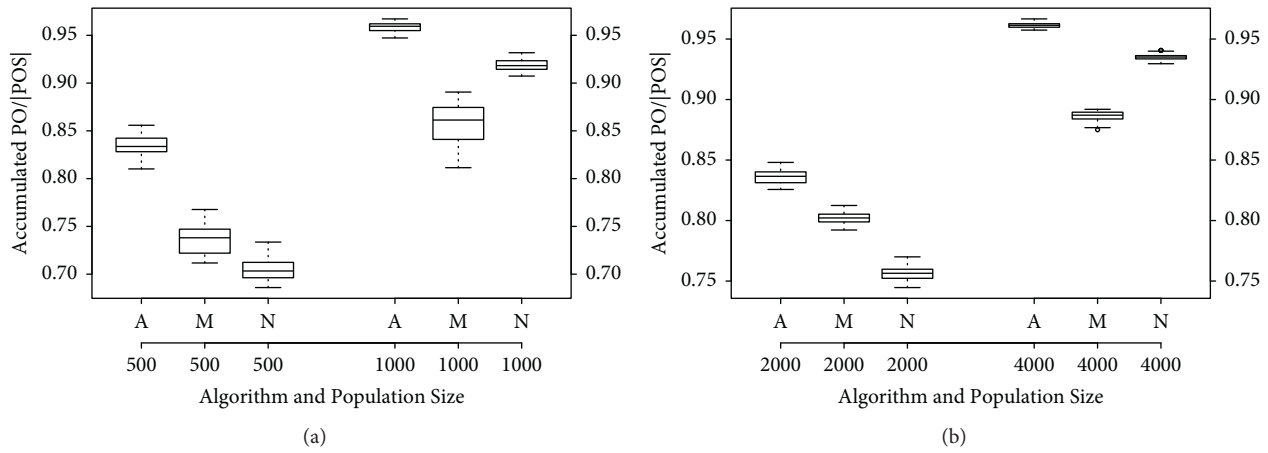


Figure 2: Ratio of accumulated number of Pareto optimal solutions found to the size of the POS. Population sizes {500, 1000} and {2000, 4000} for 4 and 5 objectives, respectively. Algorithms AεSεH (A), MOEA/D (M), and NSGA-II (N). (a) $M = 4$ objectives and (b) $M = 5$ objectives.

between algorithms' mechanisms is very close. On the other hand, even if some methods can still work with smaller populations, they too suffer a considerable loss in performance if the ratio ends up being too small.

While it is difficult or impossible to know the POS of some problems, a possible method to determine if we are close to a good enough population size for a problem could be as follows. Choose two algorithms, start them on the same population size, run for the same number of generations, and evaluate them on a performance metric. The population size that makes the gap closer will indicate a good enough choice for that particular problem.

4.2. Relationship of Resolution and Performance. Since in our resolution index we are looking at the ratio between the accumulated number of PO solutions across all generations, which can be used to evaluate the performance of an algorithm, we want to compare it with a more commonly used performance indicator, the hypervolume [19], and see if they share some relationship.

The hypervolume gives the multidimensional volume of the objective space that is dominated by a nondominated set and enclosed by a reference point. Given that our objectives are in the range [0,1] we set this point at the origin (all zeros).

For our analysis, we first compute the accumulated set of nondominated solutions found until generation t, for each generation and run, followed by the computation of the hypervolume for each of these sets. Boxplots for the 3 objectives and population size of 200 are reported in Figure 3.

By comparing the accumulated number of PO solutions with the hypervolume of the accumulated nondominated set in Figure 3, we can see that the trend and growth rate in both are similar for the AεSεH algorithm. From generation 1 to 30, we see rapid growth of the hypervolume and the number of PO solutions found. After that, these values stagnate. A similar trend is observed for the other algorithms.

We calculate the correlation between the hypervolume values for each configuration and the number of PO solutions found until a given generation using Spearman's correlation coefficient. For the example shown, the coefficients are 0.897, 0.786, 0.844, and 0.795 with $p < 2.2 \times 10^{-16}$ for NSGA-II, AεSεH, IBEA_{HV}, and MOEA/D, respectively, which shows that both values are correlated.

Knowing that there is a positive correlation between the accumulated number of PO solutions and the hypervolume value, we verify if ranking algorithms according to each of these metrics would yield similar results. For this purpose, we report in Table 3 results by the algorithms obtained with population size {50, 100, 200} on problems with 3, 4, and 5 objectives. The table shows the average values of accumulated PO solutions found by the algorithm and the hypervolume of the joint set of nondominated solutions found over all generations. The last two columns also include a ranking between algorithms from 1 (best) to 4 (worst) according to the measured value of accumulated PO solutions and hypervolume, respectively. When all rankings agree for a given algorithm, we show them in bold. The hypervolume values for a given number of objectives and population size are checked for a statistically significant difference using the Mann–Whitney test with a 95% confidence interval. If the hypervolume values are not statistically different, the corresponding algorithms are given the same ranking.

Note from this table that there is a very good agreement between the rankings given by the measured number of PO solutions and the hypervolume of all nondominated solutions found by the algorithm. Thus, the partial orderings given by this feature are useful, giving us a good idea of the relative performance of the algorithms.

That there is a correlation between both metrics is expected since an algorithm able to keep discovering new Pareto optimal solutions will indeed improve its hypervolume evaluation. How much it improves will depend on where this newly discovered solution is in the objective space and how much they improve the already attained distribution of solutions. On that note, this very reason is why we can only have a partial ranking with accumulated Pareto

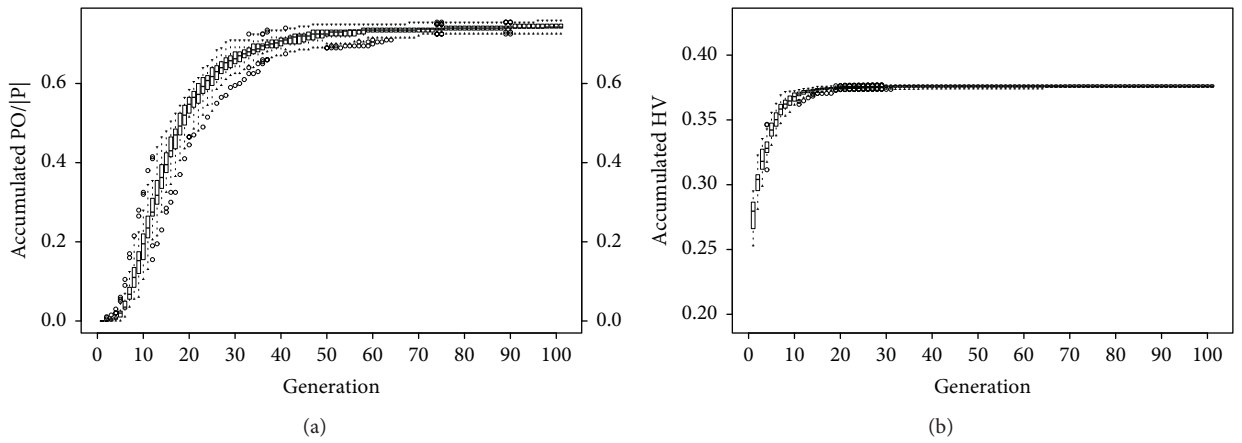


Figure 3: Accumulated number of Pareto optimal solutions and hypervolume of the accumulated joint nondominated sets. Algorithm AεSεH, $M = 3$ objectives, population size 200, and number of generations 100. (a) Accumulated PO solutions and (b) hypervolume.

