Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2025) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

29th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2025)

Adaptive Metaheuristic Selection in Island-Based Optimization

Adam Żychowski[a,*], Xin Yao[b], Jacek Mańdziuk[a,c]

[a] *Warsaw University of Technology, Faculty of Mathematics and Information Science, Koszykowa 75, Warsaw, Poland*
[b] *School of Data Science, Lingnan University, 8 Castle Peak Road, Tuen Mun, Hong Kong SAR*
[c] *Faculty of Computer Science, AGH University of Krakow, Mickiewicza 30, Krakow, Poland*

Abstract

The optimization of complex problems remains a significant challenge across various domains of science and engineering. This paper introduces a novel approach to island-based optimization that dynamically adapts metaheuristic selection during runtime, extending the Diversity-driven Cooperating Portfolio of Metaheuristics (DdCPM) framework. Our method integrates additional metaheuristics beyond the original implementation and proposes adaptation strategies that dynamically reconfigure the algorithm portfolio based on performance indicators and population characteristics. Experimental results across both discrete and continuous optimization benchmarks demonstrate that adaptive metaheuristic selection enhances solution quality and convergence rates compared to static approaches. The proposed framework represents an advancement in hybrid optimization systems, offering improved performance through intelligent adaptation mechanisms that correspond to the evolving state of the search process.

Keywords: Island-based algorithm; Optimization metaheuristic; Distributed optimization;

1. Introduction

Optimization problems are a key part of many real-world challenges, such as optimizing energy consumption in smart grids [2], financial portfolio optimization [31], traffic flow management [14], vehicle routing [21, 22], scheduling [37], or designing efficient manufacturing processes [38]. These problems are particularly challenging due to high dimensionality, non-smooth objectives, and complex constraints. Traditional methods often struggle with these difficulties, leading to the development of population-based metaheuristics as powerful and flexible alternatives.

Population-based metaheuristics employ a collection of candidate solutions to search the problem space, sometimes using mechanisms inspired by natural processes. Over the decades, researchers have developed numerous metaheuristic algorithms [26], each with its own strengths and weaknesses. For example Genetic Algorithms (GA) [5] excel at global exploration but may struggle with fine-tuning the solutions. Particle Swarm Optimization (PSO) [16] offers fast

* Corresponding author.
E-mail address: adam.zychowski@pw.edu.pl

1877-0509

convergence but is susceptible to premature convergence in complex landscapes. Ant Colony Optimization (ACO) [7] demonstrates effectiveness in discrete optimization problems but can be computationally intensive. Differential Evolution (DE) [32] provides robust performance for continuous optimization but may require careful parameter tuning. Simulated Annealing (SA) algorithms [36] offer a good balance between exploration and exploitation but may converge slower than other methods.

The complementary nature of these algorithms has motivated research into hybrid approaches that leverage multiple metaheuristics simultaneously. Island-based models represent a prominent paradigm for such hybridization. In island-based evolutionary algorithms [39], the population is divided into subpopulations (islands), each evolving independently with periodic migration of individuals between islands. This structure maintains diversity across the population and allows subpopulations to specialize in different regions of the search space.

Recent advancements in island-based optimization have focused on heterogeneous models where different islands employ distinct metaheuristics. The Diversity-driven Cooperating Portfolio of Metaheuristics (DdCPM) [41] represents a contribution in this domain by incorporating an adaptive diversity-driven migration mechanism that manages the information flow between islands running different metaheuristics. DdCPM demonstrates superior performance compared to both single-algorithm implementations and homogeneous island-based variants across diverse optimization problems.

Motivation. While heterogeneous island-based models like DdCPM show promising results, they typically employ a static assignment of metaheuristics to islands throughout the optimization process. This fixed assignment fails to account for the dynamic nature of optimization problems, where different algorithms may perform better at different stages of the search or in different regions of the solution space. The effectiveness of metaheuristics can vary significantly as the search progresses: some algorithms excel at exploration during early stages, while others are more effective at exploitation during later stages.

The performance of island-based models is highly dependent on the migration strategy employed. While DdCPM addressed fundamental questions regarding *when, what,* and *where* to migrate, it did not explore the question of *how* to adapt the metaheuristic portfolio itself. Given that different problems and even different stages of solving the same problem may benefit from different metaheuristic configurations, there is a clear need for mechanisms that can dynamically adjust the metaheuristic portfolio based on performance feedback.

These limitations motivate our research into adaptive metaheuristic selection for island-based optimization. By developing methods to dynamically decide which metaheuristic operates on which island during the algorithm's execution, we aim to create a more responsive and more effective optimization framework that can adapt to the specific characteristics of the problem being solved and to the changing landscape as the search progresses.

Contribution. The key contributions of the paper are as follows:
- **Dynamic metaheuristic adaptation**. We propose novel strategies for adaptive metaheuristic selection that enable the algorithm to dynamically reconfigure the assignment of metaheuristics to islands during runtime. We present 3 strategies for selecting islands for metaheuristic replacement and 4 strategies for choosing a metaheuristic to replace the current one on a given island.
- **Extended metaheuristic portfolio**. We expand the DdCPM framework by incorporating additional metaheuristics, specifically Differential Evolution (DE) and Simulated Annealing (SA) algorithms, creating a more diverse portfolio of optimization methods. We conduct extensive experiments to evaluate different island configurations and metaheuristic combinations, identifying optimal setups.
- **Extensive empirical evaluation**. We validate our approach through comprehensive experiments on two distinct problem domains: the Black Box Optimization Benchmarking (BBOB) suite for continuous optimization and Traveling Salesman Problem (TSP) instances for discrete optimization.

