Logic Journal of the IGPL, 2021, pp. 1–12

doi: DOI HERE
Advance Access Publication Date: Day Month Year
Paper

PAPER

Design and implementation of Parallel self-adaptive Differential Evolution for global optimization

Iztok Fister,[1] Andres Iglesias,[2] Akemi Galvez,[2] Dušan Fister[3] and Iztok Fister Jr.[1,*]

[1]Faculty of Electrical Engineering and Computer Science, University of Maribor, Koroška cesta 46, 2000 Maribor, Slovenia, [2]Dpt. of Applied Mathematics and Computational Sciences, University of Cantabria, Avenida de los Castros, s/n, 39005 Santander, Spain and [3]Faculty of Economics and Business, University of Maribor, Razlagova 14, 2000 Maribor, Slovenia

[*]Iztok Jr. Fister. iztok.fister1@um.si

FOR PUBLISHER ONLY Received on Date Month Year; revised on Date Month Year; accepted on Date Month Year

Abstract

The results of evolutionary algorithms depend on population diversity that normally decreases by increasing the selection pressure from generation to generation. Usually, this can lead the evolution process to get stuck in local optima. This study is focused on mechanisms to avoid this undesired phenomenon by introducing parallel self-adapted differential evolution that decomposes a monolithic population into more variable-sized sub-populations, and combining this with the characteristics of evolutionary multi-agent systems into a hybrid algorithm. The proposed hybrid algorithm operates with individuals having some characteristics of agents, e.g., they act autonomously by selecting actions, with which they affect the state of the environment. Additionally, this algorithm incorporates two additional mechanisms: aging, and adaptive population growth, which help the individuals by decision-making. The proposed parallel differential evolution was applied to the CEC'18 benchmark function suite, while the produced results were compared with some traditional stochastic nature-inspired population-based and state-of-the-art algorithms.

Key words: Differential Evolution, Variable population size, Aging mechanism, Autonomous agent

Introduction

Nowadays, nature-inspired population-based algorithms have achieved an important role in solving real-world problems. These algorithms typically take the inspiration for their work either in the evolution theory of Darwin [7] or the intelligent behavior of social living animals or insects [3]. The former inspiration has led to emergence of Evolutionary Algorithms (EA), while the latter to the advent of Swarm Intelligence (SI) based algorithms. Indeed, the nature-inspired algorithms are also applicable to domains, where no domain-specific knowledge has been discovered. Obviously, the quality of solutions are mainly dependent on the operations of the two components of the evolutionary search process [9]: exploration and exploitation. Exploration refers to searching in undiscovered regions of the search space, while exploitation to discovering the search in the vicinity of the known good solutions. Intensification of the exploration component typically causes losing the diversity of the population, and directs the search into premature convergence.

This phenomenon is especially undesirable in conditions of open-ended evolution [13] and artificial life [2], where an agent must operate continuously without any breaks. Therefore, a lot of approaches have been proposed for avoiding this phenomenon, such as, for instance by [19], and [10]. The further step in mastering the problem in open-ended evolution and artificial life, obviously, is the development of Parallel Evolutionary Algorithms (PEA) [12] and of Evolutionary MultiAgent Systems (EMAS) by [6]. The PEAs decompose the monolithic population in EAs into more sub-populations, where selection and reproduction are limited to individuals inhabiting one region, and a migration operator is used to move selected individuals from one region to another. The EMAS maintain a population of agents during an evolutionary cycle. Each agent represents a solution of the problem to be solved. During its life-time, starting with birth and ending with death, the agent is able to reproduce or clone. The reproduction is similar to crossover while cloning to mutation in classical EAs. The selection is responsible for preserving the best solution on the one hand, and to eliminating the less fit solutions via the action death from the EMAS on the other.

The purpose of this study is to combine the characteristics of the PEAs with those of EMAS into a hybrid algorithm that could confront the problem of premature convergence effectively on the one hand, and keep the quality of results obtained by solving the global optimization problems at a sufficiently high level on the other. Actually, we started with the self-adaptive Differential Evolution (jDE) [5] using a monolithic population. In the next phase, the parallel jDE (i.e., PjDE) was developed, where this population is decomposed into more sub-populations evolving in parallel. The advantage of using the jDE in place of the original DE [17] is the fact that the control of the algorithm's parameters is moved in jDE from the algorithm's level to the individual level. Finally, the concept of agents is borrowed from EMAS and applied to individuals in the population of solutions. This means that individuals in the PjDE have some characteristics of agents, because they act autonomously in deciding which action to select in order to affect the current problem state. Consequently, some components of the classical jDE algorithm are either eliminated (e.g., survivor selection), or redefined in new way (e.g., variation operators).

On an individual's level, the following actions can be executed: replication, clone, migrate, death, and rebirth. The actions affect the state of the individual, which is reflected in its fitness function. In our study, individuals can interact with all the other individuals within the island, and this topology determines the neighborhood of each individual. Furthermore, they are also able to change their relative positions within the particular island. Using the topology, a parent selection can be performed using local information only. The parent selection is necessary for implementation of replication and clone, based on DE mutation strategies (i.e., variation operators). The migrate action enables individuals to move from one island to another according to migration probability.

Indeed, the PjDE incorporates two new mechanisms: aging, and adaptive population growth. The mechanisms affect the population model. The former influences the size of island by action death, and controls decreasing the number of individuals with regard to the feedback obtained from the population in the last generation, while the latter helps individuals by deciding whether to apply a reproduction or a clone action. However, the reproduction causes growing the population size, while clone keeps the population size intact.

As a result, the Parallel self-adaptive Differential Evolution with Variable population size for global optimization (gPVajDE) was proposed, and applied to the CEC-18 benchmark function suite. The results of comparison with the classical EAs, like DE, and its self-adaptive variants jDE and SaDE [16], and the state-of-the-art algorithms, like LShade [18] and jSO [4], showed the great potential of the proposed system, especially when comparing them with the traditional ones, and encouraged us to continue with the research in the direction of improving them, also in the future. Let us mention that the purpose of the study was not to improve the results of the state-of-the-art algorithms at any cost, but to show that the proposed concept could also be used for solving the hardest optimization problems.

The structure of the remainder of the paper is as follows: Background information is reviewed in Sect. 2. In line with this, the basics of Differential Evolution and its self-adaptive variant jDE are highlighted. The design and implementation of EMAS for global optimization is the subject of Sect. 3. Sect. 4 illustrates experiments and the obtained results. The paper concludes with Sect. 5, which summarizes the performed work, and outlines potential directions for the further research.

Basic information

Information needed for understanding the subject that follows is reviewed in this section. At first, the basics of DE are discussed, because this is a well-known algorithm for global optimization, and serves in this study as an inspiration for implementing the actions of reproduction and cloning. The section concludes with a description of its self-adaptive variant jDE, that is capable of storing local copies of control parameters into representations of individuals.

Differential Evolution

Variation operators in DE base on difference of vectors maintained in the population of individuals, where each vector represents a solution of the problem of interest. Despite its strong mathematical definition, the algorithm belongs to a class of stochastic nature-inspired population-based algorithms, because the DE variation operators reflect similar characteristics as found in crossover and mutation in Darwinian evolution [7]. It has developed by Storn and Price in 1995 [17] and was applied quickly for solving many continuous, as well as discrete problems. Due to its good results, many variants of this algorithm have arisen since then.

