General Subpopulation Framework and Taming the Conflict Inside Populations

Danilo Vasconcellos Vargas vargas@cig.ees.kyushu-u.ac.jp
Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka, 819-0395, Japan

Junichi Murata murata@cig.ees.kyushu-u.ac.jp
Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, 819-0395, Japan

Hirotaka Takano takano@cig.ees.kyushu-u.ac.jp
Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, 819-0395, Japan

Alexandre Cláudio Botazzo Delbem acbd@icmc.usp.br
Institute of Mathematics and Computer Science, University of São Paulo, São Carlos, 13566-590, Brazil

Abstract

Structured evolutionary algorithms have been investigated for some time. However, they have been under-explored specially in the field of multi-objective optimization. Despite their good results, the use of complex dynamics and structures make their understanding and adoption rate low. Here, we propose the general subpopulation framework that has the capability of integrating optimization algorithms without restrictions as well as aid the design of structured algorithms. The proposed framework is capable of generalizing most of the structured evolutionary algorithms, such as cellular algorithms, island models, spatial predator-prey and restricted mating based algorithms under its formalization. Moreover, we propose two algorithms based on the general subpopulation framework, demonstrating that with the simple addition of a number of single-objective differential evolution algorithms for each objective the results improve greatly, even when the combined algorithms behave poorly when evaluated alone at the tests. Most importantly, the comparison between the subpopulation algorithms and their related panmictic algorithms suggests that the competition between different strategies inside one population can have deleterious consequences for an algorithm and reveal a strong benefit of using the subpopulation framework.

Keywords

Structured Evolutionary Algorithms, Parallel Evolutionary Algorithms, Hybridization, Multi-objective Algorithms, Novelty Search, General Subpopulation Framework, General Differential Evolution.

1 Introduction

Although particle swarm optimization algorithms, differential evolution and genetic algorithms follow different lines of thought, they can all be seen from the same framework or structure. Not only these types but most of the algorithms in evolutionary computation share the same framework. They are based on a single population of individuals, which interacts in some form to produce new ones inside the same population.

To these types of algorithms, it is usually given the name of unstructured EAs or panmictic [43].

On the other hand, island based models and cellular algorithms achieved relevant improvements, indicating that the evolutionary bioinspiration, when extended to include concepts of subpopulation and neighborhood aspects, can be advantageous [47]. These types of algorithms are called structured models.

Nonetheless, the use of structured algorithms in multi-objective optimization has been under-explored [38]. Lately, we researchers start asking ourselves what could be the next step (future research trends) [5], since very simple and effective algorithms were developed, and it is hard to improve them without losing any of their benefits. This article tackles this problem from a different perspective. Here, we switch the focus from algorithms to frameworks.[1] Moreover, when changing from a panmictic to a structured framework, small and simple changes may give relevant improvements to the algorithms of the state of the art.

This article proposes the subpopulation framework which has the following features:

- *Integration Capability* - It allows for the addition of any number of algorithms which are integrated as subpopulations of the framework. Although this feature is not new, for example it was explored similarly in island models [32], here we show that not only evolutionary algorithms (EAs) but any optimization algorithm can be integrated in this framework. It is not required for these algorithms to be population based either (examples of how this can be constructed are given in Section 6.2).

- *General Formulation* - This framework is a general case for most of the structured approaches including but not limited to cellular algorithms and island based models (Section 6.1). The formalized subpopulation framework also generalizes the panmictic framework, because the panmictic framework is its special case when the number of subpopulations is fixed to 1 and the $\mathcal{IM}$ matrix set (that describes the interaction among subpopulations of the proposed framework, further explained in Section 6) can be ignored.

- *State of The Art Solutions* - Experimentally, it was shown that algorithms based on the subpopulation framework can achieve state of the art results (Section 9). In fact, results with the Subpopulation Algorithm based on Novelty (SAN) (Section 7.2) can be reasonably regarded as one of the most robust algorithms to date in multi-objective optimization, solving different types of problems in bi-objective and many objective settings with excellent results and surpassing the third version of the Generalized Differential Evolution algorithm (in short GDE3, currently one of the most suitable MOEA of the state of the art [11]) in most of the tests.

Experiments are conducted with two novel algorithms that implement the proposed subpopulation framework. These algorithms are developed based on single population ones (panmictic). The chosen panmictic algorithms, which were also used for comparison, are the GDE3 and a simple novelty search algorithm called Multi-Objective Novelty Algorithm (which is also a contribution of this article, described in Section 5.2) Here, the intention is to choose algorithms as different as possible to show some aspects of the subpopulation framework and its applicability to any type

[1] The definition of framework used in this article refers to a basic structure underlying a set of algorithms that is formalized and exemplified, enabling the understanding and analysis of a class of algorithms rather than a single one.

of algorithm. Notice that the dissimilarities in the GDE3 and Multi-Objective Novelty Algorithm arise from the fact that the former is objective-based while the later is novelty-based (further explanation of novelty search is given on Section 5). In fact, it will be shown that *the differences present in strategies of two or more subpopulations benefits their integration, in contrast with the competition which arises when different strategies are present in a single population*.

This article shows that simple subpopulations dynamics can give relevant improvements when combined with an algorithm of the state of the art in the proposed framework, demonstrating the strong benefits of the subpopulation framework. Additionally, the competition between different strategies inside the traditional single-population framework can have deleterious consequences for an algorithm. This is analyzed and verified experimentally in Section 9.5. Such problems confronted by the panmictic algorithms are similar to the ones confronted by the objective-based algorithms when contrasted with novelty-search based algorithms [30], since they are easily trapped in deceptive fitness landscapes. The solution provided by the subpopulation framework is that the presence of multiple populations with different dynamics will let the algorithm be less sensitive to local optima.

Finally, this article presents a discussion over an unexpected result, where the experimental results with a combination of three simple subpopulations achieved state of the art quality in the WFG Toolkit [21] (presented and explained in Sections 9.3 and 9.5).

Sections 2, 3, 4 and 5 review briefly the literature respectively in similar structured EAs, differential evolution in single-objective optimization, differential evolution multi-objective algorithms and novelty search areas. Thereafter, Section 6 proposes the general subpopulation framework. Section 7 describes two subpopulation algorithms which use as basis the general subpopulation framework. Section 8 presents the methodology used for comparison, Section 9 describes the problems' characteristics and shows the results obtained on them. Lastly, the conclusions are presented in Section 10.

2 Structured EAs

On one hand, the usual type of EAs pertain to a class of single population algorithms, which we call here single-population framework. But they are also known as panmictic EAs. On the other hand, there are other algorithms which spread their population into a structure with some defined interrelationship [1]. This paper will follow the definition that structured algorithms are any procedure which may have its population formulated with subpopulation groups, with the number of possible non-trivial subpopulation groups necessarily greater than one. For example, the simple EA can not be seen as a structured algorithm, since the number of possible subpopulation groups can never be formulated as greater than one [15]. Multi-objective ELSA is a local selection algorithm which also cannot be seen as a structured algorithm [36]. Note that some procedures, such as the restricted mating, fit in the previous definition of structured algorithms [53]. Therefore, restricted mating based algorithms can be seen as structured algorithms (see Section 6.1 for the complete description).

Parallel EAs are usually examples of structured EAs which are sometimes divided into three classes [17], [43]:

1. Island Model: The basic structure used by this model consists of multiple subpopulations, where a limited amount of genetic information is exchanged between any of them arbitrarily;

2. Stepping Stone Model: In this model a neighborhood relation is defined, where only the adjacent subpopulations can exchange information. Aside from that, it is defined in the same way as the Island Model;

3. Neighborhood Model: A complex single population structure, where individuals interact only with adjacent individuals.

The cellular algorithm [35] (also called fine grained model or lattice model) for example pertains to the third class.

According to [6], Parallel MOEA models can be divided also into three classes: global parallelization, coarse grain and fine grain. Global parallelization does not present any structured population aspect, while coarse grain (also called island GAs) and fine grain (also called cellular GAs) are parallel versions of structured algorithms already mentioned before. In [46], the classifications of the parallel models differ from the previous three classes, though from a population structure point of view they can still be converted to the previous three classes.

Other types of EAs were also developed where the evolutionary conditions differed from subpopulation to subpopulation. These were called nonstandard structured EAs and they were reviewed by Alba and Tomassini in [1]. Another extensive review of single-objective structured EAs can be found in the book of Tomassini [47].

Regarding multi-objective algorithms, there are also some algorithms which are structured. To cite some: multi-objective cellular algorithms [38], some rudimentary subpopulation algorithms [42], [8], spatial predator-prey MOEA [28] and multi-colony ant algorithms [23]. Spatial predator-prey MOEA defines an adjacency matrix with edges as solutions where the predator makes a random walk and erases the worst solution in the neighborhood which is related to a given objective [28]. The number of predators walking are as much as there are objectives. Ant colony optimization algorithms construct a population of solutions by sampling from a probabilistic model (usually in the form of a matrix of pheromone). This matrix of pheromone is constantly updated by the ants. Although they can not be defined as structured algorithms by the definition above, their multi-colony version can be defined. Multi-colony optimization algorithms use normally multiple matrices of pheromone with some rules to decide how and which pheromone matrix to be updated/used.

Moreover, a generalized framework of the structured algorithms is still non existent. This article fills this gap by formalizing a general unifying framework capable of representing most if not all of these structured models.

2.1 Related Methods

Although being a single population algorithm, AMALGAM is related to the proposed framework since both can be used to integrate algorithms. AMALGAM is a panmictic multiobjective algorithm that create a number of offspring points using genetic operators from different algorithms. Fast nondominated sorting is used to rank the offsprings together with the previous population, subsequently defining the next population [50].

As told before, one important difference between AMALGAM and the proposed framework is that the first is panmictic. Therefore, it has the disadvantage that multiple algorithms joined together may conflict with each other in the single pool of solutions. Another important difference is that AMALGAM can only define the integration of algorithms with biological models for population evolution, since genetic operators are necessary for the integration. Here, the proposed framework define the integration of any optimization algorithm.

The portfolio design proposed by [16] runs different algorithms (strategies) or copies of the same strategy with the objective of selecting the best strategy for the given problem. Details of how the selection and evaluation of strategies as well as the strategies themselves are dependent on the problem at hand [19]. The strategies run without communication between each other. Therefore, when considered under the light of the framework described here, the set of interaction matrices is null and can be ignored (interaction matrices are part of the frameword defined in Section 6). The similarities between this method and the proposed framework are limited to the use of multiple algorithms together.

3 Differential Evolution

The Differential Evolution (DE) is a meta-heuristic contained in the subfield of evolutionary computation, which can be employed for optimizing multi-dimensional real-value functions, where these functions are neither required to be continuous nor to be differentiable. It solves problems using a simple algorithm similar to the ones used by EAs, but the operators used by the DE are not based on the evolution of species [44]. The algorithm is described succinctly in Table 1 and the procedures of mutation, crossover and selection are explained in the following subsections.

Table 1: Differential Evolution Algorithm

1. Initialize population with random samples uniformly distributed over the search space
2. Repeat for each individual until a criterion of convergence is met
   (a) Mutation
   (b) Crossover
   (c) Selection
3. Return solution

3.1 Mutation

For each vector $x_{i,g}$, where i is the index of this vector (which relates to the individual index in the population, since each individual has its own vector) and g is the current generation where the vector takes place, the mutation is applied by creating a mutant vector based on a numerical operator described in Equation 1.

$$v_{i,g+1} = x_{r1,g} + F(x_{r2,g} - x_{r3,g}), \tag{1}$$

where $r1$, $r2$ and $r3$ are randomly selected individuals of the population, which must differ from the individual i. F is a parameter which should meet the condition $F \in [0, 2]$.

3.2 Crossover

During the crossover, trial vectors $u_{i,g+1}$ are created from a combination of the mutation vector $v_{i,g+1}$ and the original vector $x_{i,g}$. The trial vector created is expressed in

Equation 2.

$$u_{i,j,g+1} = \begin{cases} x_{i,j,g} & \text{if } rand() > CR \text{ and } j \neq rnd_i; \\ v_{i,j,g+1} & \text{if } rand() \leq CR \text{ or } j = rnd_i, \end{cases} \quad (2)$$

where $rand() \in [0, 1]$ is an uniformly distributed random number, $CR \in [0, 1]$ is a parameter passed to DE, j is the vector component index and rnd_i is a randomly chosen index, with the objective of choosing at least one component from the vector $v_{i,j,g+1}$.

