JAISCR, 2023, Vol. 13, No. 4, pp. 289 – 306
10.2478/jaiscr-2023-0020

A NEW APPROACH TO DETECTING AND PREVENTING POPULATIONS STAGNATION THROUGH DYNAMIC CHANGES IN MULTI-POPULATION-BASED ALGORITHMS

Krystian Łapa[1,*], Danuta Rutkowska[2], Aleksander Byrski[3], Christian Napoli[4]

[1] *Częstochowa University of Technology*
42-200 Częstochowa, Poland

[2] *University of Social Sciences*
90-113, Łódź, Poland

[3] *AGH University of Kraków*
30-059, Krakow, Poland

[4] *Christian Napoli*
Sapienza University of Rome
00185 Rome, Italy

[*] *E-mail: krystian.lapa@pcz.pl*

Submitted: 6th May 2023; Accepted: 11th October 2023

Abstract

In this paper, a new mechanism for detecting population stagnation based on the analysis of the local improvement of the evaluation function and the infinite impulse response filter is proposed. The purpose of this mechanism is to improve the population stagnation detection capability for various optimization scenarios, and thus to improve multi-population-based algorithms (MPBAs) performance. In addition, various other approaches have been proposed to eliminate stagnation, including approaches aimed at both improving performance and reducing the complexity of the algorithms. The developed methods were tested, among the others, for various migration topologies and various MPBAs, including the MNIA algorithm, which allows the use of many different base algorithms and thus eliminates the need to select the population-based algorithm for a given simulation problem. The simulations were performed for typical benchmark functions and control problems. The obtained results confirm the validity of the developed method.

Keywords: multi-population-based algorithms, migration topologies, population stagnation

1 Introduction

Population-based algorithms allow for iterative processing of a population of solutions (called individuals) to find the most optimal solution. The solutions are encoded by the parameters of the individuals, while their quality is determined by the evaluation function associated with the problem under consideration. The parameters encoded by individuals are modified by various mechanisms (operators) that can be local (exploitation), global (exploration), or everything in between. In modifying the parameters of one individual, only random or auxiliary variables (e.g. mutation or momentum, respectively) or parameters of other individuals (e.g. crossover) may be used. The use of parameters of other individuals, or the use of small changes in parameters, can cause the whole population to become very similar in terms of these parameters, and the solutions converge to one area of the parameter space (search space). Thus populations can often stuck in the local optimum. Several mechanisms can be used to prevent this, and one of them is the simultaneous use of multiple populations, the approach considered in this paper.

Multi-population-based algorithms (MPBAs) use multiple populations (also called islands or sub-populations) that are usually processed independently, while in certain situations individuals between them are exchanged (migrated). The methods of sub-populations processing are determined by the training plans. The selection of the populations for the exchange of individuals is determined by the topology of their connections (migration topology). The number of exchanged individuals and the method of their selection is determined by the migration strategy, while the replacement period is determined by the frequency of migration. Thus, many factors can affect the performance of multi-population-based algorithms (including selecting basic optimization parameters), which makes the problem of their configuration complicated and may lead to premature stagnation. The more details on the problems associated with MPBAs are described in the next section. In this paper, an attempt was made to further improve MPBAs in various aspects, proposing more universal solutions, and including using mechanisms that try to prevent premature convergence.

1.1 Motivation

The high popularity of population-based and multi-population-based algorithms has its advantages and disadvantages. On the one hand, new and better algorithms appear frequently, and algorithms are adapted to more and more complex simulation problems (see e.g. [14]). On the other hand, the multitude of algorithms and their versions makes it difficult to choose the best algorithm for a given simulation problem (see e.g. [7, 15, 18, 43]). A large number of population-based algorithms also makes it difficult to compare them, and most often in the papers on a new version of an algorithm, it is compared only with related algorithms (or their modifications) or original versions of other algorithms (see e.g. [24, 37]). This also applies to various mechanisms used in multi-population-based algorithms, such as the development of new topologies and migration strategies, mechanisms to prevent stagnation, etc., which are sometimes only tested for one population-based algorithm or tested for multiple algorithms but not compared with other strategies. Such papers, however, then focus on other aspects of the operation of the considered algorithms (see e.g. [1, 17]). Of course, these problems arise mainly due to the difficulty of running such complex simulations and keeping track of all the latest versions of various population-based algorithms, and thus they are practically inevitable.

In this paper, it was decided to focus on universality. Firstly, the MNIA algorithm is further developed, in which many sub-populations based on different population-based algorithms can cooperate. Secondly, new approaches were proposed for stagnation detection and the dynamic changes in active populations and the number of their individuals. These solutions are designed to apply to most migration topologies and various MPBAs. Thirdly, the proposed approaches, in addition to a comparison with known mechanisms, will be tested for various migration topologies and various sub-population algorithms. Such assumptions, besides the proposal of new mechanisms, should not only find more efficient configurations of algorithms that give better optimization results but also determine the further possible direction of development of such algorithms.

1.2 Contribution

The contribution of this paper to the development of population-based algorithms is as follows: a) directing further development of more universal methods that allow to eliminate the problem of selecting a specific population-based algorithm, in particular, the development of the MNIA algorithm [40]; b) proposing and applying known and new mechanisms related to the dynamic change in the number of populations and their individuals, aimed at improving the accuracy of population-based algorithms; c) proposing and applying known and new mechanisms related to the dynamic change in the number of populations and their individuals, aimed at improving the complexity of population-based algorithms; d) development and testing of a new criterion to determine the level of population stagnation, in particular for use in a multi-population-based algorithms; e) a detailed and comprehensive comparison of different methods for changing the number of populations and the number of individuals used with different migration topologies and for different multi-population-based algorithms.

1.3 Paper Structure

The structure of the paper is as follows: Section 2 presents a review of the literature, Section 3 describes the proposed method, Section 4 presents considered simulation variants and obtained results, and Section 5 draws final conclusions.

2 Background

According to the no-free lunch theory [2], there is no single population-based algorithm that is best suited for all simulation problems. This, on the one hand, causes a problem because a lot of new population-based algorithms are being developed, which is already criticized (see e.g. [7, 15, 43]). An overview of various algorithms can be found e.g. in [15, 23, 38, 45]. On the other hand, new algorithms are developed in such a way as to improve the search mechanisms and avoid premature convergence of the algorithm (stagnation of the population at the local optimum), and therefore this is the positive part. New algorithms have more configuration options, although there are exceptions, e.g. [31]). Moreover, new algorithms can be more computationally complex, which can be critical when optimizing more complicated simulation problems (e.g. control systems problems [25, 32], fuzzy systems [41, 47]). Moreover, the spectrum of applications of these algorithms is constantly increasing (see e.g. [48]). At the same time, some algorithms combine the advantages of different algorithms and become universal, partially eliminating the problem of choosing a specific population-based algorithm for a specific simulation problem (see e.g. [40]). Despite the rapid development of population-based algorithms, the problem of premature convergence (stagnation) still occurs and is crucial.

One way to prevent population stagnation in population-based algorithms is to diversify the behavior of the population. In the simplest approaches with one population individuals can be divided according to certain criteria. Examples are Grey Wolf Optimizer, in which three main individuals and the rest of the herd are distinguished [31], Bison Algorithm, in which there are individuals with two different behaviors [21], or Termite Queen Algorithm, in which there are as many as five different types of individuals (queen, flying worker, foraging worker, serving worker, and soldier [9]). In some cases, distinguished individuals can be modified using different methods of changing parameters that originate from different population-based algorithms, and thus hybridization occurs (see e.g. [11, 14, 27]).