The population in the DE algorithm consists of Np real valued vectors that undergo operations of variation operators, like mutation, crossover, and selection. In the basic mutation, two solutions are selected randomly, and their scaled difference is added to the third solution, as follows:

$$\mathbf{u}_i^{(t)} = \mathbf{x}_{r0}^{(t)} + F \cdot (\mathbf{x}_{r1}^{(t)} - \mathbf{x}_{r2}^{(t)}), \quad \text{for } i = 1, \ldots, Np, \tag{1}$$

where $F \in [0.1, 1.0]$ denotes the scaling factor that scales the rate of modification, while Np represents the population size and $r0$, $r1$, $r2$ are randomly selected values in the interval $1, \ldots, Np$. Note that the proposed interval of values for parameter F was enforced in the DE community.

The mentioned mutation strategy is dedicated primarily for exploring a search space. However, the exploitation component of the search space is also needed for efficient searching of the solution space. In line with this, the following mutation strategy has been developed:

$$\mathbf{u}_i^{(t)} = \mathbf{x}_{best}^{(t)} + F \cdot (\mathbf{x}_{r1}^{(t)} - \mathbf{x}_{r2}^{(t)}), \quad \text{for } i = 1, \ldots, Np, \tag{2}$$

where $\mathbf{x}_{best}^{(t)}$ is the current best individual, and $r1$, $r2$ are randomly selected values in the interval $1, \ldots, Np$. Let us emphasize that a balance between exploration and exploitation can be achieved by mixed use of both strategies [19].

DE employs a binomial (denoted as 'bin') or exponential (denoted as 'exp') crossover. The trial vector is built from parameter values copied from either the mutant vector generated by Eq. (1) or parent at the same index position laid i-th vector. Mathematically, the binomial crossover can be expressed as follows:

$$w_{i,j}^{(t)} = \begin{cases} u_{i,j}^{(t)} & \text{rand}_j(0,1) \leq CR \vee j = j_{rand}, \\ x_{i,j}^{(t)} & \text{otherwise}, \end{cases} \tag{3}$$

where $CR \in [0.0, 1.0]$ controls the fraction of parameters that are copied to the trial solution. The condition $j = j_{rand}$ ensures that the trial vector differs from the original solution $\mathbf{x}_i^{(t)}$ in at least one element. Mathematically, the selection can be expressed as follows:

$$\mathbf{x}_i^{(t+1)} = \begin{cases} \mathbf{w}_i^{(t)} & \text{if } f(\mathbf{w}_i^{(t)}) \leq f(\mathbf{x}_i^{(t)}), \\ \mathbf{x}_i^{(t)} & \text{otherwise}. \end{cases} \tag{4}$$

The selection is usually called 'one-to-one', because both trial and corresponding vector laid on the i-th position in the population compete for surviving into the next generation, where the better according to the fitness function takes wins.

Several ways exist in which operation of variation operators are applied to the population of solutions. Therefore, a specific notation was introduced to describe the varieties of these methods (also mutation strategies), in general. For example, 'rand/1/bin', defined by Eq. (1), denotes that the base vector is selected randomly, and one vector difference is added to it, while 'best/1/bin', defined by Eq. (2) designates that one vector difference is added to the current best vector. In both cases, the number of modified parameters in the trial/offspring vector follows a binomial distribution.

jDE algorithm

The original DE works with fixed setting of control parameters. As a matter of fact, the results of optimization can be improved significantly (especially in dynamic and uncertain environments), when the parameters are adapted or self-adapted during the evolutionary run. One of the first self-adaptive DE variants (jDE) was proposed by Brest et al. [5] that was applied effectively, especially, on continuous optimization problems. In this algorithm, two control parameters, i.e., scale factor F and crossover rate CR, are added to the representation of an individual, and undergo operation of the variation operators together with the problem variables.

As a result, the individual in jDE is represented as follows:

$$\mathbf{x}_i^{(t)} = (x_{i,1}^{(t)}, x_{i,2}^{(t)}, ..., x_{i,D}^{(t)}, F_i^{(t)}, CR_i^{(t)}).$$

Let us mention that the variation operators are not controlled using the parameters F and CR on the algorithm's level as in the original DE, but on an individual level. This means that the jDE maintains parameters F_i and CR_i for each i-th individual separately, and modifies them according to the following equations:

$$F_i^{(t+1)} = \begin{cases} F_l + \text{rand}_1 * (F_u - F_l), & \text{if } \text{rand}_2 < \tau_1, \\ F_i^{(t)}, & \text{otherwise}, \end{cases} \tag{5}$$

and

$$CR_i^{(t+1)} = \begin{cases} \text{rand}_3, & \text{if } \text{rand}_4 < \tau_2, \\ CR_i^{(t)}, & \text{otherwise}, \end{cases} \tag{6}$$

where $\text{rand}_{i=1...4} \in [0,1]$ are randomly generated values drawn from uniform distribution in the interval $[0,1]$, τ_1 and τ_2 are learning steps, and F_l and F_u lower and upper bounds for parameter F, respectively.

Parallel self-adaptive Differential Evolution

The aim of developing the gPVajDE is to combine the best characteristics of the PEA with those of the EMAS into one hybrid algorithm. Thus, the authors wish to keep the performance of this algorithm comparable with the other nature-inspired algorithms. The development of it can be divided into two parts: (1) Dividing the monolithic population of size Np into n interrelated sub-populations such that the sum of their population sizes is equal to $\sum_{i=1}^{n} Np_i = Np$, and (2) Providing the individuals within the population independently in decision-making, which action should be the most adequate answer to the current problem state.

Additionally, two mechanisms are incorporated within the algorithm, i.e., aging and adaptive population growth. The former introduces the concept of an individual's age, replacing the original 'one-to-one' selection and changes this with a more natural paradigm, asserting that when people are old enough they must die. The latter directs individuals by making the decision whether replacement or clone should be performed in a particular situation. In summary, the aging mechanism takes care about reducing the island size, while the adaptive population growth enriches the island with the new individuals.

Aging mechanism

The aging mechanism presents one of the more popular concepts of adapting the population size during the evolutionary cycle in the EA community, and was used in the Genetic Algorithm with Varying Population Size (GAVaPS) [14]. This mechanism introduced the concept of an individual's "age", which counts the number of generations the individual stays "alive".

The aging mechanism operates as follows: Each individual in a population lives the number of generations (ages) determined by its lifetime. This parameter depends on the fitness of the corresponding individual, i.e., the higher the fitness of an individual, the higher the lifetime granted to it. Mathematically, the lifetime LT is defined as:

$$LT = \begin{cases} MinLT + K \cdot \frac{f_i - MinFit}{AvgFit - MinFit}, & \text{if } AvgFit \geq f_i, \\ \frac{1}{2}(MinLT + MaxLT) + K \cdot \frac{f_i - AvgFit}{MaxFit - AvgFit}, & \text{if } AvgFit < f_i, \end{cases} \tag{7}$$

where $MinLT$ and $MaxLT$ denotes the minimum and maximum available lifetime values, respectively, $AvgFit$, $MinFit$, and $MaxFit$ are the average, minimum, and maximum values of fitness in the current population, while the coefficient is expressed as $K = \frac{1}{2}(MaxLT - MinLT)$.