3.3 Selection

The selection is the last step of the generation, where it is determined for each vector if the trial vector $u_{i,g+1}$ will substitute the original vector $x_{i,g}$ or not. For this, both the $u_{i,g+1}$ and $v_{i,g}$ are evaluated and the vector with better fitness function is kept, forming the next generation vector $x_{i,g+1}$.

3.4 Comparison with other Evolutionary Algorithms

The DE algorithm and its variations are known by their robustness, quality of the solutions, short running time, easy use and application to a wide range of applications not limited by the type of the objective function [44], [3].

Promising results were obtained in numerous different experiments. Two variations of it achieved the best solutions on all problems from ICEC'96 [45]. In the work of [49] it was shown to achieve more accurate solutions, faster and with greater robustness than Particle Swarm Optimization (PSO) and Evolutionary Algorithms (EAs). At the state of the art, the DE is still compared on equal grounds to complex optimization algorithms (e.g., Estimation of Distribution Algorithms) [14].

4 Differential Evolution based Multi-objective Methods

The DE was shown to achieve significant improvements over other single-objective [49] as well as in multi-criteria optimization algorithms [48], [11]. The reason behind this overall better results lies partially on the rotational invariant behavior of DE's operators, which adapts to the fitness landscape when compared with NSGA-II's genetic operators and other algorithms with similar genetic operators [22]. Recent studies show that in multi-objective-problems, DE is one of the best approaches when the problem size increases in scale [11].

There are various multi-objective methods based on differential evolution [4]. They can be divided into old versions of algorithms which used only Pareto dominance to select individuals and modern methods which use the Pareto dominance and a diversity measure for selection [48]. It is generally accepted that the last version of the generalized evolution algorithm (the GDE3 [25]) and the differential evolution multi-objective algorithm (DEMO) [41] are the representatives of the modern class of multi-objective algorithms based on DE [48], [11]. By taking into account that DEMO [41] is also similar to GDE3 [25] algorithm, without constraint handling and a fallback to the original DE in the case of single-objective, we will conduct the comparison and study on GDE3 solely.

Recently, a comparison between eight modern multi-objective algorithms was made [11]. They showed evidences that GDE3 is currently one of the most suitable MOEA of the state of the art. Among the results, it is stated that GDE3 tends not only to be faster, but also scales better in relation to the number of decision variables. In the tests made, there was only one other algorithm of the state of the art based on the PSO approach with similar performance.

4.1 General Differential Evolution 3

GDE3 has the same basic loop as the DE, with a modification in the selection phase and an addition of a pruning stage. In the selection phase, the algorithm considers the Pareto dominance and the constraints. Let s and t correspond respectively to the solution and its respective trial solution. Then, in the selection phase the following statements apply:

- If both s and t are not feasible. The trial solution t substitute s only when it dominates the solution s in unconstrained space;
- If one solution is feasible and the other is not feasible, the feasible solution is chosen;
- Finally, if both solutions are feasible, the solution which dominates the other is kept. However, if neither one dominates the other, both solutions are added to the next population, increasing the size of the population.

As a consequence of the modifications in the selection stage, the pruning stage was added to keep the population to a minimum because GDE3's selection phase described above can make the population increase in size. The pruning stage consists of sorting based on a diversity measure, consecutively selecting the first individuals to fill the next population size.

In the first version, GDE3 used crowding distance as its diversity measure [25], similar to NSGAII [7]. But in its most recent version the k-nearest neighbors measure was used as a distance measure. This metric was shown to be more consistent than the crowding distance measure when the number of objectives is greater than two [26]. The experiments conducted in this paper use GDE3 with a k-nearest neighbors measure.

5 Novelty Search

In nature, evolution is usually observed as an open-ended process which continually creates individuals with greater complexity and diversity [34]. Novelty search is a method developed by Lehman and Stanley that mimics the open-ended evolutionary process with a simple novelty metric [30], [29], rewarding novel individuals with a direct measure of novelty.

Moreover, in the perspective of optimization, problems are sometimes deceptive. This is usually the case for real world problems, because when problems increase in size and complexity it is improbable that a fitness function exists which can drive the algorithm directly to the goal. Novelty search aids the optimization in these deceptive spaces by identifying stepping stones, which are the novel individuals found by the novelty metric.

Recently, novelty search was used in very distinct areas such as neuro-evolution [37], [30], genetic programming [31], multi-objective evolution [37] and robotics [9], [10]. Moreover, there are an ever increasing number of articles with further evidence of novelty search benefits in deceptive problems. Some papers even showed the astonishing find that novelty search can be used sometimes as a substitute of objective-based search [30], [51]. The good results of novelty search in relation to objective-based search revealed that objective-based search may have deleterious effects on search.

5.1 Novelty Metric

For measuring the novelty of a solution, the novelty search relies on a metric which can be any equation capable of describing how much an individual is novel in comparison

with the past individuals of the archive. The usual metric used is the k-nearest neighbors which was also employed by Lehman and Stanley in their pioneering work on novelty search[29]. The following equation defines it exactly:

$$p(x) = \frac{1}{k}\sum_{i=1}^{k} dist(x, \mu_i), \tag{3}$$

where k is a parameter defined arbitrarily, μ_i is the i-th nearest neighbor of x according to the distance measure $dist()$. The distance measure is problem dependent. Usually, it is calculated in the behaviors space rather than fitness space, where behaviors space is composed as the small set of features which identifies a unique behavior (reducing the search space and differing in this way from exhaustive enumeration). The archive is an incremental set of individuals, receiving new individuals only if they surpass a novelty threshold n_{min} adjusted automatically by some rule.

It goes often unnoticed, but one of the problems of this novelty metric lies on its dynamic adjustment, i.e., the parameters used to update the archive. The following are the dynamics commonly used to update the metric:

- if more than n_a individuals entered the archive, multiply n_{min} by n_{inc};
- if n_r individuals did not enter in the archive, multiply n_{min} by n_{dec};

where n_a, n_r are positive integers (refers to the number of individuals), n_{inc}, $n_{dec} \in \mathbb{R}, n_{inc} > 1, 0 < n_{dec} < 1$ (refers to values of the novelty metric). These parameters define the rate of individuals which enter the archive. It follows that the bigger the archive is the more sensitive the novelty metric is to identify new individuals, because the higher the number of points, the less separated the points will be from each other. Then, a bigger archive is a direct result from a smaller n_{min} and consequently a more sensitive search with less chances of letting new individuals go unnoticed. On the other hand, a bigger archive makes the metric evaluation slower.

5.2 Multi-objective Novelty Algorithm (MONA)

In this Section, we propose MONA. The first algorithm to use novelty in a multi-objective context. The algorithm uses solely novelty search. Therefore, this algorithm follows the same line as the Lehman and Stanley study [30], hypothesizing that an algorithm based on the novelty alone might be better than objective based methods. MONA is a very simple algorithm proposed in this article, where the space of all the objectives is taken to be the behavior space of the novelty, differently from the Mouret approach [37] where novelty was seen as an additional objective. Table 2 describes the algorithm.

The purpose of this algorithm is to be a very simple algorithm, which will be compared as well as used in the general subpopulation framework, showing that from very simple bases efficient and robust algorithms can be constructed.

6 General Subpopulation Framework

The General Subpopulation Framework (GSF) is proposed here as an underlining structure of a class of multi-objective algorithms which unifies a number of structured EAs in its formalization. Additionally, it is capable of integrating different optimization algorithms without restrictions. This flexible ability of joining algorithms together is

Table 2: Multi-Objective Novelty Algorithm

1. Initialize population with random samples uniformly distributed over the search space
2. Repeat for each individual in the population until a criterion of convergence is met
   (a) Apply the same mutation and crossover operators as used by DE
   (b) Calculate the novelty metric
   (c) Verify if its novelty metric is above the n_{min} threshold, if it is above insert it on the archive (unlimited in size)
   (d) Update n_{min} (see Section 5.1)
   (e) Create a new population by sampling uniformly with replacement from the archive
3. Return the archive's non-dominated solutions as the solution set

important as it will be shown in the experiments. Mostly, because this type of cooperation between algorithms can sum their benefits while the competition between them in each subpopulation is decreased to a minimum.

In this context, we define:

Definition 1 Subpopulation
A subpopulation is a finite set of individuals related with a group of well defined dynamics. These dynamics are usually (although not necessarily) composed of interactions of these individuals with either themselves or individuals of other subpopulations. But they are not in any way limited to it.

When connecting these subpopulations together, a new matrix appears. To this matrix is given the name $\mathcal{IM}$. It is formally defined as follows:

Definition 2 $\mathcal{IM}$ - Subpopulation Interaction Probability Matrix Set
The subpopulation interaction probability matrix set $\mathcal{IM}$ is a set of matrixes of the form:

$$\mathcal{IM} = \{IM_1, IM_2, ..., IM_m\}, \tag{4}$$

where m is the number of types of interactions used in an optimization algorithm. And each IM_i corresponds to the following matrix:

$$IM_i = \begin{pmatrix} p_{i,1,1} & p_{i,1,2} & \cdots & p_{i,1,s} \\ p_{i,2,1} & p_{i,2,2} & \cdots & p_{i,2,s} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i,s,1} & p_{i,s,2} & \cdots & p_{i,s,s} \end{pmatrix}, \tag{5}$$

where s is the number of subpopulations and $p_{i,a,b}$ is the probability of an interaction i occurring in subpopulation a and taking as parameters the individuals of subpopulation b or the subpopulation b itself.

The evolutionary operators are examples of interactions. For example in the case of a subpopulation based version of DE's operators, let us assume their interactions are described by IM_d. Then, the trial vector would be, for each individual of this subpopulation, composed of three individuals chosen based on the probabilities of the IM_d matrix. Recall that the $\mathcal{IM}$ matrix set can be ignored in the case of only one subpopulation and this is why it can be ignored for panmictic algorithms.

Notice also that the interaction of each subpopulation can also differ from subpopulation to subpopulation. In the case of just one subpopulation k having an interaction i, IM_i would be of the following form:

$$IM_i = \begin{pmatrix} 0 & 0 & \cdots & 0, \\ \vdots & \vdots & \ddots & \vdots \\ p_{i,k,1} & p_{i,k,2} & \cdots & p_{i,k,s} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}. \tag{6}$$

Naturally, more complicated global dynamics might also be present, such as dynamical probabilities that depend on time t:

$$IM_i = \begin{pmatrix} p_{t,i,1,1} & p_{t,i,1,2} & \cdots & p_{t,i,1,s} \\ p_{t,i,2,1} & p_{t,i,2,2} & \cdots & p_{t,i,2,s} \\ \vdots & \vdots & \ddots & \vdots \\ p_{t,i,s,1} & p_{t,i,s,2} & \cdots & p_{t,i,s,s} \end{pmatrix}. \tag{7}$$

Additionally, the population size variable is extended to a vector version. Because the proposed framework has a number of subpopulations, each with a given size. This vector is hereby called $\mathcal{S}$ and is defined as follows:

Definition 3 $\mathcal{S}$ - Vector of Subpopulation Sizes
The subpopulations' sizes are defined by vector $\mathcal{S}$, which corresponds to:

$$S = (n\check{p}_1, n\check{p}_2, ..., n\check{p}_s), \tag{8}$$

where $n\check{p}_a$ is the size of subpopulation a. The *total subpopulation size* (ts) is naturally:

$$ts = \sum_{j=1}^{s} n\check{p}_j. \tag{9}$$

An equivalent and more convenient representation exists which is independent of the total subpopulation size. Let $np_a = \frac{n\check{p}_a}{ts}$, corresponding to the ratio of the total subpopulation. Then, the following representation is also verified:

$$\begin{aligned} &np_a \in \{x \in \mathbb{R} : 0 < x < 1\} \\ &\sum_{j=1}^{s} np_j = 1. \end{aligned} \tag{10}$$

With the previous definitions it is possible to describe explicitly the GSF:

Definition 4 GSF - General Subpopulation Framework
Suppose we have s subpopulations, then $\mathcal{P}$ is the set of subpopulations $\mathcal{P} = \{P_1, ..., P_s\}$

and $\mathcal{A}$ is the set of panmictic algorithms $\mathcal{A} = \{A_1, ..., A_s\}$ where a subpopulation P_i is constructed by an algorithm (strategy) A_i. Therefore, the GSF is defined as a 4-tuple $< \mathcal{P}, \mathcal{A}, \mathcal{S}, \mathcal{IM} >$, where $\mathcal{S}$ and $\mathcal{IM}$ were previously defined as respectively the vector of subpopulation sizes and the set of interaction probability matrices.