The advanced approach consists of dividing the population into separated sub-populations (islands) that exchange solutions in certain circumstances. Usually, islands operate based on the same population-based algorithm, and the method of modifying individuals on each island may be identical (see e.g. [4]), or depending on the parameters of a given island (see e.g. [36]). There are also solutions where the behavior of each island can be based on different operators or different population algorithms (see e.g. [40]). When only two or three sub-populations are used and a different population-based algorithm is used for each of them, such solutions are also called hybrid (see e.g. [33]). In the case of a larger number of sub-populations, it becomes crucial to select the topology of their connections determining the way of exchanging individuals and other mechanisms described later in this section, broken down into topologies, migration strategies, parameters, adaptive mechanisms, and hybridizations with other, non-population-based, algorithms. Each of these mechanisms can have a significant impact on the algorithm's effectiveness and thus is important from the point of view of this paper.

2.1 Migration topologies

There are many ways to improve the performance of multi-population-based algorithms. The first one is the selection of the appropriate migration topology. Different topologies have different effects on population diversification and search intensification. The selection of the appropriate topology can affect the balance between exploration and exploitation and also affect it differently depending on the simulation problem under consideration.

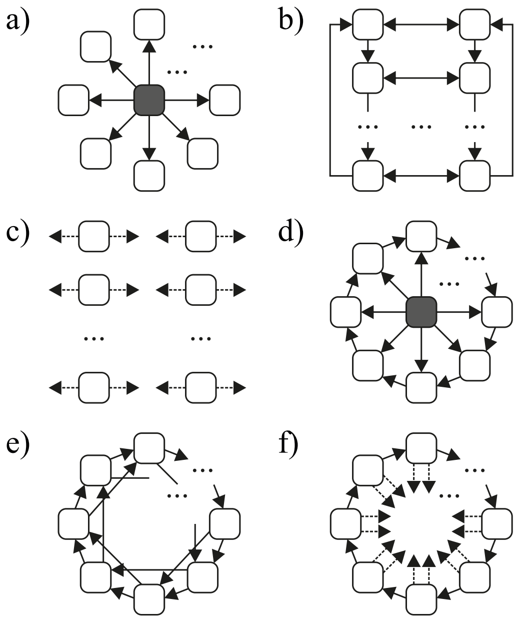

Figure 1. Efficient migration topologies according to [12]: a) inverted star with the best population in the middle (ISBM), b) ladder with a down-top connection (LDTC), c) topology with two random connections for each population (TTRC), d) inverted star as in a) combined with ring (IRBM) e) ring with additional connections between neighbors (RWAC), f) ring with additional random connections, also called small world (SMWD). Where: rectangles stand for populations, dark rectangles stand for populations that are conditionally replaced with best population, arrows stand for migration directions, and dotted arrows stand for random connections.

The most common topologies according to [19] are star and ring, while others worth noting are torus and lattice. Some topologies also change dynamically, examples are randomly changed topologies (see e.g. [8]) or topologies where the best or worst population is placed in a specific location (see e.g. [12]). Moreover, different approaches, such as a fitness-based Migration Policy designed to promote the maintenance of diversity through a mechanism that combines groups of individuals to alternate between exploration and exploitation proposed in [6], can be used instead.

An interesting overview and performance tests of 36 different migration topologies is presented in [12]. The best topologies, regardless of other configurations and simulation problems used, turned out to be: an inverted star with a mechanism for setting the best population in the middle of topology (see Figure 1.a)), a ladder with a down-top connection (see Figure 1.b)), and a topology in which each population has two random connections with others (see Figure 1.c)). Due to their universality, these topologies will be used in the simulations in this paper.

2.2 Migration strategies

Sub-populations migration is a key element of island algorithms. This mechanism may relate to the frequency of migration, the number of individuals to be exchanged, and the method of selection of individuals for exchange. Not only does population diversification depend on these configurations, but also the accuracy of the search. In addition to the standard configuration described above and specifying methods for selecting and replacing individuals, other solutions are also used. In [35] the individuals of the two populations are mixed randomly at each iteration of the algorithm, and in [13] individuals are moved into a dynamic number of sub-populations instead. In [49] a grouping mechanism is used to divide the population into two subswarms. In the [30] the authors use a superior population from which they send non-dominated solutions to subpopulations.

The paper [12] shows the influence of migration parameters on a given migration topology. The best parameters turned out to be the frequency of replacement every 10 iterations of the algorithm, and the selection of an individual for replacement using the roulette wheel method. With such parameters, the following topologies turned out to be worth nothing: an inverted star with a ring and placing the best population in the middle (see Figure 1.d)), a ring with additional connections between neighboring populations (see Figure 1.e)) and a ring with two additional random connections (see Figure 1.f)).

These topologies were also considered in this paper's simulations.

2.3 Parameter adjustment

The selection of parameters for MPBAs is the third important factor in fine-tuning these algorithms. This includes, among others, the selection of the number of islands and the number of individuals in these islands. Choosing the number of individuals is not an easy task, because each population-based algorithm works differently for different numbers of individuals (some algorithms need fewer individuals and more iterations, and vice versa). The parameters of specific islands are also important, which may be the same for each island or different. What is more, the operation of each island can be based on mechanisms from other population-based algorithms.

Parameter tuning can be done offline (selection of parameters before applying the algorithm) or online where parameters are changed or adapted during the simulations [16]. In the case of offline parameter selection, they can be set by trial and error to static values, or values depending on a given step of the algorithm (see e.g. [11]). In the case of MPBAs, different parameters can be set for each population and thus increase the chance of adapting the algorithm to a given problem (see e.g. [40]). Another parameter optimization idea is meta-optimization, which is the optimization of parameters for some underlying optimizer using a different algorithm [16, 26]. This approach is also used in multi-criteria algorithms, where, for example, algorithm parameters are optimized in such a way as to obtain the best results of multi-criteria performance metrics [26].

The paper [12] shows that, regardless of the migration topology and the simulation problem under consideration used, a good configuration is a topology with 8 islands composed of 32 individuals each, while the paper [40] shows that algorithms in which islands operate based on different population algorithms allow obtaining better and more stable results.

2.4 Adaptive mechanisms

Adaptive mechanisms mean dynamic adjustment of the algorithm operation through all configurations (migration topology, migration strategy, and algorithm parameters). Modification of the above settings is most often done based on the detection of population stagnation and optimization progress analysis. Some of these mechanisms are also used for common single-population-based algorithms.

In [46] Differential Evolution parameters are adaptively adjusted according to the statistical information learned from the previous searches in generating improved solutions. In [5] fuzzy system is used to control population diversity at decision variable space by self-adapting the crossover rate control parameter. In [22] an adapted crossover rate value is assigned to each individual according to individual fitness value. It is also worth mentioning the adaptive mechanisms of re-initializing the population (or part of it), which allow one to search subsequent areas of search space and thus to improve the results (see e.g. [29]). Adaptive mechanisms can be focused also on the complexity of the optimization - e.g. by dynamically changing the number of populations or individuals, as well as a dynamic selection of surrogate solutions (see e.g. [11]).

In the [28], individual island training plans (both population algorithms and their parameters) are changed based on the optimization progress, which brought a significant improvement. In the paper [29], when stagnation was detected, various mechanisms were used to re-initialize those populations that performed the worst. In this case, the best solution turned out to be re-initialization involving the creation of new individuals based on the mechanisms of differential evolution and the use for re-initialization the individuals from other populations. In this paper, it is planned to test the use of different approaches to adaptive changes in the MPBAs.

