Enhancing a Hierarchical Evolutionary Strategy Using the Nearest-Better Clustering *

Hubert Guzowski[1][0000−0002−9678−0384], Maciej Smołka[1][0000−0002−3386−0555], and Libor Pekař[2][0000−0002−2401−5886]

[1] AGH University of Krakow, Kraków, Poland
{guzowski,smolka}@agh.edu.pl
https://www.informatyka.agh.edu.pl/
[2] Tomas Bata University in Zlín, Zlín, Czech Republic
pekar@utb.cz
https://fai.utb.cz/

Abstract. A straightforward way of solving global optimization problems is to find all local optima of the objective function. Therefore, the ability of detecting multiple local optima is a key feature of a practically usable global optimization method. One of such methods is a multi-population evolutionary strategy called the Hierarchic Memetic Strategy (HMS). Although HMS has already proven its global optimization capabilities there is an area for improvement. In this paper we show such an enhancement resulting from the application of the Nearest-Better Clustering. Results of experiments consisting both of curated benchmarks and a real-world inverse problem show that on average the performance is indeed improved compared to the baseline HMS and remains on par with state-of-the-art evolutionary global optimization methods.

Keywords: evolutionary algorithm · global optimization · continuous domain· Nearest-Better Clustering

1 Introduction

Real-world engineering applications often require solving a global optimization problem. A vast part of them features various kinds of multimodality. Hill climber optimization methods are naturally ill-suited to this type of task, but even single-population stochastic methods having the theoretical asymptotic guarantee of success tend to locate only one solution in practically available time. To tackle this challenge, various niching techniques are used which, by reducing information exchange or competition between groups of individuals, isolate so-called species aimed at exploiting different optima of the objective function. Examples

* The research presented in this paper was partially supported by the Polish National Science Center under grant No. 2020/39/I/ST7/02285, by the funds of the Polish Ministry of Science and Education assigned to the AGH University of Krakow, by The Czech Science Foundation under grant No. GAČR 21-45465L, and the internal grant No. RVO/CEBIA/2021/001 by TBU in Zlín.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

include various augmentations of Particle Swarm Optimization and Differential Evolution algorithms [26,24,16]. The Hierarchic Memetic Strategy (HMS) [23] is another optimization algorithm aimed at the multimodal optimization. It achieves the isolation between groups of individuals by running multiple optimization processes in parallel. In order to prevent several populations from converging to the same optima, they are organized in a tree structure, where higher-level processes decide whether and where to start lower-level processes. This operation is called sprouting.

HMS already proved its capabilities in finding multiple optima of the objective for both benchmarks and real-world inverse problems [25,22]. The method was also equipped with special mechanisms addressing the problem of optima located in the flat regions of the objective landscape [21]. These are not used in this paper. Here we present a technique based on the Nearest Better Clustering (NBC) aimed at improving the HMS sprouting. NBC is a well-established niching technique used both in multimodal optimization, dynamic optimization and even in Exploratory Landscape Analysis (ELA) [15]. We conjecture that by combining the unique structure of HMS with the ability of NBC to locate funnel structures in the target function landscape, we obtain a highly effective multimodal optimization method. The extensive tests we present later in this article are intended to test the new mechanism against several criteria:

- Does the application of NBC lead to better location of several target function optima?
- Does the application of NBC improve performance in terms of the best quality solution found?
- Are the parameters of the new mechanism better at adapting to the target problem?

2 Background

In the field of optimization, it has become customary to talk about exploration and exploitation features of various algorithms. The exploration is tied to discovering promising areas of the objective function domain where we expect to find an optimum. These areas are called peaks, funnels or basins of attraction, depending on the nature of problem solved or algorithm used. The exploitation refers to locating the solution as accurately as possible in an already defined area. For a single population of individuals that exchange knowledge between each other, either tendency for exploration or exploitation will at some point outweigh the other [23]. In the case of exploitation, convergence occurs and in the case of exploration we arrive at a method similar to the pure random search. Niche methods partition the population so that its parts running in parallel can converge to detected optima without depriving the algorithm of its ability to explore. This behavior is particularly beneficial in multimodal optimization problems when we can distinguish multiple equally good global optima, but it also has benefits for standard single-solution optimization problems [11]. To date,

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

a number of niche techniques have been presented that differ in the way the division is achieved. One of these is a clustering based on the distance between individuals, exemplified by the NBC method [15]. The main observation behind the NBC is that the distance between an individual and its nearest neighbor with a better fitness value will be greater for the best individual in a given basin of attraction than for its surrounding individuals. Thus, a spanning tree where edges connect nearest better individuals with the best solution overall at its root is constructed. Then longer edges are removed creating subtrees representing species meant to occupy different niches.