Table 3: Algorithm comparison by accumulated PO solutions (APOS, feature) and hypervolume (HV, performance metric).

Algorithm	M	P	Measured		Rank	
			APOS	HV	APOS	HV
NSGA-II	3	50	105.93	0.373968	**3**	**3**
AεSεH			121.97	0.374248	**1**	**1**
$IBEA_{HV}$			90.37	0.372361	**4**	**4**
MOEA/D			114.00	0.374201	2	1
NSGA-II		100	137.97	0.375304	2	3
AεSεH			141.700	0.375287	**1**	**1**
$IBEA_{HV}$			123.70	0.373812	**4**	**4**
MOEA/D			135.50	0.375422	3	1
NSGA-II		200	148.03	0.375640	1	3
AεSεH			148.67	0.375904	**1**	**1**
$IBEA_{HV}$			136.57	0.375140	**4**	**4**
MOEA/D			144.67	0.375757	3	1
NSGA-II	4	50	166.80	0.194207	4	3
AεSεH			346.77	0.196897	**2**	**2**
$IBEA_{HV}$			195.50	0.191004	3	4
MOEA/D			436.10	0.197264	**1**	**1**
NSGA-II		100	343.23	0.197154	**3**	**3**
AεSεH			581.77	0.198445	2	1
$IBEA_{HV}$			302.40	0.193841	**4**	**4**
MOEA/D			656.27	0.198507	**1**	**1**
NSGA-II		200	608.87	0.198481	**3**	**3**
AεSεH			863.77	0.199031	**1**	**1**
$IBEA_{HV}$			468.10	0.197055	**4**	**4**
MOEA/D			857.50	0.199034	1	1
NSGA-II	5	50	157.80	0.146900	4	3
AεSεH			526.10	0.153434	**2**	**2**
$IBEA_{HV}$			190.17	0.139987	3	4
MOEA/D			904.10	0.154429	**1**	**1**
NSGA-II		100	341.60	0.151905	**3**	**3**
AεSεH			986.43	0.156632	**2**	**2**
$IBEA_{HV}$			329.70	0.145065	3	4
MOEA/D			1479.13	0.157620	**1**	**1**
NSGA-II		200	718.50	0.155398	**3**	**3**
AεSεH			1631.47	0.158275	**2**	**2**
$IBEA_{HV}$			602.30	0.150032	**4**	**4**
MOEA/D			2219.17	0.159051	**1**	**1**

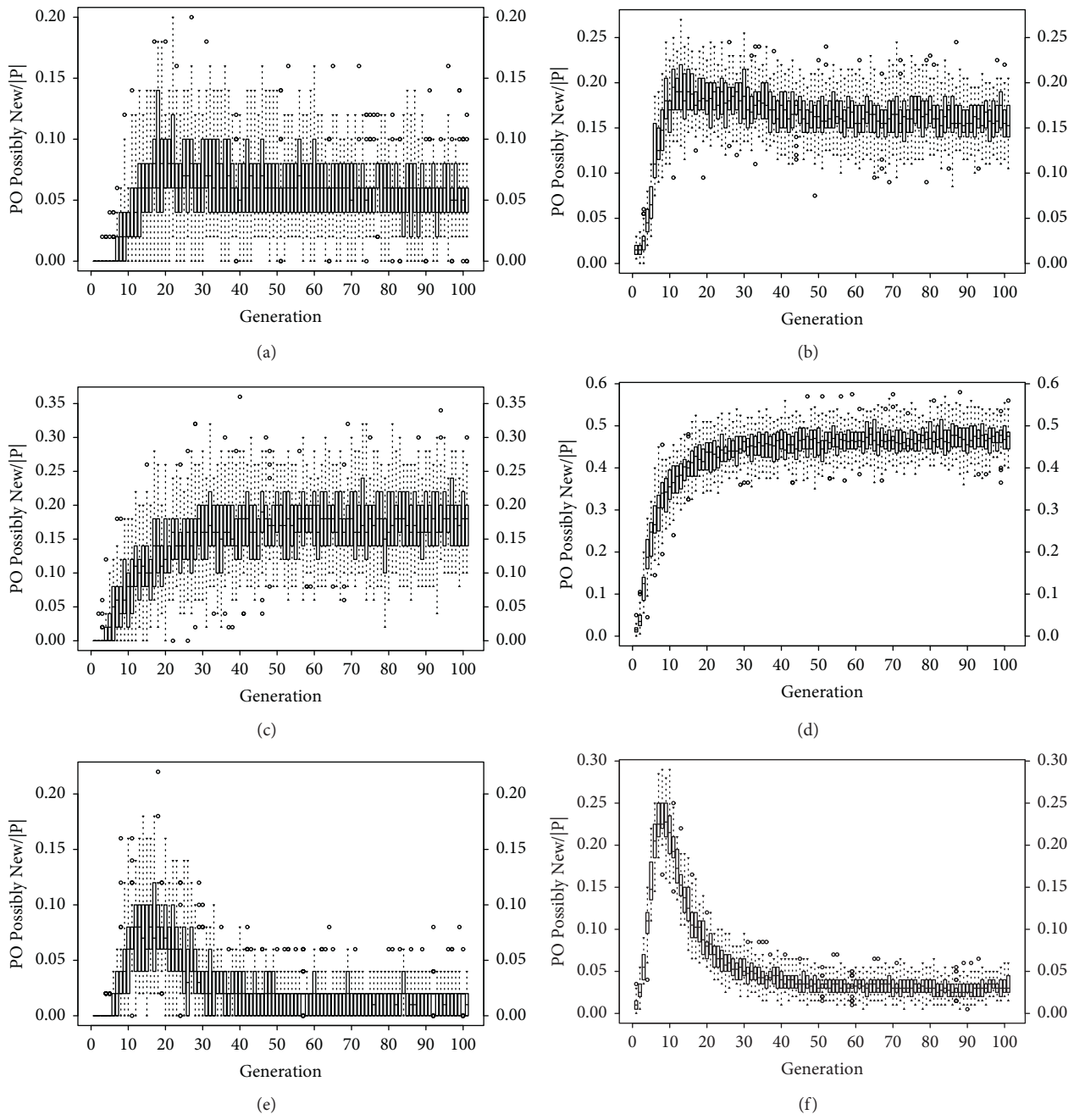


Figure 4: Pareto optimal solutions in the population that are new respect to the previous generation. Population sizes 50 and 200 for 3 and 6 objectives, respectively. Algorithms AϵSϵH, MOEA/D, and IBEA$_{HV}$. (a) AϵSϵH, $|P| = 50$, $M = 3$; (b) AϵSϵH, $|P| = 200$, $M = 6$; (c) MOEA/D, $|P| = 50$, $M = 3$; (d) MOEA/D, $|P| = 200$, $M = 6$; (e) IBEA I_{HD}, $|P| = 50$, $M = 3$; (f) IBEA I_{HD}, $|P| = 200$, $M = 6$.