2.5 Hybridizations

In addition to the aforementioned hybridization consisting of the simultaneous use of different population-based algorithms or operators derived from them, it is worth mentioning that the literature is full of methods in which population-based algorithms are combined with other families of algorithms. Hybridizations worth mentioning are the combination of population-based algorithms with memetic algorithms, which are also evolutionary algorithms but focused on local search (see e.g. [34, 44]) or algorithms based on backpropagation (see e.g. [3, 20]). This shows how important the population-based methods are and that there is still a huge potential for their development. Such approaches will not be considered in this paper due to their complexity, however, it is possible to explore such an interesting topic in the future.

3 Proposed approach

Section 2 shows how many different approaches can be used to improve the performance of MPBAs, how many parameters influence the performance of these algorithms, and what approaches allow them to obtain promising results. This section describes the proposed approach and its configuration with its division into the mechanisms of changing the number of populations and the number of individuals within the populations, mechanisms of preventing population stagnation, and a complete description of the proposed algorithm. The proposed approach has been developed in such a way that it is as universal as possible and can be used for many topologies, algorithms, and problems.

3.1 Dynamic number of populations and individuals within populations

There are many approaches for changing population and individual numbers (see also Section 2.2). The purpose of such changes may be to increase the number of individuals in more promising populations, remove less promising populations or add new populations, increase the diversity of individuals, or reduce computational complexity, all to improve the performance of MPBAs. Some methods are typical for specific algorithms and do not allow their use in different topologies. For example, the ICA [4] algorithm has individuals grouped into empires with an imposed migration topology, in which the best population gains an individual from the worst population, thus the number of groups in the population may slowly decrease and thus it is not possible to apply other topologies directly.

In this paper, the use of the following universal approaches is proposed allowing for their use in most migration topologies: a) disabling or enabling of specified populations - in this case, the disabled populations are not iterated, but to ensure compliance with the migration topology, they may continue to participate in the transfer of individuals, b) dynamic changes in the number of individuals of selected populations - if there is a need to reduce the number of individuals, the worst solutions in terms of adopted evaluation function are removed, and if there is a need to increase the number of individuals, new ones are created based on tournament selection of individuals from other populations and the use of a mutation operator derived from the genetic algorithm (to maintain population diversity). The above mechanisms can be used in any combination (see Table 1) and when certain conditions occur (e.g. stagnation detection). The idea of their use is as follows: populations are sorted according to the evaluation function of the best individuals, the number of population individuals is checked against the number of individuals resulting from the adopted strategy (see Table 1), if individuals need to be added, removed or a given population needs to be disabled or enabled, the approaches described above are used.

The proposed approaches were divided into two categories: focused on improving accuracy (IA) and focused on reducing complexity (RC). Their combinations are presented in Table 1. The first approach involved increasing individuals in $Nchg$ populations while extinguishing individuals in other $Nchg$ populations (for the details see DBDSW and DSDSB in Table 1). The next goal is just to extinguish individuals in $Nchg$ populations and thus reduce complexity (see DSBST and DSWRS in Table 1). Then, the use of a solution was proposed in which there are no disabled populations but only changes in the number of individuals - increasing the number of individuals in the $2 \cdot Nchg$ populations and reducing them in the second $2 \cdot Nchg$ populations (see LWIBS and LBIWR in Table 1). This variant was also presented similarly in the version with complexity reduction (see LWRWR and LWRBS in Table 1). The latest variants were designed with a more linear change in the number of individuals in the analogous matter (see LDFBW, LDFWB, LDRBW, and LDRWB in Table 1). The above changes were designed so that the number of individuals in the IA approach is constant, while in the RC approach, it was reduced by 25% compared to the variant aimed at improving accuracy. It is worth noting, that despite assumptions about how to add or remove indi-

Table 1. Mechanisms for changing the number of populations and individuals within populations used to improve accuracy or complexity (see Section 3.1). Examples are given for $Npop = 8, Nind = 32, Nchg = 2$.

strategy	aim	description	examples of population sizes ordered from best to worst	processed individuals
DBDSW	IA	disable *Nchg* best populations, double size of *Nchg* worst populations	-, -, 32, 32, 32, 32, 64, 64	100%
DWDSB	IA	disable *Nchg* worst populations, double size of *Nchg* best populations	64, 64, 32, 32, 32, 32, -, -	100%
DSBST	RC	disable *Nchg* best populations	-, -, 32, 32, 32, 32, 32, 32	75%
DSWRS	RC	disable *Nchg* worst populations	32, 32, 32, 32, 32, 32, -, -	75%
LWIBS	IA	lower $2 \cdot Nchg$ worst, increase size of $2 \cdot Nchg$ best	48, 48, 48, 48, 16, 16, 16, 16	100%
LBIWR	IA	lower $2 \cdot Nchg$ worst, increase size of $2 \cdot Nchg$ best	16, 16, 16, 16, 48, 48, 48, 48	100%
LWRWR	RC	lower $2 \cdot Nchg$ worst individuals	32, 32, 32, 32, 16, 16, 16, 16	75%
LWRBS	RC	lower $2 \cdot Nchg$ best individuals	16, 16, 16, 16, 32, 32, 32, 32	75%
LDFBW	IA	linear individuals distribution, from best to worst	60, 52, 44, 36, 28, 20, 12, 4	100%
LDFWB	IA	linear individuals distribution, from worst to best	4, 12, 20, 28, 36, 44, 52, 60	100%
LDRBW	RC	half-linear distribution with reduction, from best to worst	32, 32, 32, 32, 28, 20, 12, 4	75%
LDRWB	RC	half-linear distribution with reduction, from worst to best	4, 12, 20, 28, 32, 32, 32, 32	75%

viduals, the proposed general approach does not determine a rigid definition of these mechanisms and any other approaches can be used instead.

3.2 Prevention of population stagnation

The mechanisms presented in Section 3.1 can be used in various situations. In this paper, it was assumed that their main purpose was to improve MPBAs and to prevent stagnation. Therefore, it is crucial to determine at what point in the optimization they should be used.

In the basic approach (STA), dynamic change mechanisms are triggered after a certain number of iterations of the algorithm. In more advanced approaches, various mechanisms for detecting stagnation can be used. One of them is detecting whether there has been an improvement in the evaluation function over the $Nimp$ iterations (SFF). In addition to detecting stagnation, it is also possible to use population convergence techniques that are based on the similarity of individuals. However, all of the mentioned mechanisms may be disturbed, because even a small improvement in accuracy results in the lack of use of given mechanisms. The solution is to check the improvement above a certain threshold or to calculate the average fitness of the population, thus the problem of determining the threshold for a given simulation problem and the problem of disturbance of the average fitness by migrations arise, respectively. In addition, in the case of multi-population-based algorithms, it is necessary to choose whether these mechanisms should be used separately for each population or whether to test the general stagnation of all populations. In the first approach, it is not possible to use some approaches to prevent stagnation (e.g. in the case of a change in the number of individuals of all populations), thus it is not used in this paper.