2. Related Work

Island-based optimization algorithms have emerged as an effective approach for tackling complex optimization problems by maintaining multiple subpopulations (islands) that evolve independently and periodically exchange information through migration mechanisms. Early studies on island models primarily focused on genetic algorithms (GAs). Whitley et al. [39] analyzed the impact of separability, population size, and convergence properties in island model genetic algorithms, highlighting their potential to maintain diversity and improve convergence rates. Alba and Tomassini [3] provided a comprehensive overview of parallelism in evolutionary algorithms, emphasizing the advantages of island-based models in terms of scalability and performance enhancement. Harada and Alba [12] surveyed parallel genetic algorithms, discussing various migration strategies and their influence on algorithm effectiveness. Other studies have shown that carefully chosen migration conditions such as migration sizes [30], migration timing [43], or migrants selection [42] significantly influence the final results of island-based models.

Adaptive metaheuristic selection has increasingly gained attention as a promising direction to improve optimization outcomes dynamically. Akay et al. [1] empirically studied parallel population-based algorithm portfolios, showing that adaptive selection mechanisms significantly outperform static configurations by dynamically responding to changing problem characteristics. Similarly, distributed adaptive frameworks have been developed to effectively coordinate metaheuristics across multiple computational nodes. The Distributed Adaptive Metaheuristic Selection (DAMS) framework [6] employs a three-layer architecture allowing nodes to autonomously decide on local information exchange and algorithm application during optimization. The GAPSO [35] and M-GAPSO [25] methods utilize DE, different PSO variants, and local search techniques and propose the concept of information sharing between algorithms. Thanks to knowing each other positions and best-so-far solutions, the algorithms can coordinate their search directions and explore the search space more effectively. [15] proposed a new model based on the fitness cloud, leading to theoretical and empirical insights on when an online adaptive strategy can be beneficial. A constrained hybrid metaheuristic algorithm (cHM) has been proposed in [18] with application to Probabilistic Neural Networks learning.

The Diversity-driven Cooperating Portfolio of Metaheuristics (DdCPM), recently introduced by Żychowski et al. [41], advances heterogeneous island-based optimization frameworks by incorporating diversity-driven migration mechanisms that adaptively manage information flow between islands running distinct metaheuristics. Despite its demonstrated superiority over homogeneous approaches, DdCPM maintains static metaheuristic assignments throughout its execution.

This limitation motivates our research into adaptive metaheuristic selection strategies that dynamically modify metaheuristic-to-island assignment during runtime based on performance indicators and population characteristics. The work extends the existing literature by introducing adaptive metaheuristic selection specifically designed for island-based optimization.

3. Adaptive Metaheuristic Selection in Island-Based Optimization

In this section, we first summarize the baseline Diversity-driven Cooperating Portfolio of Metaheuristics (DdCPM) [41] framework, and then introduce the proposed adaptation strategies that dynamically modify the metaheuristic configuration during runtime.

3.1. Baseline Solution

The DdCPM framework is a hybrid island-based optimization approach that cooperatively combines multiple metaheuristic algorithms. The method considers a set of n_i islands, each running a distinct metaheuristic algorithm to evolve its subpopulation. These islands periodically exchange individuals through migration mechanisms, fostering collaboration between different optimization strategies.

Each island I within the set of islands has three primary attributes: $I_{population}$, representing the set of candidate solutions (individuals) on the island; $I_{metaheuristic}$, specifying the algorithm used to evolve the population; and $I_{neighbors}$, defining the set of neighboring islands from which migration can occur.

Migration plays a crucial role in the effectiveness of DdCPM, as it enables cooperation between islands running different metaheuristics. DdCPM addresses three fundamental aspects of migration: *when* to migrate, *what* to migrate, and *where* to migrate. The extensive empirical analysis demonstrated that the optimal configuration combines: (1) triggering migration when either fitness metrics stagnate or diversity fails to increase; (2) selecting individuals for migration based on a weighted combination of fitness and diversity contribution; and (3) implementing either a *ring* or *clique* topology for inter-island connections. This configuration demonstrated superior performance compared to single-algorithm implementations and homogeneous island-based variants across diverse optimization problems. We use DdCPM as the baseline framework for our modifications.

The baseline DdCPM implementation contains three widely recognized metaheuristic algorithms: Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO). In this paper we incorporate two additional metaheuristics into the portfolio - Differential Evolution (DE) and population-based version of Simulated Annealing (SA) [13, 4]. The selection of metaheuristics was based on their popularity — the five chosen metaheuristics were the most frequently cited as of December 31, 2022 [26].