optimal solutions. They do not take into account the distribution, only the total number, giving place to situations where an algorithm with a lower number of accumulated Pareto optimal solutions could have a better hypervolume than an algorithm that has found more Pareto optimal solutions. Nevertheless, it is still interesting to look at both metrics and use them to see if our algorithm is achieving not only a good resolution but also a well-distributed one.

4.3. Feature-Based Analysis of Dynamics. To understand the effects of the population size and selection mechanism on the ability of the algorithm to achieve the resolutions observed in Figure 1, we now concentrate on the dynamics of the population tracking some of its features.

First, we focus on $M = 3$ objectives with population size $|P| = 50$, where $|P|$ is 32.9% of the |POS| and $M = 6$ with $|P| = 200$, where $|P|$ is 1.2% of the |POS|. In Figure 4, we see

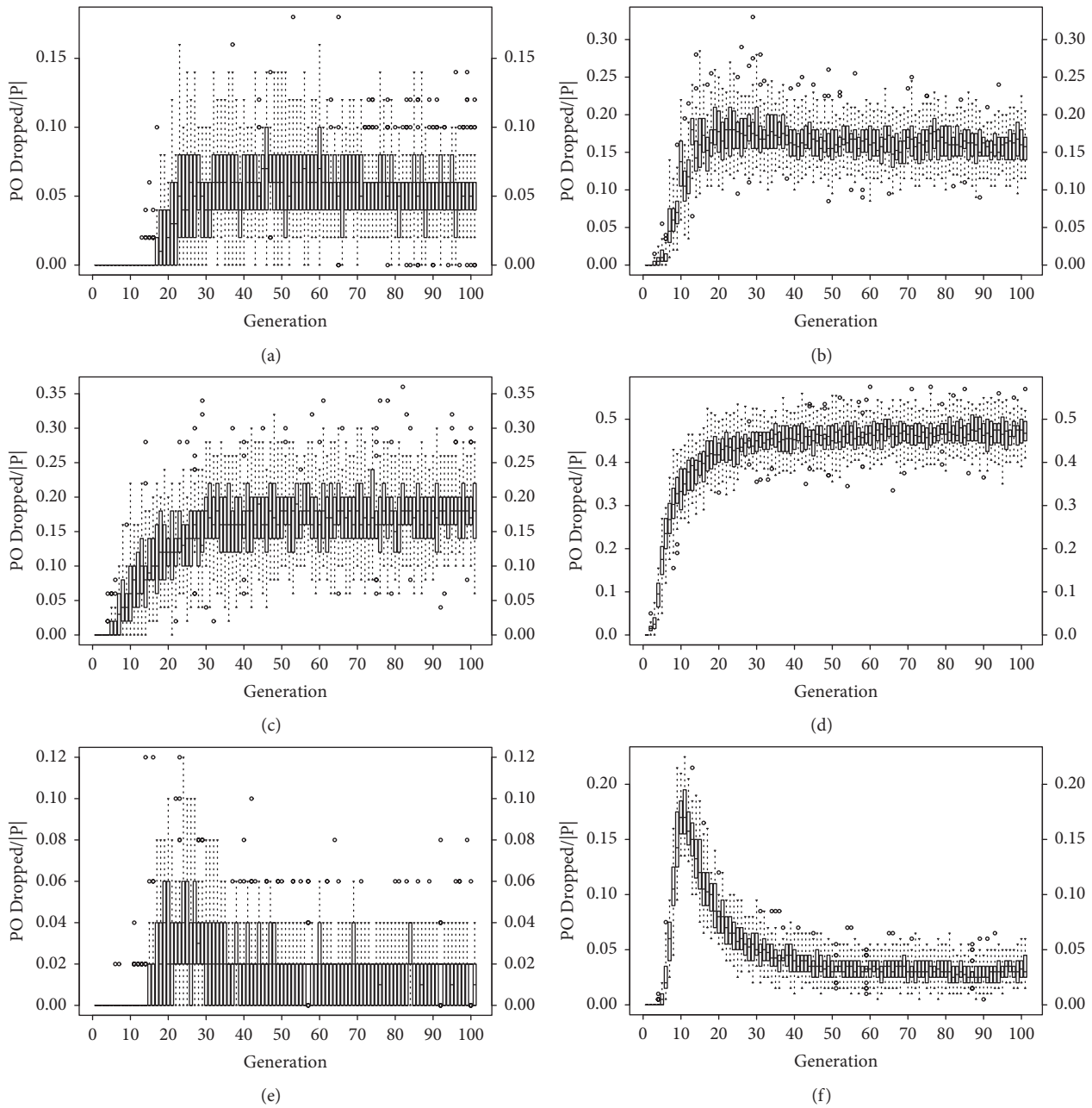


Figure 5: Pareto optimal solutions dropped from the population. Population sizes 50 and 200 for 3 and 6 objectives, respectively. Algorithms AεSεH, MOEA/D, and IBEA_{HV}. (a) AεSεH, $|P| = 50$, $M = 3$; (b) AεSεH, $|P| = 200$, $M = 6$; (c) MOEA/D, $|P| = 50$, $M = 3$; (d) MOEA/D, $|P| = 200$, $M = 6$; (e) IBEA_{HV}, $|P| = 50$, $M = 3$; and (f) IBEA_{HV}, $|P| = 200$, $M = 6$.

the fraction of POpnew/$|P|$, that is, the PO solutions that are part of the population at generation t but were not part of it at generation $t - 1$ and are therefore possibly new. This fraction could include solutions found several generations ago that are being rediscovered, apart from the ones seen for the first time. AεSεH and MOEA/D reach a peak in the initial generations and remain close to this value during the remaining generations. In $M = 3$ with $|P| = 50$ (32.9 % of the |POS|) and also $M = 6$ with $|P| = 200$ (1.2% of the |POS|), the value POpnew/$|P|$ for AεSεH is around half of that of MOEA/D. As for the IBEAs, once this value reaches its peak, it drops to very small values. Thus, IBEA rediscovers and finds very few new Pareto optimal solutions after the 30-generation mark.