The same observation about the inability to preserve exploration and exploitation for optimization algorithms using a single population inspired the creation of HMS. However, its answer is not the same as of typical niching methods. Instead of separating the population operating under a single algorithm into subspecies, HMS combines multiple populations operating under different algorithms into a hierarchical relationship [23] which operation is described in Algorithm 1. At the base of the tree hierarchy is the root algorithm, which can be a global optimization engine of user's choosing, but which has to be geared predominantly towards exploration. As a result, it would not be able to find the best solution with a decent accuracy in practically achievable time, but instead is responsible for detecting candidate basins of attraction. When it finds one, it launches a new process focused on exploiting the local optimum. In general this process is operating locally or semi-locally, thus its initial population is sampled from a Gaussian distribution as opposed to a uniform distribution over the whole search space which is used for the root process. Originally, the same evolutionary algorithm with different parameters was used at all levels, but the introduction of Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) as the algorithm responsible for exploitation has significantly improved performance [22]. HMS run consists of a loop executing a fixed number of iterations for each of its subprocesses. The subprocesses can be deactivated based on their specific stopping criterion e.g. no improvement over several epochs while the global stopping condition is checked for termination of the whole algorithm. An HMS user has control over the number of tree levels, which algorithms will be run on them, what will be their parameters and stopping conditions. This flexibility makes HMS adaptable to specific real-world tasks. One example would be the usage of multiwinner voting operators to promote a diversity of solutions for lower level algorithms, which made it possible to identify shapes and sizes of function niches [5]. To facilitate the use of HMS for non-expert users, an autoconfiguration mechanism was introduced based on objective function features calculated using ELA methods [7]. Although the HMS already proved to be a versatile and efficient global optimizer, there are still clear areas for improvement. In particular, the mechanism responsible for identifying the pools of attraction used to date could be a source of some practical problems. Its operation is very simple: consider the best current solution from every higher-level subprocess and if it is far enough from any centroid of lower-level population, start a new process at its location. Although intuitive and functional, this mechanism has clear disadvan-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

Algorithm 1: High-level pseudocode of HMS

Input: Objective function f, *dimensions* and *bounds* of f, $hmsTreeConfig$
Output: $foundOptima$ of f

```
root ← initPopulation(hmsTreeConfig[0], ∅)
activePopulations ← {root}
while global stop condition is not satisfied do
    foreach p in activePopulations do
        isActive ← runMetaepoch(p)
        if not isActive then
            activePopulations.remove(p)
            if p.engine = SEA then
                p.best ← runLocalMethod(p.best)
    runSprout(activePopulations, hmsTreeConfig)
foundOptima ← []
foreach p in all populations do
    foundOptima.add(p.best)
```

tages. Firstly, the value of the *far enough* distance parameter is highly problem dependent. This does not only mean that it should be scaled proportionally when working with a larger domain. Actually, the ideal value of the parameter should be exactly the radius of the basin of attraction so as not to allow more processes to run in it, but also not to cover the other basins. In practice, working with black box problems, we will certainly not set the exact value of this parameter correctly. So we can end up with a situation where the value is too high which makes it difficult to cover optima that are close to each other, or too low which results in a redundant running multiple processes in the same niche. In addition, the check approximates the radius of attraction pools to be perfectly spherical, which may not be an appropriate shape. Secondly, the best individual of a higher-level population can stagnate in the same peak for multiple iterations. This leads to underutilization of the information contained in the population of individuals. Even if solutions of similar quality to the best individual are found, they will remain ignored.

3 HMS with NBC component

In the previous sections, we have already mentioned the idea and basics of the HMS algorithm. Apart from the new sprout mechanism, the core of the algorithm remains the same. Currently, we considered the Simple Evolutionary Algorithm (SEA) and CMA-ES as candidates for algorithm components. Additionally, if SEA is used as a lower-level method, then on its deactivation we execute an additional local optimizer. In particular, we use L-BFGS-B provided by Python SciPy library, but it can be replaced according to user's preference. The implementation used during experiments described in this paper can be found in [8], whereas the continuously developed Python implementation of the HMS is avail-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

able on GitHub [3]. The core functionality is based on the LEAP library [4] while CMA-ES implementation is sourced from pycma package [10].

Our main contribution in this paper is the NBC-based sprout mechanism, which is presented in detail in Algorithm 2. Viewed from a high level, it consists of performing NBC clustering for each active population and then filtering resultant individuals based on quality and distance to active subprocesses at the lower level of the tree. Those distances are determined based on centroids calculated at the start of the process (line 2). At the start of the procedure we apply a simple elite selection mechanism introduced in line [14] which takes only the best fraction of the population given by $truncFactor$ (line 4). Afterward a standard Nearest Better Clustering is performed for each subprocess population separately. This consists of construction of a tree where individuals are connected to their nearest better neighbor and then cutting of edges that are $nbcCut$ times longer than mean edge length. Root individuals of each subtree are tested if they are far enough from active populations centroids mentioned above. Afterward a singular best individual that passes the distance criterion is selected per each subpopulation (line 20). The last filtering mechanism is taking the number of best candidates that will not exceed a subprocess limit imposed for each level (line 21). Resulting individuals serve as starting locations for new subprocesses.

The new mechanism addresses shortcomings described at the end of Sec. 2. Firstly, thanks to the capabilities of NBC, each parent population can identify multiple basins of attraction per metaepoch. The main benefit of this feature is that during consecutive metaepochs the parent population will sprout new subprocesses in different basins of attraction even if the best individual would not change. Secondly, we transfer the responsibility for determining the radii of the basins of attraction from the user to the method. Instead the user can operate on factors that influence the mechanism on the higher level. $nbcCut$ and $farEnough$ listed in pseudocode 2 both influence the granularity of sprouting process. $truncFactor$ can shift the behavior more towards exploration or exploitation, but our experiments suggest that the 0.8 can be safely assumed as its default value. $LevelLimit$ remains more as an emergency hard cap as the NBC tends to organically limit itself. The new mechanism therefore provides the user with more parameters than the previous one. On the one hand, this may allow the algorithm to be better tuned to the problem at hand, but in practice it is easier to use parameter-free algorithms.

4 Experimental procedure

To answer the questions posed in the introduction, we prepared a series of comparisons. The first subsection below focuses on assessing the ability of the algorithm to locate multiple global optima. In the next one, the aim is to see what impact the newly applied mechanism has on the quality of the best solution found.