3.2. Adaptation Methods

While the baseline DdCPM implementation demonstrates the effectiveness of cooperating metaheuristics in an island-based model, it employs a static assignment of algorithms to islands throughout the optimization process. Our adaptation strategies extend this framework by introducing dynamic reassignment of metaheuristics during runtime, corresponding to the evolving state of the search process. The adaptation process involves three key components: a) determining which island should have its metaheuristic replaced (island selection), b) deciding which metaheuristic should replace the current one (algorithm replacement), and c) establishing when this adaptation should occur.

a) Island Selection Strategies – We propose three strategies for selecting the island for metaheuristic replacement:

A. **Random Island Selection (Random)**: This strategy randomly selects an island for metaheuristic replacement, introducing stochasticity into the adaptation process. This approach prevents deterministic biases and provides a baseline for comparison with more sophisticated selection methods. Random selection allows for the potential discovery of beneficial algorithm-problem matchings that might not be apparent from the performance metrics alone.

B. **Weakest Performance Island (Weakest)**: This strategy identifies the island with the lowest average fitness of its top K individuals. By targeting the weakest-performing island for the algorithm reassignment, we aim to improve regions of the search space where the current metaheuristic is underperforming. Parameter K can be adjusted, with smaller values emphasizing elite performance and larger ones considering broader population quality.

C. **Similarity-Based Selection (Similar)**: This strategy identifies the island whose population is most similar to populations on other islands. For each individual on an island, we find the closest individual (by means of a solution encoding distance) among all other islands, then average these minimum distances to compute a similarity score. The island with the lowest score (highest similarity) becomes a candidate for a metaheuristic replacement. The intuition is that islands with highly similar populations to others might benefit most from metaheuristic diversification, as they may be exploring redundant regions of the search space.

b) Algorithm Replacement Strategies – Once an island is selected for adaptation, we implement one of four strategies to determine which metaheuristic should replace the current one:

1. **Random Algorithm Selection (Random)**: This strategy randomly selects a metaheuristic from the available portfolio (excluding the current one on the island). This approach provides a baseline comparison for more sophisticated selection methods and enables a wider exploration of algorithm-problem pairings.

2. **Best Individual Algorithm (Best)**: This strategy adopts the metaheuristic that is currently employed on the island which contains the overall best individual. The rationale is to replicate successful metaheuristics, i.e., leverage the metaheuristic that has demonstrated superior performance in finding high-quality solutions.

3. **Highest Average Fitness Algorithm (Avg Fitness)**: This strategy selects the metaheuristic from the island with the highest average population fitness. Unlike the previous strategy that focuses on elite performance, this approach considers the overall quality of solutions generated by different algorithms, potentially favoring metaheuristics that maintain a consistently high-performing population rather than those that find isolated exceptional solutions.

4. **Upper Confidence Bound Algorithm Selection (UCB)**: This strategy applies principles from the multi-armed bandit problem [20] in algorithm selection. Each metaheuristic is assigned a score based on its historical performance and the uncertainty associated with that performance estimate. The UCB [17] approach assigns a score to each available metaheuristic algorithm using the following formula:

$$\mathrm{UCB}_i(t) = \hat{\mu}_i(t) + C\sqrt{\frac{\ln(t)}{N_i(t)}} \tag{1}$$

where: $\mathrm{UCB}_i(t)$ is the upper confidence bound of the score for metaheuristic i at iteration t; $\hat{\mu}_i(t)$ is the estimated performance up to iteration t (the average fitness improvement of metaheuristic i); $N_i(t)$ is the number of times metaheuristic i has been selected up to iteration t; C is a parameter that controls the exploration-exploitation trade-off.

The first term, $\hat{\mu}_i(t)$, represents the exploitation component, favoring algorithms that have demonstrated strong performance in previous iterations. Specifically, $\hat{\mu}_i(t) = \frac{1}{N_i(t)} \sum_{s=1}^{N_i(t)} r_i(s)$ where $r_i(s)$ is the reward (fitness improvement) observed when metaheuristic i was applied in selection s. The second term, $C\sqrt{\frac{\ln(t)}{N_i(t)}}$, represents the exploration component. It is higher for metaheuristics that have been selected less frequently (smaller $N_i(t)$) and generally grows as the total number of iterations increases. Parameter C controls the balance between exploration and exploitation: higher values of C place more emphasis on exploration, while lower values focus more on exploitation. UCB is also the main selection mechanism in the popular Monte Carlo Tree Search algorithm [33].

During the adaptive process, when an island is selected for algorithm replacement, the UCB score (eq. 1) is calculated for each available metaheuristic (excluding the one currently used on the island). The metaheuristic with the highest UCB score is then selected to replace the current one. This approach ensures that the selection process exploits metaheuristics that have proven effective in previous iterations, explores potentially promising metaheuristics that have been underutilized, and gradually shifts focus toward better-performing metaheuristics as more information (higher confidence) becomes available.

c) Adaptation Timing - The timing of metaheuristic adaptation is crucial for balancing the stability of the search process and the responsiveness to changing search conditions. We implement adaptation either **every K_i iterations** (for the island chosen based on one of aforementioned selection strategies A-C) **or when stagnation is detected** on some island (replacement takes place on this island). Stagnation is defined as the absence of improvement in either the best or average fitness for K_s consecutive iterations. This approach ensures that adaptation occurs at regular intervals while also responding dynamically to the stagnation of the search process.

We incorporated the above adaptation strategies into the DdCPM model, creating a dynamic portfolio approach that adaptively reconfigures the metaheuristic landscape based on optimization performance. Combining multiple island selection strategies, algorithm replacement methods, and adaptation timing mechanisms, led to creation of a flexible system that can adjust to the specific characteristics of the problem being solved.