Adaptive population growth

The adaptive population growth implements the so-called Non-Linear population Size Reduction (NLSR) mechanism, where the population size is adapted following the population dynamics. In population dynamics, the measure of the uncertainty in the population size is expressed as:

$$\Delta H^{(t+1)} = \log \frac{2 \cdot S^{(t)}}{Np^{(t+1)}}, \tag{8}$$

where ΔH denotes a change in the evolutionary entropy [8], $S^{(t)}$ is the number of positive variations in the last population, and $Np^{(t+1)}$ is the effective population size in the next evolutionary cycle. Entropy influences increasing/decreasing the current population size regarding the following relations:

$$\Delta_{\max} = \begin{cases} -\text{rand}(1, 2 \cdot (R^{(t+1)} - N^{(t+1)})), & \text{if } \Delta H^{(t+1)} > 0, \\ +\text{rand}(1, 2 \cdot (R^{(t+1)} - N^{(t+1)})), & \text{if } \Delta H^{(t+1)} < 0, \\ 0, & \text{if } \Delta H^{(t+1)} = 0, \end{cases} \tag{9}$$

where $\Delta_{\max}$ denotes a modification in the population size, and $R^{(t+1)}$ is a decreased, linear, reference function that reduce the

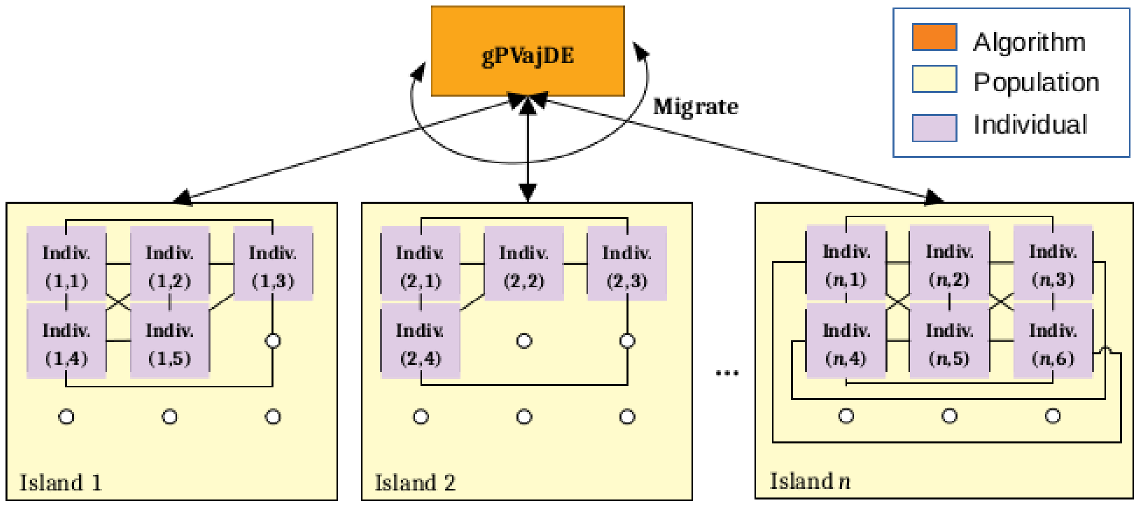


Fig. 1. The architecture of the proposed gPVajDE.

population size according to:

$$R^{(t+1)} = \left(1 - \frac{t+1}{t_{\max}}\right) \cdot (MaxNp - MinNp) + (t+1), \qquad (10)$$

where t is the generation number, $t_{\max}$ the maximum number of generations, $MaxNp$ and $MinNp$ are the maximum and minimum population sizes, respectively.

Design of gPVajDE

The architecture of the proposed gPVajDE is illustrated in Fig. 1, from which it can be seen that the algorithm consists of more islands connected in ring topology that evolve in parallel. The ring topology enables migration of selected individuals from a target island to either its predecessor's or successor's island in the ring. Each island hosts its own set of individuals. These are positioned on random positions on a predefined topology.

Obviously, there are several topologies available, like ring, grid, or complete graph, that enable the potential connections between individuals (also nodes in topology). Indeed, the topology defines the neighborhood of each node. This consists of all individuals, which can be achieved by connection locally. In line with this, the size of the neighborhood depends strongly on the topology used. For instance, the neighborhood size of individuals illustrated in the figure is four, due to the grid topology, where each node is connected with the four neighbors. However, when complete graph topology is employed, no restrictions are imposed in interactions among individuals. Obviously, the positions of individuals are not fixed, because they are able to change their positions in each generation randomly.

On the other hand, the individuals can perform autonomous actions, with which they are able to affect the current problem state. In summary, the gPVajDE operates on three levels: algorithm, population, and individual. These levels are labeled in the legend on Fig. 1.

Agent actions

Individuals use actions to react to events from the current problem state (also environment) according to the results of the making-decision process. The proposed gPVajDE supports the following set of actions:

$$Act = \{\text{REPR,CLONE,MIGR,REBIRTH,DEATH}\}, \qquad (11)$$

where

REPR	–	sexual reproduction (crossover),
CLONE	–	asequal reproduction (mutation),
MIGR	–	migration of agents,
REBIRTH	–	regeneration of the agent,
DEATH	–	killing of the agent.

Action REPR generates the trial solution using the exploration DE-mutation strategy (Eq. (2)), and keeps it within the island beside its parent to evolve into the next generation. With new individuals, the reproduction contributes to the island's growth.

Action CLONE, that generates the trial solution using the exploitation DE-mutation strategy (Eq. (1)), selects the better between the parent and trial solutions for the next generation, and, therefore, keeps the island size unchanged.

Interleaving the actions REPR and CLONE, the proposed hybrid algorithm tries to find a balance between exploration and exploitation. Interestingly, parent selection in the gPVajDE bases on the topology in place of implementation of parent selection in the original DE/jDE. This means that the selection operator selects the candidate solutions for entering in the mutation strategies randomly from neighbors in the neighborhood of the target solution.

Action MIGR enables some individuals in one island to move to the other island according to the migration probability p_m. This action depends strongly on the topology, in which islands are arranged in the communication network, and takes place on the algorithm level. If, for instance, the ring topology is used, where each node is connected by exactly two nodes, the particular individual in the target island can be migrated either to its predecessor or successor island in the ring.

Action REBIRTH is launched when the state of the environment falls into the condition "No population diversity". The condition arises when the diversity of the island sub-population is lost. In general, the population diversity is calculated according to equation [15]:

$$I_{\mathbf{c}}^{(t)} = \sum_{i=1}^{Np} \sum_{j=1}^{D} (x_{i.j} - c_j)^2, \tag{12}$$

where $I_{\mathbf{c}}^{(t)}$ represents an inertia moment, Np the population size, D the dimension of the problem, and vector $\mathbf{c} = \{c_j\}$ is the mass center expressed as:

$$c_j^{(t)} = \frac{1}{Np} \sum_{i=1}^{Np} x_{i,j}, \quad \text{for } j = 1, \ldots, D. \tag{13}$$

Actually, this condition arises when the diversity of the k-th island is lost, in other words, $I_{\mathbf{c},k}^{(t)} = 0$ for $k = 1, \ldots, n$. That is a typical case for small sized islands, where the search process achieved its matured phase. Consequently, the action causes the creation of a new individual that replaces the worst one.

Action DEATH eliminates all outdated individuals from the island's sub-population, and is, actually, triggered as the result of the aging mechanism.

Implementation of gPVajDE

Due to the autonomy of an individual within the gPVajDE, its representation is richer than those used in the original DE/jDE. An individual in gPVajDE has its own times of birth and death, and lives as long as permitted by its quality. Additionally, it is positioned in a random location within the predefined network topology, and is capable of performing actions. Therefore, the individual is represented as a tuple:

$$X_i = \langle \mathbf{x}_i, f_i, M_i, age_i, LT_i, act_i \rangle, \quad \text{for } i = 1, \ldots, Np, \tag{14}$$

where

$\mathbf{x}_i$	–	solution with elements $\{x_{i,j}\}$ for $i = 1, \ldots, D$,
f_i	–	a fitness function of the problem in question,
M_i	–	location of the individual within the island,
age_i	–	current age of an agent,
LT_i	–	calculated maximum lifetime,
act_i	–	action to be performed by the individual.