To prevent the above problems, a new criterion for stagnation detection, resistant to both minor and abrupt changes in the improvement of the evaluation function, was proposed in this paper. The use of standard improvement is difficult due to, for example, the possibility of the fitness value falling below zero (thus logarithmic approaches do not always apply) as well as the aforementioned problem of analysis of a given simulation benchmark (it is required that it works equally well at the beginning of the simulations when the changes in the fitness function values are large, as well as in the final stage of optimization, and regardless of the range of values of a given simulation problem). To solve that, first, a flexible percentage improvement in fitness values between iterations is used as a basis. The proposed flexible percentage is calculated for $Nels$ of the last iterations of each population and checks the level of improvement of the evaluation function relative to the improvement from $Nels$ optimization steps:

$$imp_i(iter) = \frac{\mathrm{ff}_i^{bst}(iter-1) - \mathrm{ff}_i^{bst}(iter)}{\mathrm{ff}_i^{bst}(iter-Nels) - \mathrm{ff}_i^{bst}(iter)}, \quad (1)$$

where $\mathrm{ff}_i^{bst}(iter)$ is the best fitness function value of individual from i-th population for $iter$ iteration step ($i = 1,...,Npop$, $Npop$ is the number of subpopulations), $Nels$ is the adopted time window. It is worth noting that in the case of the above equation, if $\mathrm{ff}_i^{bst}(iter-stp) - \mathrm{ff}_i^{bst}(iter) = 0$, the result should be equal to zero, and if $iter < stp$, the default value of the improvement γ should be used. To immunize the above improvement factor to small and large fluctuations in the improvement of the evaluation function, an infinite impulse response filter was used (see [10]). In this case, the general formula for proposed stagnation detection for single population is defined as follows:

$$stg_i(iter) = \alpha \cdot stg_i(iter-1) + (1-\alpha) \cdot stg_i(iter), \quad (2)$$

where α is a coefficient determining how much the value from the previous iteration step has an impact on the proposed criterion. In the case of a first iteration step $stg_i(1)$, the value is set to the coefficient γ value. The stagnation in the proposed approach can be detected by checking if the $stg_i(iter)$ value has dropped below a certain β value. Examples of the calculated ratio are shown in Figure 2. As can be seen in Figure 2, the proposed value $stg_i(iter)$ works well in both presented cases. It is worth noting that in case Figure 2.a) there is often a small improvement in the population, making the SFF approach ineffective and detects stagnation only towards the end, while in case of Figure 2.b) there is more downtime during optimization, sometimes after a big improvement - in this case, using the standard approaches may be too frequent. The proposed approach (ELA) is new in the literature, in partic-

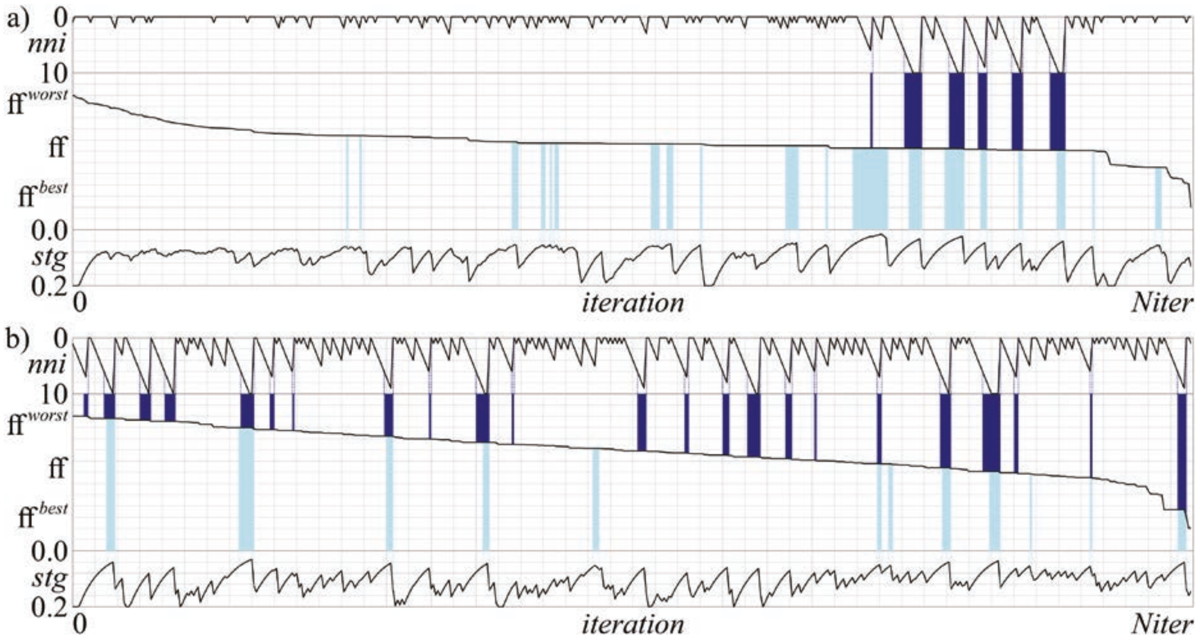

Figure 2. Example of population stagnation detection: area marked in a dark color - standard approach, area marked in a light color - proposed approach, stg stands for proposed criterion (see equation (2)), and nni stands for the number of iterations without improvement: a) slow and steady improvement in the middle of the optimization, b) more dynamic but stable changes in the fitness values.

ular in terms of performance evaluation of multi-population algorithms.

3.3 Proposed algorithm

The approach proposed in this paper has been designed universally so that most migration topologies, stagnation repair mechanisms, or population-based algorithms can be used.

The algorithm consists of the following steps:

Step 1. Initialization of $Npop$ populations and for each of them an equal number of $Nind$ individuals. Both the initialization method and the choice of population processing algorithms can be completely arbitrary. In the case of the MNIA approach, each population can be processed by a different algorithm with different parameters. In this step, individuals are also evaluated and the base number of individuals and population status (disabled/enabled) are changed according to the rules defined by Table 1 and the optimization parameters. After initialization, the populations are set up according to the adopted migration topology (see Fig 1). Setting the $iter$ variable to 0.

Step 2. In this step, the processing of individuals of each population is triggered by the adopted population-based algorithm. This step can be processed a certain number of iterations ($Niter$) or until a certain optimization fitness function value error threshold is reached. At the end of a single iteration, the $iter$ variable is incremented and the following conditions are checked:

– Checking whether the migration topology needs to be changed, e.g. in the case of the ISBM topology, the best solutions are transferred to the central point of the topology. Changes are checked and applied in each iteration of the algorithm.

– Checking if $Nmig$ iterations have elapsed. If so, the exchange of individuals between populations is by the adopted topology and migration strategy.

– Checking the conditions determining the need to change the number of active populations or the number of individuals within them according to the proposed mechanism of reducing and adding individuals (see Section 3.1). If the conditions are met for at least $Nwar$ of the sub-populations, the strategies presented in Table 1 apply. The examples of such changes are shown in Table 2.

Table 2. An example of changes in populations after stagnation is detected and the populations are sorted according to the evaluation function of their best individuals for the DWDSB case, $Npop = 8$, $Nind = 32$ and $Nchg = 2$ (see Table 1). The number of individuals and population statuses results from previous changes where the populations may have been sorted in a different order, '-' stands for disabled population.

population (i)	number of individuals	new number of individuals	action
1	32	64	add individuals
2	64	64	do nothing
3	32	32	do nothing
4	32	32	do nothing
5	64	32	remove individuals
6	-	32	enable population
7	-	-	do nothing
8	32	-	disable population

Step 3. Checking if the stop condition is reached, if not return to step 2. If so, present the best-found solution in terms of the adapted fitness function.

The algorithm considered in this paper does not require a specific approach to encoding solutions and their evaluation and these issues will not be considered in detail. However, the algorithm can be easily adapted, to solving hybrid-type problems (consisting of finding the solution structure and its parameters), etc. Then, the method of encoding solutions may be analogous to that presented in the papers [25, 40].