The subpopulations may even be used to join arbitrary algorithms which may not even be based on populations. That is, as long as each algorithm can generate a set of solutions to compose the subpopulation which is representative of its dynamics, the subpopulation framework can handle the joining process (examples are given in Sections 6.2 and 7). For example, in the case of the random search algorithm the subpopulation can be constructed from the last generated solutions. Therefore, to the knowledge of the authors, any algorithms can be joined (mixed) by using this framework. Naturally, for the inclusion of an algorithm in this framework it is also relevant but not necessary to have:

- Dynamics taking into account different individuals of its population (which can be modified to handle any individual of any population by the $\mathcal{IM}$ set of matrices);
- Different dynamics from the other subpopulations present in the framework. This can be relevant, since the higher the similarities between subpopulations are the less important the subpopulations become, in other words, multiple subpopulations with similar dynamics will produce results similar to a single population.

The following subsections demonstrate how GSF can represent most of the optimization algorithms. Section 6.1 shows how GSF can represent various types of structured EAs, while Section 6.2 gives two examples of famous algorithms (one a panmictic EA and the other a non-evolutionary algorithm) as well as shows how they can be transformed to the GSF approach without losing many of their characteristics. In Section 7, two new algorithms are proposed based on their related panmictic algorithms. This time, however, the objective is not merely illustrative, since the algorithms described possess important features described in detail later on. In fact, these important features enable them to surpass algorithms of the state of the art.

6.1 Representation Capabilities

The general subpopulation framework can represent various types of structured EAs, including:

- Island-Based Models[47] - Each panmictic island forms a subpopulation P_i with the set of algorithms $\mathcal{A}$ containing identical algorithms for all subpopulations. Let the number of panmictic islands be s, then $\mathcal{S} = (\frac{1}{s}, ..., \frac{1}{s})$ and $|A| = |P| = s$. Between the subpopulations an interaction defined by the exchange of genetic information can be formalized with an IM_1 matrix of the form:

$$IM_1 = \begin{pmatrix} 0 & p_{i,1,2} & \cdots & p_{i,1,s} \\ p_{i,2,1} & 0 & \cdots & p_{i,2,s} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i,s,1} & p_{i,s,2} & \cdots & 0. \end{pmatrix} \tag{11}$$

  That is, each individual selected for exchange must necessarily go to another subpopulation, therefore the diagonal entries are zero. This dynamic is usually the unique one which other subpopulations can participate in. Inside the algorithms other dynamics can take place (e.g., crossover) and these would have also a trivial

set of IM_is with the only non-null probabilities residing on its diagonal (i.e., the interactions happen only inside the same subpopulation).

- Cellular Algorithms[35] - This type of algorithm can be thought as the opposite line of thought in comparison with Island-Based Models, where the number of subpopulations is maximized with the minimum possible size of subpopulations, i.e., cellular algorithms can be seen as a large number of subpopulations P_i of equal size 1. Let the number of cells in a given cellular algorithm be s, then $\mathcal{S} = (\frac{1}{s}, ..., \frac{1}{s})$ and $|P| = s$ with each individual cell corresponding to a subpopulation P_i and the update of each cell can be divided into s algorithms forming the $\mathcal{A}$ set of panmictic algorithms. Consider the case of a cellular algorithm with nine individuals with a von Neumann neighborhood, then it possesses nine subpopulations and an IM_c matrix defined by:

$$IM_c = \begin{pmatrix} 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 & \frac{1}{4} & 0 & 0 & \frac{1}{4} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 \end{pmatrix}. \tag{12}$$

  Moreover, all interactions of cellular algorithms use the same neighborhood, therefore the set of matrices $\mathcal{IM}$ is given by:

$$\mathcal{IM} = \{IM_c, IM_c, ..., IM_c\}. \tag{13}$$

  In some certain cellular algorithms, a dynamical IM_c has to be used to represent the change of neighborhood of each cell.

- Restricted Mating [53] - Some procedures although not related to subpopulations at first glance, can be converted to this formalization. Restricted mating, for example, can be formalized with subpopulations. By considering each subpopulation containing only one individual, we have the restricted mating interaction defined by:

$$IM_1 = \begin{pmatrix} 0 & p_{1,2} & \cdots & p_{1,s} \\ p_{2,1} & 0 & \cdots & p_{2,s} \\ \vdots & \vdots & \ddots & \vdots \\ p_{s,1} & p_{s,2} & \cdots & 0 \end{pmatrix}, \tag{14}$$

  when for any (a, b) pair, $p_{a,b}$ becomes:

$$u_{a,b} = \begin{cases} 1 & \text{if } dist(a,b) < \sigma; \\ 0 & \text{otherwise,} \end{cases} \tag{15}$$

$$p_{a,b} = \frac{u_{a,b}}{\sum_{i=1}^{s} u_{a,i}} \tag{16}$$

  σ is an arbitrary threshold and $dist(a, b)$ is usually the Euclidean distance between solutions a and b [53].

- Spatial Predator-Prey MOEA [28] - This algorithm defines an adjacency matrix G with edges as solutions where the predator makes a random walk. This algorithm can be reformulated into the subpopulation framework by considering as interaction the replacement of the preys selected by the predators. Although the replacement can be done of multiple ways, only the edges in the predator's neighborhood

participate. Therefore, for the replacement interaction, each position (x, y) of the interaction matrix becomes:

$$IM_1(x, y) = min\{G(k, x), G(k, y)\}, \tag{17}$$

where k is the edge of the predator responsible for this interaction matrix. Basically, two solutions can only interact if they are in the k (predator's edge) neighborhood.

- Multi-colony Ant Algorithms - Ant colony optimization algorithms in general are difficult to map into the subpopulation framework because they use population models instead of the solutions themselves. This problem is faced similarly when trying to convert estimation of distribution algorithms [40], [27]. Additionally, some of these methods do not possess a population structure. For example, ant colony optimization algorithms with one colony do not use a structure approach to optimization following the definition above, i.e., although the construction of the solutions by the ants use solution components organized in a structured way, the population of solutions itself is not structurally formulated [23]. However, some of them such as the multi-colony ant algorithms do have a population structure. In this case, it is possible to approximate roughly the population model (e.g., the pheronomone matrix) as a subpopulation and consider the interrelation between them as interactions with their respective interaction matrices. That is, the pheromone matrices update interaction can be represented as:

$$IM_1 = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}, \tag{18}$$

when the update is only realized at the original colony. And when the update is done by region ($\{L_1, L_2, ..., L_s\}$) in the nondominated front, for a given solution a we have:

$$IM_1 = \begin{pmatrix} a \in L_1 & a \in L_2 & \dots & a \in L_s \\ a \in L_1 & a \in L_2 & \dots & a \in L_s \\ \vdots & \vdots & \ddots & \vdots \\ a \in L_1 & a \in L_2 & \dots & a \in L_s \end{pmatrix}. \tag{19}$$

6.2 Examples of Panmictic to GSF Conversion

This subsection shows how optimization algorithms of almost any type can be converted to multi-population versions represented by the GSF. Examples of both the simple genetic algorithm [15] and the simulated annealing [24] will be presented. Their $\mathcal{IM}$ matrix sets will be defined and, among other things, it will be shown how their dynamics could be used to affect other subpopulations. Notice that the conversions will not make explicit the vector of subpopulations sizes $\mathcal{S}$, since this parameter is not related with the representation and thus it can be established independently.

6.2.1 Simple Genetic Algorithm

There are three basic procedures in a simple genetic algorithm: crossover, mutation and selection. However, mutation does not depend on other individuals and selection is executed over a set of individuals of its own population. Then, it does not make sense

to define an interaction matrix for them. The mutation and selection can be normally applied, with the only difference from the single population version being that the target is now the current subpopulation (i.e. not the entire population). In fact, this slight modification defines the algorithm A_i which constructs its respective subpopulation P_i under the GSF formulation.

Thus, let us define the $\mathcal{IM}$ set, which consists of only the crossover interaction ($\mathcal{IM} = \{IM_1\}$). The crossover interaction matrix IM_1 defines the probabilities that an individual of a given subpopulation participate in the crossover. The exact values of the IM_1 is the trivial $IM_1 = 1$. Note that the simple GA is not a structured algorithm (there are not any other subpopulation to interact with). However, the designer might want to modify IM_1 when joining this algorithm with other algorithms.

6.2.2 Simulated Annealing

One of the main difficulties that can be spotted on the simulated annealing is that it is not a population-based algorithm. This problem can be circumvented by adding the recent modifications of the variables' values in a First In First Out data structure, creating a subpopulation derived from its dynamics. Therefore, the simulated annealing algorithm plus the creation of a subpopulation defines algorithm A_i to be applied on its created subpopulation P_i.

Lastly, the interaction matrices are defined by an empty set ($\mathcal{IM} = \{\}$), since there is no interaction between solutions in its dynamics. An empty $\mathcal{IM}$ might be unappealing at first glance, but when joined with the subpopulations of other algorithms, the subpopulation constructed by this algorithm might be used by other interactions and consequently influence the global dynamics.

6.3 What is the benefit of using GSF to describe a panmictic algorithm?

It was shown before that panmictic algorithms can be converted to the GSF. However, they possess a trivial IM and bring little explanation. Thus, one might question about the usefulness of such a conversion.

The answer is that, once converted to the GSF, any panmictic algorithms can be integrated seamlessly as a subpopulation in other GSF based algorithms. Section 7 will show some examples of algorithms constructed using the GSF.

Last but not least, the pressures of different panmictic algorithms can be compared by weighting their subpopulations' sizes. Comparison of algorithms is an important and complicated subject which is aided by GSF. GSF also enables a relatively easy evaluation of the cooperation between algorithms, facilitating the construction of hybrid algorithms with the simple addition or deletion of subpopulations.

6.4 What is the benefit of using GSF?

One feature of the subpopulation framework is the division of interactions over interaction matrices. Thus, one can separate only the interactions under interest and compare their structural behavior by looking at those matrices. For example, it is possible to see that both spatial predator-prey and cellular algorithms are similar in the sense that both use similar interaction matrices (neighborhood matrices).

Moreover, designing structured algorithms may become easier by looking at different interactions and interaction matrices instead of multiple structures and their internal behavior. The framework also aids other abstractions such as a mix between structures (i.e., sometimes the structure behave like a cellular algorithm and sometimes like a island model) by the simple inclusion of other interaction matrices. For example,

the inclusion of a cellular's interaction matrix into a island model algorithm.

7 Evaluation of General Subpopulation Algorithms (GSAs)

To evaluate the subpopulation framework appropriately, we elaborate two subpopulation algorithms: one based on GDE3 (see Section 4.1) and the other based on MONA (see Section 5.2). We will hereby call these GSAs respectively the Subpopulation Algorithm based on General Differential Evolution (SAGDE) and the Subpopulation Algorithm based on Novelty (SAN).

Both SAGDE and SAN are motivated by the fact that single-objective DEs evolved at each objective usually achieve good results. Take for example the WFG1 problem [21]. If we apply a GSA made uniquely of subpopulations of single-objective DEs, each evolving a different single objective, we achieve usually the result plotted in Figure 1. Note that the DEs achieve good results on each single objective, with the resultant individuals very close to the Pareto front, but the front is hardly covered. Then, what if another subpopulation is added to this algorithm, which might wisely "mix" these DEs solutions? The following algorithms are motivated by this question and in Section 9 an extensive answer is given based on the experiments.

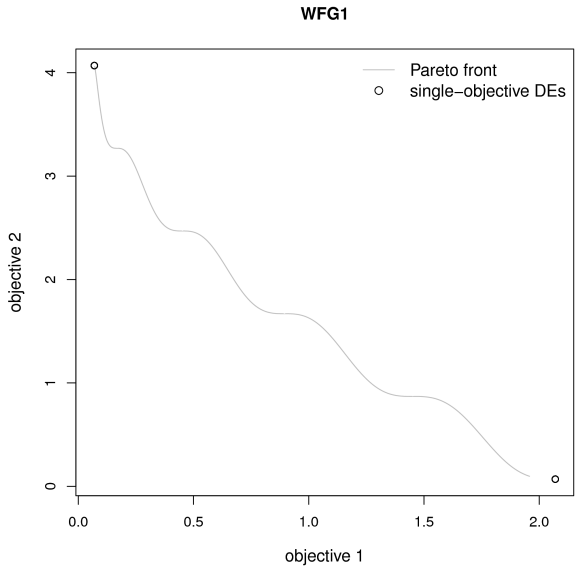

Figure 1: Solutions of single-objective DEs, each one evolving a different objective. For the test, the WFG1 problem is used with 2 objectives, 20 distance parameters and 4 position parameters. Each DE had a population of 50 individuals, with $CR = 0.6$, $F = 0.5$, and maximum number of generations of 25000.