[3] https://github.com/agh-a2s/pyhms

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

Algorithm 2: Pseudocode of NBC based HMS sprout

Input: *activePopulations*, *hmsTreeConfig*
Parameters: *truncFactor*, *nbcCut*, *farEnough*, *levelLimit*

```
sproutCandidates ← []
centroids ← calculatePopulationsCentroids(activePopulations)
foreach p in activePopulations do
    tp ← takeBest(p, truncFactor)
    T ← createEmpyTree()
    T.addNode(best individual i from tp)
    distances ← []
    foreach c in tp\{i} do
        nbi ← findNearestBetterNeighbour(c, tp)
        T.addNode(c)
        T.addEdge(c, nbi)
        distances.add(distance(c, nbi))
    cut_dist ← nbcCut · mean(distances)
    foreach e in T.edges() do
        if e.distance > cut_dist then
            Cut off e
            c ← newly created root node
            if all distances(c, centroids) > farEnough · mean(distances) then
                sproutCandidates.add(c)
sproutCandidates ← TakeOnePerDeme(sproutCandidates)
sproutCandidates ← BestPerLevelUpToLimit(sproutCandidates, levelLimit)
foreach c in sproutCandidates do
    childPopulation ← initPopulation(treeConfig[level(c) + 1], c)
    activePopulations.add({childPopulation})
```

4.1 Global optima coverage

We compare HMS with base sprout against new NBC sprout mechanism on a multimodal testbed provided by the PyDDRBG multimodal optimization benchmark [1]. We use 10 default target functions with identifiers from 1 to 10 provided by the library in a static variant. All of those target functions are 10-dimensional with box constraints between -5.0 and 5.0 for every dimension. As a quality measure we provide how many of the function's global optima were located and what was the total evaluation budget of the algorithm. The decision whether a given optimum has been located is made by checking whether any of the best solutions found by each subprocess is at a distance no greater than the specified niche radius for the given optimum. Besides sprout mechanism both versions use the 2-levels HMS structure with root SEA and CMA-ES leaves with the same hyperparameters.

In the first part of this experiment, we investigated the effect of changing the values of the sprouting mechanism parameters on the number of global optima covered and sprouted subprocesses in general. For the baseline mechanism we had only 1 parameter to test. For NBC sprout, we can manipulate 3 pa-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

rameters: the subtree cut-off factor, the distance factor and the fraction of best individuals included. Each of these parameters can be used to limit the number of subprocesses sprouted and therefore, when testing the cut-off and distance factors, we set the second parameter to the liberal value of 1.0. When testing the *truncFactor* fraction value, we set the other two to 1.5. A run of each configuration was repeated 10 times.

In the second part, based on the previous results, we selected a single configuration for both sprouting mechanisms and ran such configured HMS 50 times for each target function to calculate the average Peak Ratio and the average function evaluation budget. For simple sprouting we set the value of the distance parameter as 4.0 while for NBC sprouting we set the cut-off value as 1.5, the distance factor as 1.0 and the fraction of solutions considered as 0.8.

4.2 Best solution quality

The tests we performed comparing the quality of the best found solution use the same procedures as the work done in the earlier article [7] and include the results achieved there for the base version of HMS. The comparison is performed on the Black Box Optimization Benchmark (BBOB) single objective continuous test suite [9]. To better compare sprout mechanisms, we keep the same two-level HMS architecture for the respective BBOB function classes consisting of SEA root with either CMA-ES or SEA leaves. We first run hyperparametric optimization for each of the five classes of 10-dimensional functions defined in BBOB by minimizing the sum of the quality of the solutions obtained after 9500 function evaluations for the first instances of every function in the class. As hyperparameter optimization tool we employed SMAC3 [12] and repeated a run 5 times for each class with a budget of 3000 evaluations in total. Details of the SMAC3 scenario and the ranges of hyperparameters considered in the optimization process can be found in linked repository [8]. Afterward we selected the best performing configuration out of the resulting 5 based on 50 runs per target function both in 10 and 20 dimensions with budget of 9500 and 19000 function evaluations respectively. We did not apply explicit parameter scaling when using the obtained configurations for 20-dimensional problems.

Our comparison focuses on two versions of HMS with base sprouting mechanisms and one based on NBC. In addition, we include the results of the BIPOP-CMA-ES [13] and iL-SHADE [3] which proved to be the most competitive and versatile algorithms in article [7]. In contrast to that work, we performed the same hyperparameter optimization and configuration selection procedure as for the HMS algorithm for a fair comparison. Although both of those algorithms are highly adaptive and do not require many parameters, they still gained in performance in the tuning process. We optimized population and archive size for iL-SHADE and population size, σ_0, population increment factor on restart and function value change tolerance for termination for BIPOP-CMA-ES. iL-SHADE

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

implementation is sourced from PyADE library [4] while BIPOP-CMA-ES implementation is provided in pycma [10].

4.3 Real-world inverse problem

In addition to comparisons to the previous article, we also decided to include in the comparison a real-world parameter identification problem formulated as a global optimization task. It features a significantly different domain size relative to the problems included in the BBOB benchmark. Here we include its concise description.

Estimating the parameters of time-delay system (TDS) models constitutes a challenging task [6,27]. Its complexity arises mainly from the existence of infinite number of the so-called system modes in the TDS dynamics; however, the model is characterized via only a small set of parameters. Nevertheless, only a few dominant modes have a decisive role [18]. Hence, the identification problem can be formulated as searching for a parameter set yielding model dominant modes with the minimum distance to the actual TDS dominant modes.