Three main algorithms need to be implemented for covering the proposed three level program architecture of the gPVajDE. The algorithm CONTROL, covering the algorithm level, takes care about: creation, termination, and parallel evolving and synchronization of islands. The population level algorithm EVOLVE provides global functions for individuals, like: positioning on random locations within the network topology, aging management, and adaptive population growth. The algorithm MAKEDECISION working individual level addresses tasks with which an individual is confronted, such as: decision-making, and performing actions.

The pseudo-code of the CONTROL algorithm is presented in Algorithm 1, from which it can be seen that its first task is creating the island topology (function CREATEISLANDS). In line with this, the number of islands n and the corresponding sub-population size Np pass as parameters to the procedure. The evolutionary cycle that follows is divided into two parts: (1) parallel evolving of the islands (function EVOLVE) and (2) performing migration actions, based on emigrants collected after finishing the first part (lines 9-14). The evolutionary cycle is terminated, when the termination condition is satisfied.

The EVOLVE algorithm illustrated in Algorithm 2 addresses demands of the algorithm's level. At first, it implements

Algorithm 1 The CONTROL algorithm.

```
Require: $n$ - island number, $Np$ - island size
Ensure: $alg$ - evolved set of islands
 1: procedure CONTROL($n$, $Np$)
 2:     $alg$ = CREATEISLANDS($n$, $Np$);
 3:     $t$ = 0;
 4:     while termination_condition_not_found do        ▷ main loop
 5:         $emg$ = $\emptyset$;                          ▷ set of emigrants
 6:         parfor all $island \in alg$ do    ▷ parallel island execute
 7:             $emg \cup =$ EVOLVE($island$);              ▷ evolve island
 8:         end parfor all
 9:         while $emg \neq \emptyset$ do    ▷ performing migration actions
10:             $emg\_item$ = HEAD($emg$);          ▷ $\langle island, item \rangle$
11:             $emg$ = TAIL($emg$);
12:             $target$ = RAND(N($emg\_item.island$));
13:             DOMIGRATE($emg\_item$, $target$);
14:         end while
15:         $t$ = $t + 1$;
16:     end while
17:     return alg;
18: end procedure
```

Algorithm 2 The EVOLVE algorithm.

```
Require: $island$ - sub-population of individuals
Ensure: $emg$ - set of emigrants
 1: procedure EVOLVE($island$)
 2:     $emg = \emptyset$;
 3:     CALCLIFETIMES(island);                    ▷ update aging data
 4:     $\Delta_{\max}$ = EVOLUTIONENTROPY($island$);    ▷ pop. dynamics
 5:     $\mathbf{M}$ = GENERATETOPOLOGY($island$);    ▷ set new topology
 6:     $\langle rebirth, avg \rangle$ = POPDIVERSITY($island$);
 7:     parfor all $individual \in island$ do
 8:         $\langle \Delta_{\max}, rebirth \rangle$ =MAKEDECISION($individual$, $\Delta_{\max}$,
 9:                                                   $rebirth$, $avg$);
10:         $emg \cup =$ DOACTION;                    ▷ executing actions
11:     end parfor all
12:     return emg;
13: end procedure
```

updating the aging data (function CALCLIFETIMES), calculating the population dynamics (function EVOLUTIONENTROPY), generating random positions of the individuals within the network topology (function GENERATETOPOLOGY), and evaluating the population diversity (function POPDIVERSITY), which returns either the worst individual in the case of losing diversity or **NULL** otherwise, and the average fitness value avg in the island.

After initialization, the parallel loop is launched, in which each individual decides (function MAKEDECISION), which action is the proper answer to the current problem state. Then, performing the selected action follows. The algorithm returns a set of emigrants that are collected throughout the cycle.

Finally, Algorithm 3 depicts the procedure for making decisions by a particular individual. This algorithm is controlled by four parameters. Indeed, it consists of a set of conditions affecting the selection of the proper action. For instance, the MIGR action is selected in accordance with the probability of migration p_m on individuals with a fitness function value more, or equal to, the average fitness value. The REBIRTH action is performed, if the losing of population diversity condition is detected. The DEATH action is selected,

Algorithm 3 The MakeDecision algorithm.

```
Require: $individual$, $\Delta_{\max}$ - entropy, $rebirth$ - pop.div., $avg$
 1: procedure MakeDecision($individual$, $\Delta_{\max}$, $rebirth$, $avg$)
 2:     if Rand(0, 1)< $p_m$ and $individual.f \geq avg$ then
 3:         SetBehavior($individual$, MIGR);          ▷ migration act.
 4:     else if $rebirth == individual$ then
 5:         SetBehavior($individual$, REBIRTH);       ▷ rebirth
 6:         $rebirth$ = NULL;
 7:     else if $individual.age \leq individual.LT$ then
 8:         SetBehavior($individual$, DEATH);         ▷ death
 9:     else if $\Delta_{\max} < 0$ then
10:         SetBehavior($individual$, REPR);          ▷ sexual reprod.
11:         $\Delta_{\max} = \Delta_{\max} - 1$;
12:     else
13:         SetBehavior($individual$, CLONE);         ▷ asexual repr.
14:     end if
15:     return $\langle \Delta_{\max}, rebirth \rangle$;
16: end procedure
```

Table 1. Parameter setup of the proposed gPVajDE.

L	Parameter	Abbreviation	Value/Interval
A	Dim. of the problem	D	$[10, 30, 50]$
	Number of fit. eval.	$nFEs$	$1,000 \times D$
	Number of islands	n	$[1, 9]^1$
	Probability of migr.	p_m	$[0.001, 0.002, 0.003]^2$
P	Minimum island size	$MinNp$	$[10]^3$
	Maximum island size	$MaxNp$	$[20, 70, 100]^2$
	Minimum life time	$MinLT$	$[1]^3$
	Maximum life time	$MaxLT$	$[24, 30, 32]^2$
I	Initial scale factor	$F_i^{(0)}$	0.5
	Initial crossover rate	$CR_i^{(0)}$	0.9

[1]The interval of observed values.

[2]The best parameter setting according to the dimension.

[3]Valid for all dimensions.

when the age of an individual *individual.age* overcomes its lifetime *individual.LT*. The selection between sexual REPR and asexual CLONE reproduction depends on the entropy $\Delta_{\max}$ that determines the maximum number of newly created individuals. When this number is decremented to zero, only the asexual reproduction is allowed.

Experiments and results

The goal of this study was to move the original jDE algorithm using a monolithic population into the distributed gPVajDE, and to show that the proposed algorithm is suitable for solving the hard continuous optimization problems better than the other algorithms of the same class. Indeed, the purpose of this work can be condensed into four hypotheses, whose evidence can be shown that the proposed algorithm:

- is capable of solving the global optimization problems,
- can improve the behavior of classical linear population size reduction,
- maintains the population diversity better than the traditional EAs,
- achieves results comparable with the results of some traditional EAs.

During the experiments, the gPVajDE applied the parameter setup as illustrated in Table 1, from which it can be seen that the parameters operate on three different levels as denoted by the column 'L' in the table: 'A' - algorithm, 'P' - population, and 'I' - individual level.