7.1 SAGDE

In a problem with n objectives, SAGDE has $n + 1$ subpopulations $\mathcal{P} = \{P_1, ..., P_{n+1}\}$, where $\{A_1, ..., A_n\}$ are single-objective DEs with each one evolving a different objective and the GDE3 (multi-objective algorithm) is used as the algorithm A_{n+1} for the subpopulation P_{n+1}. The GDE3 subpopulation as well as the n single-objective differ-

ential evolution subpopulations behave in the same way as usual aside from the fact that an uniform matrix IM_1 (shown in Equation 20) is used to determine which individual will be part of the trial vector in the differential operator, i.e., $\mathcal{IM} = \{IM_1\}$ where:

$$IM_1 = \begin{pmatrix} \frac{1}{n+1} & \frac{1}{n+1} & \cdots & \frac{1}{n+1} \\ \frac{1}{n+1} & \frac{1}{n+1} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n+1} & \frac{1}{n+1} & \cdots & \frac{1}{n+1} \end{pmatrix}. \tag{20}$$

7.2 SAN

In the same way as SAGDE, SAN has $n+1$ subpopulations $\mathcal{P} = \{P_1, ..., P_{n+1}\}$ with n of them made of single-objective DEs ($\{A_1, ..., A_n\}$), where n is the number of objectives of the problem. Each single-objective DE optimizes a different objective and there is an additional subpopulation corresponding to the MONA (A_{n+1}) (multi-objective algorithm based on the novelty search approach proposed by this article, see Section 5.2).

Both MONA and the n single-objective DE subpopulations behave in the same way as usual with the unique differences being the use the same uniform matrix IM_1 described in Equation 20 (i.e., an individual chosen has an uniform probability of $\frac{1}{n+1}$ of coming from any subpopulation) to select individuals for the trial vector in the DE operator used in both algorithms. Moreover, MONA verifies any new individuals generated by any subpopulation for inclusion in the novelty archive (i.e., not only its own generated individuals). In other words, the inclusion of solutions in the novelty archive is a different interaction defined by IM_2. It is activated every time a new solution is created in any subpopulation, IM_2 matrix is defined below:

$$IM_2 = \begin{pmatrix} 0 & 0 & \dots & 1 \\ 0 & 0 & \dots & 1 \\ 0 & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}, \tag{21}$$

where the last column is referent to the MONA's subpopulation.

8 Comparison Methodology

To compare algorithms the following procedure is used:

1. Realize multiple runs of the algorithm and store the solution sets.
2. For each solution set do:
   - Compute the hypervolume indicator (Section 8.1.1);
   - Compute the ϵ indicator (Section 8.1.2);
   - Store each quality indicator result in a separate vector.
3. Algorithms are compared in three ways:
   - A group of algorithms is compared using their respective quality indicator's mean value and standard deviation. Algorithms with mean value inside the standard deviation of the best mean value are considered equally good.

- Verify the statistical significance between a pair of algorithms with a non-parametric Mann-Whitney test [20]. The alternative hypothesis that one method has a better (smaller) quality indicator than the other is accepted if the p-value is lower than 0.05.
- Calculate the 50% attainment surface (Section 8.2) based on the solution sets.

8.1 Quality Indicators

In this article, to compare the quality of the algorithms, the hypervolume indicator [52],[2] and the ϵ indicator [54] are used. These unary quality indicators were recommended by [13], since they are based on different preference information. The following subsections define these quality indicators.

8.1.1 Hypervolume indicator

The hypervolume indicator (I_h) is defined as the difference between the hypervolume of the Pareto front and the hypervolume of the non-dominated solution set in objective space [52],[2]. This indicator requires a reference point for the calculation, therefore the nadir point is used in this article.

8.1.2 ϵ indicator

The ϵ indicator (I_ϵ) is defined as the minimum factor ϵ by which a non-dominated approximation set (i.e., set of objective vectors which do not dominate each other) is worse than the Pareto optimal front. Let a and p be vectors in Z (the objective space) with $Z \subseteq \mathbb{R}^{+d}$ where d is the number of objectives, then the ϵ dominance between two vectors is defined by Equation 22.

$$a \succeq_\epsilon p \equiv \forall i \in [1, d] : a_i \leq \epsilon \cdot p_i. \tag{22}$$

Then, according to [54], the ϵ indicator is formally defined in Equation 23.

$$I_\epsilon(T) = \inf_{\epsilon \in \mathcal{R}} \{\forall p \in O \; \exists a \in T : a \succeq_\epsilon p\}, \tag{23}$$

where T is the target approximation set and O is the Pareto optimal set. In this paper O refers to a reference set which approximates the Pareto optimal set.

As shown in [39], quality indicators may be misleading. Therefore, when visually possible, attainment surfaces were also computed for the comparison.

8.2 Attainment Surfaces

Attainment surface (AS) is the boundary in objective space of the dominated area for a single run of an algorithm. They are important because such surfaces show detailed information about the performance differences between algorithms. To infer a statistically significant attainment surface, multiple runs of the algorithms are required and an approximated mean result is calculated. Usually, the 50% attainment surface is used as a mean measure approximation, which is defined as the area dominated by at least 50% of the approximation sets [18],[12]. In this paper, the code provided by [33] is used to obtain the 50% attainment surfaces.

9 Experiments

Some of the usual benchmarks of multi-objective problems poorly represent important classes such as non-separable and multimodal problems. Therefore, this paper makes use of a relatively recent set of tests called WFG [21]. The WFG set of problems present a

Table 3: Properties of the WFG test problems.

Problem	Obj.	Separable	Modality	Bias	Geometry
WFG1	$f_{1:M}$	yes	uni	polynomial,flat	convex,mixed
WFG2	$f_{1:M-1}$	no	uni	-	convex,disconnected
	f_M	no	multi	-	
WFG3	$f_{1:M}$	no	uni	-	linear,degenerate
WFG4	$f_{1:M}$	yes	multi	-	concave
WFG5	$f_{1:M}$	yes	deceptive	-	concave
WFG6	$f_{1:M}$	no	uni	-	concave
WFG7	$f_{1:M}$	yes	uni	parameter dependent	concave
WFG8	$f_{1:M}$	no	uni	parameter dependent	concave
WFG9	$f_{1:M}$	no	multi,deceptive	parameter dependent	concave

varied set of properties which can test the scalability of algorithms in both parameters and number of objectives. In Table 3 there is a summary of the characteristics of its test problems.The WFG Toolkit makes use of position and distance parameters. In one hand, when a distance parameter is modified the new solution may dominate, be dominated or be equivalent to the previous one. On the other hand, when a position parameter is modified the new solution is either incomparable or equivalent to the previous one. Tests were performed for the WFG problems with 20 distance parameters and 4 position parameters, resulting in 24 parameters to be optimized.

9.1 Results and Discussions

Each empirical attainment surface and quality indicator was calculated based on 30 solution sets, which were obtained from multiple independent runs of the algorithm in question. Different seeds were used for each algorithm run. Both the maximum number of generations and the *total subpopulation size*[2] (or population size in the case of panmictic algorithms) were fixed to respectively 25000 and 100. This fact assures that all algorithms have the same number of evaluations.

9.2 Choice of Parameters

Table 4 shows the parameters used for GDE3. They correspond to the same used by Kukkonen and Lampinen [26]. The reader may observe that when compared with usual single-objective DE's settings, the parameters of all algorithms possess a lower value of CR and F. This happens because multi-objective optimization maintain a high diversity. Therefore, it is not necessary to have a higher value of F or CR for better exploration of the search space, because individuals are different enough and the trial vectors are also suitably different. Tests with even smaller values of F were shown to improve the coverage ($F = 0.1$), but with great impacts on the distance to the Optimal Pareto Front (OPF). The gain in coverage was not enough to surpass SAN's coverage and the distance to the front was poorer enough, such that GDE3 was surpassed by SAN in all problems tested (even on some problems that it performed similarly to SAN with $F = 0.5$).

In the case of GSA's algorithms, F should be logically an even lower value. This is justified by the fact that GSA's subpopulations are usually very different from each other. We conducted preliminary tests with $F = 0.5$ and many results were the same

[2]Note that the variables *subpopulation size* and *total subpopulation size* are different from each other. The total subpopulation is defined in Section 6.

Table 4: Parameter's Table. The first two ratios of the $\mathcal{S}$ vector correspond to the subpopulations of DEs used and the third ratio is either MONA (for the SAN) or GDE3 (for the SAGDE).

	GDE3
CR	0.1
F	0.5
	MONA
CR	0.1
F	0.1
n_{inc}	1.1
n_{dec}	0.999
n_a	1
n_r	50000
	SAGDE
CR	0.1
F	0.1
$\mathcal{IM}$	uniform
$\mathcal{S}$	$(0.1, 0.1, 0.8)$
	SAN
CR	0.1
F	0.1
$\mathcal{IM}$	uniform
$\mathcal{S}$	$(0.3, 0.3, 0.4)$
n_{inc}	1.1
n_{dec}	0.999
n_a	1
n_r	50000

as the ones obtained with $F = 0.1$, though some problems showed as expected a slightly worse result. For MONA and SAN, the novelty parameters were decided upon a quality-efficiency trade-off, with both algorithms having the same fixed parameters.

Regarding the chosen subpopulation sizes of SAN and SAGDE, they are directly related to subpopulation's algorithm strength to "mix" the solutions of the single-objective DEs' subpopulations. Some subpopulations "mix" better the solutions than others (directly related to the coverage of the OPF), requiring a smaller subpopulation size (MONA subpopulation), while other subpopulations require a bigger subpopulation size to get a similar coverage (GDE3). This happens specially because GDE3 have various strategies and coverage is just one of its strategies. Recall that in SAN and SAGDE there are two single-objective DEs. These algorithms explore the problems as shown in Figure 1 and discussed in Section 7. Therefore, "mixing" the solution is necessary for coverage and this is only achieved by other subpopulations (GDE3 and MONA subpopulations for respectively SAGDE and SAN).

9.3 Study on Bi-objective Optimization

Tests were performed for the WFG problems with two objectives. Parameters used by the algorithms are fixed and summarized in Table 4.

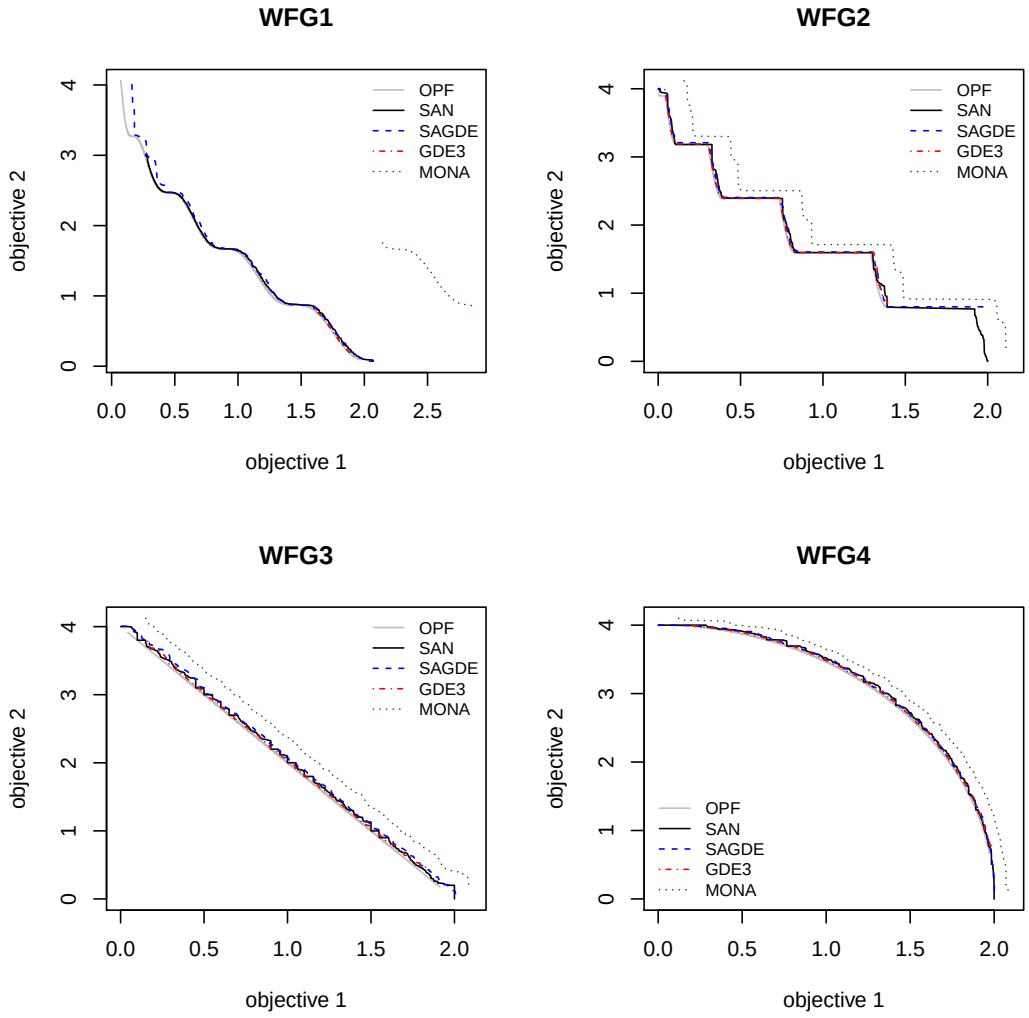

Figure 2: 50% attainment surfaces for the WFG Toolkit problems (minimization problems). Calculated for 30 independent runs.