Processes with interconnected heating-cooling loops (HCPs) are representatives of TDSs as they inherently incorporate heat and mass transfer, which causes latencies [28]. A stable laboratory HCP was investigated, e.g., in [19,20]. Despite its physical simplicity, the laboratory HCP has a relatively complex dynamics due to delays in the feedback loops. We herein focus on the relation between the input voltage to the heater $P_H(t) = u(t)$ and the outlet fluid temperature of the radiator $\vartheta_{CO}(t) = y(t)$ expressed by the delay-differential equation [19]

$$y^{(3)}(t) + a_2\ddot{y}(t) + a_1\dot{y}(t)a_0y(t) + a_{0D}y(t-\theta) = b_0u(t-\tau) + b_{0D}u(t-\tau-\tau_0) \quad (1)$$

where b_0, b_{0D}, a_2, a_1, a_0, and a_{0D} are non-delay model parameters, while τ_0, τ, and θ are positive delays. The necessary stability conditions read

$$a_2 > 0, a_1 > 0, a_0 + a_{0D} > 0 \quad (2)$$

The equivalent model transfer function to (1) reads

$$G_m(s) = \frac{Y(s)}{U(s)} = \frac{b_0 + b_{0D}e^{-\tau_0 s}}{s^3 + a_2s^2 + a_1s + a_0 + a_{0D}e^{-\vartheta s}}e^{-\tau s} \quad (3)$$

where Y and U are the Laplace transforms of y and u.

Among the large number of methods and techniques to determine unknown model parameters, a specific family of identification approaches utilizing a nonlinear element in the feedback loop exists. A subset of these approaches enables to estimate parameter values of (3) in the frequency domain [2,17]. Namely, one can obtain a set of experimentally obtained values $\widehat{G}_m(\imath\omega_j + a)$ for some suitable discrete values of $\omega_j, a \geq 0$, $j = 1, 2, ..., N$, where $\imath$ is the imaginary unit satisfying $\imath^2 = -1$. Hence, the parameter optimization task related to model (3)

[4] https://github.com/xKuZz/pyade

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

and with respect to (2) can be formulated as searching the closest distance of estimated $\widehat{G}_m(.)$ to the modeled ones $G_m(.)$ for selected ω_j, a that are decisive for dominant modes behavior. Then, the optimization problem can be expressed as follows

$$\begin{aligned} \mathbf{p}^* &= \arg\min f(\mathbf{p}) \\ f(\mathbf{p}) &= \sum_{j=1}^{N} \left| \widehat{G}_m(\imath\omega_j + a) - G_m(\imath\omega_j + a) \right|^2 \\ \mathbf{p} &= [b_{0D}, \tau_0, \tau, a_2, a_1, a_0, a_{0D}, \theta] \\ \text{such that} &: [\tau_0, \tau, \theta, a_2, a_1, a_0 + a_{0D}] > 0 \end{aligned} \tag{4}$$

It is worth noting that problem (4) is multimodal and constrained. It i.a. means that some results with even an excellent cost function value do not reflect physical reality. Therefore, it is necessary to remain close to the values obtained by physical modeling and analysis [19,20]. Using ELA we discovered that this problem is of the same type as the BBOB class 5 problems, i.e. multimodal with the weak global structure. For this reason, we chose the same algorithm configurations for this problem as for the class 5 function. We also set a budget of 9500 evaluations which is consistent with previous tests. The actual computational problem domain is specified by box constraints of $[0, 500]$, $[0, 500]$, $[0, 1000]$, $[0, 2500]$, $[0, 2500]$, $[-250, 250]$, $[-250, 250]$, $[0, 1000]$ for respective components of $\mathbf{p}$. As its size is much larger than in the benchmark case, we scaled the distance parameters of baseline sprouting, mutation standard deviation and σ_0 linearly. In the case of NBC sprout, we did not need to apply scaling. To tackle additional domain constraints, we apply penalization during optimization process and at the end consider only valid individuals for the comparison.

5 Results

This section presents the results of the experiments carried out in subsections which correspond to their descriptions in Section 4.

5.1 Global optima coverage

The results of tests examining the effect of changing parameter values on the coverage of function optima show a clear improvement after applying the new NBC-based sprout mechanism. Because of limited space available, we include the resultant graphs for 3 out of 10 target functions under examination. However, the same general trend can be observed for every target function besides the instance 9, for which both variants perform poorly. The complete results are included in Zenodo archive [8]. As can be seen in Figure 1 for the base sprout mechanism, when reducing the distance parameter below some border value the number of subprocesses increases fast, but the raise does not correspond to a similar increase in number of covered optima. Meanwhile, for the NBC sprout, we observe an even more drastic increase in number of created subprocesses with

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

a value decrease between 2.0 and 1.5. However, this time it translates into substantially higher optima coverage. It is important to note, that the recommended NBC cut value of 2.0 produces quite limited number of sprouted subprocesses and also a lower optima coverage. This may be related to the nature of the PyDDRBG [1] benchmark, which contains problems with global optima clustered inside a single peak or some yet to be discovered peculiarity in our application of NBC. Although the use of the new mechanism brings clear benefits in the number of optima covered, the way in which the amount increases rapidly in response to small changes in the value of the parameter is not desirable. Our preliminary tests on other test beds suggest, that this problem is not as drastic in the case of more easily distinguishable peaks, but it remains as an area to be addressed in the future works.

The next comparison shows the average number of covered peaks and budget expended during that process for both sprout variants. Although both mechanisms can be easily tuned to be highly exploratory or to expend the least amount of evaluations, we had to choose configurations that maximize both of those criteria. This context is relevant to the interpretation of the results in Table 1. They show higher optima coverage for 8 out of 10 functions for the new sprout mechanism. We can look more closely at the instance 6, which is the only one with worse coverage achieved with the new mechanism. As we can see on Figure 1, the steep jump in coverage for this particular instance occurs for relatively low value of parameters, which were also lower than the ones we chose for this experiment. However, if we look at the results from the perspective of coverage efficiency, the picture shown is even more favorable as the gain in exploration capabilities of the algorithm does not come at the cost of sacrificing performance in terms of evaluation budget used. On average around 1526 evaluations were required for each optimum covered in case of the base sprout while the NBC sprout required 1081 per optimum. This is a clear qualitative improvement in the performance of the algorithm under these criteria.