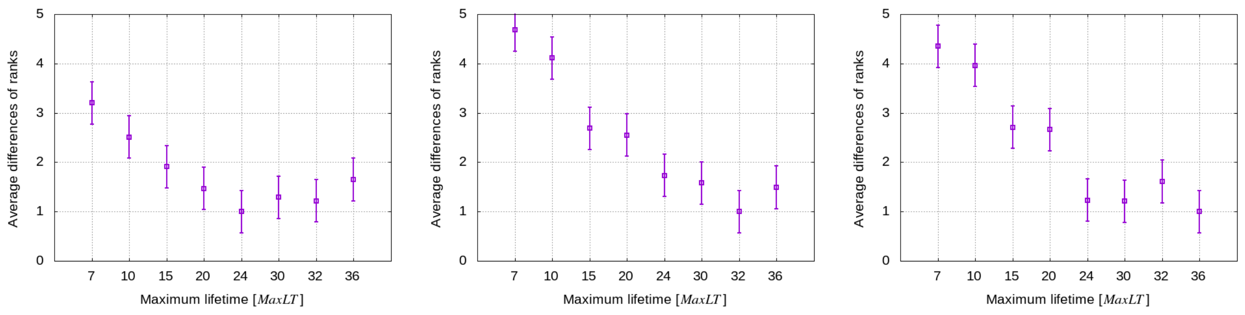
As a termination condition, the maximum number of fitness function evaluations was employed for all configurations of the algorithm. In this way, the comparisons with the results among various algorithms were performed fairly. The number of islands in the ring topology were varied in the interval $n \in [1, 9]$ in steps of 1, where configuration with $n = 1$ represents the original jDE using the monolithic population. Three optimal values of migration probability p_m are documented regarding the problem dimensions in the table. The island sub-population size was varied in the interval $Np \in [MinNp, MaxNp]$, where $MinNp$ denote the minimum sub-population size and $MaxNp$ the maximum one. Indeed, $MinNp = 10$ was applied for all configurations, while the maximum sub-population size in the table designates the best values according to the observed dimensions as found during the experimental work. The same is also true for parameters *MinLT* and *MinLT* denoting the minimum and maximum lifetime values, respectively. The complete graph topology of individuals was applied in our study. Finally, the initial values of parameters $F_i^{(0)} = 0.5$ and $CR_i^{(0)} = 0.9$ were used for the proposed algorithm during the experiments.

The results obtained by the algorithms were evaluated according to five standard statistical measures: *Best, Worst, Mean, Median,* and *StDev* values. Friedman's non-parametric statistical test [11] was conducted in order to estimate the quality of the results obtained by various nature-inspired algorithms for global optimization. This test is a two-way analysis of variances by ranks, where the null hypothesis is stated, assuming that medians between the ranks of all algorithms are equal. The second step is performed only if a null hypothesis of a Friedman test is rejected. In this step, the post-hoc tests are conducted using the calculated ranks. Indeed, the Nemenyi post-hoc test was used for graphical presentation of the results after determining the control method (i.e., the algorithm with the lowest rank). The post-hoc test was conducted using a significance level of 0.05.

The CEC'18 benchmark function suite was employed as a test-bed. It consists of 30 benchmark functions that are divided into four classes [1]: (1) unimodal functions (1–3), (2) simple multimodal functions (4–10), (3) hybrid functions (11–20), and (4) composition functions (21–30). Unimodal functions have a single global optimum and no local optima. Unimodal functions in this suite are non-separable and rotated. Multi-modal functions are either separable or non-separable. In addition, they are also rotated and/or shifted. To develop the hybrid functions, the variables are divided randomly into some sub-components and then different basic functions are used for different sub-components. Composition functions consist of a sum of two or more basic functions. In this suite, hybrid functions are used as the basic functions to construct composition functions. The characteristics of these hybrid and composition functions depend on the characteristics of the basic functions. The functions of dimensions $D = 10$, $D = 30$, and $D = 50$, were used in our experiments, while the search range of the problem variables was limited to $x_{i,j} \in [-100, 100]$.

Results

In order to confirm our hypotheses, the results of four tests are illustrated in detail:



a. Dimension $D = 10$. **b.** Dimension $D = 30$. **c.** Dimension $D = 50$.

Fig. 2. Influence of maximum lifetime $MaxLT$ by controlling the aging mechanism.

- influence of the aging mechanism,
- influence of the adaptive population growth,
- searching for the best algorithm parameter setting,
- comparative analysis.

In the remainder of the paper, the mentioned tests are described in detail.

Influence of the aging mechanism

The experiment was aimed at establishing the characteristics of the parameter $MaxLT$ by controlling the aging mechanism. The aging mechanism replaces the selection operator in traditional EAs, and favors those individuals that have the better function values. However, the mechanism depends on the lifetime parameter LT, based on minimum and maximum lifetime values, where the minimum lifetime was fixed to $MinLT = 1$, while the maximum lifetime was varied in the set of feasible values:

$$MaxLT \in \{7, 10, 15, 20, 24, 30, 32, 36\}.$$

Obviously, the initial value of this parameter is determined for each individual at the birth time randomly using the set of feasible values, while its optimal value is adapted by the evolutionary search process during the run. It turned out that the optimal value of the $MaxLT$ parameter depends strongly on the dimension of the problem in question.

The other parameters in the test were fixed as follows: $n = 1$, $Np^{(0)} = 55$, $Np^{(t)} \in [10, 100]$ for $t > 0$, and $migr\_rate = 0.000$. This means that searching for the optimal value of the parameter $MaxLT$ proceeded by the gPVajDE using the monolithic population and without migration. Due to the fact that the experiments were conducted for all three observed dimensions of the benchmark functions, there were conducted $8 \times 3 \times 25 = 600$ independent runs. In this term, the number 8 denotes the size of the domain values $|MaxLT|$, 3 the different number of dimensions, and 25 the number of independent runs per instance.

The results of experiments are depicted in Fig. 2, that consists of three diagrams according to the observed dimensions. Each graph presents the results of the Nemenyi post-hoc tests, obtained after conducting the Friedman non-parametric test, where each statistical classifier consists of $5 \times 30 = 150$ statistical measures (i.e., 5=the number of statistical measures, and 30=the number of functions) obtained after optimization of the proposed algorithm.

As can be seen from Fig. 2, the best results were obtained by searching for the parameter controlling the maximum lifetime to the value $MaxLT = 24$ by optimizing the CEC'18 benchmark functions of dimension $D = 10$. However, the value of the parameter maximum lifetime increased to $MaxLT = 32$ and $MaxLT = 36$ by dealing with functions of dimension $D = 30$ and $D = 50$, respectively.

Influence of the adaptive population growth

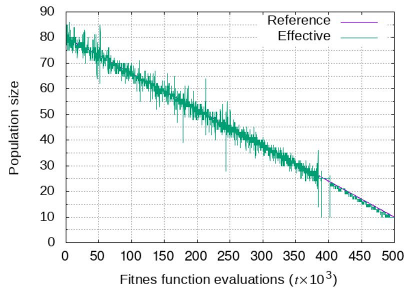
The purpose of the experiment was to establish the behavior of the Non-Linear population Size Reduction mechanism (NLSR), and to show that using it within the gPVajDE can also improve their results. This mechanism distinguishes itself from the famous Linear population Size Reduction (LSR) used in more popular stochastic nature-inspired population-based algorithms. Actually, while the LSR is capable of uniform decreasing of population size, when the search process becomes matured, the population size can also be increased by the NLSR imposing the new individuals via sexual reproduction. The population size reduction is called non-linear, because the relationship between the number of generations and population size is modeled non-linearly.