4. Experimental setup

We evaluated the proposed method on two well-established benchmark problem domains to ensure comprehensive testing across both discrete and continuous optimization challenges: Traveling Salesman Problem (TSP) and Black Box Optimization Benchmarking (BBOB) [9]. For TSP we randomly selected 10 instances from the TSPLIB library [28]. These instances consist of between 400 and 724 nodes (cities), specifically: rd400, fl417, pcb442, d493, att532, si535, u574, p654, d657, and u724. For BBOB we utilized all 24 problem instances from the COCO platform [11] dataset. The number of dimensions for all functions was set to 20.

While parameter optimization can significantly impact algorithm performance, it was not the primary focus of this study. We conducted only preliminary experiments to determine reasonable parameter settings, as our main goal was to investigate the general effectiveness of adaptive metaheuristic selection strategies in island-based optimization. Each island maintained a fixed population size of $n_I = 100$. Adaptive timing parameters are to $K_i = 20$ (fixed adaptation timing interval) and $K_s = 10$ (stagnation interval). For each metaheuristic, we used standard parameter settings as recommended in the literature, namely ACO [19]: pheromone evaporation rate of 0.1, initial pheromone value of 1.0; DE [34]: scaling factor $F = 0.5$, crossover rate $CR = 0.9$; GA [24]: crossover rate of 0.8, mutation rate of 0.01; PSO [8]: inertia weight = 0.729, cognitive and social parameters $c1 = c2 = 1.49445$; SA [10]: initial temperature of 100, cooling rate of 0.95.

All experiments were conducted independently 20 times using different random seeds. The results presented in the next section represent averages across all runs and all benchmark instances (24 for BBOB and 10 for TSP). Statistical significance was assessed using a Wilcoxon Signed-Rank test with a p-value threshold of ≤ 0.05. To ensure a fair comparison, all algorithms were allocated the same fitness function evaluation budget of 10^6 evaluations. Wall clock time was not reported, as the execution time differences between algorithm variants (adaptation strategies) and parameter settings were negligible.

We utilized standard implementations of the selected algorithms, previously proposed in the literature. Specifically, for continuous optimization, we employed ACO [19], DE [34], GA [24], PSO [8] and SA [10]. For the TSP, we used ACO [40], DE [23], GA [27], PSO [29] and SA [10]. Detailed implementation descriptions can be found in the cited papers or in our source code, which will be made publicly available upon paper acceptance.

Our primary objective was to demonstrate the potential of adaptive metaheuristic selection within the island model framework. We did not perform direct comparisons with specialized algorithms for specific problem domains, as the main assets of our approach aim to provide a general optimization framework applicable to various problem types.

5. Results

5.1. Static Metaheuristic Combinations

Table 1 presents the mean and standard deviation results for various island-based configurations of metaheuristics for BBOB and TSP benchmark datasets. For single-metaheuristic configurations, DE demonstrates the best performance for both BBOB and TSP benchmarks, while SA shows the weakest performance across both problem domains. Larger metaheuristic combinations consistently outperform configurations with fewer metaheuristics included. It suggests that the combination of diverse metaheuristic approaches creates a synergy that enhances overall performance.

Table 1. Mean and standard deviation results for island-based configurations of metaheuristics across BBOB and TSP benchmark datasets. Each metaheuristic from the island configuration runs on a separate island. The best results are **bolded**.

Islands configuration	**BBOB**	**TSP**	**Islands configuration**	**BBOB**	**TSP**
ACO	1.928 ± 0.063	35461 ± 59	ACO+DE+GA	1.678 ± 0.040	35100 ± 45
DE	1.677 ± 0.037	35239 ± 49	ACO+DE+PSO	1.668 ± 0.033	35016 ± 35
GA	1.788 ± 0.051	35830 ± 83	ACO+DE+SA	1.648 ± 0.023	34906 ± 35
PSO	1.842 ± 0.050	36006 ± 79	ACO+GA+PSO	1.788 ± 0.060	35468 ± 55
SA	1.988 ± 0.026	36193 ± 74	ACO+GA+SA	1.758 ± 0.053	35526 ± 55
ACO+DE	1.688 ± 0.038	34993 ± 39	ACO+PSO+SA	1.778 ± 0.050	35249 ± 45
ACO+GA	1.808 ± 0.059	35465 ± 60	DE+GA+PSO	1.655 ± 0.043	35176 ± 50
ACO+PSO	1.812 ± 0.055	35446 ± 55	DE+GA+SA	1.658 ± 0.035	35197 ± 50
ACO+SA	1.858 ± 0.045	35397 ± 55	DE+PSO+SA	1.645 ± 0.028	35120 ± 40
DE+GA	1.665 ± 0.048	35225 ± 50	GA+PSO+SA	1.748 ± 0.058	35760 ± 65
DE+PSO	1.672 ± 0.039	35198 ± 45	ACO+DE+GA+PSO	1.658 ± 0.030	35050 ± 35
DE+SA	1.668 ± 0.030	35096 ± 45	ACO+DE+GA+SA	1.638 ± 0.025	34960 ± 40
GA+PSO	1.768 ± 0.056	35728 ± 75	ACO+DE+PSO+SA	1.625 ± 0.015	34811 ± 30
GA+SA	1.775 ± 0.054	35646 ± 70	ACO+GA+PSO+SA	1.738 ± 0.055	35325 ± 50
PSO+SA	1.798 ± 0.050	35784 ± 65	DE+GA+PSO+SA	1.637 ± 0.023	35148 ± 45
			ACO+DE+GA+PSO+SA	$\mathbf{1.618} \pm 0.013$	$\mathbf{34778} \pm 25$