In Figure 5, we see the fraction POdrop/$|P|$, that is, the PO solutions that were dropped from the population and have been replaced by other nondominated solutions, optimal or not. It is interesting to point out that the trend of this curve is very similar to the POpnew/$|P|$ fraction. In both cases studied here, MOEA/D drops around three times as

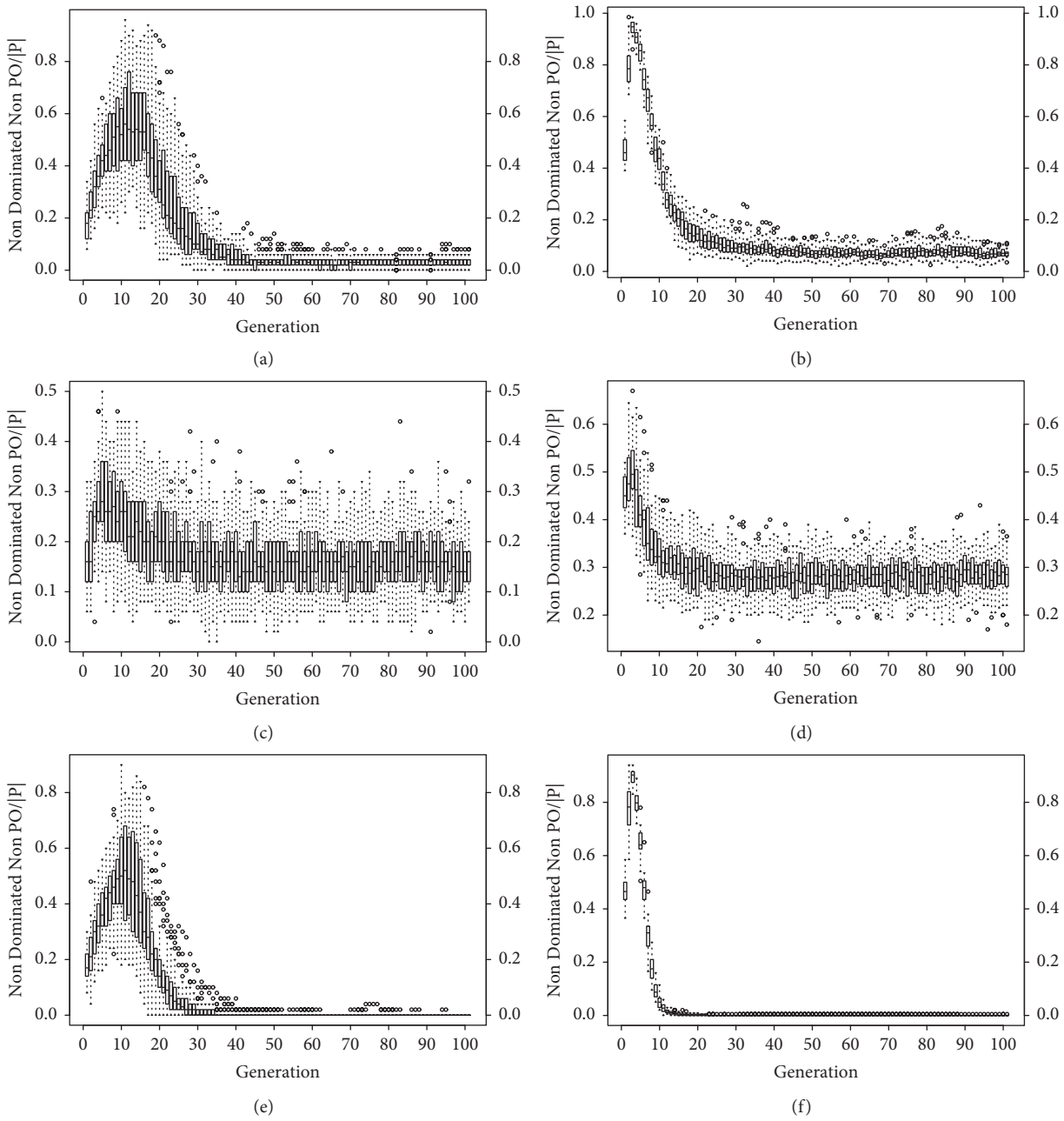


Figure 6: Nondominated solutions in the population that are not Pareto optimal. Population sizes 50 and 200 for 3 and 6 objectives. Algorithms AεSεH, MOEA/D, and IBEA_{HV}. (a) AεSεH, $|P| = 50$, $M = 3$; (b) AεSεH, $|P| = 200$, $M = 6$; (c) MOEA/D, $|P| = 50$, $M = 3$; (d) MOEA/D, $|P| = 200$, $M = 6$; (e) IBEAHV, $|P| = 50$, $M = 3$; (f) IBEAHV, $|P| = 200$, $M = 6$.

many PO solutions as AεSεH. The IBEAs at their peak drop a number of PO solutions similar to AεSεH, but afterward, it reduces rapidly to less than a third of AεSεH.

In Figure 6, we see the fraction $\text{NDNP}_t/|P|$, that is, solutions that are nondominated but not Pareto optimal. For around the first 10 or 20 generations, while it is still early in the search and few solutions are expected to be PO, the value of this indicator is two times higher in AεSεH than in MOEA/D. On the other hand, during the latest stages of the search, when it is expected that the algorithms would have accumulated more PO than NDNP solutions, this fraction is three times higher in MOEA/D than in AεSεH. In IBEA, this indicator is also higher than MOEA/D at the initial stages of the search, although smaller than AεSεH, and approaches 0 in the latest stages of the search.

From the previous plots, we can draw some conclusions in this section. Independently of the fraction of the POS that the population can hold, MOEA/D discovers and rediscovers more PO solutions than AεSεH while at the same time dropping more of these optimal solutions per generation and keeping more nondominated non-PO ones. Finding new PO solutions by dropping already found ones

can be seen as explorative behavior. However, dropped solutions can also be replaced with nonoptimal solutions. An algorithm with this ability could escape local optima to find optimal solutions unreachable unless there is a step down to inferior solutions first. Both, MOEA/D and AεSεH present this trait, with MOEA/D being more intense in its approach. This behavior on 3 objectives, where more fronts need to be climbed and also observed on 4 to 6 objectives, where there are fewer fronts, suggests that exploration could positively impact algorithm performance. However, in larger search spaces, where hitting the POS is harder, too much exploration could actually be detrimental.

Focusing on the drop of PO solutions, it is interesting to think how an algorithm can drop an already found optimal solution sometimes in favor of an inferior one. Here are some insights.

In Pareto dominance-based algorithms and their extensions, the algorithm will have to discard some nondominated solutions obtained after combining the current population and the offspring if their number is larger than the population size. Since dominance is computed within the population, and not globally, during truncation the algorithm cannot differentiate between optimal and suboptimal solutions, that is, they appear nondominated at the population level. In this case, some optimal solutions could be dropped. This happens more often when the ratio between population size and the size of the POS is too small.