In order to show the advantage of NLSR, the various configurations of gPVajDE with different numbers of islands varying from one to six (denoted as c−1 to c−6) were compared with their counterparts using the LSR. However, the other parameters were set similar as in the last experiment. The results of the tests are depicted in Table 2, where the achievements obtained by gPVajDE of different configurations are compared according to the Wilcoxon 2-paired non-parametric statistical test using a significance level of 0.05. In summary, it can be seen from the table that, the results obtained by the gPVajDEs using LSR were significantly better than those obtained by their counterparts using NLSR three times (i.e., for $D = 10$ and $D = 30$), while, in contrast, the results of the latter were even seven times better. This means that applying the NLSR in place of LSR is reasonable.

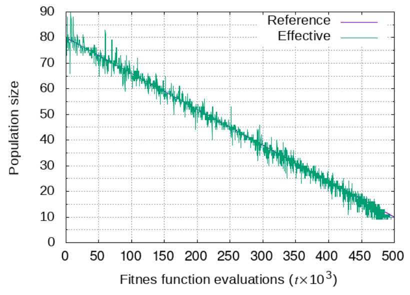
Table 2. Indicating the influence of the population diversity on the best results by gPVajDE.

Dim.	Meth.	c−1	c−2	c−3	c−4	c−5	c−6	Sum.
10	LSR	+	−	−	−	+	=	2
	NLSR	−	+	+	−	+	=	3
30	LSR	−	=	−	−	=	+	1
	NLSR	+	=	+	+	=	−	3
50	LSR	−	=	=	=	=	=	0
	NLSR	+	=	=	=	=	=	1

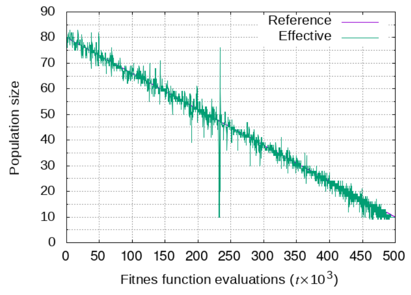
In the next test, the behavior of the NLSR feature was observed, where a composition function f_{21} of dimension



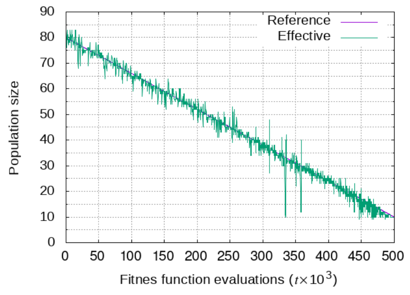
a. c−1 with one population.



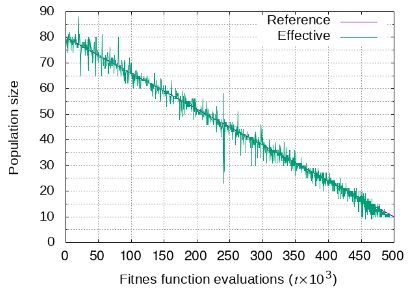
b. c−2 with first island.



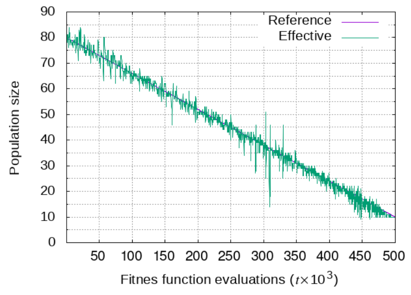
c. c−2 with second island.



d. c−3 with first island.



e. c−3 with second island.



f. c−3 with third island.

Fig. 3. Characteristics of the population size reduction obtained by optimizing CEC'18 benchmark function f_{21} of dimension $D = 50$ by gPVajDE of various configuration.

$D = 50$ was taken into consideration. The function was optimized by the proposed gPVajDE algorithm of three different configurations, i.e., using one (c−1), two (c−2), and three islands (c−3). Actually, one typical run of the function optimization was selected for each algorithm in question, where no optimal solution was achieved. Thus, the behavior of the algorithm can be monitored during the whole run. The result of the test is illustrated in Fig. 3 that is divided into six diagrams according to the particular configuration and the behavior of the population size in each island.

As can be seen from the diagrams in Fig. 3, the population size oscillated around the reference line, representing a traditional linear population size reduction. Typically, the population size increase is followed by size reduction. The increase is launched by imposing a sexual reproduction. On the other hand, reducing the population size is a consequence of the aging mechanism. Actually, this mechanism can introduce such a high selection pressure that it can eliminate the major part of individuals from the agent's population in only one cycle (e.g., look Fig. 3c). Consequently, imposing the new individuals by sexual reproduction in the next cycle causes replacing the vacant places in the island's sub-population.

Searching for the best gPVajDE parameter setting

As can be seen from Table 1, there are four algorithm parameters: While the first two (i.e., dimensions of the functions D and the maximum number of fitness function evaluations $nFEs$) are prescribed by the benchmark suite, the last two (i.e., the number of islands n and migration probability p_m) depend on the algorithm's configuration and, therefore, their proper setting is the subject of experimental work.

Indeed, this test focused on finding the gPVajDE configuration with the optimal number of islands n, migration probability p_m, and parameters $MaxNp$ and $MaxLT$, respectively. The best parameter setting was searched in two phases: After conducting the first test, in which the optimal values of the $MaxLT$ parameters were determined for each of the observed function dimensions, the second phase launched, where the parameter $MaxNp$ was varied in the set of the following values:

$$MaxNP \in \{20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 130, 150, 170\},$$

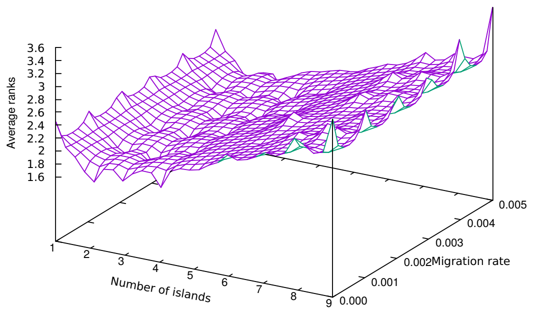
and the parameter was fixed to $MinNp = 10$. Thus, the initial value of the sub-population size was calculated as:

$$Np^{(0)} = \frac{MaxNp - MinNp}{2}. \qquad (15)$$

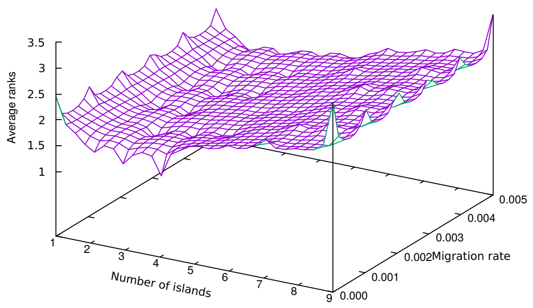
The other parameters, i.e., n and p_m, were varied as illustrated in Table 1. In summary, $9 \times 3 \times 13 \times 6 = 2.106$ different instances were taken into consideration. If we assume that each instance demanded 25 independent runs, the total number of independent runs was increased to $52,650$.

The huge number of results obtained after optimization were compared using the Friedman non-parametric statistical tests. Indeed, the results of these tests (i.e., ranks) were analyzed as follows: At first, the best interval of sub-population sizes $Np_k \in [MinNp, MaxNp_k]$ for $k = 1, \ldots, |MaxNp|$ needed to be calculated for each of the observed number of islands n, and migration probabilities p_m. Then, the corresponding ranks obtained by the particular pairs of parameters $\langle n, p_m\rangle$ was put into 3-dimensional space. In this manner, the so-called rank landscape was constituted with peaks and valleys. Obviously, the best ranks were identified as valleys.