Table 1. Comparing the efficiency of finding global optima between HMS with base and NBC sprout with set parameters on PyDDRBG benchmark. Values are averaged over 50 runs.

Instance		HMS base sprout		HMS NBC sprout	
Id	N optima	Optima covered	Evaluation budget	Optima covered	Evaluation budget
1	4	2.32	3091.2	2.34	2797.2
2	2	1.88	4018.4	1.96	3715.8
3	3	2.04	3515.0	2.88	5161.6
4	3	1.62	3023.4	1.86	3170.8
5	4	2.14	3700.2	3.82	4976.8
6	16	3.34	3003.0	2.74	2704.8
7	8	3.5	3785.2	4.82	4442.6
8	9	2.48	3351.6	6.38	5367.6
9	18	0.42	2944.2	0.42	3099.6
10	16	2.42	3385.2	10.6	5430.6

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

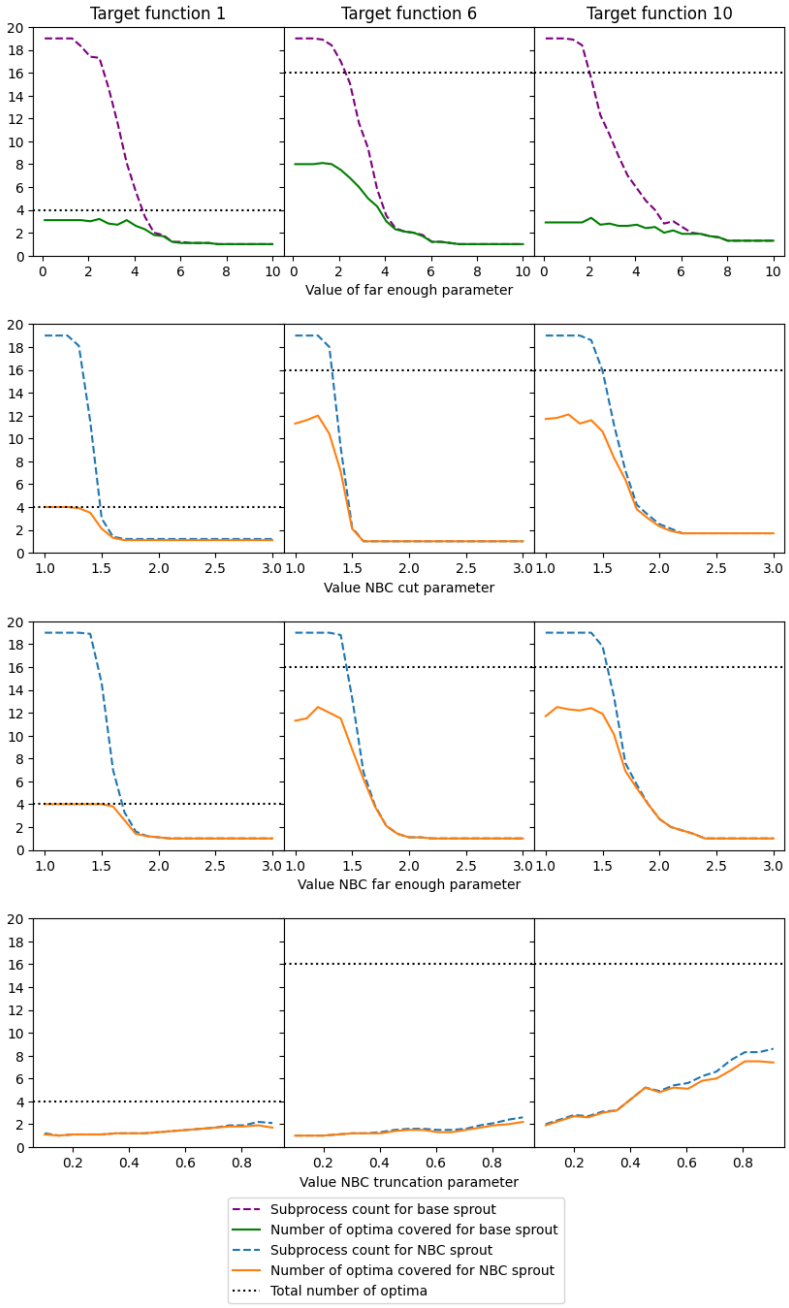

Fig. 1. Coverage difference for DDRB instance 5, 8 and 10. The first row corresponds to base sprout mechanism while rows 2-4 represent the change of one of the parameters in new NBC-based sprout mechanism. Results are averaged over 10 runs per value of a specific parameter.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

5.2 Best solution quality

Comparison of solution quality is included in Table 2. In 27 out of 48 cases the new variant of HMS with NBC sprout outperforms the previous version of the algorithm, while in the other 16 it performs worse. As the hyperparameter tuning produced different configurations with different performances on the same functions, some fluctuations in solution quality are to be expected. However, the number of improved results is substantially higher and also in 8 cases the improvement is more than tenfold. Compared to the original sprout mechanism the gains are most visible for classes of functions that are less multimodal (f1-14). We attribute this results to the improved filtering of unnecessary new subprocesses introduced in the new sprout mechanism. At the same time, when looking at the bigger picture of HMS in comparison to other algorithms, its purpose remains as the method to tackle highly multimodal problems (f20-24). Although the new mechanism certainly patched the performance on some of the previously problematic instances, making the algorithm more universal in usage (f2,f6).