For decomposition, the subproblem strategy imposes a stricter order between solutions. In combinatorial problems, hopefully, the solutions regarded as optimal for the subproblem are also Pareto optimal. This is a key difference between dominance-based algorithms and decomposition-based ones. There could be cases where a solution is globally superior or even optimal in a multiobjective sense, that is, in terms of Pareto dominance, but the subproblem formulation ranks it lower and replaces it with some other solution that is better for the local subproblem.

For indicator-based algorithms, since they try to induce a total ordering between solutions according to their contribution to the performance indicator, they converge to the subset of solutions with the highest rank, which cardinality is the size of the population. Once the algorithm finds this subset, the operators of variation could discover other Pareto optimal solutions. However, these solutions will have a rank inferior to those already in the population; therefore, no replacement of optimal solutions will occur. Eventually, the algorithm will stagnate not being able to find new solutions. These traits could be helpful to converge on larger subspaces, although diversity could still be an issue. In IBEA, the fitness of a solution x represents the loss of quality that will be incurred if this solution is eliminated from the population. It is given by $\text{Fitness}(x) = \sum_{x' \in P \setminus \{x\}} - e^{-I(x',x)/\kappa}$, where I is the indicator function and $\kappa > 0$ a scaling factor set by the user. Changing the values of κ may induce different dynamics for IBEA.

From the point of view of the continued discovery of Pareto optimal solutions, being able to drop even an optimal solution makes sense if we wish to keep exploring and hitting new and hopefully also optimal solutions or rediscovery of previously found ones. However, just looking at common performance metrics is difficult to notice this type of behavior, which makes a strong argument for more population features and to look not only at the final approximation but also at all the approximation sets that the algorithm produces while converging.

This particular feature also tells us that some kind of archiving should be prepared to avoid losing interesting solutions that are discarded to make space in the population.

5. Conclusions

In this work, we studied the population dynamics of several multi- and many-objective optimizers to understand better the mechanisms that allow generating a high-resolution approximation of the Pareto optimal set.

Particularly, we investigated population sizes relative to the size of the Pareto optimal set and looked at different selection mechanisms used by representative algorithms of the main approaches to multi- and many-objective optimization, that is, Pareto dominance-, extensions of Pareto dominance-, decomposition-, and indicator-based MOEAs. The selected algorithms were NSGA-II, $A\varepsilon S\varepsilon H$, MOEA/D, and IBEA with the hypervolume and ε indicators.

To track the dynamics, we used as features several generational assessment indices that count the number of solutions in the population that are Pareto optimal, separating them between those that are new and those that have already been seen. We also tracked the Pareto optimal solutions dropped from the population. The feature that keeps track of the newly discovered Pareto optimal solutions allowed us to link population dynamics to algorithm performance in terms of the accumulated number of Pareto optimal solutions found by the algorithm, which gives a direct measure of the resolution of the approximation.

Using these features, we derived some important conclusions. (i) Independently of the population size and a number of objectives, the dynamics of the algorithms show that MOEA/D will discover and rediscover more Pareto optimal solutions than $A\varepsilon S\varepsilon H$ and IBEA. (ii) Independently of the population size and the number of objectives, the dynamics of the algorithms show that MOEA/D will drop more Pareto optimal solutions than $A\varepsilon S\varepsilon H$ and IBEA, in this order. (iii) How these characteristics of the algorithms transform into performance depends on population size. More precisely, it depends on the ratio between the population size |P| and the size of the Pareto optimal set |POS|, that is, |P|/|POS|. For very small ratios of$|P|/|\text{POS}|$, the resolution of the approximation α, that is, the accumulated number of Pareto optimal solutions, found by MOEA/D is larger than $A\varepsilon S\varepsilon H$, IBEA, and NSGA-II, in that order, that is, $\alpha_{\text{MOEA/D}} > \alpha_{A\varepsilon S\varepsilon H} > \alpha_{\text{IBEA}} > \alpha_{\text{NSGA}-II}$. However, for larger ratios, the resolution of the approximation α found by $A\varepsilon S\varepsilon H$ improves and becomes larger than MOEA/D. From our experiments, $\alpha_{A\varepsilon S\varepsilon H} \sim \alpha_{\text{MOEA/D}}$ for $|P|/|\text{POS}| = 0.13$ in 4 objectives and $\alpha_{A\varepsilon S\varepsilon H} > \alpha_{\text{MOEA/D}}$ for a ratio $(|P|/|\text{POS}|) > 0.3$ independently of the number of objectives. NSGA-II requires a ratio of $(|P|/|\text{POS}|) > 0.6$ to get a resolution of the approximation better than MOEA/D. (iv) Replacing the

indicator in IBEA, hypervolume or ε, does not substantially change the dynamics nor its relative performance compared to other algorithms.

The analysis based on features indicates that there is valuable information that can be obtained by looking at the population and how its compositions change in each generation. Just looking at the final population and performance metrics is not enough if we want to keep improving our algorithms. Designers can look at the behavior of new selection and variance operators through features and use them to describe and characterize their dynamics. While practitioners could include these metrics to feedback the algorithm or even choose what operators should suit the problem better.

In future works, we would like to explore these paths and move towards real scenarios, where the search space is larger and convergence becomes a challenge for the algorithm. For this to happen, we require a new set of features that can be computed without the need for the Pareto optimal set. We also think that to further take advantage of the features' analytical power, the next step is to use them to make predictions on the algorithm's behavior and performance and take the appropriate decisions that can lead to a better approximation.

Appendix

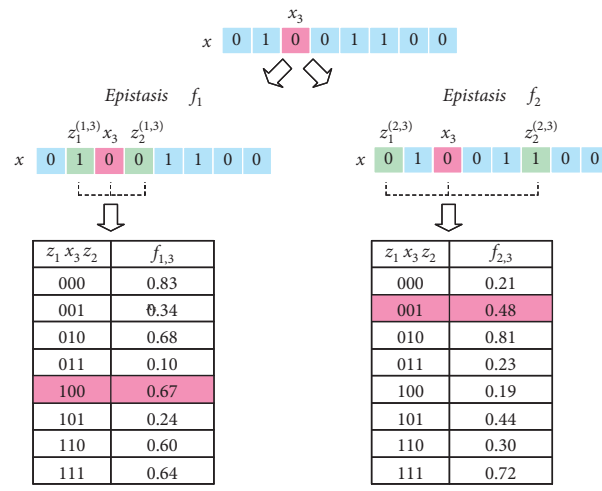
A. Test Problem Generator

Looking into the algorithms' behavior postconvergence and analyzing its ability to discover Pareto Optimal solutions require a problem that can be easily tuned and versatile enough to generate these situations.

MNK-landscapes [12] is a multiobjective extension of Kauffman's NK-landscapes [27] that can generate adjustable instances of multi- and many-objective optimization problems given some parameters. The number of objectives can be controlled with M, N defines the number of bits of the string that represents the decision variables, while K controls the ruggedness of the landscape by determining the epistatic interactions between the decision variables.