4 Simulations

The simulations were carried out in such a way as to test both the strategies from Table 1 and compare different approaches to preventing stagnation (including the proposed one), all for different migration topologies, different population algorithms, and various simulation problems.

The assumptions and parameters of the simulation are as follows: topologies tested: ISBM, LDTC, TTRC, IRBM, RWAC, SMWD (see Figure 1), all strategies tested from Table 1, for the migrations one worst individual from one population is replaced by one individual selected by roulette wheel from other randomly selected population. Used multi-population-based algorithms: island algorithms with same population-based in all populations MPGA, MPDE, MPCS, MPGWO (with consecutively base algorithms: GA, DE, CS, GWO with default parameters from the literature), MNIA (GA+DE+CS+GWO). Base number of populations being changed after stagnation detection $Nchg$ = 2, number of iterations for STA approach $Nsta$ = 25, number of iterations without improvements for SFF approach $Nimp$ = 6, parameters for proposed method ELA: $\alpha = 0.9$, $Nstp = 10$, $\beta = 0.06$, $\gamma = 0.2$, number of populations with stagnation to apply prevention mechanisms $Nwar$ = 2, number of islands $Npop$ = 8, based number of individuals in islands $Nind = 32$, number of iterations $Niter = 1000$, number of simulation repetitions $Nrep$ = 50, migration interval $Nmig$ = 10. The mentioned parameters of the above approaches were determined by trial and error method (see also Figure 3) or based on the values suggested in the literature, however, in the future, it is possible to test the meta-optimization approach, in which the parameters will be selected automatically.

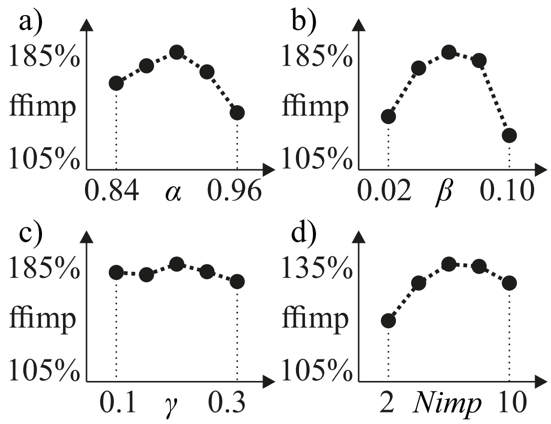

Figure 3. Influence of parameters of ELA and SFF approaches on fitness function values improvement (ffimp). In the case of a), b) and c), parameters other than those tested were set as follows: $\alpha = 0.9$, $\beta = 0.06$, $\gamma = 0.2$.

Table 3. Results averaged across all benchmark functions and all multi-population-based algorithms. The results show the improvement in results (in percent) relative to the same methods without the use of stagnation detection and prevention mechanisms. SM stands for stagnation mechanisms, SS stands for stagnation strategy. The best results for the accuracy approach are in bold, and for the complexity reduction approach are underlined.

SM	SS	ISBM	LDTC	TTRC	IRBM	RWAC	SMWD	AVG
STA	DBDSW	147	173	143	243	254	199	193
	DWDSB	216	118	120	150	-154	69	86
	DSBST	33	41	50	50	52	47	46
	DSWRS	52	49	62	77	59	85	64
	LWIBS	185	148	139	222	201	244	190
	LBIWR	773	337	282	-111	-94	39	204
	LWRWR	111	96	94	141	123	129	116
	LWRBS	160	129	90	154	142	147	<u>137</u>
	LDFBW	74	173	150	252	281	181	185
	LDFWB	259	233	241	407	640	338	**353**
	LDRBW	122	113	90	162	124	125	123
	LDRWB	132	107	117	152	134	113	126
SFF	DBDSW	118	140	130	147	145	144	137
	DWDSB	86	69	89	96	87	88	86
	DSBST	46	42	47	51	57	89	55
	DSWRS	118	57	75	64	66	53	72
	LWIBS	152	137	125	193	183	165	159
	LBIWR	181	203	520	269	257	-26	234
	LWRWR	114	105	86	160	117	108	115
	LWRBS	107	115	100	180	156	149	134
	LDFBW	187	161	131	269	190	194	189
	LDFWB	173	288	-579	286	301	215	114
	LDRBW	111	114	86	146	132	114	117
	LDRWB	155	148	115	166	159	262	<u>168</u>
ELA	DBDSW	649	261	142	709	207	250	**370**
	DWDSB	287	136	-59	259	147	111	147
	DSBST	34	33	129	46	58	49	58
	DSWRS	62	66	52	259	70	63	95
	LWIBS	204	155	152	121	220	199	175
	LBIWR	201	589	313	32	165	238	256
	LWRWR	109	97	83	146	136	108	113
	LWRBS	121	138	113	165	146	145	138
	LDFBW	98	172	169	279	290	203	202
	LDFWB	312	664	362	-188	557	576	**381**
	LDRBW	135	120	95	153	145	128	130
	LDRWB	221	114	106	147	211	154	<u>159</u>
	AVG	173	162	116	168	166	153	156

Table 4. Results for the proposed stagnation method ELA averaged across all multi-population-based algorithms and stagnation strategies for all benchmark functions. The results show the improvement in results (in percent) relative to the same methods without the use of stagnation detection and prevention mechanisms. SP stands for simulation problem, BASE stands for averaged normalized results, and allows comparing which topology is best (lower values stand for better results). The best results are in bold.

SP	ISBM	LDTC	TTRC	IRBM	RWAC	SMWD	AVG
C01	184	220	30	-20	26	-6	72
C02	87	27	34	-20	102	70	50
C03	103	111	72	182	163	171	133
C04	217	188	39	439	176	158	203
C05	179	180	226	235	220	195	206
C06	321	361	328	-108	505	545	**325**
C07	44	93	49	58	87	82	69
C08	526	566	348	650	323	273	**448**
C09	164	163	119	180	163	180	162
C10	205	202	35	218	108	79	141
C11	123	104	120	102	169	124	124
C12	198	227	201	40	309	366	224
C13	136	128	46	226	122	123	130
C14	379	347	309	407	292	218	**325**
C15	236	280	206	39	343	354	243
C16	119	157	41	131	97	78	104
C17	346	366	246	370	264	190	297
C18	172	184	172	108	233	283	192
C19	188	167	43	236	120	105	143
C20	236	230	214	266	212	158	219
C21	217	231	213	38	336	345	230
C22	120	150	41	191	109	96	118
C23	274	353	302	72	306	291	266
C24	280	213	125	148	156	241	194
C25	152	155	41	343	52	82	138
C26	114	151	40	148	105	74	105
C27	211	222	188	78	290	238	204
C28	145	165	42	199	118	95	127
AVG	**203**	**212**	138	177	197	186	185
BASE	**0.03**	0.71	0.41	**0.00**	1.00	0.93	

Table 5. Results for the proposed stagnation method ELA averaged across all multi-population-based algorithms and stagnation strategies for the control problems. The results show the improvement in results (in percent) relative to the same methods without the use of stagnation detection and prevention mechanisms. SP stands for simulation problem. The best results are in bold.

SP	ISBM	LDTC	TTRC	IRBM	RWAC	SMWD	AVG
MSD	5	4	1	0	0	2	2
WTT	3	2	2	-2	4	3	2

The considered methods were tested using the CEC2013 benchmark functions (improved CEC2005 benchmark set as used in [42]), hereinafter referred to as C01-C28 [39]. In addition, problems of selecting the parameters of control systems based on PID elements: MSD, WTT (see details in [25]) were also considered.