5.3 Real-world optimization problem

Results included in Table 3 show improved results for the new version of HMS compared to the base one. Given that the real-world problem is similar in landscape to class 5 BBOB functions for which the new version have not performed better, these results hint at the better adaptability of the new mechanism. In the wider context of the HMS algorithm, we can see that the configurations trained for BBOB problems transferred very well to a real-world problem and remain competitive for BIPOP-CMA-ES and iL-SHADE. This is a very good sign for the use of default HMS configurations and the overall quality of the algorithm.

6 Conclusion

In this paper we have introduced and tested a new Nearest Better Clustering based component for the Hierarchic Memetic Strategy. The new mechanism improves the algorithm's exploration capabilities by making a better use of the information contained in higher-order populations. At the same time, we have shown improvement in terms of the best solution found for a wide range of optimization problems included in the BBOB benchmark and for a real-world problem where HMS performed on a par with state-of-the-art global optimizers. The parameters of the new mechanism are less problem-dependent making the determination of universal default values simpler. However, the increase in the number of parameters and mechanisms complexity may make the new mechanism more difficult to use from the perspective of an unfamiliar user. Thus, in future works we plan to combine the new functionalities with configuration auto-selector proposed in [7]. Another promising area for improvement is the parameter adaptation during the runtime, which seems to be a natural direction of development for HMS, yet still remains unexplored.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

Table 2. Comparison results between HMS versions and algorithms in portfolio on BBOB benchmark. Each column contains 50 runs average value of fitness (adjusted by value of global optima) and its standard deviation in parentheses.