The comparison between the 50% attainment surfaces of SAN, SAGDE, GDE3 and MONA is shown in Figures 2 and 3. Before discussing the results it is necessary to shown Tables 5 and 6 with the mean and standard deviation (sd) of ϵ and hypervolume quality indicators as well as Tables 7 and 8 with the statistical significance of both quality indicators. Most of the time the tables and figures agree with each other. Therefore, when not stated otherwise, the discussion concerns the overall behavior of all three comparisons (attainment surfaces, mean/sd and statistical hypothesis testing) For more information on the construction of these tables and figures please refer to Section 8 or to the tables and figures themselves.

Regarding the comparison between SAGDE and GDE3. SAGDE is significantly better than the GDE3 in the WFG1 for both quality indicators (clearly observable in

Table 5: ϵ indicator's mean and standard deviation for SAN, SAGDE, GDE3 and MONA. For each problem the best mean value as well as the other mean values inside the standard variation of the best mean value are marked in **bold**.

	SAN	SAGDE	GDE3	MONA
Problems	mean (sd)	mean (sd)	mean (sd)	mean (sd)
WFG1	**0.20(0.03)**	**0.18(0.09)**	1.53(0.02)	2.06(0.06)
WFG2	**0.05(0.01)**	0.10(0.11)	0.42(0.37)	0.18(0.02)
WFG3	**0.07(0.01)**	0.14(0.03)	0.29(0.13)	0.19(0.02)
WFG4	**0.07(0.01)**	0.10(0.03)	0.28(0.23)	0.14(0.01)
WFG5	**0.12(0.01)**	0.17(0.03)	0.37(0.10)	0.22(0.02)
WFG6	**0.11(0.01)**	0.16(0.04)	0.44(0.30)	0.19(0.02)
WFG7	**0.08(0.01)**	0.10(0.02)	0.41(0.16)	0.16(0.01)
WFG8	**0.23(0.01)**	0.29(0.10)	0.41(0.31)	0.33(0.02)
WFG9	**0.09(0.01)**	0.15(0.06)	0.15(0.18)	0.18(0.01)

Table 6: Hypervolume indicator's mean and standard deviation for SAN, SAGDE, GDE3 and MONA. For each problem the best mean value as well as the other mean values inside the standard variation of the best mean value are marked in **bold**.

	SAN	SAGDE	GDE3	MONA
Problems	mean (sd)	mean (sd)	mean (sd)	mean (sd)
WFG1	**0.21(0.04)**	**0.24(0.12)**	3.37(0.08)	6.23(0.15)
WFG2	**0.02(0.01)**	0.10(0.10)	0.25(0.20)	0.67(0.11)
WFG3	**0.13(0.02)**	0.23(0.04)	**0.13(0.02)**	0.75(0.15)
WFG4	**0.09(0.01)**	0.11(0.01)	**0.08(0.01)**	0.46(0.05)
WFG5	**0.37(0.02)**	0.42(0.02)	**0.35(0.02)**	0.86(0.10)
WFG6	**0.34(0.02)**	**0.37(0.03)**	**0.32(0.23)**	0.78(0.12)
WFG7	**0.10(0.01)**	**0.12(0.01)**	**0.10(0.03)**	0.51(0.07)
WFG8	0.87(0.05)	0.88(0.24)	**0.62(0.08)**	1.28(0.10)
WFG9	**0.21(0.02)**	0.35(0.20)	**0.15(0.12)**	0.72(0.03)

Table 7: P-values of comparison between SAN, SAGDE, GDE3 and MONA algorithms with Mann-Whitney significance test using the ϵ indicator. Results are marked in **bold** when the null hypothesis is rejected with a significance level of $\alpha = 0.05$. The alternative hypothesis is that the algorithm in the row is statistically better (smaller quality indicator) than the algorithm in the column.

Algorithm	Problem	SAN	SAGDE	GDE3	MONA
SAN	WFG1		0.99	$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG2		$\mathbf{0.01}$	$\mathbf{1.7e-4}$	$\mathbf{8.4e-18}$
	WFG3		$\mathbf{4.4e-11}$	$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG4		$\mathbf{4.1e-8}$	$\mathbf{1.2e-11}$	$\mathbf{8.4e-18}$
	WFG5		$\mathbf{3.9e-11}$	$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG6		$\mathbf{1.5e-10}$	$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG7		$\mathbf{2.1e-7}$	$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG8		0.35	$\mathbf{2.1e-4}$	$\mathbf{8.4e-18}$
	WFG9		$\mathbf{1.3e-6}$	$\mathbf{5.4e-4}$	$\mathbf{8.4e-18}$
SAGDE	WFG1	$\mathbf{5.0e-3}$		$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG2	0.98		$\mathbf{2.6e-3}$	$\mathbf{1.3e-10}$
	WFG3	0.99		$\mathbf{1.2e-7}$	$\mathbf{9.3e-8}$
	WFG4	0.99		$\mathbf{9.9e-5}$	$\mathbf{2.4e-6}$
	WFG5	0.99		$\mathbf{4.8e-14}$	$\mathbf{9.4e-8}$
	WFG6	0.99		$\mathbf{8.3e-13}$	$\mathbf{1.2e-4}$
	WFG7	0.99		$\mathbf{5.6e-16}$	$\mathbf{9.1e-12}$
	WFG8	0.65		$\mathbf{0.02}$	$\mathbf{2.3e-3}$
	WFG9	0.99		0.95	$\mathbf{0.01}$
GDE3	WFG1	1	1		$\mathbf{8.4e-18}$
	WFG2	0.99	0.99		0.63
	WFG3	1	0.99		0.99
	WFG4	0.99	0.99		0.92
	WFG5	1	0.99		0.99
	WFG6	1	0.99		0.99
	WFG7	1	1		0.99
	WFG8	0.99	0.97		0.12
	WFG9	0.99	$\mathbf{0.04}$		$\mathbf{5.4e-7}$
MONA	WFG1	1	1	1	
	WFG2	1	0.99	0.37	
	WFG3	1	0.99	$\mathbf{4.1e-4}$	
	WFG4	1	0.99	0.07	
	WFG5	1	0.99	$\mathbf{5.4e-10}$	
	WFG6	1	0.99	$\mathbf{1.7e-10}$	
	WFG7	1	0.99	$\mathbf{1.8e-12}$	
	WFG8	1	0.99	0.87	
	WFG9	1	0.98	0.99	

Table 8: P-values of comparison between SAN, SAGDE, GDE3 and MONA algorithms with Mann-Whitney significance test using the hypervolume indicator. Results are marked in **bold** when the null hypothesis is rejected with a significance level of $\alpha = 0.05$. The alternative hypothesis is that the algorithm in the row is statistically better (smaller quality indicator) than the algorithm in the column.

Problem	Algorithm	SAN	SAGDE	GDE3	MONA
SAN	WFG1		0.40	$\mathbf{8,4e-18}$	$\mathbf{8.4e-18}$
	WFG2		$\mathbf{3.7e-13}$	$\mathbf{7.6e-12}$	$\mathbf{8.4e-18}$
	WFG3		$\mathbf{7.8e-14}$	0.53	$\mathbf{8.4e-18}$
	WFG4		$\mathbf{1.6e-4}$	0.99	$\mathbf{8.4e-18}$
	WFG5		$\mathbf{4.5e-12}$	0.99	$\mathbf{8.4e-18}$
	WFG6		$\mathbf{1.6e-4}$	0.99	$\mathbf{8.4e-18}$
	WFG7		$\mathbf{6.8e-4}$	0.98	$\mathbf{8.4e-18}$
	WFG8		0.74	0.99	$\mathbf{8.4e-18}$
	WFG9		0.46	0.99	$\mathbf{8.4e-18}$
SAGDE	WFG1	0.60		$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG2	0.99		0.1	$\mathbf{3.1e-15}$
	WFG3	0.99		0.99	$\mathbf{8.4e-18}$
	WFG4	0.99		0.99	$\mathbf{8.4e-18}$
	WFG5	0.99		1	$\mathbf{8.4e-18}$
	WFG6	0.99		0.99	$\mathbf{8.4e-18}$
	WFG7	0.99		0.99	$\mathbf{8.4e-18}$
	WFG8	0.26		0.99	$\mathbf{1.2e-9}$
	WFG9	0.53		0.99	$\mathbf{8.4e-18}$
GDE3	WFG1	1	1		$\mathbf{8.4e-18}$
	WFG2	0.99	0.89		$\mathbf{2.5e-16}$
	WFG3	0.46	$\mathbf{2.9e-14}$		$\mathbf{8.4e-18}$
	WFG4	$\mathbf{1.6e-4}$	$\mathbf{9.3e-10}$		$\mathbf{8.4e-18}$
	WFG5	$\mathbf{5.1e-4}$	$\mathbf{1.0e-16}$		$\mathbf{8.4e-18}$
	WFG6	$\mathbf{2.6e-8}$	$\mathbf{3.3e-11}$		$\mathbf{2.4e-13}$
	WFG7	$\mathbf{0.01}$	$\mathbf{4.6e-4}$		$\mathbf{8.4e-18}$
	WFG8	$\mathbf{1.9e-13}$	$\mathbf{1.0e-9}$		$\mathbf{8.4e-18}$
	WFG9	$\mathbf{7.1e-10}$	$\mathbf{2.3e-10}$		$\mathbf{5.9e-17}$
MONA	WFG1	1	1	1	1
	WFG2	1	0.99	1	
	WFG3	1	1	1	
	WFG4	1	1	1	
	WFG5	1	1	1	
	WFG6	1	1	0.99	
	WFG7	1	1	1	
	WFG8	1	0.99	1	
	WFG9	1	1	1	

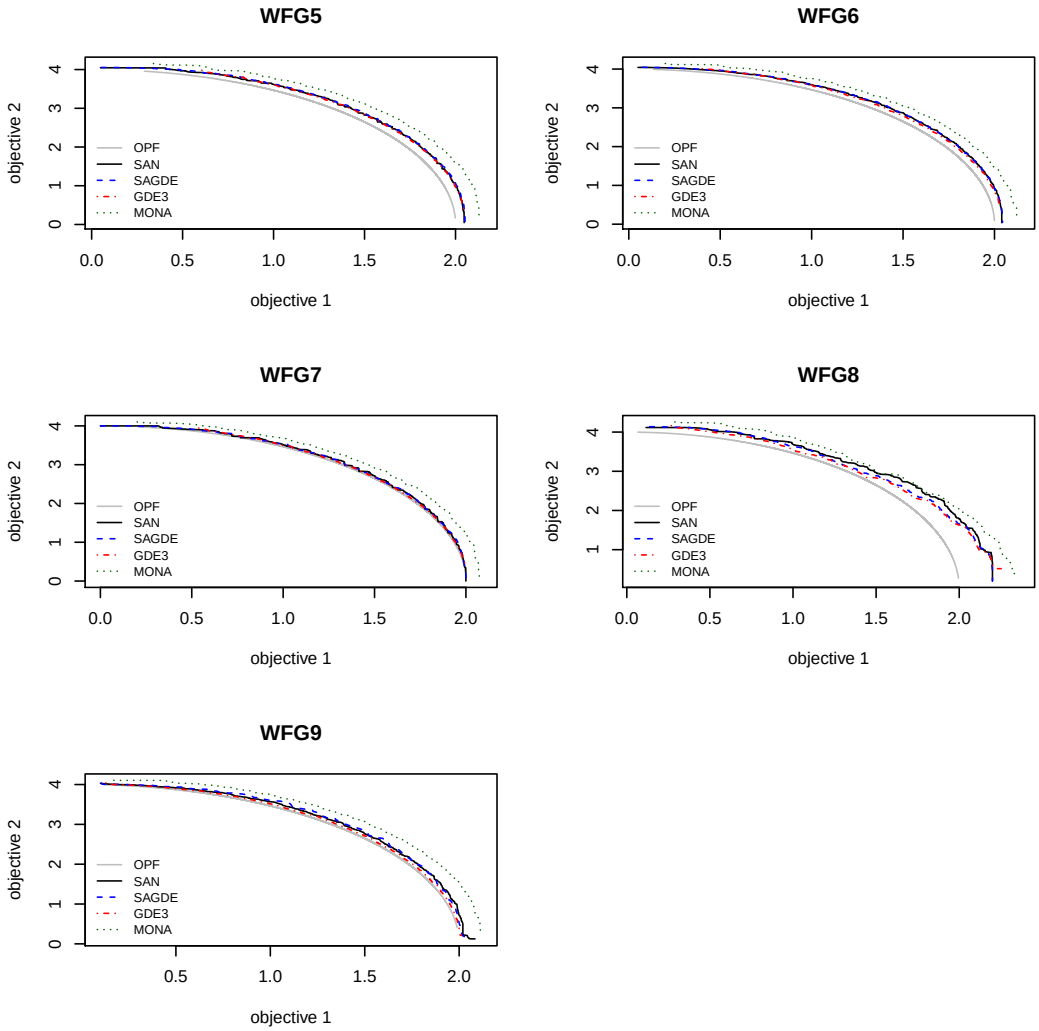

Figure 3: 50% attainment surfaces for the WFG Toolkit problems (minimization problems). Calculated for 30 independent runs.