4.1 Simulation results

For a clearer presentation of the results and to show the capabilities of the mechanisms used, all results are shown as a percentage improvement over the case where no stagnation detection and prevention mechanisms were used. For example, a value of 100% means an improvement in error value from 1.0 to 0.5, a value of 200% from 1.0 to 0.25, and a value of -100% a deterioration from 1.0 to 2.0 (all problems concerned the minimization of the value of the evaluation function).

Results detailing all simulation cases are shown in Table 3, results detailing simulation problems for the proposed ELA stagnation detection method are shown in Table 4, results detailing various multi-population-based algorithms are shown in Table 6, and general summary of the individual methods is presented in Table 7. In addition, Figure 3 presents the impact of the α, β, γ, and $Nimp$ values on the results obtained for the ELA method and SFF method respectively. Results for control problems are shown in Table 5.

4.2 Simulation conclusions

The best results in terms of fitness function values were achieved for the proposed stagnation detection strategy (ELA) and the LDFWB and DBDSW stagnation prevention mechanisms, while the proposed LDFWB mechanism also performed very well in the standard approach (STA) - see bold values in Table 3. For variants with less computational complexity, the best mechanism is LDRWB, but in this case, the stagnation detection approach based on the lack of improvement of the evaluation function (SFF) worked slightly better (see LDRWB on SFF and ELA in Table 3. For each migration topology a satisfactory improvement over the lack of stagnation detection mechanisms was achieved, the only exception is the TTRC topology - see Table 3.

According to the no-free lunch theory, different degrees of improvement were achieved for each simulation problem using the ELA mechanism and the stagnation prevention mechanisms. For some configurations, there was no improvement in the results (see negative values in Table 4). The ELA mechanism improved the performance of the optimization for each of the topologies considered, with the best topology (IRBM) improving by 177%, and the next topology (ISBM) improving by 203%, thus giving the best results overall - see Table 4. The ELA approach also improved the results for control problems (see Table 5). In the case of non-synthetic tests, the improvement was significantly smaller, as expected, but for the PID control problems, even a few percent improvement means a significant improvement for the system (depending on the selected evaluation function). Details of the tested control systems can be found, e.g. in [25].

The best base results were achieved for the MNIA algorithm, and the proposed mechanisms allowed for an additional improvement of 152%. The greatest improvement of results by the stagnation detection and prevention mechanisms was achieved for the MPGWO algorithm - 217% improvement compared to the base approach - see Table 6. Nevertheless, the use of the MNIA algorithm with different population-based algorithms as islands gives very satisfactory results and at the same time eliminates the need to choose a specific population-based algorithm - thus it is a more universal solution. Stagnation detection improved most of the simulation variants, with the worst improvement being achieved for the MPCS algorithm, with an average improvement of 107% - see Table 6.

Regardless of the approach, algorithm, and strategy, the proposed stagnation detection criterion achieved the best results (see ELA in Table 7). In the case of the best-accuracy strategy (IA), LDFWB worked best (see LDFWB in Table 7), while the complexity-oriented strategy (RC) also achieved an improvement in accuracy - in this case, the LDRWB mechanism turned out to be the best (see LDRWBin Table 7).

The influence of the parameters α, β, and γ of the proposed ELA approach and $Nimp$ of the SFF approach may be of key importance for various simulation problems and is a topic worth addressing in future research - see Figure 3. Research on other

Table 6. Results averaged across all benchmark functions and migration topologies. The results show the improvement in results (in percent) relative to the same methods without the use of stagnation detection and prevention mechanisms. SM stands for stagnation mechanisms, SS stands for stagnation strategy, BASE stands for averaged normalized results, and allows comparing which topology is best (lower values stand for better results). The best results for the accuracy approach are in bold, and for the complexity reduction approach are underlined.

SM	**SS**	**MNIA**	**MPGA**	**MPDE**	**MPCS**	**MPGWO**	**AVG**
STA	DBDSW	364	6	83	107	407	193
	DWDSB	101	25	76	50	180	86
	DSBST	88	-69	62	26	122	46
	DSWRS	111	-36	79	20	145	64
	LWIBS	139	357	179	66	207	190
	LBIWR	53	337	256	119	257	204
	LWRWR	87	150	178	37	126	116
	LWRBS	115	142	205	60	163	<u>137</u>
	LDFBW	177	346	128	86	188	185
	LDFWB	464	283	112	195	712	**353**
	LDRBW	101	167	152	48	145	123
	LDRWB	95	153	166	72	143	126
SFF	DBDSW	142	29	109	184	222	137
	DWDSB	117	30	70	76	136	86
	DSBST	89	-44	75	30	126	55
	DSWRS	116	-22	98	23	146	72
	LWIBS	84	337	111	134	130	159
	LBIWR	125	376	324	217	127	234
	LWRWR	99	143	145	56	133	115
	LWRBS	97	145	203	109	119	134
	LDFBW	92	360	127	221	143	189
	LDFWB	82	241	24	112	110	114
	LDRBW	85	174	139	79	109	117
	LDRWB	96	148	317	137	140	<u>168</u>
ELA	DBDSW	773	28	86	151	811	**370**
	DWDSB	157	24	75	76	403	147
	DSBST	82	-71	73	33	173	58
	DSWRS	194	-42	79	22	224	95
	LWIBS	144	268	126	120	217	175
	LBIWR	9	412	263	549	48	256
	LWRWR	79	148	164	55	119	113
	LWRBS	116	146	165	98	165	138
	LDFBW	203	215	191	184	216	202
	LDFWB	374	461	277	88	704	**381**
	LDRBW	109	170	143	80	145	130
	LDRWB	96	150	292	121	136	<u>159</u>
	AVG	152	158	149	107	217	156
	BASE	0.00	1.00	0.47	0.89	0.18	

Table 7. Summary of tested mechanisms averaged for all benchmark functions. The results show the improvement in results (in percent) relative to the same methods without the use of stagnation detection and prevention mechanisms. SM stands for stagnation mechanisms, SS stands for stagnation strategy. The best results are in bold.

SM/SS	ISBM	LDTC	TTRC	IRBM	RWAC	SMWD	AVG
STA	189	143	132	158	147	143	152
SFF	129	132	77	169	154	130	132
ELA	203	212	138	177	196	185	**185**
DBDSW	305	191	139	366	202	198	233
DWDSB	196	108	50	169	27	89	106
LWIBS	180	146	139	178	201	203	175
LBIWR	385	377	371	63	109	83	232
LDFBW	119	169	150	267	254	193	192
LDFWB	248	395	8	169	500	376	**283**
DSBST	37	39	75	49	56	62	53
DSWRS	77	57	63	133	65	67	77
LWRWR	111	99	88	149	125	115	115
LWRBS	129	127	101	166	148	147	136
LDRBW	123	116	90	154	134	123	123
LDRWB	170	123	113	155	168	176	**151**

stagnation detection mechanisms, in particular hybrid mechanisms, is also worth considering and is not excluded in the future.

5 Conclusions

Mechanisms of stagnation detection and prevention allow for achieving better results in the case of multi-population-based algorithms in most cases. In particular, the best results in averaging the tested topologies, algorithms, simulation problems, and stagnation prevention mechanisms were achieved for the proposed stagnation detection approach (ELA). This approach is based on the analysis of the local improvement of the evaluation function and the infinite impulse response filter. The considered mechanisms for the appropriate management of populations and their sizes also contributed to the improvement of results and additionally enabled the simultaneous reduction of computational complexity. A significant improvement in the results was also achieved for the MNIA algorithm, which uses different baseline population algorithms within the islands. The achieved results point to the possibility of developing multi-population-based algorithms without the use of new ideologies, which is criticized, and at the same time provide a further field for the development of research in which other similar solutions can be used.