	HMS	**HMS with NBC**	**BIPOP-CMA-ES**	**iL-SHADE**
	10 dimensions			
f1	9.4e-15 (7.8e-15)	**2.3e-15 (5.2e-15)**	8.8e-13 (8.6e-13)	3.1e-15 (5.9e-15)
f2	7.7e-01 (1.2e+00)	**1.1e-14 (2.3e-14)**	8.1e-13 (6.2e-13)	8.1e-12 (7.2e-11)
f3	8.4e+00 (3.1e+00)	1.3e+01 (5.0e+00)	8.6e+00 (3.9e+00)	**8.3e+00 (3.7e+00)**
f4	1.4e+01 (4.7e+00)	1.8e+01 (7.4e+00)	**1.1e+01 (3.6e+00)**	**1.1e+01 (3.9e+00)**
f5	6.8e-15 (2.4e-14)	**2.8e-16 (2.8e-15)**	6.7e-13 (4.0e-13)	2.8e-15 (1.2e-14)
f6	7.2e-01 (3.7e+00)	4.4e-03 (3.0e-02)	**2.0e-13 (1.7e-13)**	1.7e-05 (4.9e-05)
f7	1.1e+00 (6.5e-01)	1.3e+01 (5.4e+00)	**1.3e-02 (1.0e-01)**	7.8e-02 (2.4e-01)
f8	1.2e-08 (1.1e-08)	**7.5e-09 (7.8e-09)**	8.0e-02 (5.6e-01)	3.1e+00 (1.5e+00)
f9	3.7e-08 (3.7e-08)	**2.3e-08 (2.5e-08)**	1.9e-05 (1.9e-04)	4.4e+00 (1.2e+00)
f10	1.0e+00 (6.8e+00)	**2.2e-03 (2.2e-02)**	8.7e-01 (3.9e+00)	7.0e-02 (1.9e-01)
f11	1.5e+00 (5.9e+00)	2.3e-01 (6.6e-01)	5.4e-01 (1.7e+00)	**1.2e-02 (8.3e-02)**
f12	1.8e+00 (2.6e+00)	**3.0e-01 (1.1e+00)**	4.2e-01 (1.4e+00)	7.8e-01 (1.1e+00)
f13	7.8e-03 (2.9e-02)	1.2e-05 (1.2e-04)	**2.6e-08 (1.4e-07)**	8.7e-03 (1.6e-02)
f14	**1.1e-11 (9.4e-12)**	1.2e-11 (8.9e-12)	3.5e-10 (2.0e-10)	2.2e-06 (3.1e-06)
f15	1.0e+01 (5.2e+00)	9.5e+00 (4.5e+00)	**3.6e+00 (3.9e+00)**	1.3e+01 (5.5e+00)
f16	3.0e-01 (3.7e-01)	3.2e-01 (3.5e-01)	**7.4e-02 (1.5e-01)**	8.7e-01 (9.6e-01)
f17	9.5e-02 (1.7e-01)	1.4e-02 (4.0e-02)	**2.6e-05 (2.4e-05)**	7.2e-03 (3.6e-02)
f18	6.4e-01 (1.1e+00)	1.8e-01 (3.1e-01	**2.9e-03 (2.8e-02)**	1.4e-01 (3.4e-01)
f19	2.1e+00 (1.4e+00)	2.4e+00 (1.6e+00)	**1.2e+00 (7.3e-01)**	1.8e+00 (4.6e-01)
f20	**1.2e+00 (2.3e-01)**	**1.2e+00 (2.4e-01)**	1.4e+00 (3.3e-01)	1.3e+00 (2.8e-01)
f21	1.4e+00 (1.4e+00)	**1.2e+00 (1.6e+00)**	3.5e+00 (4.4e+00)	4.8e+00 (7.1e+00)
f22	**1.3e+00 (8.6e-01)**	1.4e+00 (1.2e+00)	5.4e+00 (1.1e+01)	2.8e+00 (1.7e+00)
f23	**1.0e+00 (4.4e-01)**	1.2e+00 (4.1e-01)	1.3e+00 (6.3e-01)	1.5e+00 (3.6e-01)
f24	2.0e+01 (5.8e+00)	2.4e+01 (7.4e+00)	**1.9e+01 (8.5e+00)**	2.6e+01 (8.4e+00)
	20 dimensions			
f1	1.6e-14 (4.6e-15)	**1.4e-14 (3.1e-15)**	9.4e-13 (4.0e-13)	4.3e-08 (3.9e-07)
f2	7.4e+01 (3.4e+01)	**9.3e-14 (4.0e-13)**	1.0e-12 (5.1e-13)	3.2e-02 (2.5e-01)
f3	2.7e+01 (7.0e+00)	2.9e+01 (8.0e+00)	**1.9e+01 (6.1e+00)**	3.2e+01 (1.0e+01)
f4	3.9e+01 (1.0e+01)	4.3e+01 (1.2e+01)	**2.2e+01 (4.1e+00)**	4.2e+01 (1.3e+01)
f5	**8.5e-14 (0.0e+00)**	**8.5e-14 (0.0e+00)**	1.0e-12 (4.1e-13)	9.7e-14 (2.8e-14)
f6	7.8e+02 (5.4e+06)	2.5e-03 (6.9e-03)	**4.7e-13 (3.1e-13)**	2.1e-02 (6.5e-02)
f7	8.8e+00 (1.1e+01)	7.8e+01 (2.3e+01)	**1.3e+00 (1.0e+00)**	2.9e+00 (2.4e+00)
f8	**2.5e-08 (2.4e-16)**	**2.5e-08 (1.8e-08)**	3.4e-01 (1.1e+00)	1.5e+01 (1.1e+01)
f9	8.4e-08 (3.2e-15)	**8.1e-08 (5.3e-08)**	1.3e-01 (6.8e-01)	1.6e+01 (1.3e+00)
f10	7.2e+00 (1.8e+01)	**2.6e+00 (1.0e+01)**	1.2e+01 (2.6e+01)	1.9e+02 (1.9e+02)
f11	2.3e+00 (7.9e+00)	**4.6e-01 (2.1e+00)**	2.7e+00 (9.1e+00)	3.0e+00 (2.5e+00)
f12	2.2e-01 (8.6e-01)	**4.8e-02 (1.6e-01)**	1.5e-01 (1.2e+00)	1.0e+00 (2.5e+00)
f13	2.5e-01 (6.7e-01)	6.1e-02 (2.1e-01)	**3.5e-02 (1.2e-01)**	1.6e+00 (1.7e+00)
f14	3.9e-09 (3.1e-09)	**1.1e-10 (1.0e-10)**	2.9e-09 (2.1e-09)	3.2e-04 (1.9e-04)
f15	2.5e+01 (7.3e+00)	**2.4e+01 (6.8e+00)**	8.6e+00 (3.0e+00)	6.1e+01 (1.4e+01)
f16	**5.1e-01 (4.0e-01)**	1.3e+00 (3.8e+00)	7.4e-01 (3.7e+00)	3.7e+00 (2.8e+00)
f17	3.4e-02 (5.8e-02)	2.5e-02 (6.6e-02)	**6.1e-05 (4.3e-04)**	3.7e-01 (3.7e-01)
f18	4.6e-01 (7.2e-01)	1.9e-01 (1.7e-01)	**1.6e-02 (6.7e-02)**	1.7e+00 (1.7e+00)
f19	4.2e+00 (1.6e+00)	4.1e+00 (2.0e+00)	**2.9e+00 (1.1e+00)**	3.2e+00 (5.6e-01)
f20	**1.5e+00 (1.5e-01)**	**1.5e+00 (2.1e-01)**	1.6e+00 (2.2e-01)	1.7e+00 (2.5e-01)
f21	**1.9e+00 (2.2e+00)**	2.0e+00 (2.5e+00)	6.1e+00 (6.3e+00)	5.7e+00 (8.1e-01)
f22	3.3e+00 (4.4e+00)	3.4e+00 (6.3e+00)	2.1e+01 (2.1e+01)	**1.3e-02 (1.8e-02)**
f23	**1.5e+00 (5.0e-01)**	1.6e+00 (5.0e-01)	2.3e+00 (6.9e-01)	2.0e+00 (3.8e-01)
f24	6.9e+01 (1.3e+01)	7.5e+01 (1.7e+01)	**5.6e+01 (3.2e+01)**	7.5e+01 (1.5e+01)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

Table 3. Comparison results between HMS versions and algorithms in portfolio on real-world problem. Each column contains 50 runs average value of fitness and its standard deviation in brackets.

HMS	HMS with NBC	BIPOP-CMA-ES	iL-SHADE
6.5e-04 (3.0e-04)	5.6e-04 (3.6e-04)	4.0e-03 (8.7e-19)	3.5e-04 (2.1e-04)