A comparative analysis of performance gains when a specific metaheuristic is added to an island-based setup is presented in Table 2. The results clearly indicate that DE contributes the most significant performance improvements across both problem domains, with a 9.50% gain for BBOB and 1.60% gain for TSP. This finding aligns with the strong individual performance of DE observed in Table 1 and reinforces its importance within heterogeneous island-based optimization frameworks. ACO, GA, and PSO offer moderate improvements, while SA contributes only slightly to performance improvement. The relatively lower contribution of SA might be attributed to its typically slower convergence rate, though it still provides useful diversity in the search process. These results highlight the varying degrees of impact different metaheuristics can have on the overall performance, confirming the potential benefits of dynamic algorithm allocation.

5.2. Adaptive Metaheuristic Selection Strategies

For the adaptive experiments, we utilized a set of 10 islands. As a baseline for evaluating our adaptive strategies, we first identified the optimal static configurations. To determine the best configuration, we conducted exhaustive

Table 2. A comparative analysis of performance gains (percent and absolute values) when a given metaheuristic is added to an island-based setup, compared to the same setup without it (e.g. ACO+GA+PSO vs. ACO+PSO for GA) across all possible metaheuristic combinations (cf. Table 1). These differences highlight each metaheuristic's effectiveness in enhancing overall performance. The best results are **bolded**.

Metaheuristic	BBOB		TSP	
	% gain	absolute gain	% gain	absolute gain
ACO	2.00	0.034	0.89	323.8
DE	**9.50**	**0.157**	**1.60**	**567.5**
GA	2.70	0.047	0.24	24.3
PSO	2.30	0.041	0.18	78.5
SA	0.80	0.015	0.05	113.1

testing by running all possible configurations and analyzing their performance. The number of possible combinations of $n = 5$ metaheuristics (ACO, DE, GA, PSO, SA) on $m = 10$ islands is $\binom{n+m-1}{m} = \binom{14}{10} = 1001$.

Table 3 presents the five highest-performing static configurations along with their results, providing a basis for comparison with our adaptive strategies. For BBOB problems, the optimal configuration consists of 2 ACO, 5 DE, 2 GA, 1 PSO, and 0 SA islands, achieving the result of 1.547. For TSP problems, the best configuration is 3 ACO, 4 DE, 2 GA, 1 PSO, and 0 SA, with the result of 34724.

Several patterns emerge from these optimal configurations. DE consistently represents the highest proportion of islands (4-6 out of 10), reflecting its strong individual performance and significant contribution to hybrid setups. ACO islands appear more beneficial for TSP problems (2-3 islands) than for BBOB problems (1-2 islands), likely due to ACO's established effectiveness in discrete optimization. SA is notably absent or minimally present (0-1 islands) in the top configurations, suggesting its limited effectiveness within this island-based framework.

Table 3. The best static combinations of tested 5 metaheuristics for 10 islands setup averaged over all BBOB and TSP instances, e.g. 2ACO+5DE+2GA+1PSO+0SA means 2 islands running ACO, 5 running DE, 2 GA, and 1 PSO. The best results are **bolded**.

	BBOB		TSP	
	Top 5 best combinations	**Result**	**Top 5 best combinations**	**Result**
1.	2ACO+5DE+2GA+1PSO+0SA	**1.547** ± 0.023	3ACO+4DE+2GA+1PSO+0SA	**34724** ± 17
2.	1ACO+6DE+2GA+1PSO+0SA	1.548 ± 0.029	2ACO+5DE+2GA+1PSO+0SA	34727 ± 23
3.	2ACO+4DE+2GA+1PSO+1SA	1.551 ± 0.021	3ACO+3DE+3GA+1PSO+0SA	34729 ± 19
4.	1ACO+5DE+2GA+1PSO+1SA	1.553 ± 0.019	2ACO+4DE+2GA+1PSO+1SA	34731 ± 22
5.	2ACO+5DE+1GA+2PSO+1SA	1.556 ± 0.022	2ACO+4DE+1GA+2PSO+1SA	34737 ± 20

Table 4 presents the results of our adaptive metaheuristic selection strategies for the same 10-island setup. In all cases, the initial configuration was 2 islands assigned for each of 5 metaheuristics. Then, due to adaptive strategies, this configuration was dynamically changed during algorithm runtime. Thus, Table 4 also shows the average number of islands running each metaheuristic over all iterations and algorithm runs.

The UCB + Weakest strategy achieves the best performance for both BBOB and TSP problems, outperforming even the best static configuration from Table 3. These results demonstrate that adaptive metaheuristic selection can indeed improve optimization outcomes compared to static assignments. The success of the UCB + Weakest strategy can be attributed to its effective balance between exploration and exploitation in metaheuristic selection. The UCB component ensures that algorithm selection exploits successful metaheuristics while still exploring potentially promising alternatives, while the Weakest island selection strategy targets underperforming regions of the search space for algorithm replacement. The dominance of the UCB as a replacement strategy and Weakest as a selection strategy is also visible in other configurations (with other selection/replacement strategies).