Acknowledgment

We gratefully acknowledge the funding support by the project ARTIQ - AI Centers of Excellence ARTIQ/0004/2021.

References

[1] Abed-alguni, B. H. (2019). Island-based cuckoo search with highly disruptive polynomial mutation. International Journal of Artificial Intelligence, 17(1), 57-82.

[2] Adam, S. P., Alexandropoulos, S. A. N., Pardalos, P. M., & Vrahatis, M. N. (2019). No free lunch theorem: A review. Approximation and optimization: Algorithms, complexity and applications, 57-82.

[3] Al-Andoli, M. N., Tan, S. C., & Cheah, W. P. (2022). Distributed parallel deep learning with a hybrid backpropagation-particle swarm optimization for community detection in large complex networks. Information Sciences, 600, 94-117.

[4] Atashpaz-Gargari, E., & Lucas, C. (2007, September). Imperialist competitive algorithm: an algo-

rithm for optimization inspired by imperialistic competition. In 2007 IEEE congress on evolutionary computation (pp. 4661-4667). Ieee.

[5] Brindha, S. (2021). A robust and adaptive fuzzy logic based differential evolution algorithm using population diversity tuning for multi-objective optimization. Engineering Applications of Artificial Intelligence, 102, 104240.

[6] Boiani, M., Parpinelli, R. S., & Dorn, M. (2022, November). A Multi-population Schema Designed for Biased Random-Key Genetic Algorithms on Continuous Optimisation Problems. In Brazilian Conference on Intelligent Systems (pp. 444-457). Cham: Springer International Publishing.

[7] Campelo, F., & Aranha, C. (2021, November). Sharks, zombies and volleyball: Lessons from the evolutionary computation bestiary. In LIFELIKE Computing Systems Workshop 2021. CEUR-WS. org.

[8] Chen, H., Heidari, A. A., Chen, H., Wang, M., Pan, Z., & Gandomi, A. H. (2020). Multi-population differential evolution-assisted Harris hawks optimization: Framework and case studies. Future Generation Computer Systems, 111, 175-198.

[9] Chen, P., Zhou, S., Zhang, Q., & Kasabov, N. (2022). A meta-inspired termite queen algorithm for global optimization and engineering design problems. Engineering Applications of Artificial Intelligence, 111, 104805.

[10] Cpalka, K., Łapa, K., & Przybył, A. (2018). Genetic programming algorithm for designing of control systems. Information Technology and Control, 47(4), 668-683.

[11] Cpałka, K., Słowik, A., & Łapa, K. (2022). A population-based algorithm with the selection of evaluation precision and size of the population. Applied Soft Computing, 115, 108154.

[12] Cpałka, K., Łapa, K., & Rutkowski, L. (2022, June). A multi-population-based algorithm with different ways of subpopulations cooperation. In International Conference on Artificial Intelligence and Soft Computing (pp. 205-218). Cham: Springer International Publishing.

[13] Das, S. R., Mishra, D., & Rout, M. (2019). A hybridized ELM using self-adaptive multi-population-based Jaya algorithm for currency exchange prediction: an empirical assessment. Neural Computing and Applications, 31(11), 7071-7094.

[14] Dziwiński, P., Przybył, A., Trippner, P., Paszkowski, J., & Hayashi, Y. (2021). hardware implementation of a Takagi-Sugeno neuro-fuzzy system optimized by a population algorithm. Journal of Artificial Intelligence and Soft Computing Research, 11(3), 243-266.

[15] Ezugwu, A. E., Shukla, A. K., Nath, R., Akinyelu, A. A., Agushaka, J. O., Chiroma, H., & Muhuri, P. K. (2021). Metaheuristics: a comprehensive overview and classification along with bibliometric analysis. Artificial Intelligence Review, 54, 4237-4316.

[16] Huang, C., Li, Y., & Yao, X. (2019). A survey of automatic parameter tuning methods for metaheuristics. IEEE transactions on evolutionary computation, 24(2), 201-216.

[17] Ishibuchi, H., Mihara, S., & Nojima, Y. (2012). Parallel distributed hybrid fuzzy GBML models with rule set migration and training data rotation. IEEE Transactions on fuzzy systems, 21(2), 355-368.

[18] Jia, F., Luo, S., Yin, G., & Ye, Y. (2023). A novel variant of the salp swarm algorithm for engineering optimization. Journal of Artificial Intelligence and Soft Computing Research, 13.

[19] Karaboga, D., & Aslan, S. (2015, November). A new emigrant creation strategy for parallel artificial bee colony algorithm. In 2015 9th International Conference on Electrical and Electronics Engineering (ELECO) (pp. 689-694). IEEE.

[20] Kassaymeh, S., Al-Laham, M., Al-Betar, M. A., Alweshah, M., Abdullah, S., & Makhadmeh, S. N. (2022). Backpropagation Neural Network optimization and software defect estimation modelling using a hybrid Salp Swarm optimizer-based Simulated Annealing Algorithm. Knowledge-Based Systems, 244, 108511.

[21] Kazikova, A., Pluhacek, M., Senkerik, R., & Viktorin, A. (2019). Proposal of a new swarm optimization method inspired in bison behavior. In Recent Advances in Soft Computing: Proceedings of 23rd International Conference on Soft Computing (MENDEL 2017) Held in Brno, Czech Republic, June 20-22, 2017 (pp. 146-156). Springer International Publishing.

[22] Li, S., Gu, Q., Gong, W., & Ning, B. (2020). An enhanced adaptive differential evolution algorithm for parameter extraction of photovoltaic models. Energy Conversion and Management, 205, 112443.

[23] Liu, Q., Li, X., Liu, H., & Guo, Z. (2020). Multi-objective metaheuristics for discrete optimization problems: A review of the state-of-the-art. Applied Soft Computing, 93, 106382.

[24] Long, W., Cai, S., Jiao, J., Xu, M., & Wu, T. (2020). A new hybrid algorithm based on grey wolf optimizer and cuckoo search for parameter extraction of solar photovoltaic models. Energy Conversion and Management, 203, 112243.

[25] Łapa, K., & Cpałka, K. (2017). Flexible fuzzy PID controller (FFPIDC) and a nature-inspired method for its construction. IEEE Transactions on Industrial Informatics, 14(3), 1078-1088.

[26] Łapa, K. (2019). Meta-optimization of multi-objective population-based algorithms using multi-objective performance metrics. Information Sciences, 489, 193-204.

[27] Łapa, K., Cpałka, K., & Słowik, A. (2021, June). Population Management Approaches in the OPn Algorithm. In International Conference on Artificial Intelligence and Soft Computing (pp. 402-414). Cham: Springer International Publishing.

[28] Łapa, K., Cpałka, K., Kisiel-Dorohinicki, M., Paszkowski, J., Dębski, M., & Le, V. H. (2022). Multi-population-based algorithm with an exchange of training plans based on population evaluation. Journal of Artificial Intelligence and Soft Computing Research, 12(4), 239-253.

[29] Łapa, K. 2023, June. Multi-population-based Algorithms with Different Migration Topologies and Their Improvement by Population Re-initialization. In International Conference on Artificial Intelligence and Soft Computing, 399-414

[30] Mansour, I. B., Basseur, M., & Saubion, F. (2018). A multi-population algorithm for multi-objective knapsack problem. Applied Soft Computing, 70, 814-825.

[31] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.