References

1. Ahrari, A., et al.: PyDDRBG: A Python framework for benchmarking and evaluating static and dynamic multimodal optimization methods. SoftwareX **17**, 100961 (2022). https://doi.org/https://doi.org/10.1016/j.softx.2021.100961
2. Bi, Q.A., Wang, Q.G., Hang, C.C.: Relay-based estimation of multiple points on process frequency response. Automatica **33**, 1753–1757 (1997)
3. Brest, J., Maučec, M.S., Bošković, B.: iL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization. In: 2016 IEEE Congress on Evolutionary Computation (CEC). pp. 1188–1195. IEEE (2016). https://doi.org/10.1109/CEC.2016.7743922
4. Coletti, M.A., Scott, E.O., Bassett, J.K.: Library for evolutionary algorithms in Python (LEAP). In: GECCO '20 Companion: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion. pp. 1571–1579. ACM (2020). https://doi.org/10.1145/3377929.3398147
5. Faliszewski, P., et al.: Multiwinner voting in genetic algorithms. IEEE Intelligent Systems **32**(1), 40–48 (2017). https://doi.org/10.1109/MIS.2017.5
6. Gupta, S., Gupta, R., Padhee, S.: Parametric system identification and robust controller design for liquid–liquid heat exchanger system. IET Control Theory & Applications **12**, 1474–1482 (2018). https://doi.org/10.1049/iet-cta.2017.1128
7. Guzowski, H., Smołka, M.: Configuring a hierarchical evolutionary strategy using exploratory landscape analysis. In: GECCO '23 Companion: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. pp. 1785–1792. ACM (2023). https://doi.org/10.1145/3583133.3596403
8. Guzowski, H., Smołka, M., Pekař, L.: Experimental software used for Enhancing a Hierarchical Evolutionary Strategy Using the Nearest-Better Clustering article for ICCS 2024 conference. Zenodo (2024). https://doi.org/10.5281/zenodo.10730541
9. Hansen, N., et al.: Real-Parameter Black-Box Optimization Benchmarking 2009: Experimental Setup. Research Report RR-6828, INRIA (2009), https://hal.inria.fr/inria-00362649
10. Hansen, N., et al.: CMA-ES/pycma. Zenodo (2023). https://doi.org/10.5281/zenodo.2559634
11. Li, X., et al.: Seeking multiple solutions: An updated survey on niching methods and their applications. IEEE Transactions on Evolutionary Computation **21**(4), 518–538 (2017). https://doi.org/10.1109/TEVC.2016.2638437
12. Lindauer, M., et al.: SMAC3: A versatile Bayesian optimization package for hyperparameter optimization. Journal of Machine Learning Research **23**(54), 1–9 (2022), http://jmlr.org/papers/v23/21-0888.html
13. Loshchilov, I., Schoenauer, M., Sèbag, M.: Bi-population CMA-ES agorithms with surrogate models and line searches. In: Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation. pp. 1177–1184. GECCO '13 Companion, ACM (2013). https://doi.org/10.1145/2464576.2482696

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43

14. Luo, W., et al.: Identifying species for Particle Swarm Optimization under dynamic environments. In: 2018 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 1921–1928 (2018). https://doi.org/10.1109/SSCI.2018.8628900
15. Luo, W., et al.: A survey of Nearest-Better Clustering in swarm and evolutionary computation. In: 2021 IEEE Congress on Evolutionary Computation (CEC). pp. 1961–1967 (2021). https://doi.org/10.1109/CEC45853.2021.9505008
16. Luo, W., et al.: Hybridizing niching, Particle Swarm Optimization, and evolution strategy for multimodal optimization. IEEE Transactions on Cybernetics **52**(7), 6707–6720 (2022). https://doi.org/10.1109/TCYB.2020.3032995
17. Miguel-Escrig, O., et al.: Multiple frequency response points identification through single asymmetric relay feedback experiment. Automatica **147**, 110749 (2023). https://doi.org/10.1016/j.automatica.2022.110749
18. Özer, M.S., Íftar, A.: Eigenvalue optimisation-based centralised and decentralised stabilisation of time-delay systems. International Journal of Control **95**, 2245–2266 (2022). https://doi.org/10.1080/00207179.2021.1906446
19. Pekař, L.: Modeling and identification of a time-delay heat exchanger plant. In: Pekař, L. (ed.) Advanced Analytic and Control Techniques for Thermal Systems with Heat Exchangers. pp. 23–48. Academic Press (Elsevier) (2020). https://doi.org/10.1016/B978-0-12-819422-5.00002-5
20. Pekař, L., et al.: Further experimental results on modelling and algebraic control of a delayed looped heating-cooling process under uncertainties. Heliyon **9**, e18445 (2023). https://doi.org/https://doi.org/10.1016/j.heliyon.2023.e18445
21. Sawicki, J., et al.: Approximating landscape insensitivity regions in solving ill-conditioned inverse problems. Memetic Computing **10**(3), 279–289 (2018). https://doi.org/10.1007/s12293-018-0258-5
22. Sawicki, J., et al.: Using Covariance Matrix Adaptation Evolutionary Strategy to boost the search accuracy in hierarchic memetic computations. Journal of Computational Science **34**, 48–54 (2019). https://doi.org/https://doi.org/10.1016/j.jocs.2019.04.005
23. Sawicki, J., et al.: Understanding measure-driven algorithms solving irreversibly ill-conditioned problems. Natural Computing **21**, 289–315 (2022). https://doi.org/10.1007/s11047-020-09836-w
24. Sheng, W., et al.: Adaptive memetic differential evolution with niching competition and supporting archive strategies for multimodal optimization. Information Sciences **573**, 316–331 (2021). https://doi.org/10.1016/j.ins.2021.04.093
25. Smołka, M., et al.: An agent-oriented hierarchic strategy for solving inverse problems. International Journal of Applied Mathematics and Computer Science **25**(3), 483–498 (2015). https://doi.org/10.1515/amcs-2015-0036
26. Wang, R., et al.: Adaptive niching particle swarm optimization with local search for multimodal optimization. Applied Soft Computing **133**, 109923 (2023). https://doi.org/10.1016/j.asoc.2022.109923
27. Wurm, J., Bachler, S., Woittennek, F.: On delay partial differential and delay differential thermal models for variable pipe flow. International Journal of Heat and Mass Transfer **152**, 119403 (2022). https://doi.org/10.1016/j.ijheatmasstransfer.2020.119403
28. Zítek, P., Hlava, J.: Anisochronic internal model control of time-delay systems. Control Engineering Practice **9**(5), 501–516 (2001). https://doi.org/10.1016/S0967-0661(01)00013-2

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63759-9_43