Tables 7 and 8 but also present in the other tables and figures). However, the quality indicators do not agree in the remaining problems, which suggests that there is just a trade-off but not an explicit advantage in these problems. SAGDE tends to achieve a better coverage of the OPF, while GDE3 is closer to the OPF albeit having a slightly poorer coverage of the front. Consequently, depending on whether coverage or proximity to the front is more important, the algorithm designer may choose one or the other algorithm.

MONA achieved poor outcomes on all problems against all algorithms. Maybe the exceptions are the better coverage when compared against GDE3 in WFG3, WFG5, WFG6 and WFG7 problems (see Table 7 and Table 5). Even so, the combined subpopulations of MONA and the single-objective DEs in the SAN obtained state of art quality.

Notice also that inside the SAGDE and SAN there is respectivelly a GDE3 and a MONA subpopulation. The GDE3 subpopulation inside SAGDE is bigger than the MONA subpopulation inside SAN, however, GDE3 subpopulation still "mix" the solutions worse than MONA (resulting in poorer coverage). Demonstrating MONA's good ability of expanding and mixing results.

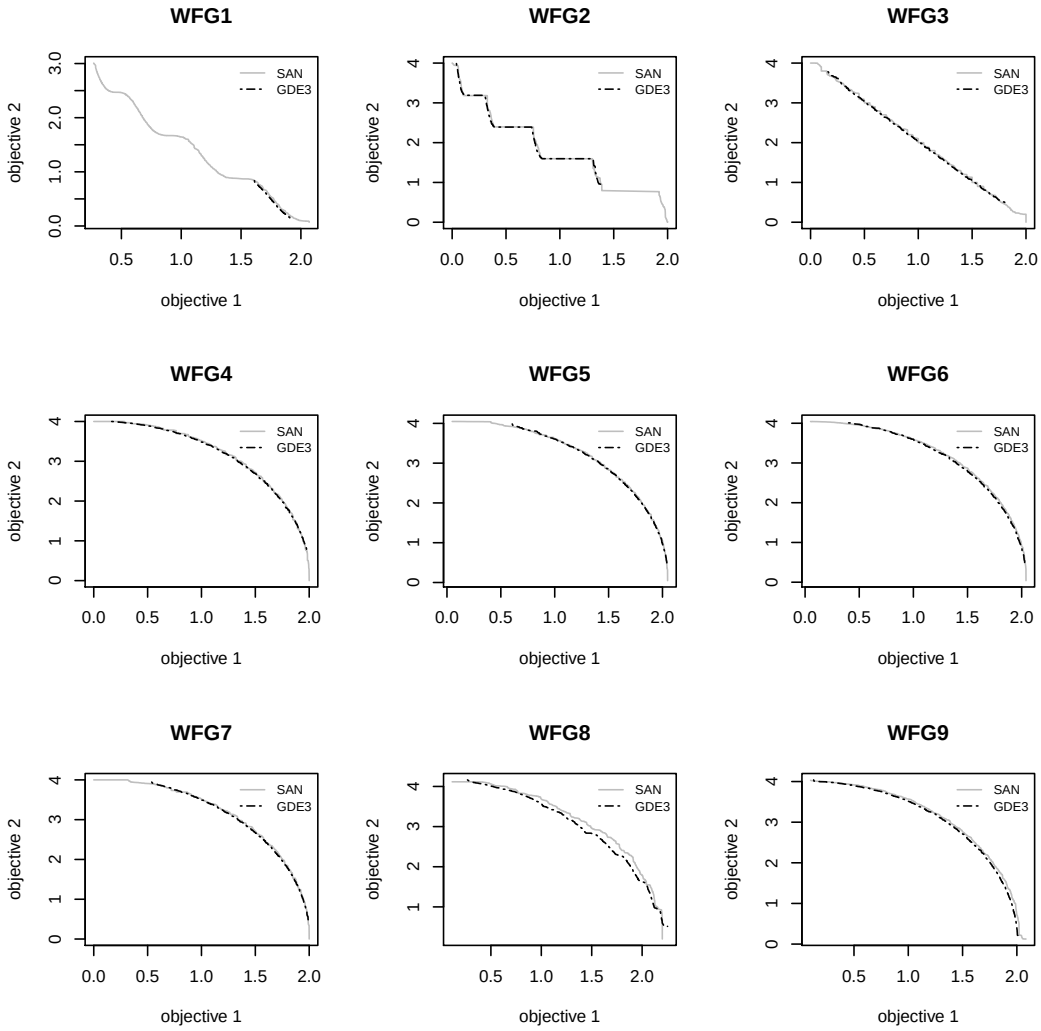

Figure 4: 50% attainment surfaces for the WFG Toolkit problems (minimization problems). Calculated for 30 independent runs.

Concerning the comparison of both GSAs, the experiments demonstrate a surprisingly better overall result of the SAN over the SAGDE, as the SAN is simpler and based on the MONA, an algorithm which achieved poor results on all tests. This fact might seem surprising at first glance, but looking from a different point of view, it is possible to understand those results if we take into account the GSF's structure. Recall that the more different two strategies are, the more the subpopulation's framework benefits

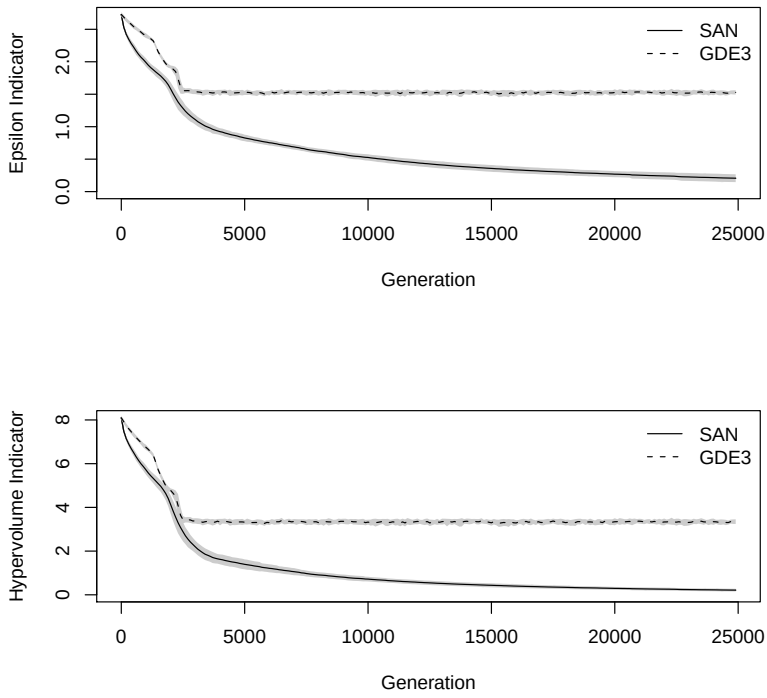

Figure 5: Hypervolume and ϵ indicators throughout the generations for both SAN and GDE3 algorithms in the WFG1 problem (the confidence interval of one standard deviation away from the mean is shown in grey). The curve was averaged over 30 independent runs.

from it. This happens because a similar strategy will also produce similar individuals in different subpopulations, using more resources for less exploration and diversity.

The comparison between SAN and GDE3 is a bit more complicated. First of all, Tables 7 and 8 show that SAN outperforms GDE3 according to both quality indicators in WFG1, WFG2 and WFG3 while the remaining problems have contrasting results of ϵ and hypervolume indicators. Consequently, GDE3 is comparable with SAN only in the concave problems, which have easier shapes of Pareto front. However, the statistical hypothesis testing does not tell us by how much is the difference (it only tells if it is bigger or not with some significance). Table 5 shows unsurprisingly that SAN outperforms GDE3 by a great difference in respect to the ϵ indicator. But according to Table 6, in all problems where GDE3 surpassed SAN statistically, GDE3 is shown to be close (inside GDE3's standard deviation) to the SAN in all problems but WFG8 (the reason why this happens is show on Section 9.6, where it is demonstrated that both algorithms have not converged yet in WFG8). In fact, to the knowledge of the authors, SAN achieved the best performance to date over all of the WFG's problems with two objectives. Additionally, for a clearer analysis, Figure 4 shows only SAN and GDE3 attainment surfaces and Figure 5 delineates the behavior of the quality indicators throughout the evolution

Table 9: Parameter's Table. The first five ratios of the $\mathcal{S}$ vector correspond to the subpopulations of the DEs used and the last ratio corresponds to the MONA.

	GDE3
CR	0.1
F	0.5

	SAN
CR	0.1
F	0.1
$\mathcal{IM}$	uniform
$\mathcal{S}$	$(0.1, 0.1, 0.1, 0.1, 0.1, 0.5)$
n_{inc}	1.1
n_{dec}	0.999
n_a	1
n_r	50000

of problem WFG1. It can be seen that both SAN and GDE3 start with similar quality (i.e., no initialization difference). However, SAN is always superior to GDE3 in quality soon after the starting point.

Both of the subpopulation algorithms presented here are strongly based on the division of the related panmictic algorithm's strategies into different subpopulations and with the results showing strong benefits of the subpopulation algorithms over the panmictic ones. This raises the question of weather the competition between different strategies inside one population can have deleterious consequences for an algorithm. In fact, SAN, which has entirely different subpopulations in terms of objectives (small or inexistent conflict inside the same subpopulation), was able to achieve the best results. Contrast this with the GDE3, which has three conflicting objectives inside its panmictic population (the two objectives of the problem and the diversity objective). Section 9.5 will touch this hypothesis more extensively and with an experimental test.

9.4 Study on Many-objective Optimization

In this study, we increased the number of objectives to five. Aside from that, the same WFG problems were used and all other problem's parameters were kept as before. Most of the algorithms' parameters remained the same as well with the only exception being vector $\mathcal{S}$, which depends on the number of objectives. The new set of parameters is shown in Table 9.

Tests with many-objective problems were realized using the SAN, the most prominent algorithm in the bi-objective study from Section 9.3, and a reference from the state of the art, GDE3.

Tables 10 and 11 display the mean and standard deviation values, while Table 12 shows the statistical results of the comparison. SAN is able to converge better in all problems according to both quality indicators except WFG8, where the quality indicators differed in the results (even so, WFG8's hypervolume indicator mean values of SAN and GDE3 are close to each other) Moreover, in all other problems SAN had very small p-values. The negative values of the hypervolume indicator means that the samples acquired from the Pareto optimum front dominate a hypervolume smaller than the SAN's dominated hypervolume. This result may be related to the number and distribution of samples in the OPF generated by the WFG toolkit. The same OPF's samples

Table 10: ϵ indicator's mean and standard deviation for SAN and GDE3. For each problem the best mean value as well as the other mean values inside the standard variation of the best mean value are marked in **bold**.