Examining the average island distribution for the best strategy (UCB + Weakest), we observe that it broadly aligns with the best static configurations identified in Table 3. DE is the most commonly chosen metaheuristics and SA is the least selected one. This shows that our adaptive approach successfully identifies effective metaheuristic allocations during runtime. The distributions resulting from the adaptation methods are more flattened compared to the optimal static configurations. This occurs because the algorithm initially assigns an equal distribution, with two islands allocated to each metaheuristic, requiring several iterations to adapt. Despite this, some adaptation strategies achieved superior final results compared to static configurations. This can be attributed to the varying importance of different

Table 4. Results for the proposed adaptive metaheuristic selection strategies in a 10-island setup for BBOB and TSP problems. The last five columns show the number of islands running a given metaheuristic, averaged over all iterations and 20 algorithm runs. The best results are **bolded**. Results that are statistically significantly better than the best static approach from Table 3 are shaded.

BBOB							
Replacement strategy	**Selection strategy**	**Result**	**ACO**	**DE**	**GA**	**PSO**	**SA**
Random	Random	1.604 ± 0.048	2.04	2.26	1.95	1.82	1.93
Random	Weakest	1.576 ± 0.044	1.82	3.64	2.03	1.58	0.93
Random	Similar	1.587 ± 0.041	1.91	3.49	2.11	1.47	1.02
Best	Random	1.592 ± 0.043	1.67	3.82	2.08	1.45	0.98
Best	Weakest	1.558 ± 0.038	1.54	4.25	2.01	1.32	0.88
Best	Similar	1.573 ± 0.037	1.61	3.93	2.07	1.49	0.90
Avg Fitness	Random	1.579 ± 0.040	1.73	3.78	2.04	1.38	1.07
Avg Fitness	Weakest	1.545 ± 0.046	1.49	4.31	1.98	1.36	0.87
Avg Fitness	Similar	1.567 ± 0.037	1.58	3.89	2.12	1.41	1.00
UCB	Random	1.563 ± 0.042	1.63	4.07	2.05	1.35	0.90
UCB	Weakest	**1.534** ± 0.035	1.66	4.43	1.95	1.09	0.87
UCB	Similar	1.539 ± 0.038	1.55	4.13	2.01	1.34	0.97

TSP							
Replacement strategy	**Selection strategy**	**Result**	**ACO**	**DE**	**GA**	**PSO**	**SA**
Random	Random	34761 ± 40	2.06	2.11	1.99	1.96	1.88
Random	Weakest	34752 ± 31	2.74	3.68	2.11	0.97	0.50
Random	Similar	34755 ± 35	2.95	3.25	2.35	0.85	0.60
Best	Random	34755 ± 32	3.00	3.41	2.06	1.00	0.52
Best	Weakest	34758 ± 23	2.84	3.47	2.21	0.94	0.43
Best	Similar	34753 ± 26	3.07	3.24	2.13	0.90	0.55
Avg Fitness	Random	34750 ± 33	3.15	3.63	1.93	0.83	0.66
Avg Fitness	Weakest	34748 ± 25	3.28	3.81	1.78	0.68	0.45
Avg Fitness	Similar	34745 ± 22	3.17	3.50	1.98	0.92	0.43
UCB	Random	34754 ± 29	2.95	3.70	1.94	0.89	0.52
UCB	Weakest	**34705** ± 23	3.35	4.03	1.70	0.45	0.47
UCB	Similar	34716 ± 21	3.11	3.70	2.01	0.67	0.51

metaheuristics at different phases of the algorithm's runtime, suggesting that a dynamic distribution may be more effective than maintaining a fixed optimal static configuration throughout the entire execution.

It is noteworthy that ACO receives significantly more islands in the TSP domain compared to BBOB, which aligns with ACO's known strength in discrete optimization problems. This demonstrates that the proposed adaptive approach effectively allocates computational resources to the most suitable metaheuristics for each problem domain despite the lack of prior domain knowledge.

Table 5 presents the frequency of switching metaheuristics for the best adaptation strategy (UCB + Weakest), providing valuable insights into the dynamic behavior of our adaptive approach. DE is the most frequently selected replacement algorithm which reinforces the finding that DE offers substantial benefits to the optimization process in continuous domains. The relatively low frequency of transitions to SA across both problem domains suggests that SA rarely emerges as the optimal choice during adaptation, which is consistent with its limited presence in the best static configurations. These switching patterns show that our adaptive approach dynamically selects effective metaheuristics, leveraging each algorithm's strengths for different problem domains.

Table 5. Frequency of switching metaheuristics for the best adaptation strategy: UCB + Weakest.