[32] Misaghi, M., & Yaghoobi, M. (2019). Improved invasive weed optimization algorithm (IWO) based on chaos theory for optimal design of PID controller. Journal of Computational Design and Engineering, 6(3), 284-295.

[33] Niu, B., & Li, L. (2008). A novel PSO-DE-based hybrid algorithm for global optimization. In Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence: 4th International Conference on Intelligent Computing, ICIC 2008 Shanghai, China, September 15-18, 2008 Proceedings 4 (pp. 156-163). Springer Berlin Heidelberg.

[34] Osaba, E., Del Ser, J., Cotta, C., & Moscato, P. (2022). Memetic computing: Accelerating optimization heuristics with problem-dependent local search methods. Swarm and Evolutionary Computation, 70, 101047.

[35] Saha, A. K. (2022). Multi-population-based adaptive sine cosine algorithm with modified mutualism strategy for global optimization. Knowledge-Based Systems, 251, 109326.

[36] Sayoti, F., & Essaid Riffi, M. (2016). Golden ball algorithm for solving flow shop scheduling problem.

[37] Sedighizadeh, D., Masehian, E., Sedighizadeh, M., & Akbaripour, H. (2021). GEPSO: A new generalized particle swarm optimization algorithm. Mathematics and Computers in Simulation, 179, 194-212.

[38] Stegherr, H., Heider, M., & Hähner, J. (2022). Classifying Metaheuristics: Towards a unified multi-level classification system. Natural Computing, 21(2), 155-171.

[39] Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y. P., Auger, A., & Tiwari, S. (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL report, 2005005(2005), 2005.

[40] Słowik, A., Cpałka, K., & Łapa, K. (2019). Multipopulation nature-inspired algorithm (MNIA) for the designing of interpretable fuzzy systems. IEEE Transactions on Fuzzy Systems, 28(6), 1125-1139.

[41] Talpur, N., Abdulkadir, S. J., Alhussian, H., Hasan, M. H., Aziz, N., & Bamhdi, A. (2023). Deep Neuro-Fuzzy System application trends, challenges, and future perspectives: A systematic survey. Artificial intelligence review, 56(2), 865-913.

[42] Tanabe, R., & Fukunaga, A. (2013, June). Evaluating the performance of SHADE on CEC 2013 benchmark problems. In 2013 IEEE Congress on evolutionary computation (pp. 1952-1959). IEEE.

[43] Tzanetos, A., & Dounias, G. (2021). Nature inspired optimization algorithms or simply variations of metaheuristics?. Artificial Intelligence Review, 54, 1841-1862.

[44] Voglis, C., Parsopoulos, K. E., Papageorgiou, D. G., Lagaris, I. E., & Vrahatis, M. N. (2012). MEMPSODE: A global optimization software based on hybridization of population-based algorithms and local searches. Computer Physics Communications, 183(5), 1139-1154.

[45] Wu, G., Mallipeddi, R., & Suganthan, P. N. (2019). Ensemble strategies for population-based optimization algorithms–A survey. Swarm and evolutionary computation, 44, 695-711.

[46] Xu, B., Tao, L., Chen, X., & Cheng, W. (2019). Adaptive differential evolution with multi-population-based mutation operators for constrained optimization. Soft Computing, 23, 3423-3447.

[47] Zalasiński, M., Cpałka, K., & Łapa, K. (2020, July). An interpretable fuzzy system in the on-line signature scalable verification. In 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (pp. 1-9). IEEE.

[48] Zalasiński, M., Laskowski, Ł., Niksa-Rynkiewicz, T., Cpałka, K., Byrski, A., Przybyszewski, K., ... & Dong, S. (2022). Evolutionary algorithm for selecting dynamic signatures partitioning approach. Journal of Artificial Intelligence and Soft Computing Research, 12(4).

[49] Zhao, X., Fang, Y., Ma, S., & Liu, Z. (2022). Multi-swarm improved moth–flame optimization algorithm with chaotic grouping and Gaussian mutation for solving engineering optimization problems. Expert Systems with Applications, 204, 117562.

Krystian Łapa received the M.Sc. and Ph.D. degrees from the Częstochowa University of Technology, Częstochowa, Poland, in 2010 and 2015, respectively. He received D.Sc. degree in Computer Science from Częstochowa University of Technology, Częstochowa, Poland, in 2019. He is currently an Associate Professor with the Department of Computer Engineering. Dr. Łapa has authored over 50 publications. His current research interests include computational intelligence, fuzzy systems, nature-inspired methods, and expert systems.
https://orcid.org/0000-0002-3926-5685

Danuta Rutkowska is a professor of computer science. She graduated from Wrocław University of Science and Technology, Wrocław, Poland, from where she also received her Ph.D. in automation and then D.Sc. in computer science. In 2002 she was given the title of professor conferred by the President of Poland. She is an author or co-author of numerous publications, mostly in computational intelligence, including several books, book chapters, and many scientific papers. Her research interests are in the area of artificial intelligence, especially computational intelligence, focusing on artificial neural networks, fuzzy systems, genetic/evolutionary algorithms, as well as hybrid intelligent systems, i.e. neuro-fuzzy or genetic-neuro-fuzzy systems, and their applications. After working for many years at the Częstochowa University of Technology, Częstochowa, Poland, she is currently a professor in the Institute of Information Technology at the University of Social Sciences in Łódź, Poland. Since 2022 she has been serving as vice-rector for science at this university.
https://orcid.org/0000-0003-0217-2589

Aleksander Byrski obtained Ph.D. in 2007 and D.Sc. in 2013 at the Department of Computer Science of the AGH University of Science and Technology in Krakow, Poland. His main research interests are metaheuristics, agentbased systems, high performance computing and simulation. He works as a Full Professor in the Institute of Computer Science at the AGH University of Science and Technology.
https://orcid.org/0000-0001-6317-7012

Christian Napoli is Associate Professor with the Department of Computer, Control, and Management Engineering "Antonio Ruberti", Sapienza University of Rome, since 2019, where he also collaborates with the department of Physics and the Faculty of Medicine and Psychology, as well as holding the office of Scientific Director of the International School of Advanced and Applied Computing (ISAAC).

He received the B.Sc. degree in Physics from the Department of Physics and Astronomy, University of Catania, in 2010, where he also got the M.Sc. degree in Astrophysics in 2012 and the Ph.D. in Computer Science in 2016 at the Department of Mathematics and Computer Science, he obtained the National Scientific Abilitation as associate professor in Computer Engineering (2017) and computer science (2019).

Christian Napoli has been Research Associate with the Department of Mathematics and Computer Science, University of Catania, from 2018 to 2019, while, previously, Research Fellow and Adjunct Professor with the same department from 2015 to 2018. He has been a Student Research Fellow with the Department of Electrical, Electronics, and Informatics Engineering, University of Catania, from 2009 to 2016, a collaborator of the Astrophysical Observatory of Catania and the National Institute for Nuclear Physics, since 2010.

He has been several time Invited Professor at the Silesian University of Technology, Visiting Academic at the New York University, and responsible of many different institutional topics from 2011 until now for Undegraduate, Graduate and PhD students in Computer Science, Computer Engineering and Electronics Engineering. His teaching activity focused on Artificial Intelligence, Neural Networks, Machine Learning, Computing Systems, Computer Architectures, Distributed Systems, and High Performance Computing. He is involved in several international research projects, serves as reviewer and member of the board program committee for major international journals and international conferences. His current research interests include neural networks, artificial intelligence, human-computer interation and computational neuropsychology.
https://orcid.org/0000-0002-3336-5853