The last parameter epistasis comes from biology where this term refers to the expression of one gene being masked by the genotypic effect of another one, due to the nonlinear interdependence found in genes. In terms of a combinatorial optimization problem, this translates to fitness functions and the nonlinear interdependence between variables that can make a single decision variable impact how other variables contribute to the overall fitness.

For a small example, suppose a binary decision vector, where all variables can be on or off, 0 or 1. Given a flip in one variable, whereas this could represent a gain in the contribution made by it, this change can also have a side effect of decreasing the contribution made by other variables that interact with it. In the context of solving an optimization problem, these interactions, which are commonly unknown beforehand, can impact the performance of the search for an optimal solution. To get a better idea of how this occurs, in



$z_1\, x_3\, z_2$	$f_{1,3}$
000	0.83
001	0.34
010	0.68
011	0.10
100	0.67
101	0.24
110	0.60
111	0.64

$z_1\, x_3\, z_2$	$f_{2,3}$
000	0.21
001	0.48
010	0.81
011	0.23
100	0.19
101	0.44
110	0.30
111	0.72

Figure 7: Epistatic interactions example.

Figure 7, there is a more detailed example of epistasis and epistatic interaction.

Let $f_{1,3}(x_3, z_1^{(1,3)}, z_2^{(1,3)})$ and $f_{2,3}(x_3, z_1^{(2,3)}, z_2^{(2,3)})$ be the fitness functions associated to bit x_3 contributing to the first objective function $f_1(\cdot)$ and the second one $f_2(\cdot)$, respectively, based on different epistatic models for each objective. $z_1^{(1,3)}$, $z_2^{(1,3)}$, $z_1^{(2,3)}$, and $z_2^{(2,3)}$ are determined when the instance is created, as well as the possible combinations and effects it has on each fitness function, shown in the table inside Figure 7. In $f_{1,3}$, x_3 epistatically interacts with its left and right neighboring bits, $x_2 = z_1^{(1,3)}$ and $x_4 = z_2^{(1,3)}$. While for $f_{2,3}$, x_3 epistatically interacts with its second bit to the left and with its third bit to the right, $x_1 = z_1^{(2,3)}$ and $x_6 = z_2^{(2,3)}$. In the example,$N = 8$ since there are eight decision variables and $K = 2$ since each x_i effect on the fitness function depends on two other variables.

It can be seen from the table that if x_2 and x_4 keep their values as well as x_1 and x_6, 0 is a good value for x_3, making its contribution to $f_1(\cdot)$ be 0.67 and $f_2(\cdot)$ be 0.48. If a bit flip were to happen, then the contributions will reduce to 0.60 and 0.23, respectively.

In this brief example, the following situations are made clear: deciding each variable value can depend only on the variable ($K = 0$, no interactions) or be as hard as having to consider all the other variables ($K = N - 1$). It means that for $K \geq 1$ and without prior knowledge on the underlying interactions, exploring the search space flipping one bit a time is a hard task [28]. If knowledge is available or certain conditions on the interaction of bits are given (i.e., only neighbors, not random), some approaches exist for the single-objective case [29, 30].

Once the objective number goes to $M \geq 2$, there will be a set of interaction tables for each objective, which means that the algorithm is searching for a compromise setting of the variables that optimizes all of them. This in particular makes a case to use algorithms that are able to preserve sections of the solution (crossover operators in evolutionary algorithms) and can flip several bits in one try [31]. Flipping only one bit at a time could not be as efficient and made the process longer.

Finally, it is also worth the mention that while not always a straightforward task, learning the bit interactions to avoid not improving bit flips is possible with some techniques [32].

B. Representative Multi- and Many-Objective Evolutionary Algorithms

B.1. NSGA-II. The nondominated sorting genetic algorithm II is an elitist (preserves the best solutions found so far) multiobjective evolutionary algorithm that uses Pareto dominance and density estimation, through crowding distance, to determine which solutions are retained and how parents are determined for the next generation.

During each generation, from the current population P_t, the equally sized Q_t that contains the generated offspring is created. After evaluating both sets according to the objectives functions, they are joint to be classified in nondominated Fronts = $\{\text{Front}_1, \text{Front}_2, \ldots, \text{Front}_n\}$. Inside each Front, a value called crowding distance is calculated that allows estimating for a particular solution the density of solutions surrounding it. To create the new population P_{t+1}, solutions in each Front_i are merged into P_{t+1} if this operation does not overfill it. When one of them does , that Front_i is sorted according to its crowding distance, and only the necessary number of solutions to complete the population size are copied.

Fitness is determined by a tuple formed by the front number that the solution belongs and its crowding distance. Lower front numbers, which indicate a better rank, are preferred, and in case of a tie, higher crowding distance is preferred. This allows the retaining of solutions that cover a zone in the objective space with a low density of solutions. Selection of parents is done by a binary tournament between randomly chosen individuals from the population using rank and crowding distance information. In Figure 8, the algorithm is presented in pseudocode. For a more detailed explanation of the algorithm, in particular how the fast nondominated sorting procedure is done, consult Deb et al. [22].

```
1  t ← 0
2  P_0 ← InitializePopulation(popSize)
3  Evaluate(P_0)
4  NonDominatedSort(P_0)
5  while ¬ StoppingCondition() do
6      Parents ← SelectParentsByRankAndDistance(P_t)
7      Q_t ← CrossoverAndMutation(Parents)
8      Evaluate(Q_t)
9      R_t ← P_t ∪ Q_t
10     Fronts ←FastNonDominatedSort(R_t)
11     P_{t+1} ← ∅
12     L ← 0
13     while Front_i ∈ Fronts do
14         if Size(P_{t+1})+Size(Front_i) > popSize then
15             L ← i
16             break()
17         end
18         else
19             P_{t+1} ← Merge(P_{t+1}, Front_i)
20         end
21     end
22     if Size(P_{t+1}) < popSize then
23         Front_L ← SortByRankAndDistance(Front_L)
24         for j to popSize-Size(Front_L) do
25             P_{t+1} ← Front_L[i]
26         end
27     end
28     t = t + 1
29 end
30 return Front_1
```

Figure 8: Pseudocode of NSGA-II.

B.2. AεSεH. The Adaptive ϵ-Selection ϵ-Hood genetic algorithm is a many-objective optimization algorithm that uses Pareto dominance relaxation in the form of ϵ-dominance to determine which solutions are retained and how parents are determined for the next generation. There is not an explicit method for fitness assignment in this algorithm.

Similar to NSGA-II, during each generation, the current population P_t and the generated offsprings Q_t are joined and classified in Fronts according to standard nondomination. If $|\text{Front}_1| < \text{popSize}$, then the procedure continues normally as in NSGA-II, where solutions are copied from the Fronts, and if a Front_i were to cause an overfill, solutions are selected randomly from this last front until P_t has the correct size.