BBOB						**TSP**					
From\To	ACO	DE	GA	PSO	SA	From\To	ACO	DE	GA	PSO	SA
ACO	0.00%	16.13%	4.11%	1.59%	0.41%	ACO	0.00%	11.41%	3.96%	1.02%	0.44%
DE	5.31%	0.00%	6.55%	1.62%	0.44%	DE	10.49%	0.00%	9.47%	0.99%	0.36%
GA	3.28%	16.58%	0.00%	2.58%	0.41%	GA	5.32%	11.76%	0.00%	1.50%	0.39%
PSO	4.08%	13.86%	2.82%	0.00%	0.40%	PSO	7.46%	10.50%	2.79%	0.00%	0.33%
SA	4.77%	11.61%	2.91%	0.54%	0.00%	SA	7.63%	10.89%	2.99%	0.31%	0.00%
sum	17.44%	58.19%	16.40%	6.33%	1.66%	**sum**	30.90%	44.57%	19.20%	3.82%	1.51%

5.3. Runtime analysis

We measured the average runtime across 20 runs for both adaptive and static configurations. For the BBOB benchmark suite, static configuration took 542.6 sec. per run on average, while the adaptive setup with the UCB + Weakest strategy required 543.3 sec. on average, yielding an overhead of 0.13%. In the TSP domain, the respective average times were equal to 753.3 and 754.1 sec., leading to an overhead of 0.11%. The results confirm that the adaptation mechanism imposes a negligible computational cost, while offering significant performance benefits. Moreover, by avoiding the need to pre-evaluate all possible static configurations (e.g., 1001 possible configurations for 10 islands and 5 metaheuristics), the adaptive approach significantly reduces the time and effort required to tune the algorithm for new problem domains.

Furthermore, optimal static configurations vary by problem domain, making the results non-transferable without extensive recalculation. In contrast, adaptive strategies dynamically adjust in a single run, eliminating the need for exhaustive tests. Time advantage grows exponentially with more islands or metaheuristics - e.g. increasing from 5 to 6 metaheuristics raises the number of possible configurations from 1001 to 3003.

6. Conclusions

This paper introduces an adaptive metaheuristic selection strategy for island-based optimization, addressing the limitations of static algorithm assignments. Through extending the Diversity-driven Cooperating Portfolio of Metaheuristics (DdCPM) framework, we propose novel adaptation mechanisms that dynamically reconfigure the metaheuristic portfolio based on performance indicators and population characteristics.

Our experimental results demonstrate that adaptive selection strategies, particularly the combination of Upper Confidence Bound (UCB) for algorithm replacement and Weakest Performance for island selection, outperform the best static metaheuristic configurations. This improves solution quality while eliminating exhaustive pre-tuning and making optimization more efficient and flexible across diverse problem domains. The adaptive approach offers a flexible solution for practitioners, reducing the need for expert knowledge about the algorithm's performance. It is particularly advantageous when the optimization context changes over time (e.g. in smart grids [2] or manufacturing scheduling [37, 38]) or when the best-suited metaheuristic for a given problem is unknown.

The findings suggest that adaptive metaheuristic selection provides a promising direction for future research in hybrid optimization frameworks. Potential future work includes exploring alternative adaptation mechanisms, including the use of machine learning techniques for predictive algorithm selection.

Acknowledgements

Adam Żychowski was funded by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) program. Xin Yao's work is partially supported by an internal seed grant from Lingnan University. Jacek Mańdziuk was partially supported by the National Science Centre, Poland, grant number 2023/49/B/ST6/01404.

References

[1] Akay, R., Basturk, A., Kalinli, A., Yao, X., 2017. Parallel population-based algorithm portfolios: An empirical study. Neurocomputing 247, 115–125.

[2] Alam, M.R., St-Hilaire, M., Kunz, T., 2016. Computational methods for residential energy cost optimization in smart grids: A survey. ACM Computing Surveys 49, 1–34.

[3] Alba, E., Tomassini, M., 2002. Parallelism and evolutionary algorithms. IEEE transactions on evolutionary computation 6, 443–462.

[4] Amey, C., Machta, J., 2018. Analysis and optimization of population annealing. Physical Review E 97, 033301.

[5] Bäck, T., Schwefel, H.P., 1993. An overview of evolutionary algorithms for parameter optimization. Evolutionary computation 1, 1–23.

[6] Derbel, B., Verel, S., 2011. DAMS: distributed adaptive metaheuristic selection, in: Genetic and Evolutionary Computation Conference (GECCO), pp. 1955–1962.

[7] Dorigo, M., Birattari, M., Stutzle, T., 2006. Ant colony optimization. IEEE computational intelligence magazine 1, 28–39.

[8] El-Abd, M., Kamel, M.S., 2009. Black-box optimization benchmarking for noiseless function testbed using particle swarm optimization, in: Genetic and Evolutionary Computation Conference (GECCO), pp. 2269–2274.

[9] Finck, S., Hansen, N., Ros, R., Auger, A., 2010. Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report. Citeseer.

[10] Gilardi, G., 2025. Metaheuristic optimization using population-based simulated annealing. https://github.com/gabrielegilardi/SimulatedAnnealing.

[11] Hansen, N., Auger, A., Ros, R., Mersmann, O., Tušar, T., Brockhoff, D., 2021. Coco: A platform for comparing continuous optimizers in a black-box setting. Optimization Methods and Software 36, 114–144.

[12] Harada, T., Alba, E., 2020. Parallel genetic algorithms: a useful survey. ACM Computing Surveys 53, 1–39.

[13] Hukushima, K., Iba, Y., 2003. Population annealing and its application to a spin glass, in: AIP Conference Proceedings, pp. 200–206.

[14] Jamal, A., Al-Ahmadi, H.M., Butt, F.M., Iqbal, M., Almoshaogeh, M., Ali, S., 2021. Metaheuristics for traffic control and optimization: Current challenges and prospects. IntechOpen.