The rank landscapes obtained using Friedman non-parametric statistical tests by optimization of CEC'18 benchmark functions of dimensions $D = 30$ and $D = 50$



a. Dimension $D = 30$ ($Np \in [10, 150]$, $lt \in [1, 32]$).



b. Dimension $D = 50$ ($Np \in [10, 170]$, $lt \in [1, 36]$).

Fig. 4. Rank landscape based on the Friedman non-parametric test obtained by optimizing the CEC'18 benchmark function suite.

are illustrated in Fig. 4, from which it can be seen that the best results were obtained using the sub-population sizes $Np^{(0)} \in [10, 150]$ and $Np^{(0)} \in [10, 170]$ by $D = 30$ and $D = 50$, respectively. The regions of the best results (i.e., the valleys in the rank landscapes) are found for the numbers of islands $n \in \{2, 3, 4\}$, while a breakaway from these numbers of islands caused bigger changes in the rank landscapes. This means that they became more hilly with increasing or decreasing the number of islands. Interestingly, the landscapes were also changed slightly according to the different migration rates. Similarly, values in the middle of the observed domain (i.e., $p_m \in \{0.001, 0.002, 0.003\}$) led to the better results. When comparing the landscape obtained by optimizing the benchmark functions of dimension $D = 30$ with the same of dimension $D = 50$, it can be seen that the landscape of the former was smoother than the latter.

The best parameter settings found during the experimental work are summarized in Table 3, where the results of optimizing the CEC'18 benchmark functions of all observed dimensions are shown. As can be seen from the table, the presented parameter

Table 3. The best parameter settings of gPVajDE obtained by optimizing the CEC'18 benchmark functions.

D	n	p_m	$MinLT$	$MaxLT$	$Np^{(0)}$	$Np-$domain
10	2	0.002	1	24	80	$[10, 150]$
30	3	0.001	1	32	80	$[10, 150]$
50	2	0.002	1	36	90	$[10, 170]$

settings justify the findings of the last test. This means that the best results were identified by gPVajDE using the smaller number of islands, except one. Also, the smaller probabilities of migration different than zero improved the results obtained by the algorithm. On the other hand, the optimal values of parameters $MinLT$ and $MaxLT$, as well as the optimal intervals of the sub-population sizes $Np-$domain and the corresponding initial values $Np^{(0)}$ are illustrated in the table.

Comparative analysis

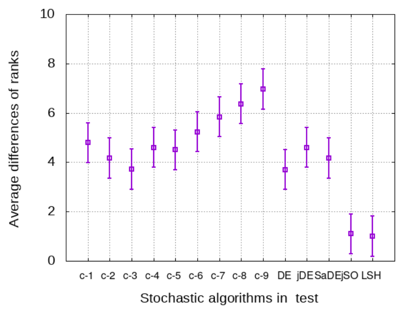
The goal of the test was to show that the results of the proposed gPVajDE algorithms were comparable with the results of the traditional stochastic nature-inspired population-based algorithms, like DE [17], jDE [5], and SaDE [16], and the state-of-the-art algorithms, like jSO [4] and LShade [18]. Thus, even nine configurations of gPVajDE algorithms with varying the number of islands from one (denoted as c−1) to nine (denoted as c−9) were taken into consideration. All the mentioned algorithms were applied for solving the CEC'18 benchmark function suite.

The results obtained from the particular algorithm were compared using the Friedman non-parametric statistical tests and refined by a Nemenyi post-hoc statistical test. In summary, the results of a comparative analysis are depicted in Fig. 5 that is divided into three diagrams according to the two parameters: the dimension of the functions D, and the corresponding migration rate p_m. However, only those settings of the parameter pair where the best results were obtained, are considered in the table.

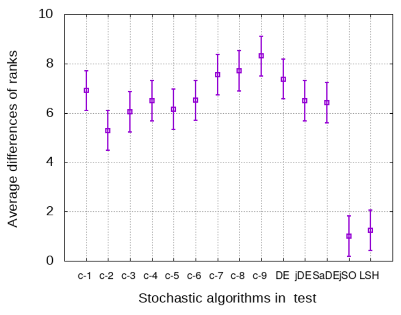
As can be seen from Fig. 5, the quality of the proposed gPVajDE algorithm depends on the number of islands. In line with this, the smaller number of islands was better than the higher. However, the gPVajDE using monolithic population (i.e., c−1) was not a preferable configuration, and must be excluded from the former assumption. The cause for this finding lies in the fact that all gPVajDE configurations used the same number of fitness function evaluations in one algorithm's run. Consequently, the gPVajDE configurations using more islands demand more small-sized sub-populations that are inefficient for the search process due to suffering a lack of population diversity. On the other hand, the configuration using a monolithic population maintains the higher population diversity, but suffers a lack of selection pressure. As a result, the proper bias between the population diversity and selection pressure ensure the optimal results of the configured gPVajDE algorithm. In our case, the reasonable number of islands must be higher than, or equal to two, but lower than, or equal to six. Increasing the population size over the maximum, or decreasing the minimum values deteriorated the results of gPVajDE drastically.

Discussion

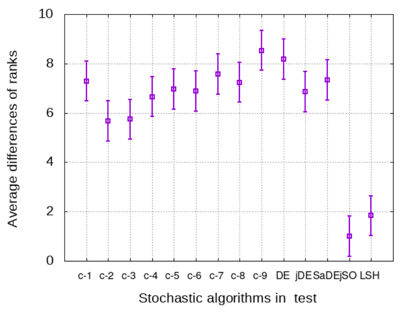
Four hypotheses were set at the beginning of the section and showing their validity was the subject of our experimental



a. Dimension $D = 10$ ($p_m = 0.002$).



b. Dimension $D = 30$ ($p_m = 0.001$).



c. Dimension $D = 50$ ($p_m = 0.002$).

Fig. 5. The results of comparative analysis using the Nemenyi post-hoc statistical test.

work. The following conclusions are summarized based on the performed work:

1. The results of applying the different gPVajDE configurations showed that this evolutionary algorithm is capable of solving the global optimization problems. However, the quality of the results depends on the proper parameter setting. Because this setup is not known in advance, extensive searching is needed for the optimal parameters.
2. Using the NLSR feature within the gPVajDE improves the results compared with its counterpart using the LSR irrespective of the problem dimension.
3. According to population diversity, it was found out that the proposed gPVajDE consisting of two and three islands maintains the population diversity longer than the traditional algorithms by optimizing the unimodal and simple multimodal functions, while the gPVajDE maintaining the monolithic population produced worse results than the traditional ones by optimizing all four groups of benchmark functions. Interestingly, the traditional algorithms, especially DE, are competitive according to the results achieved with the proposed gPVajDE, also by optimization of both hybrid and composition functions. Obviously, the main advantage of different configurations of the proposed algorithms presents the explicit control of losing the population diversity that can cause the algorithm to climb out from the local optima in certain situations.
4. In terms of comparison with the other traditional and state-of-the-art algorithms, it was found that the proposed gPVajDE algorithms are comparable, if not better, than the traditional stochastic nature-inspired population-based algorithms, like DE, jDE, and SaDE, but the results of the state-of-the-art algorithms, like jSO, and L-Shade, stayed an unreachable goal for the gPVajDE at the moment.

Under the aforementioned assumptions, the posted hypotheses hold for gPVajDE using more than one agent and less than six agents.