However, in the more common case of $|\text{Front}_1| > \text{popSize}$ in many-objective optimization, ϵ-sampling with ϵ_s as a parameter is done. This function applies random sampling to select solutions from Front_1, copying them into P_{t+1} and eliminating from Front_1 all the ϵ-dominated solutions by the chosen sample. If P_{t+1} were to be overfilled, solutions are randomly eliminated until it reaches the correct size. Otherwise, if P_{t+1} still is not complete, the remaining solutions are randomly chosen from the previously discarded ϵ-dominated ones.

For parent selection, first ϵ-hood creation creates a cluster of solutions in objective space, which is done by selecting randomly a solution from the population and creating a neighborhood around it, composed by all the ϵ-dominated solutions using ϵ_h. This process is done until all solutions belong to a neighborhood. Then the function ϵ-mating visits each neighborhood in a round-robin fashion, selecting randomly two parents from each one of them. This assures that even solutions in low populated neighborhoods have the same reproduction probability.

At each generation, the ϵ_s used during selection and the ϵ_h used during the neighborhood creation are adapted taking into account a step size Δ and the population size. In particular, ϵ_h is adapted so the number of neighborhoods is closer to the one specified by the user N_h^{Ref}. Pseudocode of the algorithm can be found in Figure 9. For a more detailed explanation of the algorithm, consult Aguirre et al. [23].

B.3. IBEA. The indicator-based evolutionary algorithm is a many-objective optimization algorithm that relies on performance indicators to determine which solutions remain in the population. The fitness for each individual represents the loss of quality that will be incurred if these solutions were to be eliminated from the population, calculated as $\text{Fitness}(x) = \sum_{x' \in P \setminus \{x\}} -e^{-I(x',x)/\kappa}$, where x is an individual, I the indicator function and $\kappa > 0$ a scaling factor set by the user.

For each generation, the current population P and its offspring Q are joint, and the fitness for each individual is

```
1  t ← 0
2  N_h^Ref ← popSize/H_size^Ref // Set reference number of neighborhoods
3  ϵ_s, ϵ_h ← 0
4  Δ_s, Δ_h ← Δ_s0, Δ_h0 // Set step adaptation
5  P = ∅
6  Q ← InitializePopulation(popSize)
7  while ¬ StoppingCondition() do
8      Evaluate(Q)
9      R ← P ∪ Q
10     Fronts ←FastNonDominatedSort(R)
11     if Size(Front_1) < popSize then
12         while Front_i ∈ Fronts do
13             if Size(P_t+1)+Size(Front_i) > popSize then
14                 L ← i
15                 break()
16             end
17             else
18                 P_t+1 ← Merge(Parents, Front_i)
19             end
20         end
21         if Size(P_t+1) < popSize then
22             for j to popSize-Size(Front_L) do
23                 P_t+1 ← Front_L[random()]
24             end
25         end
26     end
27     else
28         P_t+1,N_s ← epsilonSampling(Front_1,ϵ_s,popSize)
29         adapt(ϵ_s,Δ_s,popSize,N_s)
30     end
31     H, N_h ← epsilonHoodCreation(P_t+1,ϵ_h) //H = {H_1, H_2, ..., H_N_h}
32     adapt(ϵ_h,Δ_h,N_h^Ref,N_h)
33     Parents ← epsilonHoodMating(H,popSize)
34     Q ← CrossoverAndMutation(Parents)
35     t = t + 1
36 end
37 return F_1
```

Figure 9: Pseudocode of AεSεH.

calculated. Then solutions with the lowest fitness are eliminated until the size of the population reaches the correct size. When a solution is removed from the population, an update is done to reflect the new fitness without the eliminated solution.

During the parent selection, a binary tournament-based on fitness is done between randomly selected solutions from the current population.

Note that IBEA transforms the original multiobjective optimization problem into one, where the main goal is to obtain an improvement in the overall performance of the population according to the selected indicator.

In this work, the following two indicators are used in conjunction with IBEA: the binary additive ϵ-indicator ($I_{\epsilon+}$) and the binary hypervolume difference-indicator (I_{HD}).

$$I_{\epsilon+}(x', x)$$

$$I_{\mathrm{HD}}(x, x') = \begin{cases} H(x') - H(x), & \text{if } x' \succeq x \text{ or } x \succeq x', \\ H(x + x') - H(x), & \text{otherwise,} \end{cases} \tag{B.1}$$

where $x \succeq x'$ indicates that x dominates x' in a Pareto sense. $I_{\epsilon+}(x, x')$ gives the minimum value by which a solution x needs to be translated in the objective space so it can weekly dominate x'. $H(x)$ gives the multidimensional volume of the objective space that is dominated by x. $I_{\mathrm{HD}}(x, x')$ gives the hypervolume that x' dominates, but not by x. The algorithm in pseudocode is presented in Figure 10. Further information about the algorithm or the performance indicators used here can be found in the work done by Zitzler and Künzli [24].

```
1  t ← 0
2  Initialize population P of size n randomly
3  while ¬ StoppingCondition() do
4      EvaluateUsingIndicator(P)
5      if Size(P) > popSize then
6          RemoveLowestFitnessIndividual(P)
7          UpdateFitness(P)
8      end
9      Parents ← Selection(P)
10     Q ← CrossoverAndMutation(Parents)
11     EvaluateUsingIndicator(Q)
12     P ← Merge(P ∪ Q)
13     t ← t + 1
14 end
15 A ← NonDominated(P)
16 return A
```

Figure 10: Pseudocode of IBEA.

B.4. MOEA/D. The multiobjective evolutionary algorithm based on decomposition, as the name suggests, decomposes the problem into single-objective ones using scalarization functions and optimizing them simultaneously. Subproblems are optimized using the surrounding solutions that belong to their neighborhood.

The algorithm requires a set of weigh vectors $\Lambda \leftarrow \{\lambda_1, \lambda_2, \ldots, \lambda_N\}$ defined by the user; a scalarization function, in this case, is considered the Tchebycheff function

```
1  t ← 0
2  externalP ← ∅
3  Λ ← {λ_1, λ_2, ..., λ_N}
4  Initialize population P of size N randomly
5  Evaluate(P)
6  Randomly assign each weight λ_i a solution from P = {x_1, ..., x_N}
7  Neig ← CreateNeighborhoods(λ,Neighborhood_Size)
8  Initialize the reference point z = (z_1, ..., z_m)
9  while ¬ StoppingCondition() do
10     for λ_i ∈ Λ do
11        Parents ← SelectParents(Neig(i))
12        x'_i ← CrossoverMutation(Parents)
13        Evaluate(x'_i)
14        UpdateReferencePoint(z)
15        UpdateNeighbors(x'_i,Neig(i))
16        Update(externalP)
17    end
18    t ← t + 1
19 end
20 return externalP
```

Figure 11: Pseudocode of a MOEA/D.

and a $\text{Neighborhood}_{\text{Size}}$. First, a population of P of N individuals is created and evaluated, followed by assigning each $x_i \in P$ to a particular weigh vector λ_i. Neighborhoods are created by calculating the Euclidean distance between any pair of weigh vectors and clustering the $\text{Neighborhood}_{\text{Size}}$ ones that are closer. The reference point needed for the scalarization function is initialized, using a problem-specific technique or setting the value for each objective as the best one according to the current population.