	SAN	GDE3
Problems	mean (sd)	mean (sd)
WFG1	$\mathbf{0.55(0.06)}$	$0.97(0.10)$
WFG2	$\mathbf{0.68(0.11)}$	$0.99(0.31)$
WFG3	$\mathbf{0.40(0.07)}$	$0.61(0.10)$
WFG4	$\mathbf{0.91(0.04)}$	$1.47(0.23)$
WFG5	$\mathbf{1.27(0.11)}$	$1.67(0.23)$
WFG6	$\mathbf{1.03(0.05)}$	$1.78(0.34)$
WFG7	$\mathbf{0.94(0.04)}$	$1.75(0.20)$
WFG8	$\mathbf{1.13(0.06)}$	$1.56(0.18)$
WFG9	$\mathbf{0.94(0.06)}$	$1.55(0.20)$

Table 11: Hypervolume indicator's mean and standard deviation for SAN and GDE3. For each problem the best mean value as well as the other mean values inside the standard variation of the best mean value are marked in **bold**.

	SAN	GDE3
Problems	mean (sd)	mean (sd)
WFG1	$\mathbf{62.89(13)}$	$997.9(98)$
WFG2	$\mathbf{30.78(8)}$	$95.5(40)$
WFG3	$\mathbf{-92.1(36)}$	$79.3(30)$
WFG4	$\mathbf{-40.7(48)}$	$378.1(67)$
WFG5	$\mathbf{897.8(173)}$	$1088(53)$
WFG6	$\mathbf{636.8(62)}$	$1050(66)$
WFG7	$\mathbf{448.2(62)}$	$1911(173)$
WFG8	$\mathbf{1399(91)}$	$\mathbf{1353(59)}$
WFG9	$\mathbf{-268.0(71)}$	$526.1(201)$

Table 12: P-values of comparison between SAN and GDE3 algorithms in many-objective problems with Mann-Whitney significance test using ϵ and hypervolume indicators. Results are marked in **bold** when the null hypothesis is rejected with a significance level of $\alpha = 0.05$. The alternative hypothesis is that the algorithm in the row is statistically better (smaller quality indicator) than the algorithm in the column.

		SAN		GDE3	
Algorithm	Problem	ϵ	hypervolume	ϵ	hypervolume
SAN	WFG1			$\mathbf{1.0e-16}$	$\mathbf{8.4e-18}$
	WFG2			$\mathbf{4.7e-6}$	$\mathbf{1.6e-15}$
	WFG3			$\mathbf{1.8e-12}$	$\mathbf{8.4e-18}$
	WFG4			$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG5			$\mathbf{2.9e-13}$	$\mathbf{3.0e-7}$
	WFG6			$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG7			$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
	WFG8			$\mathbf{3.3e-17}$	0.99
	WFG9			$\mathbf{8.4e-18}$	$\mathbf{8.4e-18}$
GDE3	WFG1	1	1		
	WFG2	0.99	0.99		
	WFG3	0.99	1		
	WFG4	1	1		
	WFG5	1	1		
	WFG6	1	1		
	WFG7	1	1		
	WFG8	1	$\mathbf{0.006}$		
	WFG9	1	1		

Table 13: Comparison of the SAN and GDE3 algorithms with Mann-Whitney significance test in many-objective problems. The respective meanings of $\Uparrow$, $\downarrow$ and $\approx$ is that SAN is statistically better, worse or equal to the GDE3.

	SAN vs GDE3 (many-objective)	
Problems	I_ϵ(p-value)	I_h(p-value)
WFG1	$\Uparrow (2.029e-16)$	$\Uparrow (1.691e-17)$
WFG2	$\Uparrow (9.415e-06)$	$\Uparrow (3.297e-15)$
WFG3	$\Uparrow (3.631e-12)$	$\Uparrow (1.691e-17)$
WFG4	$\Uparrow (1.691e-17)$	$\Uparrow (1.691e-17)$
WFG5	$\Uparrow (1.691e-17)$	$\Uparrow (1.691e-17)$
WFG6	$\Uparrow (1.691e-17)$	$\Uparrow (1.691e-17)$
WFG7	$\Uparrow (1.691e-17)$	$\Uparrow (1.691e-17)$
WFG8	$\Uparrow (6.764e-17)$	$\downarrow (0.013)$
WFG9	$\Uparrow (1.691e-17)$	$\Uparrow (1.691e-17)$

were used to compare both GDE3 and SAN and therefore there is not any bias in the comparison (i.e., GDE3 could have had negative hypervolume as well).

This suggests that SAN should achieve better results when problems increase in complexity. Recall that on bi-objective problems, GDE3 was shown to be comparable with SAN only when concave Pareto fronts were present. Naturally, with the increase in the number of functions to be optimized, the number of conflicting objectives inside panmictic algorithms is expected to increase as well. This explains the better overall solutions of SAN in all the many-objective problems with many different properties (see Table 3).

9.5 Explanation

It has been argued before that the algorithms based on the GSF achieve better results since they divide different strategies (algorithms) in distinct populations which avoid both the undesirable conflicts and the prevalence of one strategy over another. Here, we will present an detailed justification.

Consider a bi-objective optimization problem being solved with SAN and GDE3. For this problem, SAN may be divided into three strategies (i.e., $|A| = 3$): one single-objective DE for each of the two objectives and MONA. GDE3 has one strategy which is composed of two steps: first selecting individuals based on Pareto dominance (main strategy) and second pruning the population based on a diversity measure (secondary strategy).

If we see the strategies as a collection of forces capable of changing the positions of solutions, it is possible to draw the most salient force vectors produced by SAN and GDE3 (Figures 6 and 7). Therefore, for GDE3, the main force points directly to the Pareto front with secondary forces pointing sideways (caused by the pruning strategy). In SAN, the single-objective DE's subpopulations' forces point directly to their respective objective's coordinate while the MONA's subpopulation points away from the previous individuals which corresponds approximately to vectors pointing in all directions with the same strength.

This analysis reveals the main problem with GDE3: its forces responsible for spreading are relatively weak. The first consequence is, for example, when the problem has a disconnected geometry or bias, the solutions may spread only over a small subset of the optimal front (see problems WFG1, WFG2 of Figures 2 and 3 or Figure 4).

Another consequence is that the necessary forces for the solution of problems depends naturally on the problems themselves and if a given problem needs more spreading forces, GDE3 presents many difficulties to spread the solutions. For example, over all the WFG's datasets the GDE3 covered poorly the extremes of the Pareto front (see Figures 2 and 3 or Figure 4) and in the case of many-objective problems, where the Pareto front becomes wider as it expands along various dimensions, it achieved poor results in all tests for both quality indicators (see Table 13).

Notice that the vectors of the GDE3 are a consequence of its panmictic design which causes inevitably one force to be stronger or weaker relative to the others. That is, this analysis is inherently connected with the conflicting strategies of panmictic algorithms.

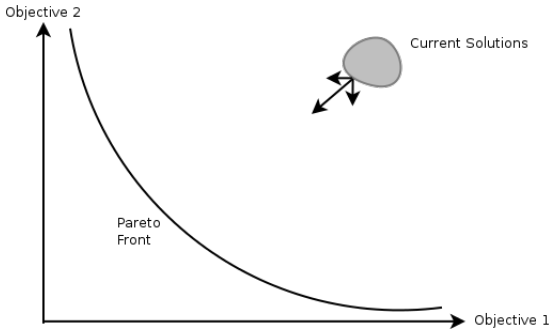

Figure 6: Diagram of the GDE3 with its strategy exposed explicitly as three components of a force. The length of the arrow is related with its intensity.

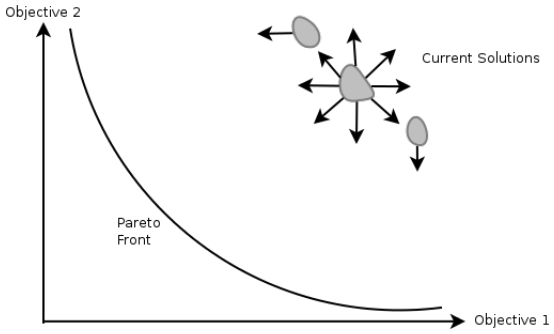

Figure 7: Diagram of the SAN with its strategies exposed explicitly as forces. The single-objective DE forces (dashed gray lines) are perpendicular to each other and the MONA force (circular dashed-point gray line) is a field-force which is stronger with the increase of the distance from the previous individuals.

9.6 Empirical Evaluation of Forces

Measuring the forces empirically can be done in various ways. If the average solution of each subpopulation in objective space is considered, it is possible to analyze the subpopulation forces throughout the evolution. However, the comparison with single population algorithms may be unfairly plotted with just one force (i.e., much of the

Table 14: Percentage of solutions which are either unfeasible or result in zero modulus forces. Both types of solutions are excluded from the calculation of forces and therefore not present in Figure 8.

Problem	Unfeasible Solutions		Zero Modulus Forces	
	SAN	GDE3	SAN	GDE3
WFG1	15.64%	9.76%	13.38%	50.45%
WFG4	15.78%	3.81%	11.27%	55.47%
WFG5	27.44%	13.85%	8.11%	44.54%
WFG8	24.42%	11.06%	0.06%	0.34%

behavior is lost with just one "mean subpopulation force"). Therefore, plotting the forces between parent and offspring in objective space seems like a better possibility, although some aspects of the global movement is lost.

Here, the forces are calculated by measuring the vector from the DE's operator main parent to its offspring in objective space (other genetic operators with no main parent might make necessary the computation of a set of forces for each individual with each force related to a parent). The experiment is composed of 2500000 evaluations samples throughout one run of the algorithm (multiple runs of the algorithm presented no significant difference from each other, as one would expect since the number of samples in one run are already representative). Figure 8 shows the accumulative angles of the forces for three problems with both GDE3 and SAN algorithms. The direction given by the 0° and 90° are respectivelly parallel to increasing x-axis and increasing y-axis (i.e., 180° is improving objective 1, 270° is improving objective 2). Regarding the measurement, it is done right before the selection phase of the differential evolution operator, otherwise the arc from 0° to 90° would be nonexistent. Naturally, a long bin means a higher number of solutions moving in that direction. Notice, however, that some histograms may have more individuals than others. This happens because two conditions cause some solutions or forces to be discarded:

- Unfeasible Solutions - They are excluded from the calculation, since unfeasible solutions can not be mapped to a point in objective space.
- Forces with Zero Modulus - In the case where the resulting child possess the same point in objective space as its main parent, the resulting force would have a zero modulus. In fact, this means that no force was applied at all and therefore it is reasonable to exclude it.

To give an idea of how many solutions were discarded and from which type (unfeasible solutions or solutions which result in a zero modulus force), Table 14 was constructed. Setting problem WFG8 aside, GDE3 has always a high number of solutions discarded (specially solutions which result in a zero modulus force). This happens because GDE3 converges prematuraly on these problems. In WFG8, however, the solutions which result in a zero modulus force are extremely small. This points to the fact that both algorithms have not yet converged in WFG8, explaining why GDE3 surpassed SAN in this problem.

Bare in mind that the forces seen are not just a "DNA" of the algorithm. They are affected intensively by the problem at hand. Therefore, the higher the bias of the problem is, the higher the influence of the problem in the measured forces becomes. The results on WFG1 and WFG8 shows exactly this interference of the problem which is

strongly biased (see Table 3 for the bias properties of all problems). Therefore, analysing the behavior on problems with less bias (such as problems WFG4 and WFG5) renders a less noisy perspective on the "DNA" of the algorithm. In fact, there are many similarities between the second and third rows of Figure 8 with Figures 6 and 7. For example, the spread of forces in all directions can be seen in SAN, i.e., every direction has a bin with noticeable longness, while GDE3 has bins on fewer directions. These results were predicted by our previous analysis.