Conclusion

The question of how to maintain the diversity of population has been followed by researchers in the evolutionary community from the beginning. In this study, this problem is addressed by introducing the parallel gPVajDE algorithm, where the individuals are capable of executing some actions, with which they modify their programming environment and, thus, have more autonomy than those in the classical EAs. Additionally, the algorithm incorporates two mechanisms: aging, and adaptive population growth. While the former controls the action death, the latter navigates between the actions reproduction and clone implemented as different DE mutation strategies.

The proposed gPVajDE was applied for solving the CEC'18 benchmark function suite of three dimensions (i.e., $D = 10$, $D = 30$, and $D = 50$) representing a test-bed for global optimization. During huge experimental work, the influence of the aging and the new adaptive population growth mechanisms were discovered in detail. Then, the configuration of the gPVajDE with the optimal setting of parameters was searched for. Finally, the results produced by various gPVjaDE configurations were compared with some traditional stochastic nature-inspired population-based algorithms, like DE, jDE, and SaDE, and some state-of-the-art algorithms, like jSO, and L-Shade. From the comparative analysis, it can be seen that the results of the proposed gPVajDE are comparable with those of the traditional algorithms in question, while needing some improvements to become also more competitive with the state-of-the-art ones.

In the future, we would like to build the gPVajDE with pure agents capable of decision-making at the individual's level. Applying the algorithm for solving the other hard optimization problems seem to be a very challenging task for the future.

Acknowledgments

Iztok Fister is grateful for the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0042 - Digital twin). Iztok Fister Jr. is grateful for the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0057). Andres Iglesias and Akemi Galvez thank the Computer Science National Program of the Spanish Research Agency and European Funds, Project #TIN2017-89275-R. (AEI/FEDER, UE), and the PDE-GIR project of the European Union's Horizon 2020 program, Marie Sklodowska-Curie Actions Grant Agreement #778035.

References

1. N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan. Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective bound constrained real-parameter

numerical optimization. Technical report, Singapore, 2016. Technical Report.

2. Wolfgang Banzhaf and Barry McMullin. *Artificial Life*, pages 1805–1834. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
3. Christian Blum and Daniel Merkle. *Swarm Intelligence: Introduction and Applications*. Springer Publishing Company, Incorporated, 1 edition, 2008.
4. J. Brest, M. S. Maučec, and B. Bošković. Single objective real-parameter optimization: Algorithm jso. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 1311–1318, June 2017.
5. Janez Brest, Sašo Greiner, Borko Bošković, Marjan Mernik, and Viljem Žumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Trans. Evolutionary Computation*, 10(6):646–657, 2006.
6. Aleksander Byrski, Rafal Drezewski, Leszek Siwik, and Marek Kisiel-Dorohinicki. Evolutionary multi-agent systems. *Knowledge Eng. Review*, 30(2):171–186, 2015.
7. Charles Darwin. *On the Origin of Species by Means of Natural Selection*. Murray, London, 1859. or the Preservation of Favored Races in the Struggle for Life.
8. Lloyd Demetrius, Stéphane Legendre, and Peter Harremöes. Evolutionary entropy: A predictor of body size, metabolic rate and maximal life span. *Bulletin of Mathematical Biology*, 71(4):800–818, May 2009.
9. A. E. Eiben and James E. Smith. *Introduction to Evolutionary Computing*. Springer Publishing Company, Incorporated, 2nd edition, 2015.
10. Iztok Fister, Andres Iglesias, Akemi Galvez, Javier Del Ser, Eneko Osaba, Iztok Fister Jr., Matjaž Perc, and Mitja Slavinec. Novelty search for global optimization. *Applied Mathematics and Computation*, 347:865–881, 2019.
11. Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Statist.*, 11(1):86–92, 03 1940.
12. Gabriel Luque and Enrique Alba. *Parallel Genetic Algorithms: Theory and Real World Applications*. Springer Publishing Company, Incorporated, 2013.
13. Michael Lynch. The evolution of genetic networks by non-adaptive processes. *Nature Reviews Genetics*, 8:803–813, 2007.
14. Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, Berlin, 2st edition, 1996.
15. Ronald W. Morrison. *Designing Evolutionary Algorithms for Dynamic Environments*. Natural Computing Series. Springer-Verlag Berlin Heidelberg, 2004.
16. A. K. Qin and P. N. Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1785–1791, Sep. 2005.
17. Rainer Storn and Kenneth Price. Differential evolution &ndash; a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, December 1997.
18. R. Tanabe and A. S. Fukunaga. Improving the search performance of shade using linear population size reduction. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 1658–1665, July 2014.
19. Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv.*, 45(3):35:1–35:33, July 2013.

Iztok Fister. Iztok Fister received his degree in Computer Science from the University of Ljubljana, and his Ph.D. degree from the Faculty of Electrical Engineering and Computer Science, University of Maribor. Since 2010, he has been a Teaching Assistant with the Computer Architecture and Languages Laboratory, Faculty of Electrical Engineering and Computer Science, University of Maribor. His research interests include computer architectures, programming languages, operational researches, artificial intelligence, and evolutionary algorithms. In his research areas, he has published more original scientific papers, review papers, book chapters, edited books, and contributed to more than 50 international scientific conferences. He is also one of the presidents in the international student conference in computer science. In many journals with impact factor, he has been promoted as an outstanding reviewer.

Andres Iglesias. Andres Iglesias works at Toho University (Funabashi, Japan) and the University of Cantabria (Santander, Spain), where he is the Director of the Computer Graphics and Artificial Intelligence (CGAI) research group. He has published more than 230 scientific papers (including journal papers in 21 categories of JCR) and 14 books, and holds two patents. He is the current chair of IFIP TC5 WG5.10 and Editorial Board Member of 15 international journals. He has been the chair/organizer of 55 international conferences and workshops and a program committee member of more than 300 international conferences. He has been a reviewer of about 200 international papers in journals (mostly in JCR journals) and more than 500 papers in international conferences, as well as an expert evaluator of projects for NSF (USA), the European Commission Horizon 2020 and FP7, and other national and regional public research agencies in several countries. His fields of interest are computer graphics, geometric modelling, artificial intelligence, and soft computing.

Akemi Galvez. Akemi Galvez works at Toho University (Funabashi, Japan) and the University of Cantabria (Santander, Spain), and is a member of the Computer Graphics and Artificial Intelligence (CGAI) research group. She has published more than 160 international papers in scientific journals and conferences, including papers in top journals of several categories of JCR, and holds two patents. She is also an Editorial Board Member of seven international journals. She has also been co-chair/co-organizer of prestigious international conferences such as ICMS'2006, ICCSA'2011, and Cyberworlds'2014, and reviewer/program committee member of several international conferences in the fields of computer graphics and artificial intelligence. Her research interests include geometric modelling and processing, shape reconstruction, artificial intelligence, soft computing, and their industrial applications.

Dušan Fister. Dušan Fister. received his B.Sc and M.Sc. in Mechatronics from the University of Maribor, Slovenia and is currently working towards his Ph.D. in Economics. His areas of interest include econometrics, data mining, optimization, robotics and controllers. He has published more than 60 research articles in referred journals, conferences and book chapters.

He is currently an assistant at the Faculty of Economics and Business, University of Maribor.

Iztok Fister Jr.. Iztok Fister Jr. received his B.Sc, M. Sc. and PhD in Computer Science from the University of Maribor, Slovenia. His areas of interest include data mining, pervasive computing, optimization and sport science. He has published more than 120 research articles in referred journals, conferences and book chapters. Furthermore, he is a member of the editorial boards of 5 different international journals, and he has acted as a program committee member in more than 30 international conferences. He is currently an assistant professor at the University of Maribor, Slovenia.