During each generation, for each subproblem with a weigh vector λ_i, a new individual x_i' is created by applying genetic operators to parents selected randomly inside the neighborhood Neig (i). If the solution dominates some in the neighborhood, a replacement is done; otherwise, it is discarded. The reference point is updated, as well as the external population. This process is repeated until all the subproblems have been visited and updated. The algorithm terminates when its stopping condition is met.

An important note here is that the replacement is done the moment a better solution is found, and it could replace any of the ones in its neighborhoods. This means that since neighborhoods overlap, the next subproblem is working with the best solutions found so far.

The pseudocode of the algorithm is shown in Figure 11. More details of the algorithm and other possible scalarization functions can be found in Zhang and Li [25].

Data Availability

Algorithms and data will be available upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Vol. 5, Springer, , Berlin, Germany, 2007.

[2] K. Deb, *Optimization for Engineering Design: Algorithms and Examples*, PHI Learning Pvt. Ltd., New Delhi, India, 2012.

[3] C. Coello and G. Lamont, "Applications of multi-objective evolutionary algorithms," *Advances in Natural Computation*, World Scientific, Singapore, 2004.

[4] M. Farina and P. Amato, "On the optimal solution definition for many-criteria optimization problems," in *Proceedings of the 2002 Annual Meeting of the North American Fuzzy Information Processing Society Proceedings. NAFIPS-FLINT 2002*, IEEE, New Orleans, LA, USA, 2002.

[5] P. J. Fleming, R. C. Purshouse, and R. J. Lygoe, "Many-objective optimization: an engineering design perspective," in *Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization*, 2005.

[6] J. G. Herrero, A. Berlanga, and J. M. M. López, "Effective evolutionary algorithms for many-specifications attainment: application to air traffic control tracking filters," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 151–168, 2009.

[7] R. J. Lygoe, M. Cary, and P. J. Fleming, "A real-world application of a many-objective optimisation complexity reduction process," in *Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization*, 2013.

[8] A. López-Jaimes, A. Oyama, and K. Fujii, "Space trajectory design: analysis of a real-world many-objective optimization problem," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, Cancun, Mexico, June 2013.

[9] Y. Nishio, A. Oyama, Y. Akimoto, H. Aguirre, and K. Tanaka, "Many-objective optimization of trajectory design for destiny mission," in *Proceedings of the Learning and Intelligent Optimization Conference*, Lille, France, 2014.

[10] R. C. Purshouse and P. J. Fleming, "Evolutionary many-objective optimisation: an exploratory analysis," in *Proceedings of the 2003 Congress on Evolutionary Computation, 2003*, vol. 3, Canberra, Australia, Dec 2003.

[11] V. Khare, X. Yao, and K. Deb, "Performance scaling of multi-objective evolutionary algorithms," in *Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization*, 2003.

[12] H. Aguirre and K. Tanaka, "Insights on properties of multiobjective MNK-landscapes," in *Proceedings of the 2004 Congress on Evolutionary Computation*, vol. 1, Portland, OR, USA, June 2004.

[13] E. J. Hughes, "Evolutionary many-objective optimisation: many once or one many?" in *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, vol. 1, Edinburgh, UK, Sep. 2005.

[14] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: a short review," in *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, Hong Kong, China, June 2008.

[15] C. von Lücken, B. Barän, and C. Brizuela, "A survey on multi-objective evolutionary algorithms for many-objective problems," *Computational Optimization and Applications*, vol. 58, no. 3, pp. 707–756, 2014.

[16] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: a survey," *ACM Computing Surveys*, vol. 48, no. 1, p. 13, 2015.

[17] C. von Lücken, C. Brizuela, and B. Barán, "An overview on evolutionary algorithms for many-objective optimization problems," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 1, Article ID e1267, 2019.

[18] H. Aguirre, A. Liefooghe, S. Verel, and K. Tanaka, "An analysis on selection for high-resolution approximations in many-objective optimization," in *Proceedings of the 13th International Conference Parallel Problem Solving from Nature*, pp. 487–497, Ljubljana, Slovenia, September 2014.

[19] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[20] H. E. Aguirre and K. Tanaka, "Working principles, behavior, and performance of MOEAs on MNK-landscapes," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1670–1690, 2007.

[21] S. A. Kauffman and E. D. Weinberger, "The NK model of rugged fitness landscapes and its application to maturation of the immune response," *Journal of Theoretical Biology*, vol. 141, no. 2, pp. 211–245, 1989.

[22] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, pp. 849–858, Paris, France, September 2000.

[23] H. Aguirre, A. Oyama, and K. Tanaka, "Adaptive ε-sampling and ε-hood for evolutionary many-objective optimization," in *Proceedings of the 7th International Conference Evolutionary Multi-Criterion Optimization*, Sheffield, UK, March 2013.

[24] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature*, pp. 832–842, Birmingham, UK, September 2004.

[25] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[26] S. Zapotecas-Martínez, H. E. Aguirre, K. Tanaka, and C. A. C. Coello, "On the low-discrepancy sequences and their use in MOEA/D for high-dimensional objective spaces," in *Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC)*, Sendai, Japan, May 2015.

[27] S. Kauffman, *The Origins of Order: Self-Organization and Selection in Evolution*, Oxford University Press, Oxford, UK, 1993.

[28] A. Wright, R. Thompson, and J. Zhang, "The computational complexity of n-k fitness functions," *IEEE Transactions on Evolutionary Computation*, vol. 4, pp. 373–379, 2000.

[29] F. Chicano, D. Whitley, G. Ochoa, and R. Tinós, "Optimizing one million variable NK landscapes by hybridizing deterministic recombination and local search," in *Proceedings of the 2017 Genetic and Evolutionary Computation Conference*, Berlin, Germany, 2017.

[30] M. Pelikan, "Analysis of estimation of distribution algorithms and genetic algorithms on NK landscapes," in *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pp. 1033–1040, Association for Computing Machinery, Atlanta, GA, USA, 2008.

[31] M. Pelikan, "NK landscapes, problem difficulty, and hybrid evolutionary algorithms," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, Portland, OR, USA, 2010.

[32] Y. Chen, T.-L. Yu, K. Sastry, and D. E. Goldberg, "A survey of linkage learning techniques in genetic and evolutionary algorithms," NCLab report, National Chiao Tung University, Hsinchu, Taiwan, 2007.