[15] Jankee, C., Verel, S., Derbel, B., Fonlupt, C., 2016. A fitness cloud model for adaptive metaheuristic selection methods, in: International Conference on Parallel Problem Solving from Nature, Springer. pp. 80–90.

[16] Kennedy, J., Eberhart, R., 1995. Particle swarm optimization, in: Proceedings of International Conference on Neural Networks, pp. 1942–1948.

[17] Kocsis, L., Szepesvári, C., 2006. Bandit based monte-carlo planning, in: European conference on machine learning, Springer. pp. 282–293.

[18] Kowalski, P.A., Kucharczyk, S., Mańdziuk, J., 2025. Constrained hybrid metaheuristic algorithm for probabilistic neural networks learning. Information Sciences 713, 122185.

[19] Liao, T., Molina, D., Stutzle, T., Montes de Oca, M.A., Dorigo, M., 2012. An aco algorithm benchmarked on the bbob noiseless function testbed, in: Genetic and Evolutionary Computation Conference (GECCO), pp. 159–166.

[20] Mahajan, A., Teneketzis, D., 2008. Multi-armed bandit problems, in: Foundations and applications of sensor management, pp. 121–151.

[21] Mańdziuk, J., 2019. New shades of the Vehicle Routing Problem: Emerging problem formulations and Computational Intelligence solution methods. IEEE Transactions on Emerging Topics in Computational Intelligence 3, 230–244.

[22] Mańdziuk, J., Żychowski, A., 2016. A memetic approach to vehicle routing problem with dynamic requests. Applied Soft Computing 48, 522–534.

[23] Mi, M., Huifeng, X., Ming, Z., Yu, G., 2010. An improved differential evolution algorithm for TSP problem, in: 2010 International Conference on Intelligent Computation Technology and Automation, IEEE. pp. 544–547.

[24] Nicolau, M., 2009. Application of a simple binary genetic algorithm to a noiseless testbed benchmark, in: Genetic and Evolutionary Computation Conference (GECCO), pp. 2473–2478.

[25] Okulewicz, M., Zaborski, M., Mańdziuk, J., 2022. Self-adapting particle swarm optimization for continuous black box optimization. Applied Soft Computing 131, 109722.

[26] Rajwar, K., Deep, K., Das, S., 2023. An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges. Artificial Intelligence Review 56, 13187–13257.

[27] Razali, N.M., Geraghty, J., et al., 2011. Genetic algorithm performance with different selection strategies in solving TSP, in: Proceedings of the world congress on engineering, International Association of Engineers Hong Kong, China. pp. 1–6.

[28] Reinelt, G., 1991. TSPLIB — A traveling salesman problem library. ORSA journal on computing 3, 376–384.

[29] Shi, X.H., Liang, Y.C., Lee, H.P., Lu, C., Wang, Q., 2007. Particle swarm optimization-based algorithms for TSP and generalized TSP. Information processing letters 103, 169–176.

[30] Skolicki, Z., De Jong, K., 2005. The influence of migration sizes and intervals on island models, in: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, pp. 1295–1302.

[31] Soler-Dominguez, A., Juan, A.A., Kizys, R., 2017. A survey on financial applications of metaheuristics. ACM Computing Surveys 50, 1–23.

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

[33] Świechowski, M., Godlewski, K., Sawicki, B., Mańdziuk, J., 2023. Monte carlo tree search: A review of recent modifications and applications. Artificial Intelligence Review 56, 2497–2562.

[34] Tanabe, R., 2024. Benchmarking parameter control methods in differential evolution for mixed-integer black-box optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 712–721.

[35] Uliński, M., Żychowski, A., Okulewicz, M., Zaborski, M., Kordulewski, H., 2018. Generalized self-adapting particle swarm optimization algorithm, in: 15th International Conference on Parallel Problem Solving from Nature, Springer. pp. 29–40.

[36] Van Laarhoven, P.J., Aarts, E.H., van Laarhoven, P.J., Aarts, E.H., 1987. Simulated annealing. Springer.

[37] Waledzik, K., Mańdziuk, J., 2018. Applying hybrid Monte Carlo Tree Search methods to risk-aware project scheduling problem. Information Sciences 460-461, 450–468.

[38] Wari, E., Zhu, W., 2016. A survey on metaheuristics for optimization in food manufacturing industry. Applied Soft Computing 46, 328–343.

[39] Whitley, D., Rana, S., Heckendorn, R.B., 1999. The island model genetic algorithm: On separability, population size and convergence. Journal of Computing and Information Technology 7, 33–47.

[40] Yang, J., Shi, X., Marchese, M., Liang, Y., 2008. An ant colony optimization method for generalized TSP problem. Progress in natural science 18, 1417–1422.

[41] Żychowski, A., Yao, X., Mańdziuk, J., 2025a. Diversity-driven cooperating portfolio of metaheuristic algorithms. Genetic and Evolutionary Computation Conference (GECCO) [accepted].

[42] Żychowski, A., Yao, X., Mańdziuk, J., 2025b. Migrant selection in island-based optimization. 6th Polish Conference on Artificial Intelligence (PP-RAI) [accepted].

[43] Żychowski, A., Yao, X., Mańdziuk, J., 2025c. Migration timing in hybrid island-based metaheuristic algorithms. International Conference on Artificial Intelligence and Soft Computing (ICAISC) [accepted].