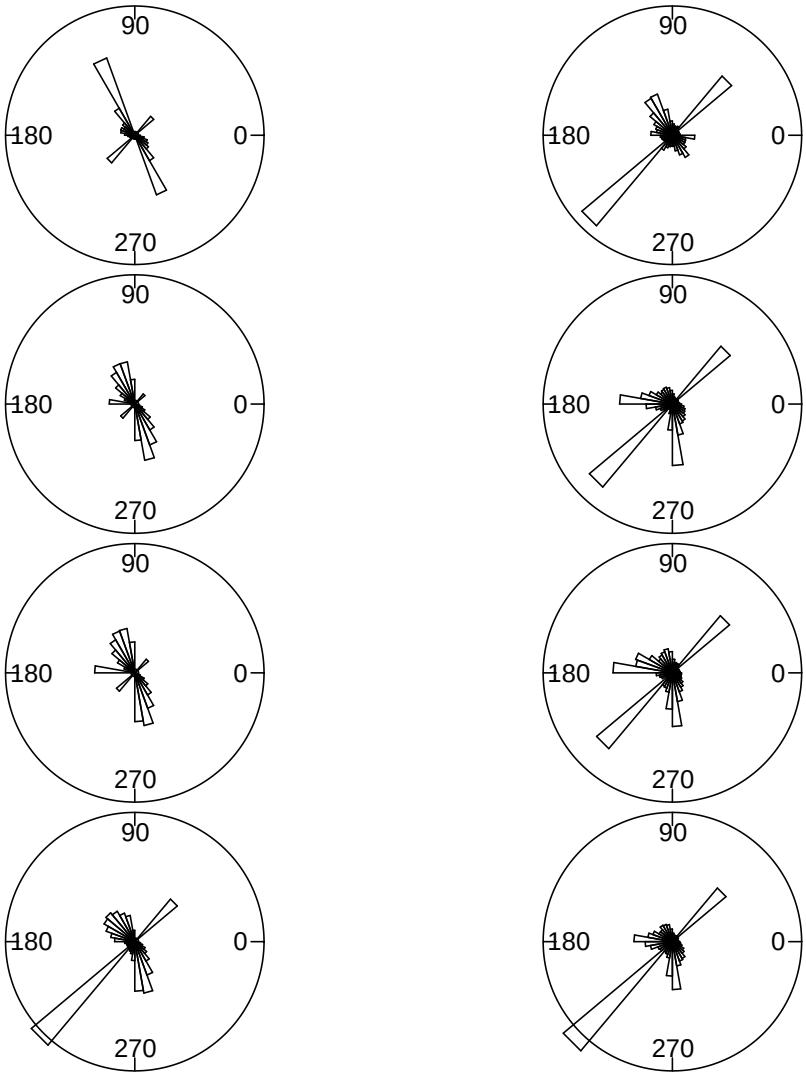

Figure 8: Accumulative angles of the forces measured for algorithms GDE3 (left column) and SAN (right column) on problems WFG1 (first row), WFG4 (second row), WFG5 (third row) and WFG8 (fourth row). The forces are measured by calculating the vector from the parent to the offspring in objective space. The scale is linear and the unfeasible solutions as well as forces with zero modulus were eliminated from the graph.

The essential idea behind all these explanations is that a panmictic population may be seen as a niche. Once no proper division is placed between strategies, no matter what strategies and procedures are involved, the population results in forces of different intensity being developed together, i.e., a conflict of forces appears inside the population. This internal population conflict is hardly solved without a division. That is, a division into subpopulations.

9.7 Further Investigations

The objective of this paper is to propose the framework together with some examples of algorithms based on it, demonstrating some of its aspects and strengths. This is however not an exhaustive exposition. There are still an extensive amount of topics to be covered. To cite some:

- Studies on the variations of $\mathcal{IM}$ and $\mathcal{S}$ as well as self-adaptive modifications;
- The effect of different and/or complex dynamics between subpopulations;
- Integration of different types of algorithms and comparison between them.

10 Conclusions

We have presented here a justification of why structured EAs, and in special the GSF, achieve better results in multi-objective optimization. This derives from the fact that *well-designed structured EAs separate better the conflicting strategies, avoiding the deleterious consequences of the competition between themselves.*

Additionally, this article presented a new framework called GSF which can aid the understanding and design of structured optimization algorithms. GSF can easily join any optimization algorithms, therefore any algorithm can be with little effort combined and tested together with others, yielding a very flexible framework.

Moreover, to the knowledge of the authors, SAN's results is the most or among the most robust algorithms of the state of the art, either surpassing GDE3 in the tests or achieving a comparable solution in terms of a trade-off between ϵ and hypervolume quality indicators. In fact, when the problems increased in the number of objectives (which also increased the number of conflicting strategies inside a panmictic algorithm) the advantage of SAN over GDE3 became more emphatic. In other words, the proposed subpopulation framework showed that with an integration of simple algorithms it was possible to achieve better solutions, surpassing or at least achieving similar performance in all tests realized with the original panmictic algorithms. Another interesting result is that a simple algorithm such as MONA, which had poor results on all tests, was shown to attain state of the art quality Pareto fronts when combined with two simple single-objective DEs in the subpopulation framework.

Thus, motivated by the population internal conflicts, structured optimization algorithms should find increasing attention of the optimization community. In this aspect, the proposed subpopulation framework will hopefully aid the development of new structured algorithms and open new possibilities for the algorithms to come. Consequently, further studies on multiple subpopulation dynamics as well as global interactions for the further understanding of the framework's frontiers is hereby encouraged.

References

[1] Alba, E. and Tomassini, M. (2002). Parallelism and evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 6(5):443–462.

[2] Beume, N., Fonseca, C., López-Ibáñez, M., Paquete, L., and Vahrenhold, J. (2009). On the complexity of computing the hypervolume indicator. *Evolutionary Computation, IEEE Transactions on*, 13(5):1075–1082.

[3] Brest, J., Greiner, S., Boskovic, B., Mernik, M., and Zumer, V. (2006). Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *Evolutionary Computation, IEEE Transactions on*, 10(6):646–657.

[4] Chakraborty, U. (2008). *Advances in differential evolution*. Springer Verlag.

[5] Coello, C. et al. (2006). Evolutionary multi-objective optimization: a historical view of the field. *Computational Intelligence Magazine, IEEE*, 1(1):28–36.

[6] De Toro, F., Ortega, J., Fernández, J., and Díaz, A. (2002). Psfga: a parallel genetic algorithm for multiobjective optimization. In *Parallel, Distributed and Network-based Processing, 2002. Proceedings. 10th Euromicro Workshop on*, pages 384–391. IEEE.

[7] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

[8] Delbem, A., de Carvalho, A., and Bretas, N. (2005). Main chain representation for evolutionary algorithms applied to distribution system reconfiguration. *Power Systems, IEEE Transactions on*, 20(1):425–436.

[9] Doncieux, S. and Mouret, J. (2010). Behavioral diversity measures for evolutionary robotics. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE.

[10] Doncieux, S., Mouret, J., and Bredeche, N. (2009). Exploring new horizons in evolutionary design of robots. In *Workshop on Exploring new horizons in Evolutionary Design of Robots at IROS*, volume 2009, pages 5–12.

[11] Durillo, J., Nebro, A., Coello, C., García-Nieto, J., Luna, F., and Alba, E. (2010). A study of multiobjective metaheuristics when solving parameter scalable problems. *Evolutionary Computation, IEEE Transactions on*, 14(4):618–635.

[12] Fonseca, C. and Fleming, P. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. *Parallel problem solving from nature-ppsn iv*, pages 584–593.

[13] Fonseca, C., Knowles, J., Thiele, L., and Zitzler, E. (2005). A tutorial on the performance assessment of stochastic multiobjective optimizers. In *Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, volume 216.

[14] García, S., Molina, D., Lozano, M., and Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. *Journal of Heuristics*, 15(6):617–644.

[15] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning.

[16] Gomes, C. and Selman, B. (1997). Algorithm portfolio design: Theory vs. practice. In *Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, pages 190–197. Morgan Kaufmann Publishers Inc.

[17] Gorges-Schleuter, M. (1991). Genetic algorithms and population structures. a massively parallel algorithm. Master's thesis.

[18] Grunert da Fonseca, V., Fonseca, C., and Hall, A. (2001). Inferential performance assessment of stochastic optimisers and the attainment function. In *Evolutionary Multi-Criterion Optimization*, pages 213–225. Springer.

[19] Guerri, A. and Milano, M. (2004). Learning techniques for automatic algorithm portfolio selection. In *ECAI*, volume 16, page 475.

[20] Hollander, M. and Wolfe, D. (1999). Nonparametric statistical methods.

[21] Huband, S., Hingston, P., Barone, L., and While, L. (2006). A review of multiobjective test problems and a scalable test problem toolkit. *Evolutionary Computation, IEEE Transactions on*, 10(5):477–506.

[22] Iorio, A. and Li, X. (2005). Solving rotated multi-objective optimization problems using differential evolution. *AI 2004: Advances in Artificial Intelligence*, pages 861–872.

[23] Iredi, S., Merkle, D., and Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms. In *Evolutionary Multi-Criterion Optimization*, pages 359–372. Springer.

[24] Kirkpatrick, S., Gelatt, C., and Vecchi, M. (1983). Optimization by simulated annealing. *science*, 220(4598):671.

[25] Kukkonen, S. and Lampinen, J. (2005). Gde3: The third evolution step of generalized differential evolution. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 443–450. IEEE.

[26] Kukkonen, S. and Lampinen, J. (2007). Performance assessment of generalized differential evolution 3 (gde3) with a given set of problems. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 3593–3600. IEEE.

[27] Larranaga, P. and Lozano, J. (2002). *Estimation of distribution algorithms: A new tool for evolutionary computation*, volume 2. Springer.

[28] Laumanns, M., Rudolph, G., and Schwefel, H. (1998). A spatial predator-prey approach to multi-objective optimization: A preliminary study. In *Parallel Problem Solving from Nature-PPSN V*, pages 241–249. Springer.

[29] Lehman, J. and Stanley, K. (2008). Exploiting open-endedness to solve problems through the search for novelty. *Artificial Life*, 11:329.

[30] Lehman, J. and Stanley, K. (2010a). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, pages 1–34.

[31] Lehman, J. and Stanley, K. (2010b). Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 837–844. ACM.

[32] Li, C. and Yang, S. (2008). An island based hybrid evolutionary algorithm for optimization. *Simulated Evolution and Learning*, pages 180–189.

[33] López-Ibánez, M., Paquete, L., and Stützle, T. (2010). Exploratory analysis of stochastic local search algorithms in biobjective optimization. *Experimental Methods for the Analysis of Optimization Algorithms*, pages 209–233.

[34] Maley, C. (1999). Four steps toward open-ended evolution. In *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference*. Citeseer.

[35] Manderick, B. and Spiessens, P. (1989). Fine-grained parallel genetic algorithms. In *ICGA'89*, pages 428–433.

[36] Menczer, F., Degeratu, M., and Street, W. (2000). Efficient and scalable pareto optimization by evolutionary local selection algorithms. *Evolutionary Computation*, 8(2):223–247.

[37] Mouret, J. and Doncieux, S. (2009). Using behavioral exploration objectives to solve deceptive problems in neuro-evolution. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 627–634. ACM.

[38] Nebro, A., Durillo, J., Luna, F., Dorronsoro, B., and Alba, E. (2006). A cellular genetic algorithm for multiobjective optimization. *NICSO 2006*, page 25.

[39] Okabe, T., Jin, Y., and Sendhoff, B. (2003). A critical survey of performance indices for multi-objective optimisation. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 2, pages 878–885. IEEE.

[40] Pelikan, M. (2005). Bayesian optimization algorithm. *Hierarchical Bayesian Optimization Algorithm*, pages 31–48.

[41] Robič, T. and Filipič, B. (2005). Demo: Differential evolution for multiobjective optimization. In *Evolutionary Multi-Criterion Optimization*, pages 520–533. Springer.

[42] Santos, A., Delbem, A., London, J., and Bretas, N. (2010). Node-depth encoding and multiobjective evolutionary algorithm applied to large-scale distribution system reconfiguration. *Power Systems, IEEE Transactions on*, 25(3):1254–1265.

[43] Sprave, J. (1999). A unified model of non-panmictic population structures in evolutionary algorithms. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 2. IEEE.

[44] Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.

[45] Storn, R. and Price, K. (2002). Minimizing the real functions of the ICEC'96 contest by differential evolution. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 842–844. IEEE.

[46] Talbi, E., Mostaghim, S., Okabe, T., Ishibuchi, H., Rudolph, G., and Coello Coello, C. (2008). Parallel approaches for multiobjective optimization. *Multiobjective Optimization*, pages 349–372.

[47] Tomassini, M. (2005). *Spatially structured evolutionary algorithms*. Springer.

[48] Tušar, T. and Filipič, B. (2007). Differential evolution versus genetic algorithms in multiobjective optimization. In *Evolutionary Multi-Criterion Optimization*, pages 257–271. Springer.

[49] Vesterstrom, J. and Thomsen, R. (2004). A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 2, pages 1980–1987. IEEE.

[50] Vrugt, J. and Robinson, B. (2007). Improved evolutionary optimization from genetically adaptive multimethod search. *Proceedings of the National Academy of Sciences*, 104(3):708–711.

[51] Woolley, B. and Stanley, K. (2011). On the deleterious effects of a priori objectives on evolution and representation. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 957–964. ACM.

[52] Zitzler, E. and Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms-a comparative case study. In *Parallel Problem Solving from Nature-PPSN V*, pages 292–301. Springer.

[53] Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *Evolutionary Computation, IEEE Transactions on*, 3(4):257–271.

[54] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., and da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *Evolutionary Computation, IEEE Transactions on*, 7(2):117–132.